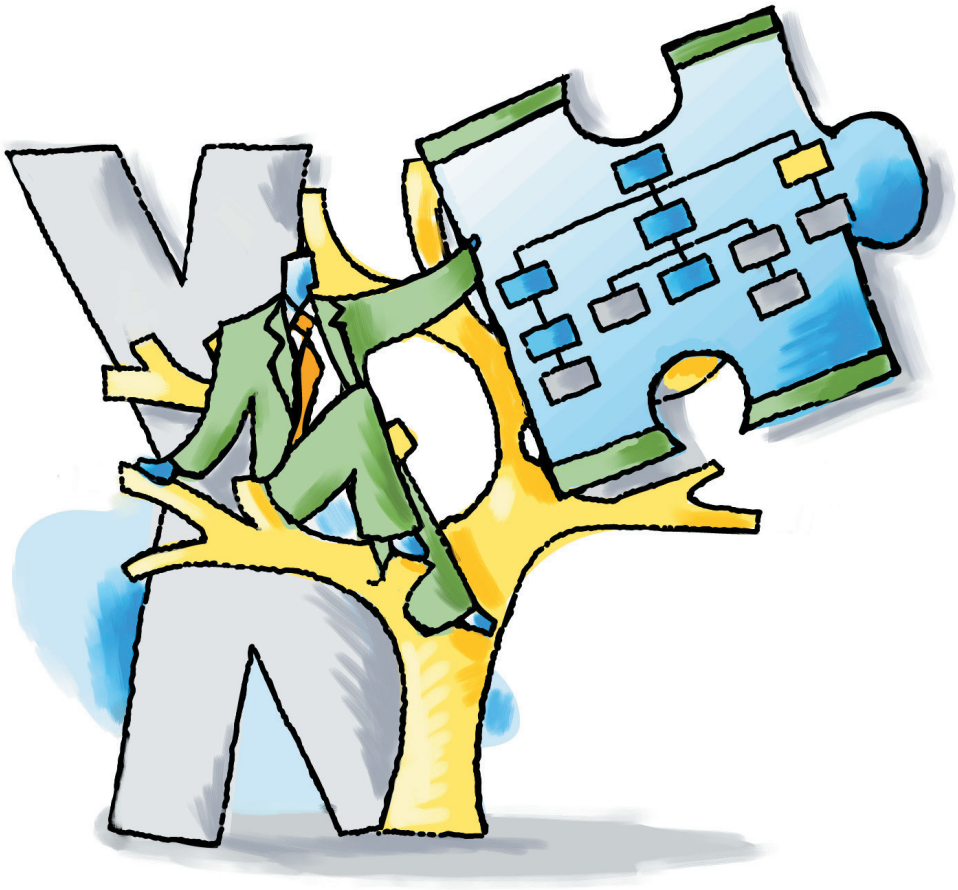
VARCHART XTree

.NET Edition 5.2
Benutzer- und Referenzhandbuch



NETRONIC

VARCHART XTree .NET Edition

Version 5.2

Benutzerhandbuch

NETRONIC Software GmbH
Pascalstraße 15
52076 Aachen
Deutschland
Tel: +49 (0) 2408 141-0
Fax: +49 (0) 2408 141-33
E-Mail sales@netronic.de
www.netronic.de

© Copyright 2020 NETRONIC Software GmbH
Alle Rechte vorbehalten.

Die Informationen im vorliegenden Benutzerhandbuch werden ohne Rücksicht auf einen eventuellen Patentschutz veröffentlicht. Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt. Bei der Zusammenstellung von Texten und Abbildungen wurde mit größter Sorgfalt vorgegangen. Trotzdem können Fehler nicht vollständig ausgeschlossen werden. Herausgeber und Autoren können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Das illegale Kopieren und Vertreiben dieses Produktes stellt einen Diebstahl geistigen Eigentums dar und wird von NETRONIC Software GmbH strafrechtlich verfolgt.

Alle Rechte vorbehalten, auch die der fotomechanischen Wiedergabe und der Speicherung in elektronischen Medien. Die gewerbliche Nutzung der in diesem Buch gezeigten Abbildungen ist nicht zulässig.

Microsoft Windows, Microsoft Explorer, Microsoft Visual Basic und Microsoft Visual Studio sind Warenzeichen der MICROSOFT Corp.

Bearbeitungsstand: 27 April 2020

Inhaltsverzeichnis

1 Einleitung 9

1.1 VARCHART XTree auf einen Blick 9
1.2 Installation 14
1.3 Lizenzierung 16
1.4 Auslieferung 17
1.5 Verwendung der deutschen Version 19
1.6 Unterstützung und Beratung 21

2 Tutorium 23

2.1 Überblick 23
2.2 VARCHART XTree in einem Formular plazieren 24
2.3 VARCHART XTree automatisch skalieren 25
2.4 Schnittstelle einrichten 26
2.5 Der erste Lauf 29
2.6 Daten aus einer Datei einlesen 32
2.7 Markierung von Knoten festlegen 34
2.8 Filter für Knoten festlegen 35
2.9 Knotenaussehen festlegen 37
2.10 Knotenformate festlegen 41
2.11 Das Aussehen von Verbindungen festlegen 44
2.12 Die Baumstruktur festlegen 45
2.13 Vertikale und horizontale Anordnung von Baumstrukturen 47
2.14 Baumstrukturen kollabieren und expandieren 51
2.15 TreeView-Stil 55
2.16 Diagramm drucken 58
2.17 Diagramm exportieren 59
2.18 Konfigurationseinstellungen speichern 60

3 Wichtige Konzepte 61

3.1 Boxen 61
3.2 Datentabellen 65
3.3 Datumsangaben und Zeitumstellung 74
3.4 Drag & Drop 76
3.5 Ereignisse 78
3.6 Filter 79
3.7 Grafikformate 80
3.8 Horizontale/vertikale Anordnung 85
3.9 Knoten 90
3.10 Knotenaussehen 94
3.11 Knotenformat 96
3.12 Kollabieren und Expandieren 99
3.13 Komplettansicht (World View) 102
3.14 Laufzeitsicherheitsrichtlinien für den Einsatz im Internet Explorer 104
3.15 Legendenansicht (Legend View) 106
3.16 Maximale Höhe des Baum-Diagramms 108
3.17 Plattformen x86 und x64 109
3.18 Schreiben von PDF-Dateien 113
3.19 Sprachanpassung von Textausgaben 116
3.20 Statuszeilentext 118
3.21 Struktur 119
3.22 Tooltips zur Laufzeit 121
3.23 TreeView-Stil 122
3.24 Vertikale Ebenen 123
3.25 Viewer Metafile (*.vmf) 125
3.26 Zuordnungstabellen 126

4 Eigenschaftenseiten und Dialogfelder 131

4.1 Allgemeines 131
4.2 Eigenschaftenseite "Allgemeines" 132
4.3 Eigenschaftenseite "Außenbereich" 140

4.4 Eigenschaftenseite "Layout" 142
4.5 Eigenschaftenseite "Knoten" 144
4.6 Eigenschaftenseite "Zusätzliche Ansichten" 147
4.7 Eigenschaftenseite "Objekte" 151
4.8 Dialogfeld "Filter verwalten" 153
4.9 Dialogfeld "Filter bearbeiten" 155
4.10 Dialogfeld "Zuordnungstabellen verwalten" 159
4.11 Dialogfeld "Zuordnungstabelle bearbeiten" 161
4.12 Dialogfeld "Zuordnung einstellen" 163
4.13 Dialogfeld "Knotenaussehen verwalten" 165
4.14 Dialogfeld "Knotenaussehen bearbeiten" 169
4.15 Dialogfeld "Boxen verwalten" 173
4.16 Dialogfeld "Box bearbeiten" 176
4.17 Dialogfeld "Boxformate/Knotenformate verwalten" 178
4.18 Dialogfeld "Boxformat bearbeiten" 180
4.19 Dialogfeld "Knotenformat bearbeiten" 183
4.20 Dialogfeld "Linienattribute bearbeiten" 188
4.21 Dialogfeld "Musterattribute bearbeiten" 189
4.22 Dialogfeld "Texte, Grafiken und Legende festlegen" 190
4.23 Dialogfeld "Legendenattribute" 194
4.24 Dialogfeld "Lizenzierung" 196
4.25 Dialogfeld "Lizenzinformationen anfordern" 198

5 Benutzerschnittstelle 199

5.1 Übersicht 199
5.2 Navigation im Diagramm 200
5.3 Zoomen 201
5.4 Knotendaten bearbeiten 203
5.5 Navigation mit Hilfe der Tastatur 205
5.6 Knoten erzeugen 206
5.7 Knoten markieren 209
5.8 Knoten löschen, ausschneiden, kopieren und einfügen 210
5.9 Knoten bearbeiten 211

5.10 Knoten mit seinem Teilbaum umhängen 212
5.11 Teilstrukturen horizontal und vertikal anordnen 215
5.12 Teilstrukturen kollabieren und expandieren 218
5.13 Seite einrichten 220
5.14 Druckvorschau 224
5.15 Kontextmenü für das Diagramm 227
5.16 Kontextmenü für Knoten 230
5.17 Kontextmenü für die Legende 233

6 Häufig gestellte Fragen 235

6.1 Was muss ich tun, wenn ich von VARCHART XGantt 4.4 für .NET nach XGantt 5.0 umsteigen möchte? 235
6.2 Was muss ich tun, wenn ich im Rahmen eines Service Releases auf einen neuen Build von VARCHART XGantt umsteigen möchte? 236
6.3 Warum erscheint eine Fehlermeldung, wenn man bei Visual Studio 2010 ein neues Projekt erzeugt und versucht, das Steuerelement auf die Form zu ziehen? 238
6.4 Wie kann das VARCHART Windows Forms neu lizenziert werden? 239
6.5 Wieso können Knoten u. U. nicht interaktiv erzeugt werden? 240
6.6 Wie verhindert man das interaktive Erzeugen von Knoten? 241
6.7 Wie lassen sich die Standard-Kontextmenüs abschalten? 242
6.8 Wie lässt sich die Performance verbessern? 243
6.9 Fehlermeldungen 244
6.10 Was tun, wenn das Steuerelement unerwartet nicht in allen Benutzerkonten eines Rechners funktioniert? 245
6.11 Sind alle Fonts verwendbar? 246

7 API Referenz 247

7.1 Objekttypen 247
7.2 VcBorderArea 248
7.3 VcBorderBox 250
7.4 VcBox 259
7.5 VcBoxCollection 272
7.6 VcBoxFormat 279

7.7 VcBoxFormatCollection 285
7.8 VcBoxFormatField 292
7.9 VcDataRecord 301
7.10 VcDataRecordCollection 307
7.11 VcDataTable 315
7.12 VcDataTableCollection 319
7.13 VcDataTableField 326
7.14 VcDataTableFieldCollection 333
7.15 VcFilter 339
7.16 VcFilterCollection 346
7.17 VcFilterSubCondition 353
7.18 VcLegendView 357
7.19 VcMap 365
7.20 VcMapCollection 372
7.21 VcMapEntry 380
7.22 VcNode 389
7.23 VcNodeAppearance 402
7.24 VcNodeAppearanceCollection 424
7.25 VcNodeCollection 430
7.26 VcNodeFormat 434
7.27 VcNodeFormatCollection 439
7.28 VcNodeFormatField 446
7.29 VcPrinter 460
7.30 VcRect 479
7.31 VcTree 483
7.32 VcWorldView 591

8 Index 601

1 Einleitung

1.1 VARCHART XTree auf einen Blick

VARCHART XTree ist eine interaktive Grafikkomponente mit deren Hilfe Sie Ihre Daten in Form von Baumdiagrammen anzeigen, bearbeiten und drucken können. Es ist das perfekte Instrument, um alle Arten von hierarchischen Strukturen, wie Materialaufstellungen, Projektstrukturpläne, Organigramme oder Dateisysteme, darzustellen. VARCHART XTree kommt vor allem in der Produktionsplanung sowie im Projektmanagement, im Prozessmanagement und in Organisationsabteilungen zum Einsatz. Die Grafiken lassen sich einfach und schnell in bestehende Anwendungen einbinden. Dadurch wird der Zeitaufwand zwischen Planung und Einsatz erheblich verringert und Sie können Ihre Zeit für andere Geschäftsabläufe nutzen.

> Kurzer Überblick über die Leistungsmerkmale

- **Beschriftungsboxen**

Zusätzlich zur Baumstruktur lassen sich Informationsboxen, die Beschriftungen und Bilder enthalten können, frei platzieren.

- **Automatisches Layout**

Die Anordnung der Baumstruktur wird nur durch Daten bestimmt und muß nicht manuell bearbeitet werden. Dies ist z.B. sehr vorteilhaft bei Teilelisten, deren Daten meist aus Datenbanken stammen, sowie für größere Unternehmen mit kontinuierlicher Mitarbeiterfluktuation, deren Organigramme automatisch akutalisiert werden.

- **Datenschnittstelle**

Nutzen Sie die flexible Datenschnittstelle um bestehende Datenstrukturen einfach anzupassen. Wie in einer relationalen Datenbank können unterschiedliche Tabellen mit bestimmten Datenfeldern definiert und miteinander verbunden werden. Es steht ein CSV- Importfilter zur Verfügung, um die Anwendungsdaten einzulesen. Die Datenfelder einer Datentabelle können verwendet werden, um Knoten in Filtern und Zuordnungstabellen zu beschriften.

- **Filter**

Mit Hilfe von Filtern können Sie Knoten oder Verbindungen auswählen, die bestimmte Kriterien erfüllen, z.B. um Knoten im Diagramm hervorzuheben.

- **Grafik-Export**

Speichern Sie das Diagramm im gewünschten Grafikformat: PNG, BMP, EMF,GIF,TIF,JPG.

- **Intuitive Interaktionen**

Überarbeiten Sie die Darstellung am Bildschirm und verändern die Basisdaten der Benutzeraktionen.

- **Sprachunterstützung**

Das Produkt und die Dokumentation sind in Englisch und Deutsch verfügbar. darüber hinaus kann jedes Textelement im Diagramm zur Laufzeit durch einen Begriff Ihrer Wahl in beliebiger Sprache ersetzt werden. Unicodezeichen werden unterstützt. Die Zeichen aller Sprachen können gleichzeitig und unabhängig vom eingesetzten Betriebssystem verwendet werden.

- **Layout**

Die Knoten lassen sich vertikal, horizontal oder in kombinierter Weise anordnen. Neben dem allgemeinen Baumstrukturlayout können zur Darstellung auch spezielle Formen wie z.B. das Aussehen des Microsoft Explorers ausgewählt werden.

- **Legende**

Die Legende kann im Außenbereich des Diagramms positioniert werden und ist im Ausdruck und in den exportierten Grafiken sichtbar. Sie kann in einem definierbaren Matrixlayout übersichtlich gestaltet werden.

- **Verbindungen**

Das Aussehen von Verbindungslinien kann eingestellt werden.

- **Mehrfache Wurzeln**

Die oberste Ebene der Baumstruktur kann aus einem oder mehreren Knoten bestehen.

- **Navigationsfenster**

Aufgrund der integrierten Komplettansicht ist die Navigation in der Grafik sehr einfach.

- **Knotenaussehen**

Knoten können in verschiedenen Formen, Farben, Farbverläufen und eigenen Bitmaps dargestellt werden. Durch individuelle Tooltiptexte lassen sie sich

hervorheben und ergänzen. Das Aussehen von Knoten kann durch den Einsatz von Filtern und Zuordnungstabellen dynamisch an Ihre Daten gebunden werden. Die Knotenbeschriftung kann aus verschiedenen Datenfeldern bestehen, die sich innerhalb oder außerhalb der Knotengrenzen platzieren lassen.

- **Outlining**

Um größere Baumstrukturen bequem zu handhaben, können Teilbäume kollabiert und expandiert werden. Das Aussehen und die Information des anzuzeigenden Stellvertreterknotens können definiert werden.

- **Drucken**

Wählen Sie das Seitenlayout und lassen es sich in der integrierten Seitenvorschau anzeigen. Wählen Sie bestimmte Bereiche des Diagramms, die auf jeder Seite wiederholt werden sollen und geben Sie die Anzahl der Seiten vor, auf die das Diagramm bei der Ausgabe verteilt werden soll.

- **Eigenschaftenseiten**

Für jedes wichtige Objekt gibt es eine Eigenschaftenseite, die den Programmieraufwand erheblich reduziert. Auf den Eigenschaftenseiten lässt sich beinahe jede Einzelheit der Komponente intuitiv anpassen. Die leistungsfähige API bietet zur Laufzeit weitere Möglichkeiten. Mit Hilfe von Ereignissen kann die Anwendung auf Benutzeraktionen in einer bestimmten Art und Weise reagieren (z.B. um eingegebene Daten zu überprüfen).

- **Strukturcode**

Die Baumstruktur kann aus einem Strukturcode oder einer Eltern-Kind-Beziehung abgeleitet werden. Der Aufbau des Strukturcodes kann frei definiert werden.

- **Beschriftungsboxen**

Zusätzlich zur Baumstruktur lassen sich Informationsboxen, die Beschriftungen und Bilder enthalten können, frei platzieren.

- **Automatisches Layout**

Die Anordnung der Baumstruktur wird nur durch Daten bestimmt und muß nicht manuell bearbeitet werden. Dies ist z.B. sehr vorteilhaft bei Teilelisten, deren Daten meist aus Datenbanken stammen, sowie für größere Unternehmen mit kontinuierlicher Mitarbeiterfluktuation, deren Organigramme automatisch akutalisiert werden.

- **Datenschnittstelle**

Nutzen Sie die flexible Datenschnittstelle um bestehende Datenstrukturen einfach anzupassen. Wie in einer relationalen Datenbank können

unterschiedliche Tabellen mit bestimmten Datenfeldern definiert und miteinander verbunden werden. Es steht ein CSV- Importfilter zur Verfügung, um die Anwendungsdaten einzulesen. Die Datenfelder einer Datentabelle können verwendet werden, um Knoten in Filtern und Zuordnungstabellen zu beschriften.

- **Filter**

Mit Hilfe von Filtern können Sie Knoten oder Verbindungen auswählen, die bestimmte Kriterien erfüllen, z.B. um Knoten im Diagramm hervorzuheben.

- **Grafik-Export**

Speichern Sie das Diagramm im gewünschten Grafikformat: PNG, BMP, EMF,GIF,TIF,JPG.

- **Intuitive Interaktionen**

Überarbeiten Sie die Darstellung am Bildschirm und verändern die Basisdaten der Benutzeraktionen. So können z. B. einzelne Knoten und Teilbäume mit Drag & Drop verschoben oder kopiert werden.

- **Sprachunterstützung**

Das Produkt und die Dokumentation sind in Englisch und Deutsch verfügbar. darüber hinaus kann jedes Textelement im Diagramm zur Laufzeit durch einen Begriff Ihrer Wahl in beliebiger Sprache ersetzt werden. Unicodezeichen werden unterstützt. Die Zeichen aller Sprachen können gleichzeitig und unabhängig vom eingesetzten Betriebssystem verwendet werden.

- **Layout**

Die Knoten lassen sich vertikal, horizontal oder in kombinierter Weise anordnen. Neben dem allgemeinen Baumstrukturlayout können zur Darstellung auch spezielle Formen wie z.B. das Aussehen des Microsoft Explorers ausgewählt werden.

- **Legende**

Die Legende kann im Außenbereich des Diagramms positioniert werden und ist im Ausdruck und in den exportierten Grafiken sichtbar. Sie kann in einem definierbaren Matrixlayout übersichtlich gestaltet werden.

- **Verbindungen**

Das Aussehen von Verbindungslinien kann eingestellt werden.

- **Mehrfache Wurzeln**

Die oberste Ebene der Baumstruktur kann aus einem oder mehreren Knoten bestehen.

- **Navigationsfenster**

Aufgrund der integrierten Komplettansicht ist die Navigation in der Grafik sehr einfach.

- **Knotenaussehen**

Knoten können in verschiedenen Formen, Farben, Farbverläufen und eigenen Bitmaps dargestellt werden. Durch individuelle Tooltiptexte lassen sie sich hervorheben und ergänzen. Das Aussehen von Knoten kann durch den Einsatz von Filtern und Zuordnungstabellen dynamisch an Ihre Daten gebunden werden. Die Knotenbeschriftung kann aus verschiedenen Datenfeldern bestehen, die sich innerhalb oder außerhalb der Knotengrenzen platzieren lassen.

- **Outlining**

Um größere Baumstrukturen bequem zu handhaben, können Teilbäume kollabiert und expandiert werden. Das Aussehen und die Information des anzuzeigenden Stellvertreterknotens können definiert werden.

- **Drucken**

Wählen Sie das Seitenlayout und lassen es sich in der integrierten Seitenvorschau anzeigen. Wählen Sie bestimmte Bereiche des Diagramms, die auf jeder Seite wiederholt werden sollen und geben Sie die Anzahl der Seiten vor, auf die das Diagramm bei der Ausgabe verteilt werden soll.

- **Eigenschaftenseiten**

Für jedes wichtige Objekt gibt es eine Eigenschaftenseite, die den Programmieraufwand erheblich reduziert. Auf den Eigenschaftenseiten lässt sich beinahe jede Einzelheit der Komponente intuitiv anpassen. Die leistungsfähige API bietet zur Laufzeit weitere Möglichkeiten. Mit Hilfe von Ereignissen kann die Anwendung auf Benutzeraktionen in einer bestimmten Art und Weise reagieren (z.B. um eingegebene Daten zu überprüfen).

- **Strukturcode**

Die Baumstruktur kann aus einem Strukturcode oder einer Eltern-Kind-Beziehung abgeleitet werden. Der Aufbau des Strukturcodes kann frei definiert werden.

Hinweis: Alle Codebeispiele in dieser Dokumentation wurden in VB.NET und C# geschrieben.

1.2 Installation

Für die Entwicklung von Anwendungen auf der Basis von .NET benötigen Sie eine Entwicklungsumgebung wie zum Beispiel Microsoft Visual Studio 2010 oder neuer. Die Entwicklungsumgebung muss mindestens .NET Framework 2.0 unterstützen und mit Mixed-Mode-.NET-Komponenten kompatibel sein. Als Betriebssystem kann nur Windows ab XP Service Pack 3 aufwärts verwendet werden und zwar in 32bit- oder 64bit-(x64)-Editionen.

Um das Steuerelement VARCHART XTree .NET auf Ihrem Rechner zu installieren, führen Sie bitte das Installationsprogramm aus.

Standardmäßig wird das Steuerelement mit allen zugehörigen Dateien unterhalb des Ordners

c:\Programme\NETRONIC (32bit-Windows) bzw.

c:\Programme (x86)\NETRONIC (64bit-Windows)

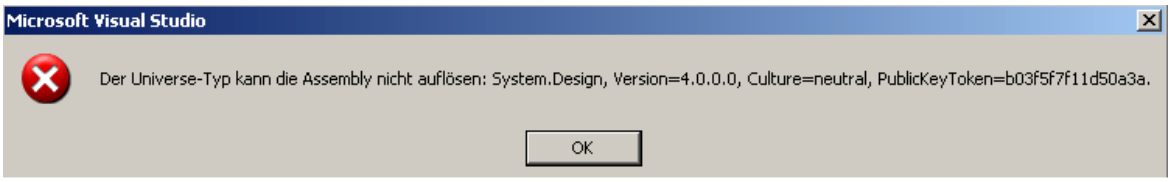
abgelegt.

Nach der Installation sollten Sie das Steuerelement in die Toolbox Ihrer Entwicklungsumgebung aufnehmen.

Wir stellen Ihnen die nötigen Schritte am Beispiel von Microsoft Visual Studio vor; andere Entwicklungsumgebungen arbeiten ähnlich:

1. Legen Sie in Visual Studio ein neues Projekt vom Typ **Windows-Anwendung** an. Es spielt dabei keine Rolle, welche Sprache Sie wählen. Wichtig ist, dass die Toolbox sichtbar ist. Sollte dies nicht der Fall sein, so können Sie sie unter **Ansicht** den Menüpunkt **Toolbox** einschalten.
2. Klicken Sie mit der rechten Maustaste auf die Toolbox und wählen im Kontextmenü **Elemente auswählen…**.
3. Mit der Schaltfläche **Durchsuchen** des Registers **.NET Framework-Komponenten** können Sie die Assembly **NETRONIC.Xtree.dll** aus dem entsprechenden Installationsordner auswählen. Nach dem Bestätigen mit **OK** erscheint das Symbol für VARCHART Xtree.NET als letzter Eintrag in der Toolbox.
4. Wichtig für die Nutzer von **Visual Studio 2010**: **Bevor** Sie das Steuerelement auf die Form ziehen, müssen Sie unter **Projekteigenschaften/Anwendung** (C#) bzw. **Erweiterte Kompilereinstellungen** (VB) das Zielframework von **.NET Framework 4 Client Profile** auf **.NET Framework 4** ändern, da ersteres nicht die von den Eigenschaftenseiten zur Designzeit benötigte System.Design.dll enthält. Falls Sie das Framework nicht umstellen,

erscheint folgende Fehlermeldung, wenn Sie das Steuerelement auf die Form ziehen möchten:



Sie können VARCHART XTree auch bedienerlos installieren. Dazu geben Sie ein:

start/wait (NameDerSetupDatei).exe /L1031 /s /V"/qn ADDLOCAL=ALL"

Damit läuft die Installation ohne jegliche Benutzereingabe und ohne Fortschrittsmeldungen auf dem Bildschirm ab. Bitte beachten Sie Folgendes:

1. Der aufrufende Prozess, z.B. eine DOS-Box, muss mit Administratorrechten gestartet werden; andernfalls erscheint eine UAC-Abfrage, die eine Benutzereingabe notwendig macht.
2. Sprachparameter: /L1031: Installation in deutsch; /L1033: Installation in englisch
3. Fortschrittsmeldungen: /qb: Es erscheinen Fortschrittsmeldungen; /qn: Es erscheinen keine Fortschrittsmeldungen, d.h. Sie sehen am Bildschirm nichts.
4. Start/wait sollte man benutzen, wenn die Installation über eine Batch-Datei abläuft; andernfalls liefe die Batch-Datei während der Installation parallel weiter.

1.3 Lizenzierung

1.3.1 Entwickler-Lizenzen

Zum Lizenzieren von VARCHART XTree .NET ziehen Sie nun das Steuerelement von der Toolbox auf die Form.

Öffnen Sie dann die **Eigenschaftenseiten**, indem Sie mit der rechten Maustaste auf das Steuerelement klicken.

Auf der Registerkarte **Allgemeines** gelangen Sie über die Schaltfläche **Lizenzierung** in den Lizenzierungsdialog.

Die Schaltfläche **Lizenzinformation von NETRONIC anfordern** führt zu einem weiteren Dialogfeld, das den Eintrag der Lizenzierungsinformationen ermöglicht.

Für die Registrierung benötigen wir drei Informationen:

- die Lizenznummer
- Ihren Namen
- den Firmennamen

Tragen Sie die erforderlichen Daten in die vorgesehenen Felder ein. Die Lizenznummer "TXnnnn" finden Sie auf dem Lieferschein zu Ihrer Bestellung.

Durch Klick auf die Schaltfläche **E-Mail an NETRONIC senden...** wird eine E-Mail erstellt, die Sie uns nur noch schicken müssen. Sie können aber auch selber eine E-Mail verfassen, in der Sie uns die benötigten Daten mitteilen. Richten Sie bitte alle Lizenzierungsanfragen an license@netronic.com.

Danach erhalten Sie von uns umgehend eine passende Lizenzdatei, die Sie bitte im Installationsordner (Ordner mit der Datei NETRONIC.XTree.Web.dll) ablegen. Der Lizenzierungsvorgang ist nun abgeschlossen.

1.4 Auslieferung

Wenn Sie eine mit VARCHART XTree .NET entwickelte Anwendung an Ihre Kunden weitergeben möchten, müssen Sie die im folgenden aufgeführten Dateien mit ausliefern. Alle anderen Dateien, die zum Produkt VARCHART XTree .NET gehören, werden nur für die Entwicklungsphase benötigt und dürfen **nicht** an Ihre Kunden weitergegeben werden.

> Framework .NET 2.0/3.0/3.5

- **In der entsprechenden Prozessorvariante für x86 oder x64**

NETRONIC.XTree.dll

NETRONIC.XTreed.dll (wenn Sie die deutsche Version nutzen möchten)

NETRONIC.XTreec.dll (wenn Sie die chinesische Version nutzen möchten)

mfc80u.dll

mfcm80u.dll

msvcp80.dll

msvcr80.dll

Die Installation der Bibliotheken *mfc80u.dll*, *msvcp80.dll*, *mfcm80u.dll* und *msvcr80.dll* muss über die im Unterverzeichnis **redist** mitgelieferten Setup-Dateien *vcredist_vs2005sp1_x86.exe* bzw. *vcredist_vs2005sp1_x64.exe* erfolgen.

Nähere Informationen dazu bietet folgende Seite:

msdn2.microsoft.com/en-us/library/ms235285(VS.80).aspx.

> Framework .NET 4.0/4.5

- **In der entsprechenden Prozessorvariante für x86 oder x64**

NETRONIC.XTree.dll

NETRONIC.XTreed.dll (wenn Sie die deutsche Version nutzen möchten)

NETRONIC.XTreec.dll (wenn Sie die vereinfachte chinesische Version nutzen möchten)

mfc100u.dll

mfcm100u.dll

msvcp100.dll

msvcr100.dll

Die Installation der Bibliotheken *mfc100u.dll, msvcp100.dll, mfcm100u.dll* und *msvcr100.dll* kann entweder direkt durch Kopieren in das Windows-Systemverzeichnis oder über die im Unterverzeichnis **redist** mitgelieferten Setup-Dateien *vcredist_vs2010_x86.exe* bzw. *vcredist_vs2010_x64.exe* erfolgen.

VARCHART XTree.NET wird für folgende Plattformen angeboten:

- Windows 8
- Windows 7
- Windows Server 2008
- Windows Vista
- Windows Server 2003
- Windows XP ab SP3

unter Verwendung des .NET Frameworks ab Version 2.0 (nähere Informationen dazu finden Sie unter

msdn.microsoft.com/netframework/technologyinfo/sysreqs/default.aspx)

Tipp

So überprüfen Sie, welches .NET Framework bereits installiert ist:

In der **Systemsteuerung** wählen Sie **Software** und suchen in der Programmliste nach dem **Microsoft .NET Framework** Eintrag.

1.5 Verwendung der deutschen Version

Die VARCHART XTree .NET Edition ist in Deutsch und Englisch verfügbar. Bei der deutschen Version wird während der Installation zusätzlich zur Control-Assembly NETRONIC.XTree.dll die Ressource-Assembly NETRONIC.XTree**d**.dll ins Installationsverzeichnis kopiert.

Verwendung zur Designzeit

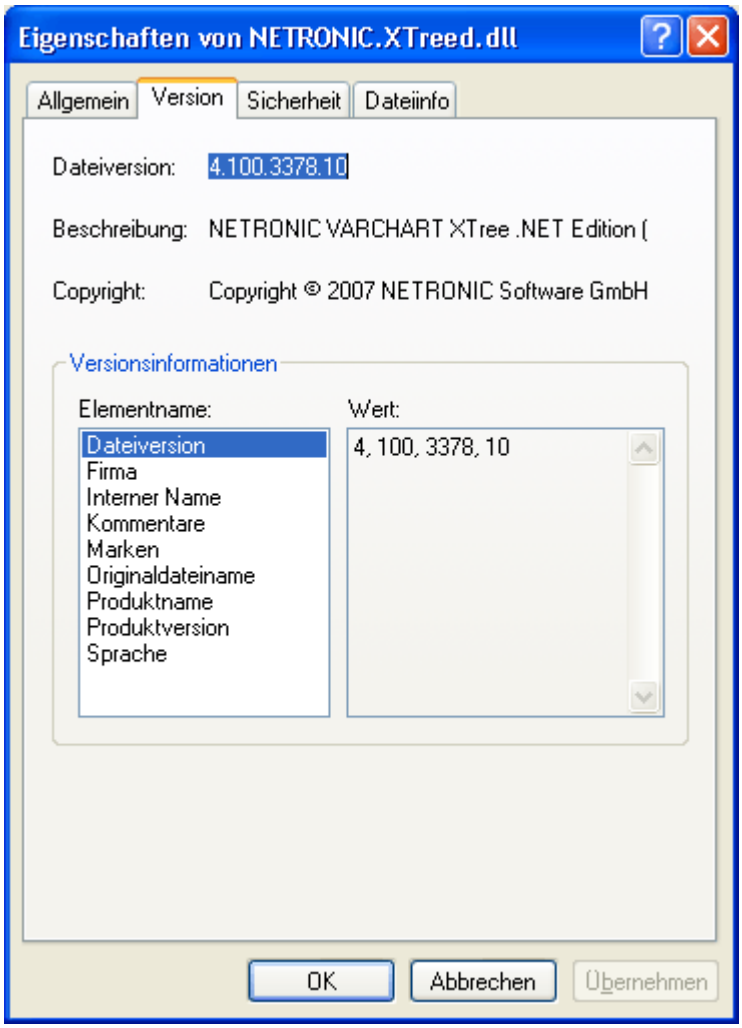
Wenn in den **Regionalen Einstellungen** (Systemsteuerung, Regions- und Sprachoptionen, Regionale Einstellungen) **Deutsch** eingestellt ist, wird die Ressource-Assembly aus dem Installationsverzeichnis geladen und die deutschen Dialoge und Eigenschaftenseiten stehen zur Designzeit zur Verfügung.

Verwendung zur Laufzeit

Um sicher zu stellen, dass auch zur Laufzeit auf die Ressource-Assembly zugegriffen wird und deutsche Dialoge zur Verfügung stehen, muss die Ressource-Assembly ins Applikationsverzeichnis kopiert werden. Dazu muss im Projekt ein Verweis auf die Assembly eingetragen werden ("Verweis hinzufügen").

Tipp: Da die Entwicklungsumgebung standardmäßig den "Copy-Local"-Parameter auf **False** setzt, muss dieser noch manuell auf **True** gesetzt werden. Wenn das Projekt dann neu erstellt wird, wird auch die Ressource-Assembly automatisch in das passende Applikationsverzeichnis kopiert und von dort geladen.

Bei eventuell auftretenden Problemen sollte überprüft werden, ob die Dateiversionsnummern der beiden Assemblys übereinstimmen (im Windows-Explorer das Kontextmenü der Datei aufrufen, Menüpunkt **Eigenschaften**, Registerkarte **Version**).


Eigenschaften von NETRONIC.XTreed.dll
Allgemein
Version
Sicherheit
Dateiinfo
Dateiversion:
4.100.3378.10
Beschreibung:
NETRONIC VARCHART XTree .NET Edition (
Copyright:
Copyright © 2007 NETRONIC Software GmbH
Versionsinformationen
Elementname:
Wert:
Dateiversion
Firma
Interner Name
Kommentare
Marken
Originaldateiname
Produktname
Produktversion
Sprache
4, 100, 3378, 10
OK
Abbrechen
Übernehmen

1.6 Unterstützung und Beratung

Sie wissen nicht, ob VARCHART XTree die speziellen Anforderungen Ihres Baumdiagramms erfüllt?

Sie wollen sich eine Vorstellung davon machen, wie viel Aufwand es ist, die eine oder andere Anforderung Ihres Baumdiagramms zu realisieren?

Sie machen gerade den Anfangstest mit VARCHART XTree und möchten eine ganz spezifische Anforderung Ihres Baumdiagramms programmieren?

Bei der Beantwortung dieser und anderer Fragen helfen wir Ihnen gerne weiter:

NETRONIC Software GmbH

Pascalstraße 15

52076 Aachen

Deutschland

Tel.: +49-2408-141-0

Fax: +49-2408-141-33

E-Mail: support@netronic.de

www.netronic.de

... übrigens, Sie können mit uns einen Wartungs- und Supportservice vereinbaren. Dann kommen Sie in den Genuss sofortiger Hilfe auch über den kostenfreien Support während der 30-tägigen Testphase hinaus. Der Wartungs- und Supportservice umfasst folgende Leistungen:

- Support-Hotline
- Qualifizierte und ausführliche Beratung in allen Anwendungsfragen
- Beseitigung von möglichen Fehlern in der gelieferten Software
- Upgrade auf ein neues Release von VARCHART XTree für Entwicklungs- und Laufzeitversionen

2 Tutorium

2.1 Überblick

In diesem Kapitel machen wir Sie mit den Grundlagen und Grundbegriffen von VARCHART XTree vertraut, die notwendig sind, um die Komponente in eigene Anwendungen integrieren zu können.

Wir werden Schritt für Schritt die bedeutsamen Aspekte von VARCHART XTree für die Anwendungsentwicklung erläutern und auf die vielfältigen Gestaltungsmöglichkeiten näher eingehen. Aus diesem Grund empfiehlt es sich, dieses Kapitel sequenziell durchzuarbeiten. Die anderen Kapitel des Handbuchs sind eher zum Nachschlagen gedacht:

- **Eigenschaftenseiten und Dialogfelder**

  Hier finden Sie umfassende Informationen zu den Eigenschaftenseiten und Dialogen, die Ihnen zur Entwurfszeit erlauben, VARCHART XTree nach Wunsch zu konfigurieren, ohne dass Sie dafür Programm-Code schreiben müssen.

- **Elemente der Benutzerschnittstelle**

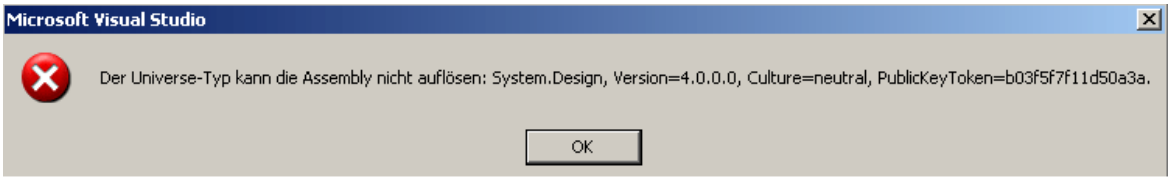
  In diesem Kapitel werden die bereits im Diagramm vorhandenen Interaktionen beschrieben. Details der Benutzerschnittstelle können individuell angepasst bzw. abgewandelt werden.

- **API-Referenz**

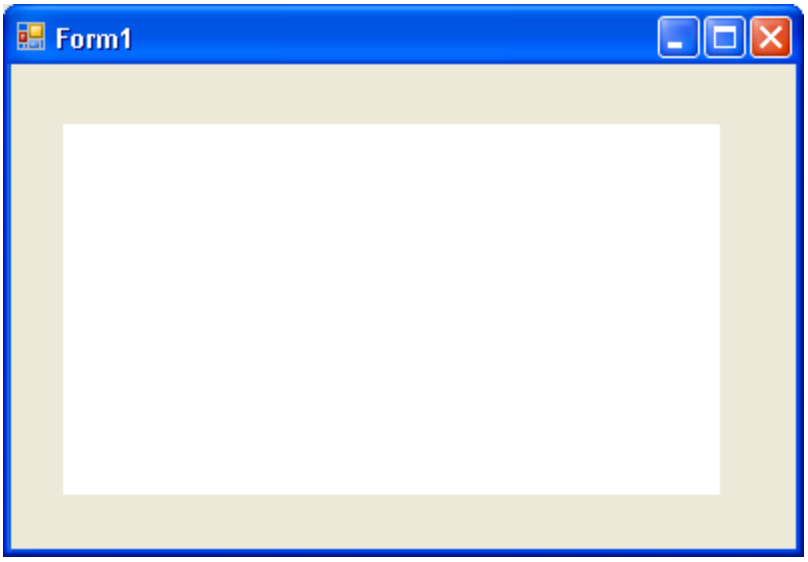
  In diesem Kapitel finden Sie ausführliche Informationen zu allen Objekten, Eigenschaften, Methoden und Ereignissen, die Ihnen VARCHART XTree bietet.

2.2 VARCHART XTree in einem Formular plazieren

Wichtig für die Nutzer von **Visual Studio 2010**: **Bevor** Sie das Steuerelement auf die Form ziehen, müssen Sie unter **Projekteigenschaften/Anwendung** (C#) bzw. **Erweiterte Kompilereinstellungen** (VB) das Zielframework von **.NET Framework 4 Client Profile** auf **.NET Framework 4** ändern, da ersteres nicht die von den Eigenschaftenseiten zur Designzeit benötigte System.Design.dll enthält. Falls Sie das Framework nicht umstellen, erscheint folgende Fehlermeldung, wenn Sie das Steuerelement auf die Form ziehen möchten:



Um VARCHART XTree in ein Formular einzufügen, klicken Sie auf das entsprechende Symbol in der Toolbox und ziehen im Formular an der gewünschten Stelle mit der Maus einen Rahmen auf. Das VARCHART-XTree-Steuerelement erscheint (zunächst als weiße Fläche) in der von Ihnen bestimmten Größe. Diese können Sie selbstverständlich mit Hilfe der Maus noch modifizieren.



2.3 VARCHART XTree automatisch skalieren

Wenn der rechte und der untere Rand des VARCHART-XTree-Steuerelement zur Laufzeit immer auf die Größe des gesamten Fensters angepasst werden sollen, können Sie das z. B. mit Hilfe des folgenden Codes erreichen:

Code-Beispiel VB.NET

```
Private Sub Form1_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    VcTree1.Width  = ClientSize.Width - VcTree1.Left
    VcTree1.Height = ClientSize.Height - VcTree1.Top
End Sub

Private Sub Form1_Resize(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Resize
    VcTree1.Width  = ClientSize.Width - VcTree1.Left
    VcTree1.Height = ClientSize.Height - VcTree1.Top
End Sub
```

Code-Beispiel C#

```
private void Form1_Load(object sender, System.EventArgs e)
   {
   vcTree1.Width  = ClientSize.Width - vcTree1.Left;
  vcTree1.Height = ClientSize.Height - vcTree1.Top;
   }

Private void Form1_Resize(object sender, System.EventArgs e)
   {
   vcTree1.Width  = ClientSize.Width - vcTree1.Left;
   vcTree1.Height = ClientSize.Height - vcTree1.Top;
   }
```

Hinweis:

Eine Namespace-Anweisung am Anfang des Programms erspart die vollständige und mitunter umständliche Referenzangabe bei der Benutzung von Datentypen und Enum-Elementen.

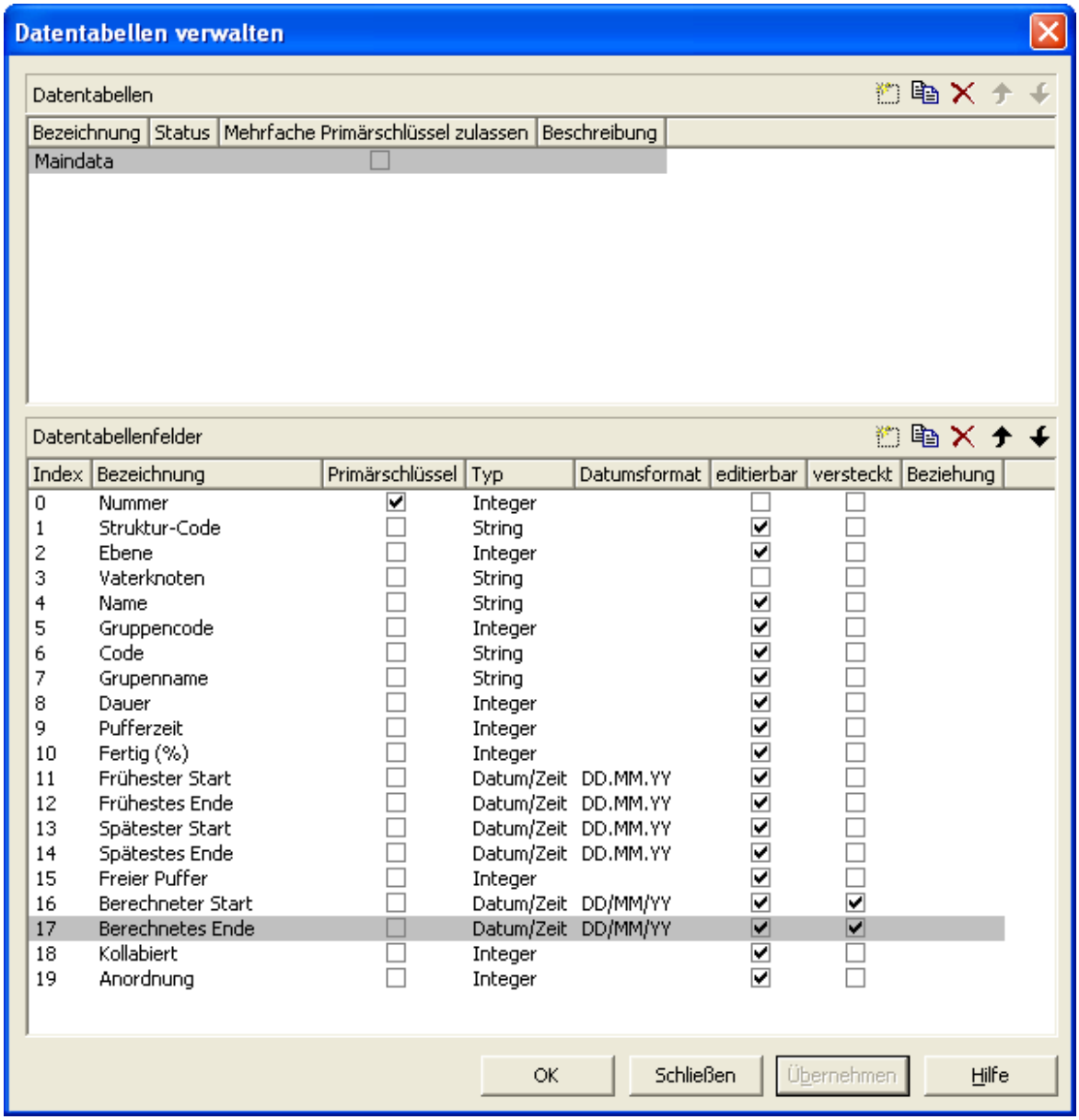
VB: Imports NETRONIC.XTree

C#: using NETRONIC.XTree;

Zum Beispiel ist dann statt **NETRONIC.XTree.VcNodeCollection** nur noch **VcNodeCollection** erforderlich.

2.4 Schnittstelle einrichten

Richten Sie nun die Schnittstelle ein, indem Sie die Datenfelder der Tabelle "Maindata" (Knotendaten) anpassen. Klicken Sie dazu bitte auf der Eigenschaftenseite **Objekte** die Schaltfläche **Datentabellen** an. Daraufhin erscheint der Dialog **Datentabellen verwalten**. Die obere Liste zeigt alle vorhandenen Datentabellen. Da noch keine neu angelegt wurde, existiert lediglich die Standard-Datentabelle mit der vordefinierten Datenfeldern. Diese werden im unteren Bereich aufgelistet.



Das Feld mit dem Index "0" hat standardmäßig den Namen "ID" und ist als "Alphanumerisch" vereinbart. Um der Beispiel-Schnittstelle gerecht zu werden, überschreiben Sie "ID" durch "Nummer" und wählen Sie den Datentyp "Integer". Aktivieren Sie das Kontrollkästchen **nur lesbar**, damit

die Nummer im Standarddialog **Vorgänge bearbeiten** nicht überschrieben werden kann.

Ein Name lässt sich editieren, indem Sie ihn doppelt anklicken bzw. ihn markieren und dann einmal anklicken. Der Typ kann aus einer Kombobox ausgewählt werden.

Ein neues Datenfeld können Sie anlegen, indem Sie das "Neu..."-Feld in der letzten Zeile editieren.

Ändern Sie nun die Felder der Maindata-Tabelle folgendermaßen:

Index	Name	Typ
0	Nummer	Integer
1	Strukturcode	String
2	Ebene	Integer
3	Vaterknoten	String
4	Name	Alphanumerisch
5	Gruppencode	String
6	Code	Integer
7	Gruppenname	String
8	Dauer	Integer
9	Pufferzeit	Integer
10	fertig (%)	Integer
11	Frühester Start	Datum/Zeit
12	Frühestes Ende	Datum/Zeit
13	Spätester Start	Datum/Zeit
14	Spätestes Ende	Datum/Zeit
15	Freier Puffer	Integer
16	Berechneter Start	Datum/Zeit
17	Berechnetes Ende	Datum/Zeit
18	Kollabiert	Integer
19	Anordnung	Integer

Aktivieren Sie für die Felder "Berechneter Start" und "Berechnetes Ende" das Kontrollkästchen **versteckt**, damit der Anwender deren Werte im Standarddialog **Vorgänge bearbeiten** nicht sieht.

Bei den **Datum/Zeit**–Feldern können Sie angeben, in welchem Format das Datum übergeben wird. Wählen Sie "DD.MM.YY".

Stellen Sie jetzt noch die Identifikation für die Knoten ein. Wählen Sie unter **Identifikation aus** das Feld "Nummer". Aus diesem Feld wird dann die Identifikation der Knoten genommen.

Hinweis: Wenn Sie einen Namen eingeben, der bereits für ein anderes Feld vergeben ist, wird dieser nicht angenommen. Der vorherige Name erscheint wieder.

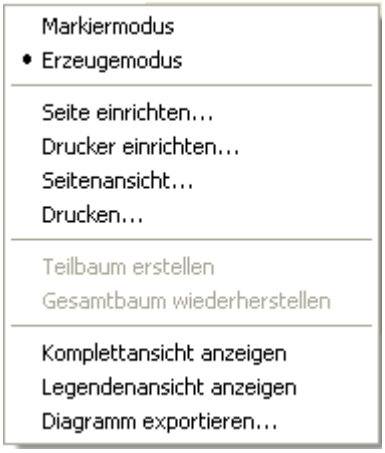
Durch Anklicken der Schaltfläche **Übernehmen** werden die Änderungen für diese Konfiguration festgeschrieben. Alternativ werden die Einstellungen auch bei **OK** und beim Wechsel der Eigenschaftenseite übernommen und stehen damit direkt in den anderen Eigenschaftenseiten zur Verfügung.

2.5 Der erste Lauf

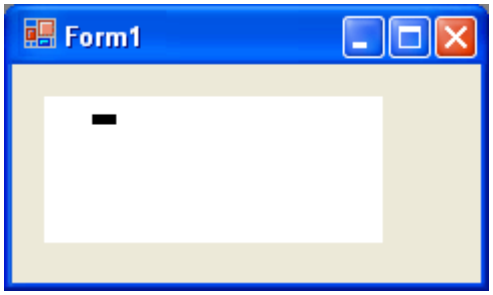
Über **Ausführen – Starten**, die Funktionstaste **F5** oder die entsprechende Visual-Basic-Ikone ( ) starten Sie nun das Programm. Das angelegte Formular erscheint mit dem leeren Diagramm.

> Knoten anlegen

Varchart XTree besitzt zur Laufzeit zwei grundlegende Modi: den Markiermodus und den Erzeugemodus. Knoten können Sie nur im Erzeugemodus anlegen. Um in den Erzeugemodus zu wechseln, klicken Sie mit der rechten Maustaste in den freien Diagrammbereich und wählen Sie im Kontextmenü den Befehl **Erzeugemodus**.

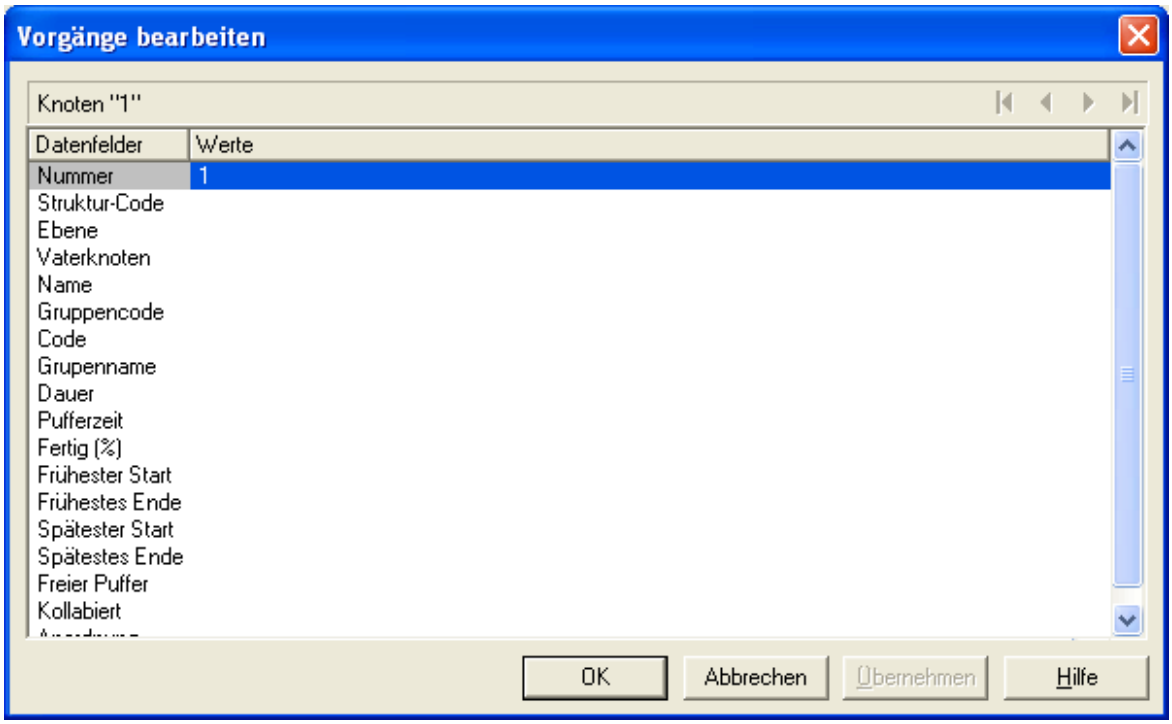


Im Erzeugemodus wird der Mauszeiger im leeren Diagrammbereich zu einem Knotenphantom in der Form eines kleinen schwarzen Rechtecks.

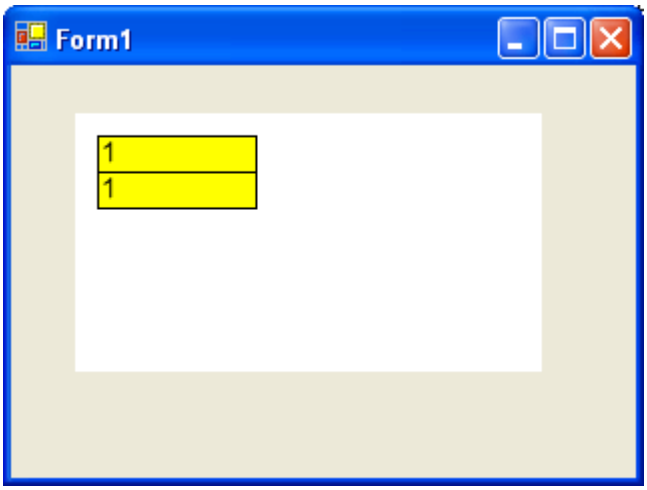


Klicken Sie nun einmal mit der linken Maustaste.

Wenn auf der Eigenschaftenseite **Allgemeines** die Option **Erzeugung neuer Knoten mit Dialog** aktiviert ist, erscheint zunächst das Dialogfeld **Vorgänge bearbeiten**, in dem die Daten dieses Knotens angezeigt werden.

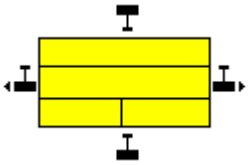


Zunächst ist nur das Datenfeld "Nummer" vorbesetzt (beim ersten Knoten mit dem Wert "1"). Sie können ggf. weitere Daten ergänzen, z. B. die Termindaten und eine Beschreibung. Sobald Sie die Daten mit **OK** bestätigen, wird der erste Knoten erzeugt.



War auf der Eigenschaftenseite **Allgemeines** die Option **Erzeugung neuer Knoten mit Dialog** nicht aktiviert, wird der erste Knoten angelegt, sobald Sie im Erzeugemodus mit der linken Maustaste in den Diagrammbereich klicken. Das Dialogfeld **Vorgänge bearbeiten** erscheint dann nicht.

Weitere Knoten können Sie durch Anlagern an einen bereits vorhandenen Knoten erzeugen. Wenn Sie den Mauszeiger im Erzeugemodus in die Nähe eines Knotens führen, verändert er seine Form und zeigt an, wo der neue Knoten angelegt würde (als Vater, Sohn oder linker bzw. rechter Bruder des Bezugsknotens).



> Knoten bearbeiten

Einen Knoten bearbeiten können Sie, indem Sie im Markiermodus mit einem Doppelklick auf den Knoten das Dialogfeld **Vorgänge bearbeiten** öffnen. Sie finden hier die Datenfelder wieder, die Sie auf der Eigenschaftenseite **Datendefinition** vereinbart haben. (Die Datenfelder, die dort als **versteckt** definiert worden sind, erscheinen nicht in diesem Dialogfeld. Die Datenfelder, die dort als **nur lesbar** definiert worden sind, können in diesem Dialogfeld nicht bearbeitet werden.)

> Zurück zum Design-Modus

Beenden Sie nun den ersten Lauf, indem Sie das Formular schließen.

2.6 Daten aus einer Datei einlesen

Um VARCHART XTree für die nächsten Schritte mit Daten zu füllen, können Sie die mitgelieferte Datei *tutorial.tre* beim Start automatisch laden. (*tutorial.tre* ist eine CSV-Datei entsprechend Ihrer eingestellten Schnittstelle. Zu Veränderungen der Schnittstelle siehe "Tutorium: Schnittstelle einrichten".)

Dazu reagieren Sie auf das Ereignis Form_Load:

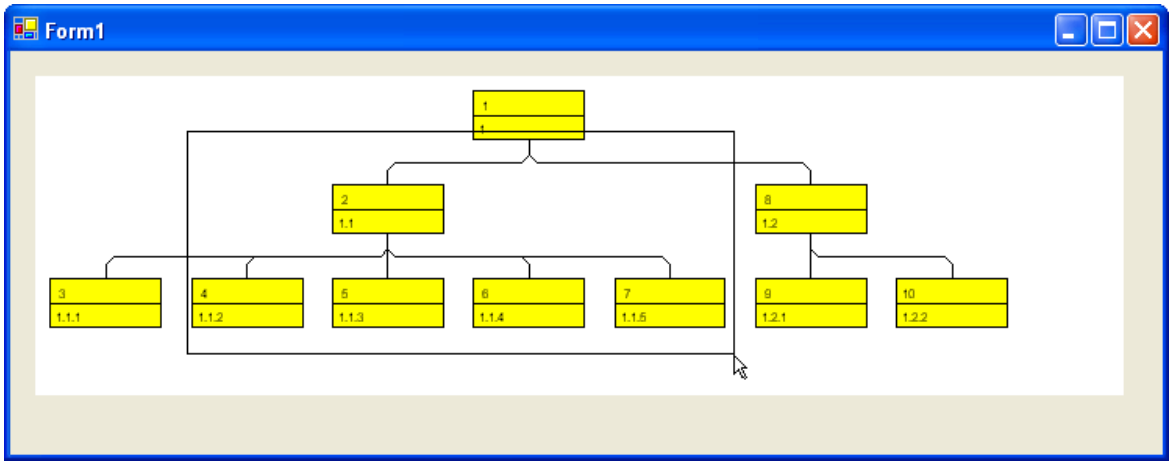
Code-Beispiel

```
Private Sub Form_Load()
    VcTree1.Open "C:\Programme\Varchart\xtree\tutorial.tre"
End Sub
```

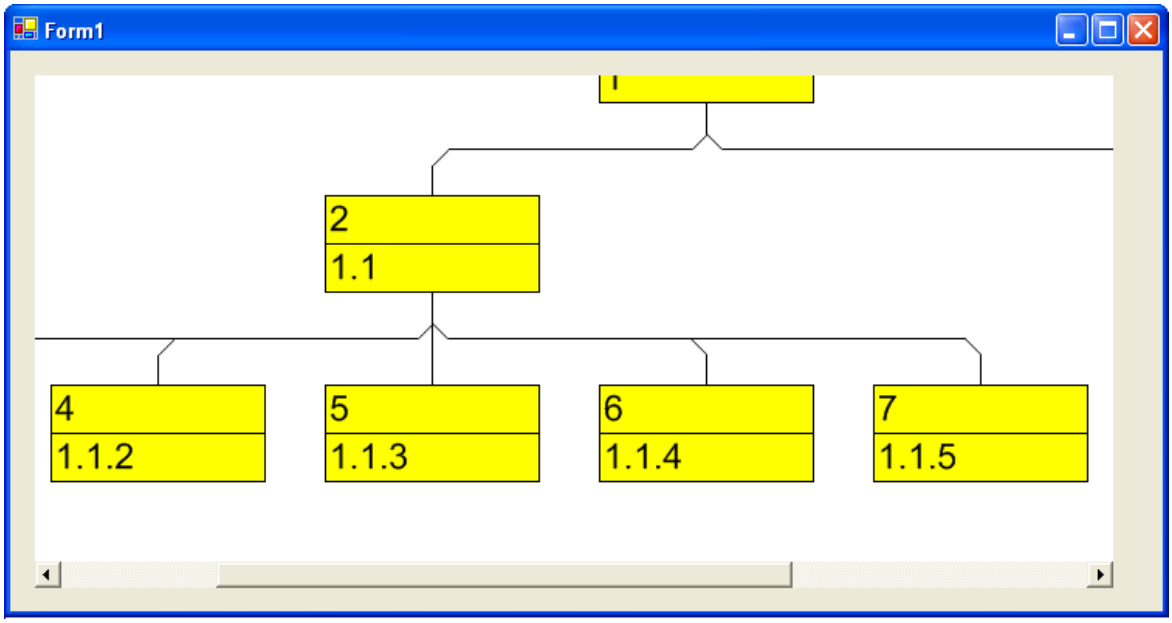
Die Pfadangabe ist abhängig von Ihrer Installation. Speichern Sie nun das Projekt. Wenn Sie nun das Programm starten, werden die Knoten des Projektes angezeigt.

VARCHART XTree stellt ein Baum-Diagramm komplett dar.

Sie können einen Ausschnitt Ihres Diagramms bildschirmfüllend darstellen lassen, indem Sie mit gedrückter linker Maustaste den zu vergrößernden Ausschnitt markieren und dann (bei noch gedrückter linker Maustaste) auf die rechte Maustaste klicken.



Dann wird der gewählte Ausschnitt bildschirmfüllend dargestellt. Mit Hilfe der Bildlaufleisten können Sie dann das Fenster wie eine Lupe über der Darstellung verschieben und so auch die anderen Bereiche der Darstellung in derselben Vergrößerung betrachten.



Kehren Sie nun wieder zum Design-Modus zurück. Ergänzen Sie ggf. den folgenden Code, damit Scrollbars in X- und Y-Richtung ausgegeben werden. (Die Ausgabe von Scrollbars hängt von dem gewählten Zoomfaktor ab.)

Code-Beispiel

```
Private Sub Form_Load()
   VcTree1.Zoomfactor = 100
End Sub
```

Soll VARCHART XTree das gesamte Formular ausfüllen, dann muss folgendes eingestellt werden:

- Stellen Sie sicher, dass die Eigenschaften **Top** und **Left** den Wert 0 haben. Damit wird VARCHART XTree oben links im Formular positioniert.
- Tragen Sie für die VARCHART-XTree-Eigenschaften **Width** und **Height** die Formularwerte von **ScaleWidth** und **ScaleHeight** ein. (Dieser Schritt ist überflüssig, wenn Sie das VARCHART XTree wie oben beschrieben automatisch reskalieren lassen.)

2.7 Markierung von Knoten festlegen

Auf der Eigenschaftenseite **Knoten** können Sie aus der Kombobox **Markierungstyp** auswählen, in welcher Weise Knoten markiert werden sollen.

Starten Sie das Programm und legen Sie im Erzeugemodus einige Knoten an. Klicken Sie nun im Markiermodus einmal auf einen Knoten, um ihn zu markieren.

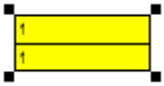
Mehrere Knoten können Sie markieren und toggeln, indem Sie sie bei gedrückter Strg-Taste mit der linken Maustaste anklicken. Jeder Klick auf einen markierten Knoten demarkiert diesen, jeder Klick auf einen nicht markierten Knoten markiert diesen.

Einen Teilbaum markieren Sie, indem Sie bei gedrückter Shift-Taste den Vaterknoten des Teilbaums anklicken.

Sie können alle markierten Knoten demarkieren, indem Sie mit der linken Maustaste in den leeren Diagrammbereich klicken.

Probieren Sie verschiedene Alternativen für den Knotenmarkierungstyp aus.

In der Abbildung sehen Sie einen durch Pickmarks markierten Knoten:

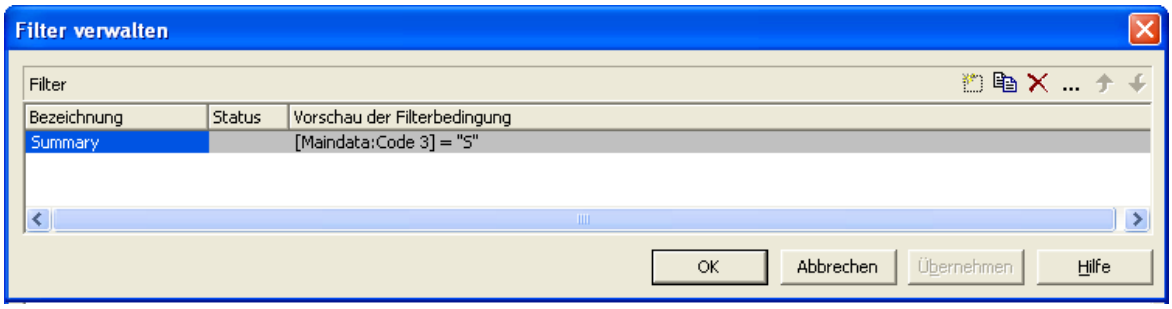


2.8 Filter für Knoten festlegen

Ein Filter enthält Kriterien zur Auswahl von bestimmten Daten, beispielsweise von Knoten.

Wenn Sie Filter für das Knotenaussehen verwenden, erhalten genau die Knoten, die die Filterkriterien eines bestimmten Knotenaussehens erfüllen, dieses Aussehen.

Um einen Filter für ein Knotenaussehen zu bearbeiten, klicken Sie auf der Eigenschaftenseite **Objekte** auf die Schaltfläche **Filter**. Sie gelangen dann in das Dialogfeld **Filter verwalten**, in dem Sie Filter umbenennen, bearbeiten, neu definieren, kopieren und löschen können.



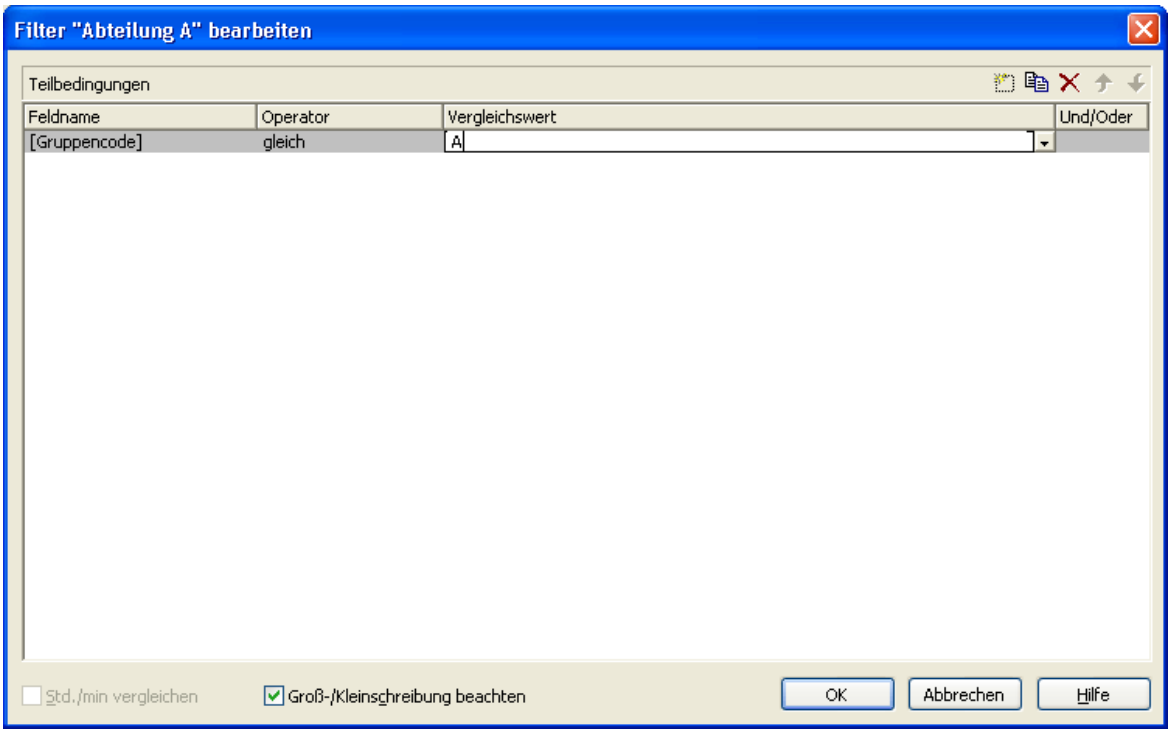
> Schaltflächen im Dialogfeld "Filter verwalten"

- Filter hinzufügen
- Filter kopieren
- Filter löschen
- Filter bearbeiten

> Filter erstellen und bearbeiten

Lernen Sie nun, wie Sie neue Filter erstellen und bearbeiten können. Klicken Sie dazu auf die Schaltfläche **Filter hinzufügen**. Der neue Filter wird am Ende der Liste angefügt. Ändern Sie seinen Namen in "Abteilung A" um.

Bearbeiten Sie nun den neuen Filter. Klicken Sie dazu auf die Schaltfläche **Filter bearbeiten**. Sie gelangen dann in das gleichnamige Dialogfeld. Nehmen Sie hier die folgenden Einstellungen vor:



In der Kopfzeile wird der aktuelle Filter angezeigt, den Sie hier bearbeiten können.

Unter **Feldname** wird das Datenfeld angezeigt, dessen Inhalt mit dem Vergleichswert verglichen werden soll. Wählen Sie hier das Datenfeld "Gruppenname".

Unter **Operator** wird der aktuelle Vergleichsoperator angezeigt. Welche Operatoren möglich sind, hängt vom Typ des gewählten Datenfeldes ab. Wählen Sie hier "gleich".

Im Feld **Vergleichswert** können Sie angeben, mit welchem Wert der Eintrag des Datenfeldes verglichen werden soll. Der Typ des Vergleichwertes muss mit dem Typ des Datenfeldes übereinstimmen, das unter **Feldname** angegeben wurde. Wählen Sie hier "A".

Unter **Und/Oder** können Sie eine Verknüpfung auswählen, wenn Sie mehrere Kriterien definieren möchten.

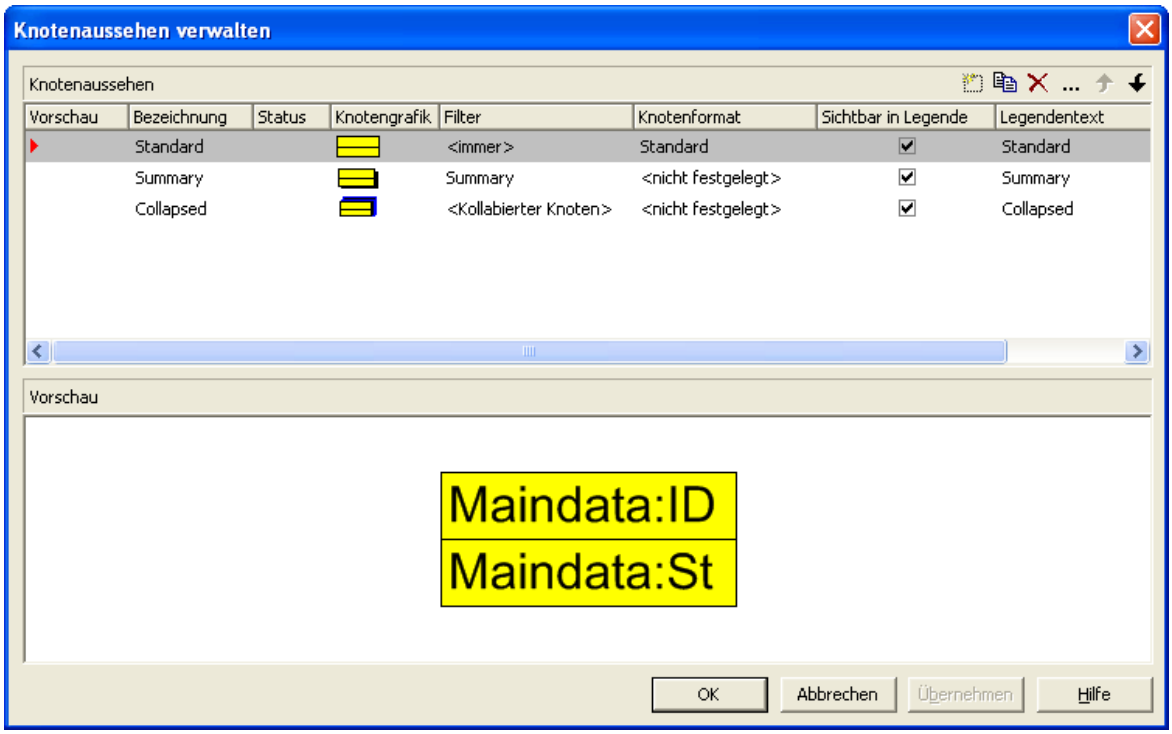
Definieren Sie den Filter wie in der Abbildung. Verlassen Sie dann das Dialogfeld **Filter bearbeiten** mit **OK**. Sie kehren in das Dialogfeld **Filter verwalten** zurück.

2.9 Knotenaussehen festlegen

VARCHART XTree bietet eine Vielzahl von Gestaltungsmöglichkeiten für das Aussehen von Knoten. Sie können das Aussehen von Knoten in Abhängigkeit von deren Daten festlegen. Beispielsweise können Sie festlegen, dass die Knoten für unterschiedliche Abteilungen jeweils unterschiedliche Farben erhalten.

Diese grafischen Attribute werden als Knotenaussehen bezeichnet.

Öffnen Sie nun die Eigenschaftenseite **Objekte** und klicken Sie auf die **Knotenaussehen**-Schaltfläche. Sie gelangen in das Dialogfeld **Knotenaussehen verwalten**.



In der **Knotenaussehen**-Tabelle werden alle aktuell vorhandenen Knotenaussehen angezeigt. Wenn Sie ein Aussehen anklicken, wird es im Vorschaufenster angezeigt.

Jedes Knotenaussehen ist mit einem Filter und einem Knotenformat verbunden. (Ausnahme: Das Knotenaussehen "Standard" ist nicht mit einem Filter verbunden.)

Der Filter gibt die Bedingungen an, unter denen ein Knoten dieses Knotenaussehen erhält. Beispielsweise ist das Knotenaussehen "Markiert" mit dem Filter "Markiert" verbunden. Dieser Filter selektiert alle markierten Knoten.

Erfüllt ein Vorgang die Filterkriterien mehrerer Knotenaussehen, werden diese Knotenaussehen für den Knoten grafisch überlagert. Begonnen wird dabei jeweils mit dem Knotenaussehen, das in der Liste ganz oben steht. Das Knotenaussehen, das ganz unten steht, wird als letztes zugewiesen und überlagert daher alle anderen.

Die niedrigste Position hat i. d. R. das Knotenaussehen "Standard", das normalerweise ganz oben in der Liste steht. Das Knotenaussehen "Standard" hat keinen Filter und wird auf alle Knoten angewendet.

Sie können die Abarbeitungsreihenfolge der Knotenaussehen mit Hilfe der Pfeil-Schaltflächen verändern.

> Knotenaussehen hinzufügen, kopieren, löschen und bearbeiten

Sie können im Dialogfeld **Knotenaussehen verwalten** mit Hilfe der folgenden Schaltflächen Knotenaussehen hinzufügen, kopieren, löschen und bearbeiten:

Knotenaussehen hinzufügen

Knotenaussehen kopieren

Knotenaussehen löschen

Knotenaussehen bearbeiten

Hinweis: Alle Knotenaussehen außer den Standard-Knotenaussehen können gelöscht werden. Bevor das markierte Knotenaussehen tatsächlich gelöscht wird, erfolgt eine Rückfrage des Programms.

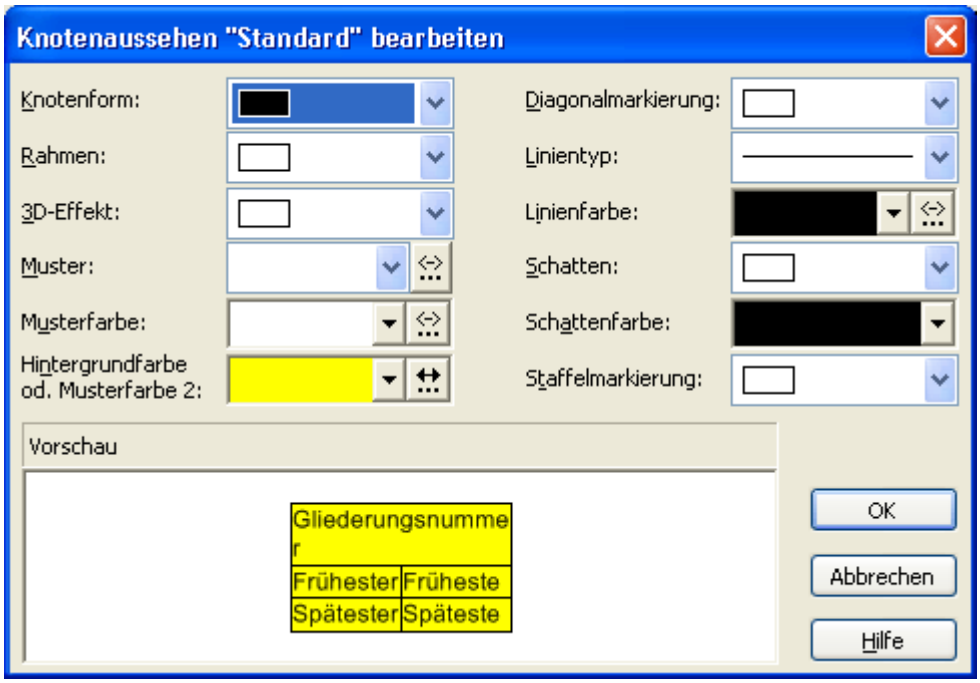
> Knotenaussehen und Filter verwenden

Lernen Sie nun den Umgang mit Knotenaussehen und Filtern an einem Beispiel kennen. Erzeugen Sie die neuen Knotenaussehen "Abteilung A" und "Abteilung B" als Kopien des Knotenaussehens "Standard".

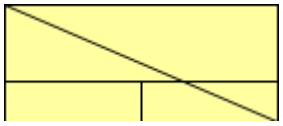
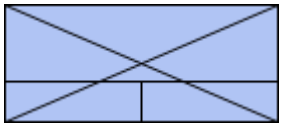
Weisen Sie dem Knotenaussehen "Abteilung A" die höchste und dem Knotenaussehen "Abteilung B" die zweithöchste Priorität zu.

Bearbeiten Sie nun diese beiden Knotenaussehen. Markieren Sie das zu bearbeitende Knotenaussehen im Dialogfeld **Knotenaussehen verwalten** und klicken Sie auf die **Knotenaussehen bearbeiten**-Schaltfläche. Sie gelangen in das Dialogfeld **Knotenaussehen bearbeiten**. In der Kopfzeile wird der Name des aktuellen Knotenaussehens angezeigt. In diesem Dialogfeld können Sie die grafischen Attribute der Knoten festlegen. Außerdem

können Sie hier festlegen, mit welchem Knotenformat und mit welchem Filter das Knotenaussehen kombiniert werden soll.



Legen Sie nun folgendes fest:

Knotenaussehen	Abteilung A	Abteilung B
Filter	Abteilung A	Abteilung B
Filterbedingung	Gruppenname gleich A	Gruppenname gleich B
Hintergrundfarbe	hellgelb	blau
Diagonalmarkierung	abwärts	gekreuzt
Aussehen		

Bestätigen Sie die Einstellungen mit **OK** und starten Sie das Programm. Erzeugen Sie einen Knoten, klicken ihn doppelt an und bearbeiten seine Daten im Dialogfeld **Vorgänge bearbeiten** in den folgenden Schritten:

- Geben Sie für "Gruppenname" den Wert "A" an: Der Knoten erscheint mit dem "Abteilung A"-Knotenaussehen, also in rot und abwärts durchgestrichen.
- Geben Sie nun für "Gruppenname" den Wert "B" an: Der Knoten erscheint mit dem Knotenaussehen "Abteilung B", also in blau und durchgestrichen mit gekreuzten Linien.

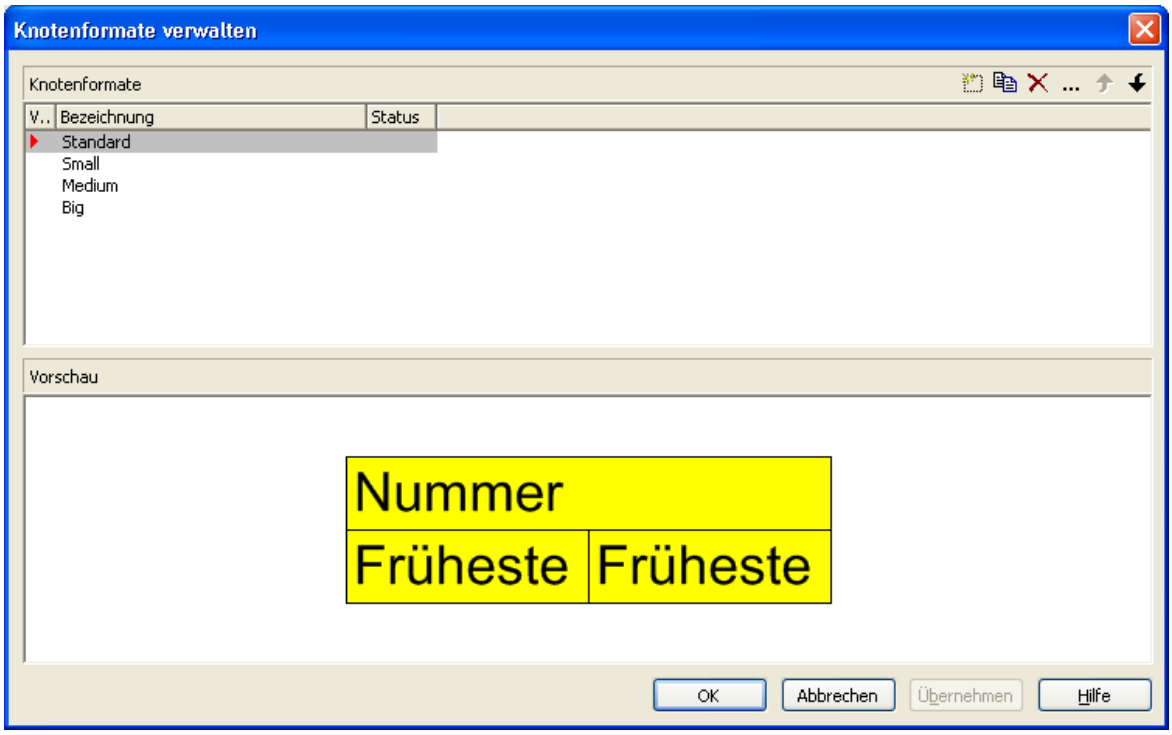
> Knotenaussehen datenabhängig festlegen

Muster, Musterfarbe, Hintergrundfarbe und die Linienfarbe eines Knotenaussehens können mit Hilfe von Zuordnungstabellen in Abhängigkeit von den Daten der Knoten festgelegt werden. Siehe hierzu das Kapitel "Wichtige Begriffe: Zuordnungstabellen".

2.10 Knotenformate festlegen

Jedes Knotenaussehen ist mit einem Knotenformat verbunden. Sie können die Knotenformate selbst festlegen.

Klicken Sie auf der Eigenschaftenseite **Objekte** auf die Schaltfläche **Knotenformate**. Sie gelangen dann in das Dialogfeld **Knotenformate verwalten**.



In der **Knotenformate**-Tabelle werden alle vorhandenen Knotenformate aufgeführt. Gehen Sie die Tabelle durch, um im Vorschaufenster das Aussehen des jeweils markierten Knotenformats zu betrachten.

Sie können im Dialogfeld **Knotenformate verwalten** mit Hilfe der folgenden Schaltflächen Knotenformate hinzufügen, kopieren, löschen und bearbeiten:

Knotenformat hinzufügen

Knotenformat kopieren

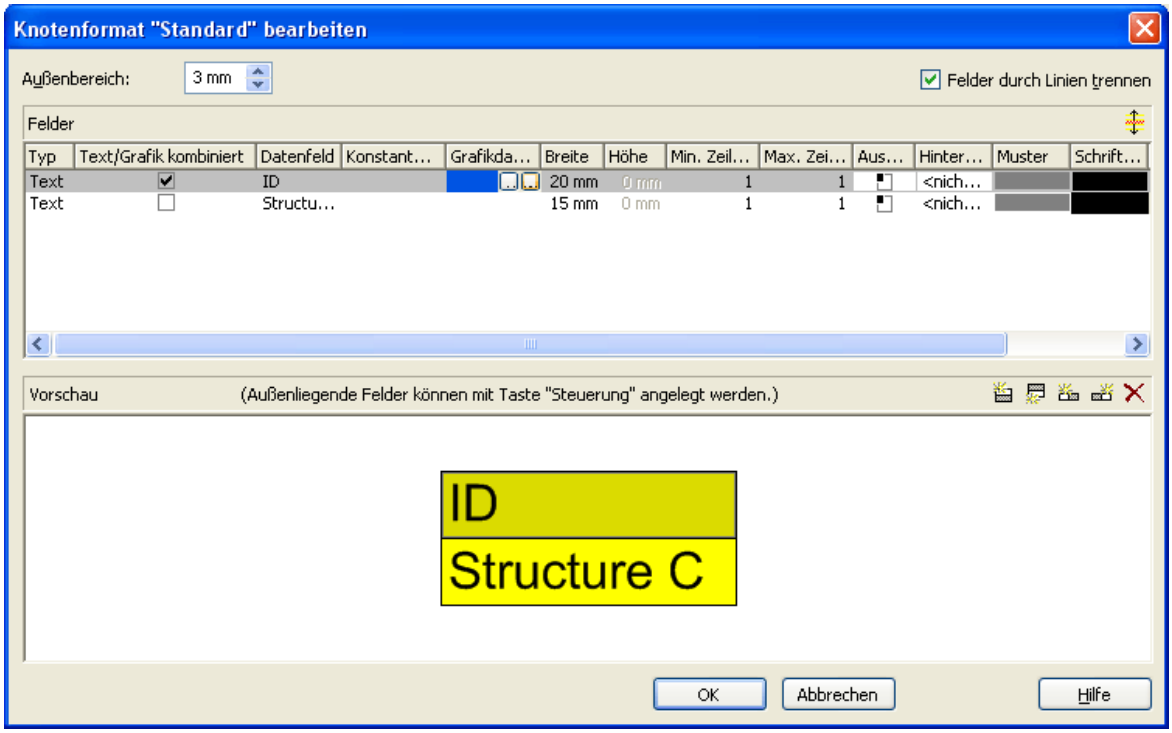
Knotenformat löschen

Knotenformat bearbeiten

Hinweis: Das Knotenformat "Standard" sowie jedes in einem Knotenaussehen verwendete Knotenformat können nicht gelöscht werden.

> Knotenformat bearbeiten

Um ein Knotenformat zu bearbeiten, markieren Sie es und klicken Sie auf die Schaltfläche **Knotenformat bearbeiten**. Das folgende Dialogfeld erscheint:



In diesem Dialog können Sie für das gewählte Knotenformat Folgendes festlegen:

- ob die Knotenfelder durch Linien getrennt werden sollen
- den Außenbereich (Abstand in Millimetern, den Knoten mit diesem Knotenformat zu benachbarten Knoten und zum Rand der Darstellung halten)
- den Typ des aktuellen Knotenfeldes (Text oder Grafik)
- für den Typ Text: das Datenfeld, dessen Inhalt in dem aktuellen Knotenfeldes ausgegeben werden soll, oder einen konstanten Text
- für den Typ Grafik: Name und Pfad der Grafikdatei, die in dem gewählten Knotenfeld dargestellt wird
- die Breite und Höhe des markierten Knotenfeldes
- die maximale Anzahl von Textzeilen im aktuellen Knotenfeld
- die Ausrichtung des Textes bzw. der Grafik des markierten Knotenfeldes
- die Hintergrundfarbe des Knotenfeldes
- das Muster des Knotenfeldes
- die Schriftart und -farbe des Knotenfeldes

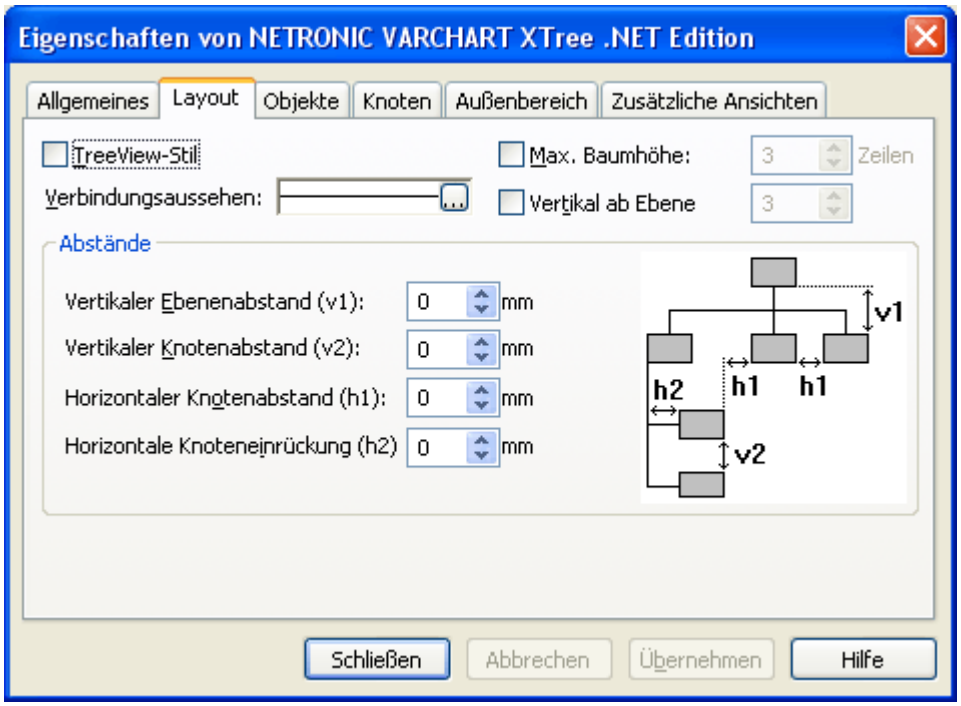
> Darstellung von Grafiken in Knotenfeldern

Sie können für ein Knotenfeld des Typs Grafik eine Grafikdatei wählen, indem Sie auf die Schaltfläche **Grafikdatei auswählen** (...) klicken und dann im gleichnamigen Windows-Dialog eine Datei wählen.

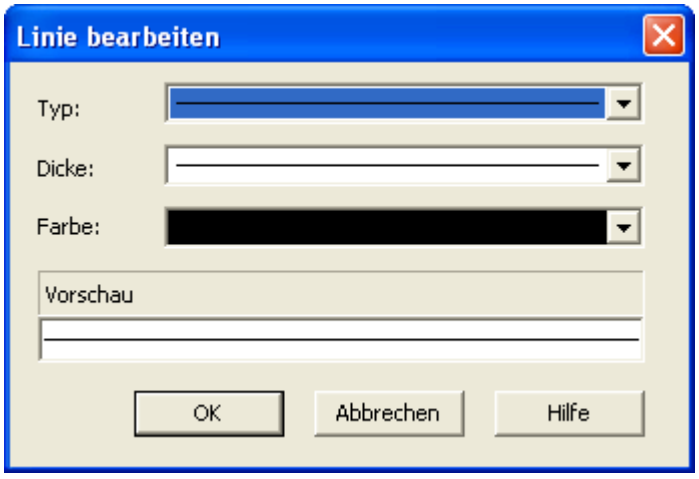
Sie können aber auch eine Zuordnung zwischen den Einträgen eines Datenfeldes und Grafikdateien herstellen. Klicken Sie dazu auf die Schaltfläche **Zuordnungen einstellen** (), um den gleichnamigen Dialog zu öffnen. Wenn eine Zuordnung vorgenommen worden ist, wird das durch ein Symbol neben dem Grafikdateinamen dargestellt (). Einzelheiten hierzu finden Sie in den Kapiteln "Eigenschaftenseiten und Dialogfelder" und "Wichtige Begriffe: Zuordnungstabellen".

2.11 Das Aussehen von Verbindungen festlegen

Auf der Eigenschaftenseite **Layout** wird im Feld **Verbindungsaussehen** das aktuelle Verbindungsaussehen angezeigt.

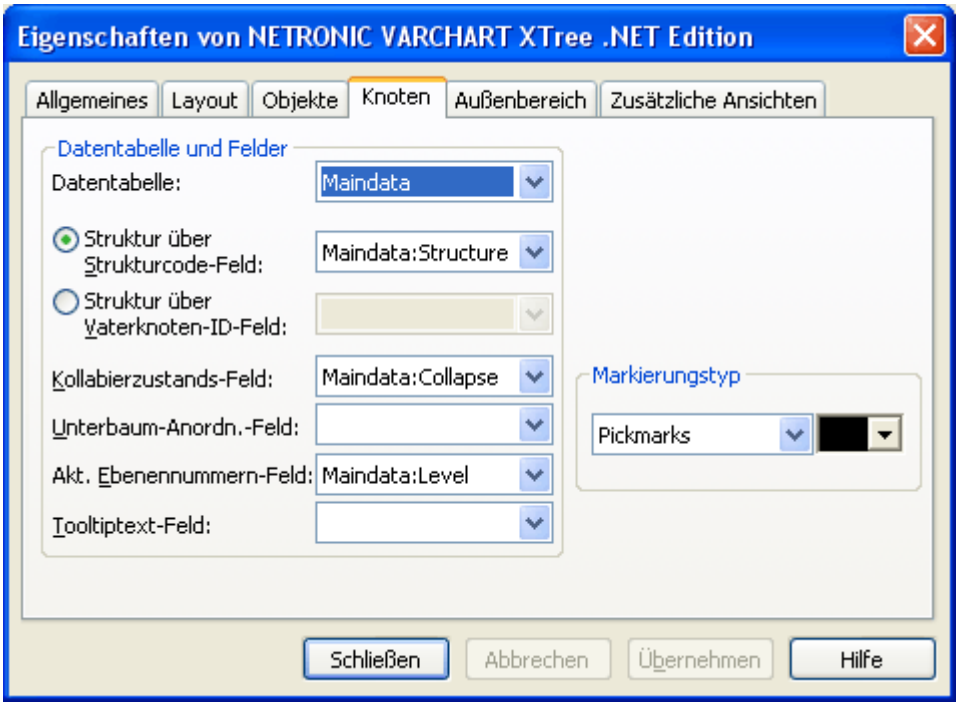


Um es zu ändern, klicken Sie auf die **Bearbeiten**-Schaltfläche. Sie gelangen dann in das Dialogfeld **Linie bearbeiten**, in dem Sie Typ, Dicke und Farbe der Verbindungslinien festlegen können.



2.12 Die Baumstruktur festlegen

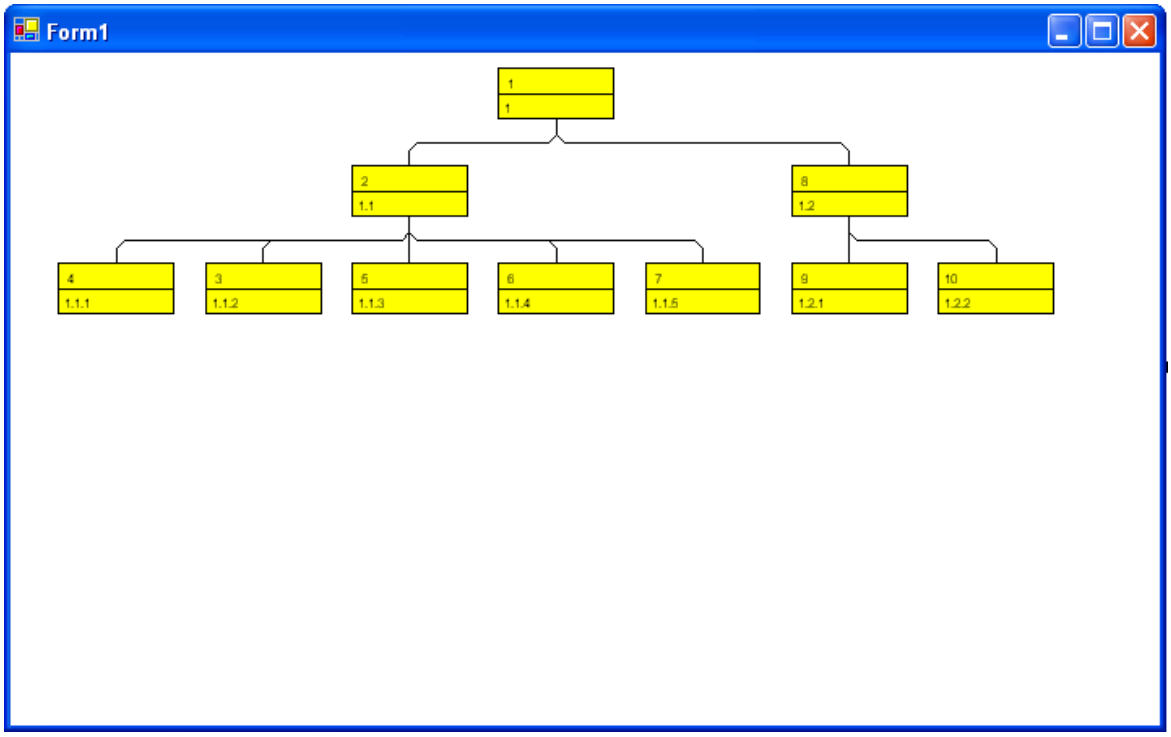
Auf der Eigenschaftenseite **Knoten** können Sie die Struktur der Baum-Diagramme festlegen.



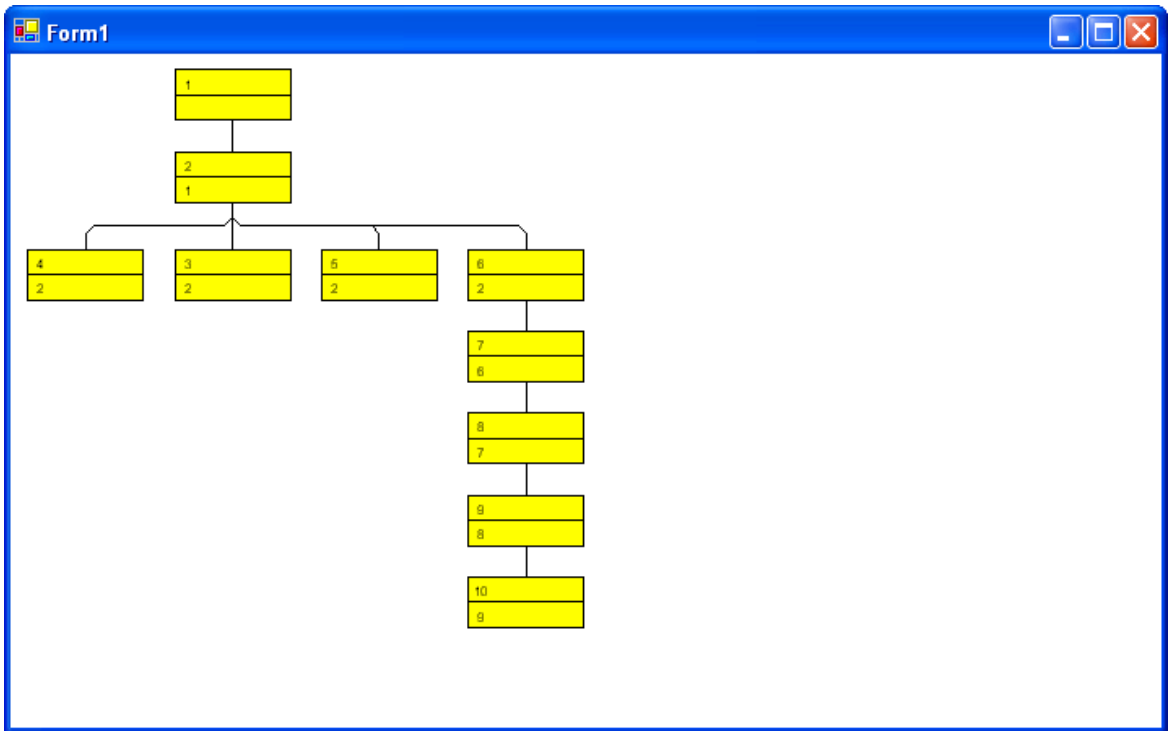
Dabei gibt es zwei grundsätzliche Alternativen:

1. **Struktur über Strukturcode-Feld:** Der Baum wird gemäß eines Strukturcodes aufgebaut. Sie können dann ein Datenfeld auswählen, das den Wert des Strukturcodes enthält. Die Ebenen werden durch ein Trennzeichen (Punkt) voneinander getrennt.
2. **Struktur über Vaterknoten-ID-Feld:** Der Baum wird durch die ID des Vaterknotens jedes Knotens definiert. Sie können das Datenfeld wählen, das die ID des Vaterknotens enthält.

Wählen Sie zunächst die Option **Struktur über Strukturcode-Feld** und wählen Sie das Feld "Strukturcode" aus. Der Wert dieses Feldes bestimmt dann die Baumstruktur. Starten Sie nun das Programm. Sie erhalten folgendes Resultat:



Kehren Sie nun zum Design-Modus zurück und wählen Sie die Option **Struktur über Vaterknoten-ID-Feld**. Wählen Sie als Datenfeld, das die ID des Vaterknotens enthält, das Feld "Vaterknoten". Starten Sie nun das Programm. Sie erhalten folgendes Resultat:

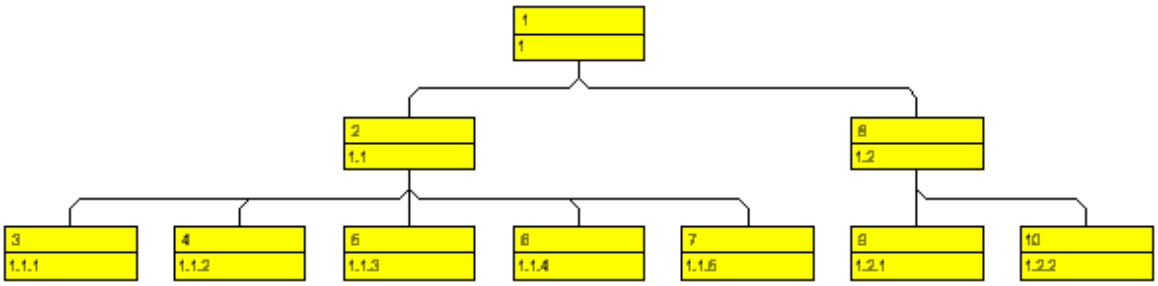


2.13 Vertikale und horizontale Anordnung von Baumstrukturen

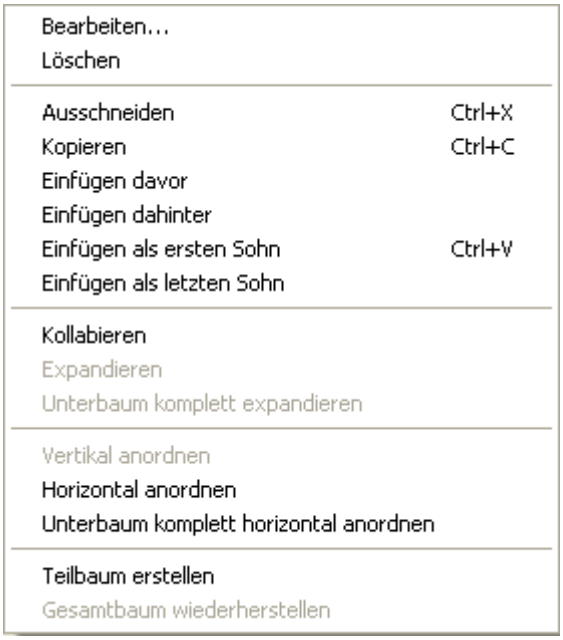
Erfahren Sie nun, wie Sie Ihr Baum-Diagramm durch die Kombination horizontal und vertikal angeordneter Teilstrukturen optimieren können.

- *Horizontale Anordnung:* Durch eine horizontale Anordnung lässt sich die Höhe eines Baum-Diagramms verringern. Alle Knoten einer Ebene werden dabei nebeneinander angeordnet. Der Port (Anknüpfungspunkt) der Verbindungslinie liegt dann mittig am unteren Rand des Vaterknotens und mittig am oberen Rand des Sohnknotens.
- *Vertikale Anordnung:* Durch eine vertikale Anordnung lässt sich die Breite des Baum-Diagramms verringern. Alle Knoten einer Ebene und deren Unterebenen werden dabei untereinander angeordnet. Der Port der Verbindungslinie liegt dann in der unteren linken Ecke des Vaterknotens und in der Mitte des linken Randes des Sohnknotens.

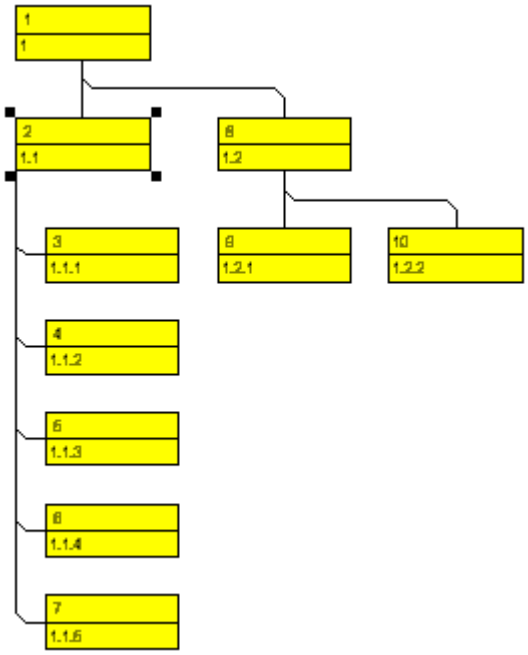
Legen Sie auf der Eigenschaftenseite **Knoten** fest: **Strukturcode in Feld:** "Strukturcode". Starten Sie dann das Programm mit den Daten der Datei *tutorial.tre* (vgl. "Tutorium: Daten aus einer Datei einlesen").



Markieren Sie nun den ersten Knoten der zweiten Ebene und klicken Sie auf die rechte Maustaste. Das folgende Kontextmenü erscheint, wobei jeweils nur die verfügbaren Befehle aktiviert sind.



Wählen Sie den Befehl **Vertikal anordnen**. Dadurch wird der Teilbaum unter dem markierten Knoten vertikal angeordnet.



Falls der Baum nicht vollständig vertikal angeordnet wird, wechseln Sie wieder zum Designmodus, öffnen Sie die Eigenschaftenseite **Layout** und prüfen Sie dort die vereinbarte maximale Baumhöhe. Die Gesamthöhe eines Baum-Diagramms (in Zeilen) bei der vertikalen Anordnung wird durch die Angabe unter **Max. Baumhöhe** begrenzt. Diese Einstellung wirkt sich nur bei vertikaler Anordnung aus. Falls in einem vertikal angeordneten Ast mehr Ebenen vorhanden sind, erfolgt ein Umbruch, und ein neuer Ast wird am Vaterknoten erzeugt.

Aktivieren Sie das Kontrollkästchen **Max. Baumhöhe** und wählen Sie den Wert "10".

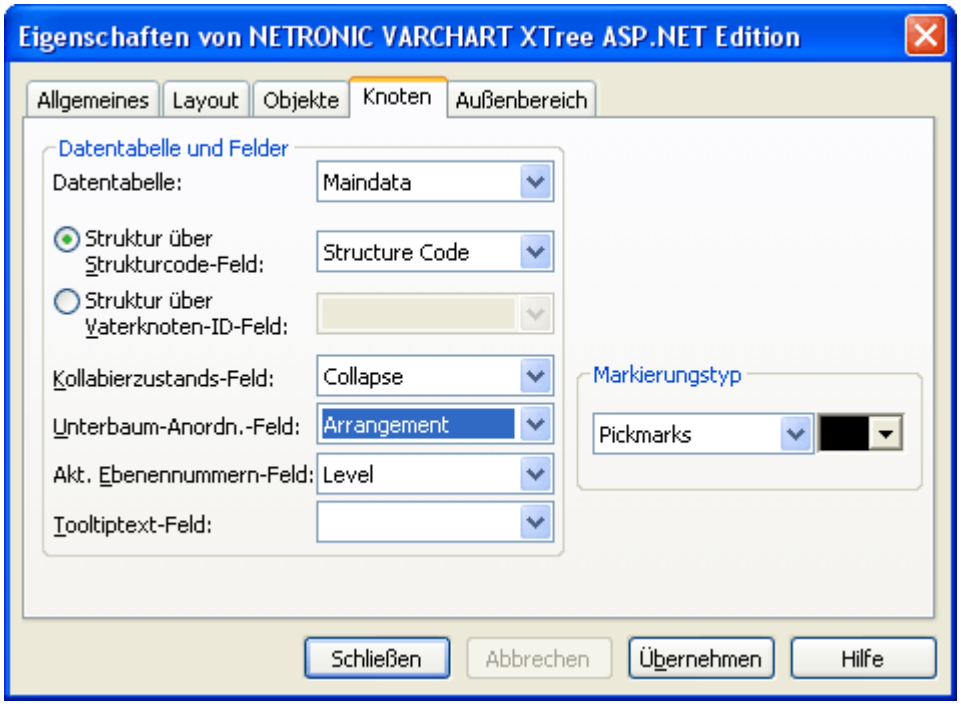
Um die vertikal angeordnete Teilstruktur zur Laufzeit horizontal anordnen zu lassen, markieren Sie den obersten Knoten der vertikal angeordneten Teilstruktur und wählen Sie anschließend im Kontextmenü den Befehl **Horizontal anordnen**. Die Teilstruktur des markierten Knotens wird dann horizontal angeordnet, aber nur eine Ebene tief. Auf die Anordnungen in den nächst tieferen Ebenen hat der Befehl **Horizontal anordnen** keine Auswirkung.

Mit dem Befehl **Unterbaum komplett horizontal anordnen** werden die Teilbäume unter den markierten Knoten vollständig, d. h. über alle Ebenen, horizontal angeordnet.

> Unterbaum-Anordnung in Datenfeld speichern

Sie können die Unterbaum-Anordnung, also den Zustand "Teilbaum horizontal angeordnet" bzw. "Teilbaum vertikal angeordnet", in einem Datenfeld speichern.

Kehren Sie dazu zum Designmodus zurück und öffnen Sie die Eigenschaftenseite **Knoten**.



Aktivieren Sie hier das Kontrollkästchen **Unterbaum-Anordnung in Feld** und wählen Sie aus der Kombobox das Datenfeld "Anordnung" aus. Dann wird die Orientierung eines Teilbaums synchron in dem Datenfeld "Anordnung" gehalten. Mögliche Inhalte des Datenfelds sind "0" (Teilbaum horizontal angeordnet) und "1" (Teilbaum vertikal angeordnet). Die

horizontale Anordnung ist nur sichtbar, wenn der direkte oder indirekte Vaterknoten selbst eine vertikale Anordnung besitzt.

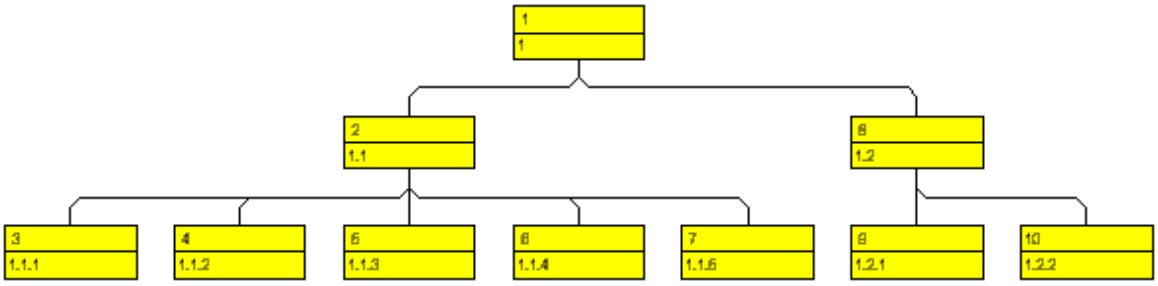
Starten Sie nun das Programm. Markieren Sie einen Knoten und klicken Sie doppelt auf die linke Maustaste, um den Dialog **Vorgänge bearbeiten** zu öffnen. Ist die Teilstruktur unterhalb des Knotens horizontal angeordnet, steht hier unter **Anordnung** der Wert "0". Ist die Teilstruktur unterhalb des Knotens vertikal angeordnet, steht hier unter **Anordnung** der Wert "1".

2.14 Baumstrukturen kollabieren und expandieren

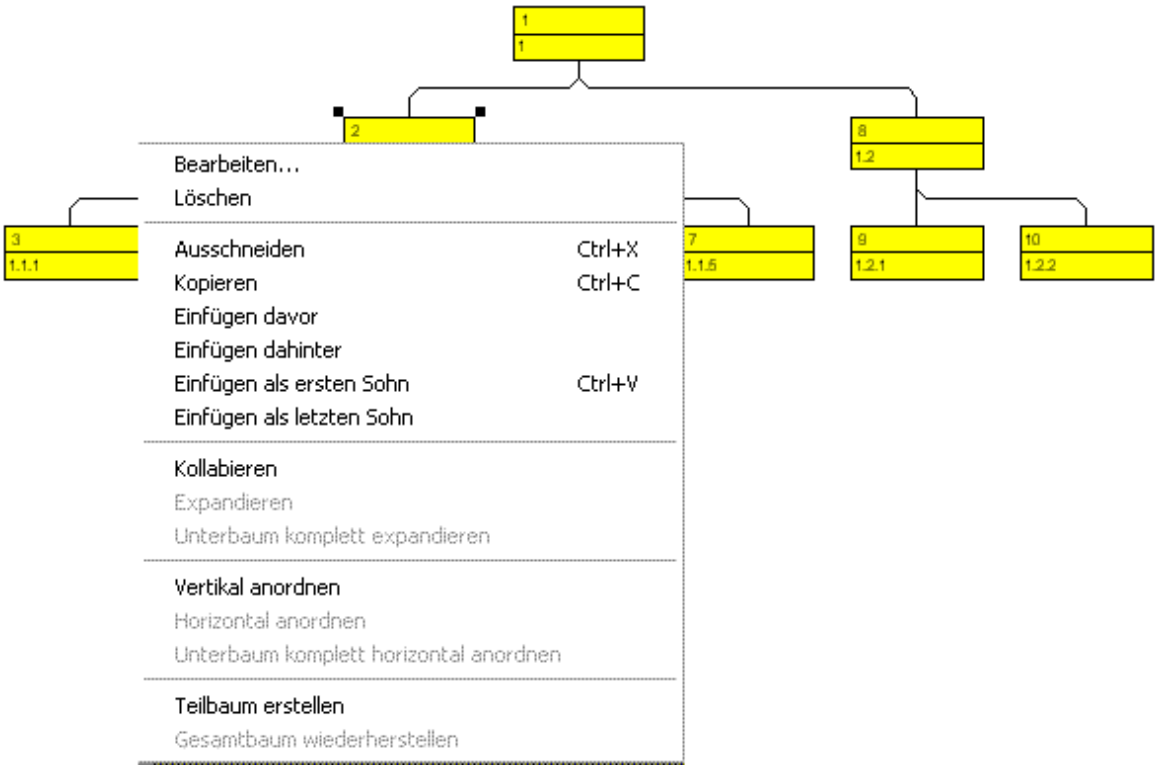
In diesem Kapitel erfahren Sie, wie Sie Baumstrukturen kollabieren und wieder expandieren können. Teilstrukturen von Baum-Diagrammen lassen sich kollabieren (wegklappen) und wieder expandieren (aufklappen). Jede beliebige Teilstruktur kann auf ihren obersten Knoten, den Gliederungsknoten, kollabiert werden. Als Gliederungsknoten wird jeweils der Wurzelknoten der Teilstruktur, die kollabiert werden soll, verwendet. Die kollabierte Teilstruktur kann ggf. wieder in ihre ursprüngliche Form expandiert werden.

Durch das Kollabieren von Teilstrukturen lassen sich auch sehr komplexe Strukturen übersichtlich gestalten. Wenn Sie beispielsweise nur bestimmte Teilstrukturen präsentieren möchten, kollabieren Sie einfach alle anderen Teilstrukturen. Dadurch, dass die kollabierten Teilstrukturen erhalten bleiben, geht die Information über die Gesamtstruktur nicht verloren.

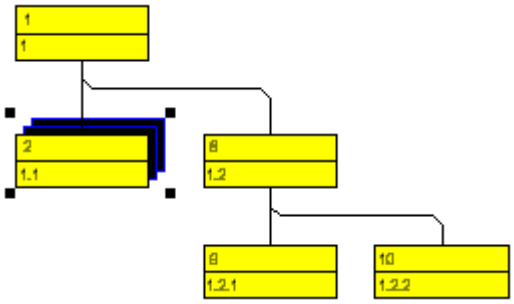
Starten Sie nun das Programm mit den Daten der Datei *tutorial.tre* (vgl. "Tutorium: Daten aus einer Datei einlesen").



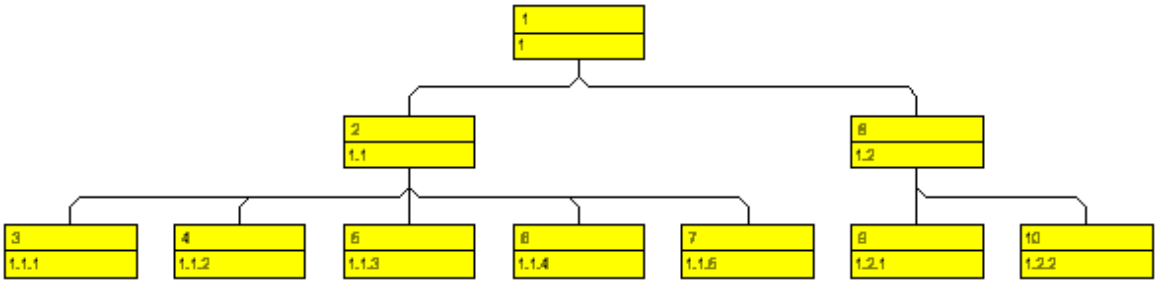
Markieren Sie nun den ersten Knoten der zweiten Ebene und klicken Sie auf die rechte Maustaste. Das Kontextmenü für Knoten erscheint.



Wählen Sie den Befehl **Kollabieren**. Nun werden alle Teilstrukturen, die zu dem markierten Wurzelknoten gehören, kollabiert. Die markierten Wurzelknoten verwandeln sich damit in Gliederungsknoten, die jeweils die nun verborgenen Teilstrukturen repräsentieren.

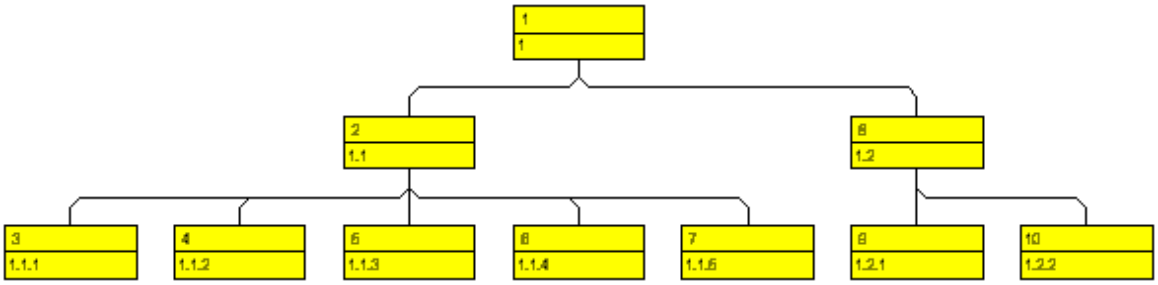


Wählen Sie nun den Befehl **Expandieren** aus dem Kontextmenü, um die Teilstrukturen wieder zu expandieren. Es werden jeweils nur die kollabierten Knoten expandiert. Befinden sich in den Unterbäumen weitere kollabierte Knoten, bleiben diese kollabiert.



Mit dem Befehl **Unterbaum komplett expandieren** aus dem Kontextmenü können Sie die kollabierten Teilstrukturen vollständig expandieren, d. h. dass dabei alle kollabierten Knoten in den Unterbäumen expandiert werden. Um diesen Befehl auszuprobieren, kollabieren Sie nun den ersten Knoten der zweiten Ebene und anschließend den Knoten der obersten Ebene.

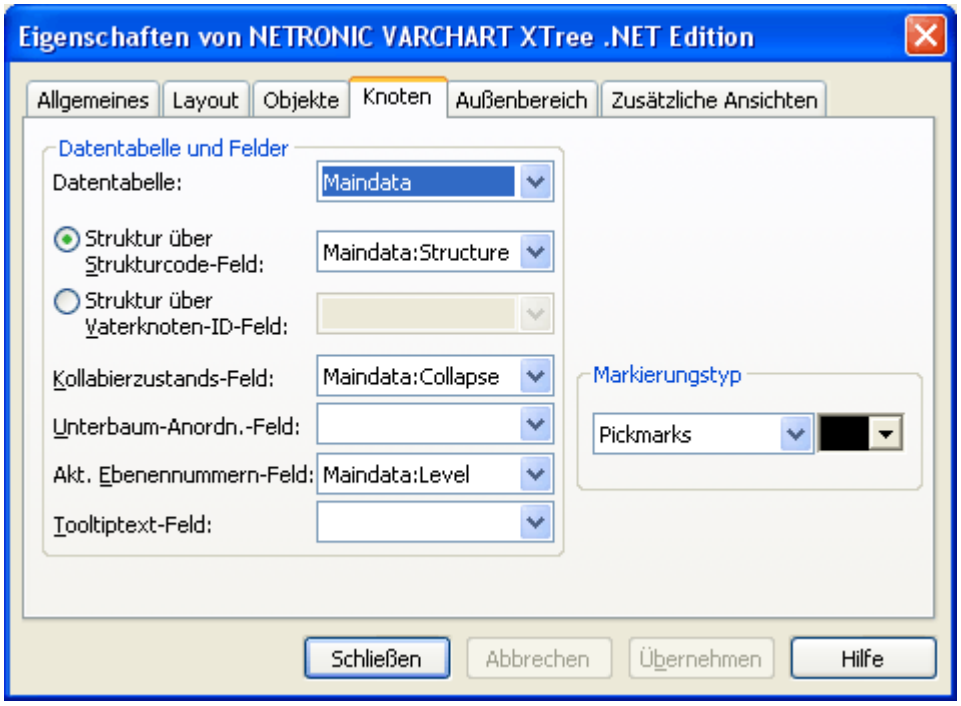
Wählen Sie nun den Befehl **Unterbaum komplett expandieren**. Nun wird der Baum wieder komplett expandiert.



> Kollabierzustand in Datenfeld speichern

Sie können den Kollabierzustand der Knoten in einem Datenfeld speichern.

Kehren Sie dazu zum Designmodus zurück und öffnen Sie die Eigenschaften-seite **Knoten**.



Aktivieren Sie hier das Kontrollkästchen **Kollabierzustands-Feld** und wählen Sie aus der Kombobox das Datenfeld "Kollabiert" aus. Dann wird der

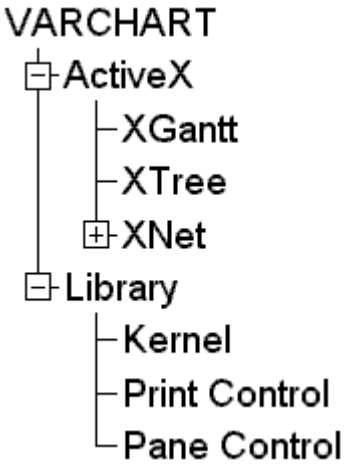
Kollabierstatus jedes Knotens synchron in dem Datenfeld "Kollabiert" gehalten. Mögliche Inhalte des Datenfelds sind "0" (Knoten expandiert) und "1" (Knoten kollabiert).

Starten Sie nun das Programm. Markieren Sie einen Knoten und klicken Sie doppelt auf die linke Maustaste, um den Dialog **Vorgänge bearbeiten** zu öffnen. Ist die Teilstruktur unterhalb des Knotens kollabiert, steht hier unter **Kollabiert** der Wert "1". Ist die Teilstruktur unterhalb des Knotens expandiert, steht hier unter **Kollabiert** der Wert "0".

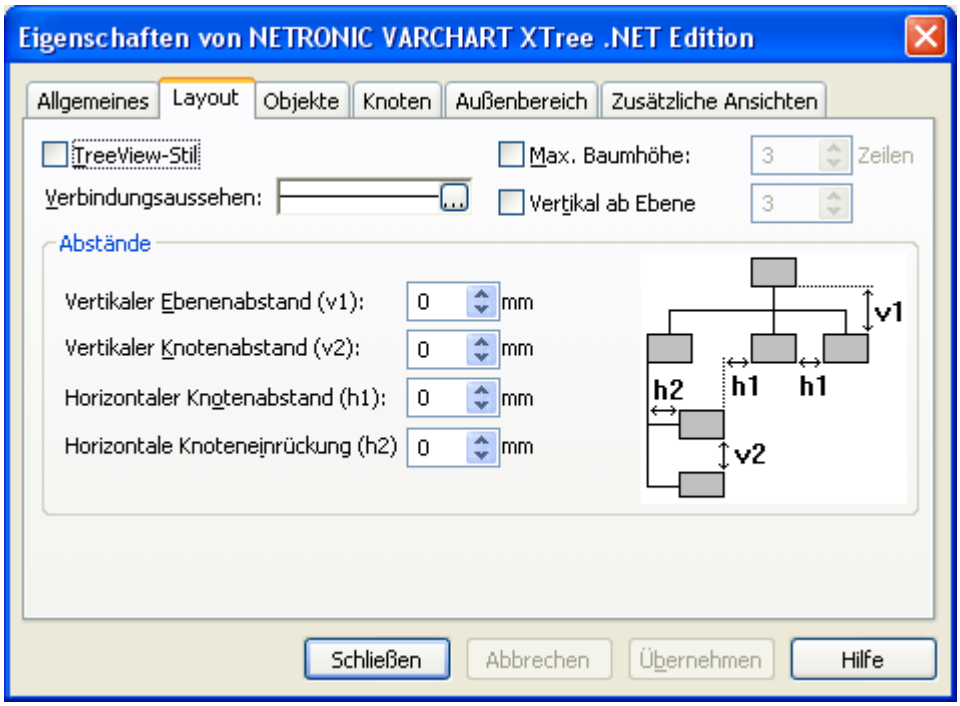
2.15 TreeView-Stil

Erfahren Sie nun, wie Sie Knoten im TreeView-Stil anordnen lassen können. Bei der TreeView-Ansicht werden vertikale Ebenen wie in TreeView-Controls (z. B. bekannt aus der Verzeichnisbaumansicht des Microsoft Explorers) mit Plus- oder Minus-Zeichen dargestellt. Dabei bedeutet ein Plus-Zeichen, dass der Knoten auf der gleichen Ebene kollabiert ist, und ein Minus-Zeichen, dass er expandiert ist. Die Zeichen werden nur bei Knoten angezeigt, die keine Blattknoten sind, d. h. die Sohnknoten besitzen. Ein Mausklick auf eines der beiden Zeichen überführt den daneben stehenden Knoten in den jeweils anderen Kollabierzustand.

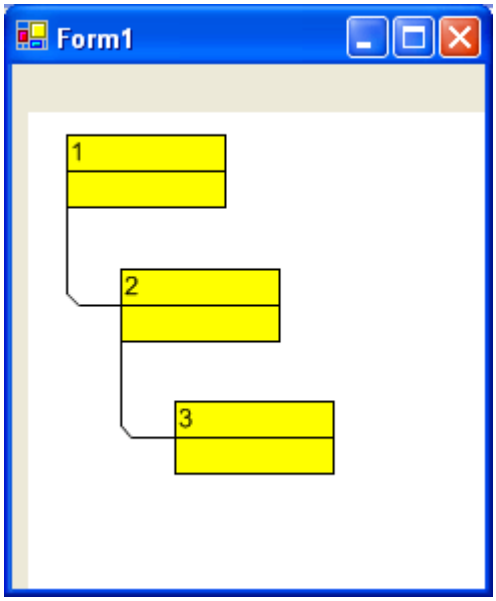
Ein Beispiel für den TreeView-Stil zeigt die folgende Abbildung:



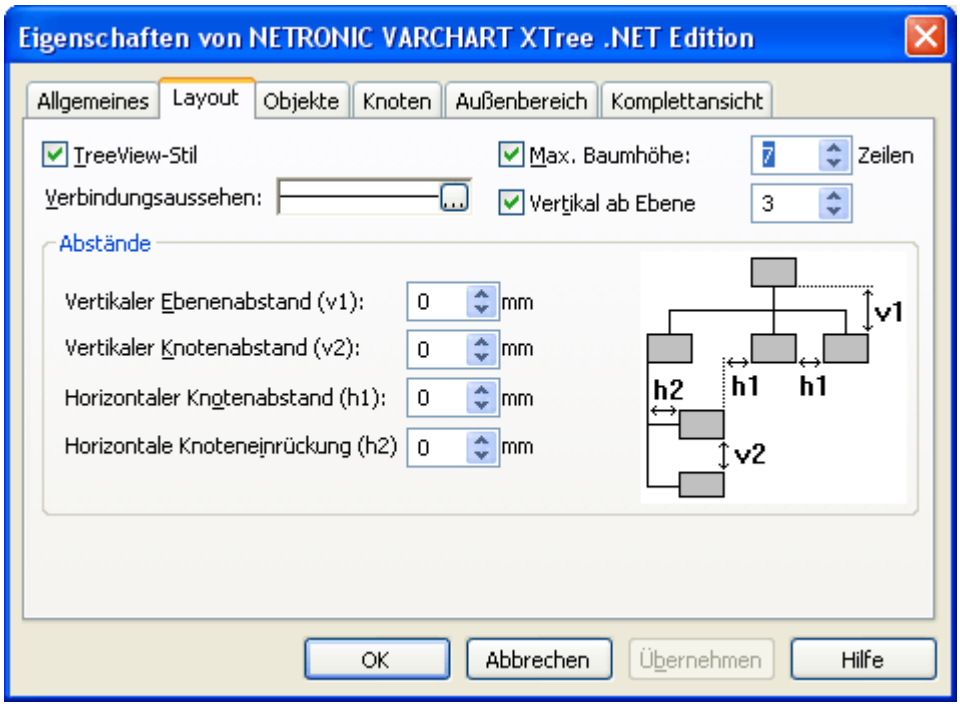
Das Kontrollkästchen **TreeView-Stil** auf der Eigenschaftenseite **Layout** bestimmt, ob die Knoten im TreeView-Stil angeordnet werden.



Deaktivieren Sie das Kontrollkästchen zunächst und starten Sie dann das Programm. Sie erhalten eine Anordnung ohne TreeView-Stil:



Wechseln Sie nun wieder zum Design-Modus und aktivieren Sie auf der Eigenschaftenseite **Layout** das Kontrollkästchen **TreeView-Stil**, um die Knoten jetzt im TreeView-Stil anordnen zu lassen:


Eigenschaften von NETRONIC VARCHART XTree .NET Edition
Allgemeines
Layout
Objekte
Knoten
Außenbereich
Komplettansicht
TreeView-Stil
Max. Baumhöhe:
7
Zeilen
Verbindungsaussehen:
Vertikal ab Ebene
3
Abstände
Vertikaler Ebenenabstand (v1):
0
mm
Vertikaler Knotenabstand (v2):
0
mm
Horizontaler Knotenabstand (h1):
0
mm
Horizontale Knoteneinrückung (h2)
0
mm
v1
h2
h1
h1
v2
OK
Abbrechen
Übernehmen
Hilfe

2.16 Diagramm drucken

Wenn Sie Ihr Diagramm nach Ihren Vorstellungen gestaltet haben, können Sie es schließlich ausdrucken. Wählen Sie dazu zur Laufzeit den Befehl **Drucken** des Kontextmenüs (rechter Mausklick im freien Diagrammbereich).

Alternativ können Sie auch die Methode **ShowPrintDialog** des Objektes VcTree verwenden.

Sie gelangen dann in das Windows-Dialogfeld **Drucken**. Gegebenenfalls können Sie zur Laufzeit auch die Druckereinstellungen bearbeiten, indem Sie den Befehl **Drucker einrichten** des Kontextmenüs aufrufen und damit das entsprechende Windows-Dialogfeld aufrufen.

Mit der Methode **PrintEx** des Objektes VcTree können Sie das Diagramm direkt ausdrucken, ohne dass zuvor ein Dialogfeld erscheint.

Wenn Sie zur Laufzeit die Seiteneinstellungen verändern möchten,können Sie aus dem Kontextmenü den Befehl **Seite einrichten** auswählen, oder auf

Seitenansicht klicken und dort auf die Schaltfläche **Seite einrichten**.

Alternativ können Sie auch die Methode **ShowPageSetupDialog** des Objektes VcTreet verwenden, um das entsprechende Dialogfeld aufzurufen.

Im **Seite einrichten**-Dialogfeld können Sie Einstellungen zur Skalierung, zur Seitenaufteilung, zur Seitennummerierung, zu den Seitenrändern etc. vornehmen. Weitere Informationen hierzu finden Sie im Kapitel 5.16, "Seite einrichten".

2.17 Diagramm exportieren

Sie können Ihr Diagramm auch als Grafikdatei exportieren. Dazu gibt es folgende Möglichkeiten:

- Wählen Sie bitte den Befehl **Grafik exportieren** des Standard-Kontextmenüs. Dann gelangen Sie in das Windows-Dialogfeld **Speichern unter**, in dem Sie das dargestellte Diagramm als Grafikdatei speichern können.
- Verwenden Sie die API-Methode **ShowExportGraphicsDialog** bzw. **ExportGraphicsToFile**

Detaillierte Erläuterungen zu den Grafikformaten finden Sie im Kapitel: **Wichtige Konzepte: Grafikformate**.

2.18 Konfigurationseinstellungen speichern

Alle Einstellungen, die Sie auf den Eigenschaftenseiten zur Design-Zeit vornehmen, werden als Ressource Ihrem Projekt hinzugefügt. Vorgenommene Änderungen werden erst wirksam, wenn Sie Ihr Projekt speichern, denn durch das Speichern wird die eingebettete Ressource aktualisiert.

Tipp: Aus diesem Grunde bietet es sich an, dass Sie in Microsoft Visual Studio .NET 2005 unter **Extras >Optionen > Umgebung > Projekte und Projektmappen > Erstellen und Ausführen** (diese Auswahl ist nur verfügbar, wenn **Alle Einstellungen anzeigen** ausgewählt ist) die Option **Alle Änderungen speichern** aktivieren, damit Ihre Einstellungen vor dem Kompilieren automatisch gespeichert werden.

Sollten Sie diese Option nicht aktiviert haben, dann müssen Sie Ihr Projekt immer manuell speichern, wenn Sie möchten, dass die in den Eigenschaftenseiten vorgenommenen Änderungen im Programm verwendet werden.

Alle Einstellungen der Eigenschaftenseiten können Sie auch jederzeit in Form einer Konfiguration außerhalb Ihres Projektes speichern und nach Bedarf wieder einlesen. Dies ist sehr praktisch, wenn Sie zu einem früheren Stand der Einstellungen zurückkehren oder die gleichen Einstellungen für andere Projekte verwenden möchten.

Eine gespeicherte Konfiguration besteht aus zwei Dateien mit gleichem Namen aber unterschiedlichen Dateiendungen. Zu einer Konfiguration gehört jeweils eine ini- und eine ifd-Datei, die beide zwingend benötigt werden.

So speichern Sie Ihre aktuelle Konfiguration:

Klicken Sie auf der Eigenschaftenseite **Allgemeines** auf die Schaltfläche **Exportieren als...** und vergeben im folgenden Dialog den gewünschten Dateinamen für die ini-Datei. Die ifd-Datei wird automatisch erzeugt.

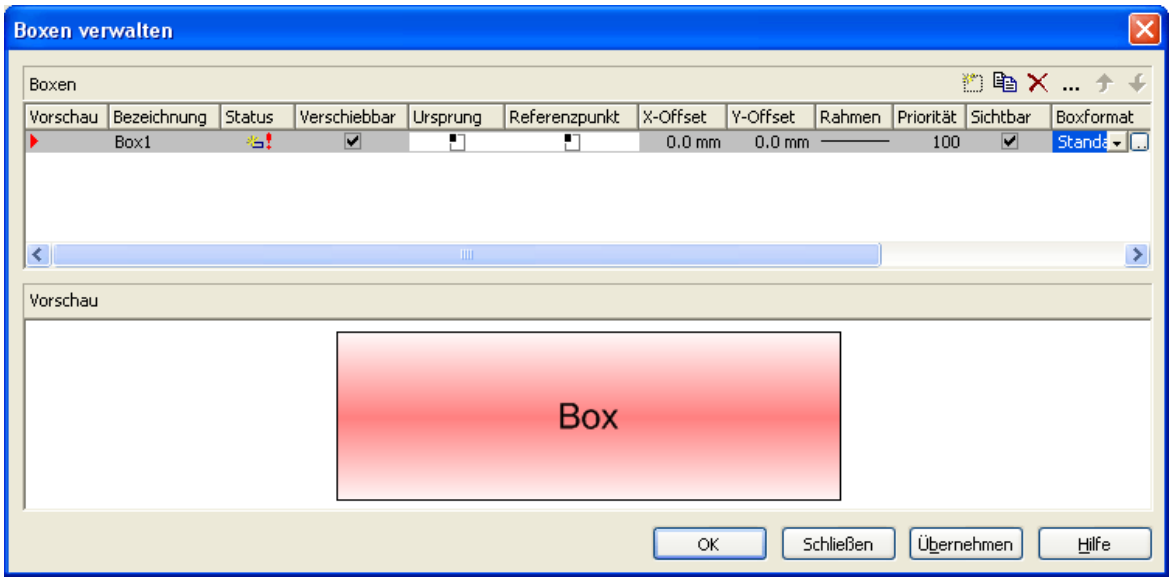
So lesen Sie eine bereits gespeicherte Konfiguration wieder ein:

Klicken Sie auf der Eigenschaftenseite **Allgemeines** auf die Schaltfläche **Import...** und wählen die gewünschte Datei aus.

3 Wichtige Konzepte

3.1 Boxen

Im Diagrammbereich können beliebig viele Boxen, die Text oder Grafiken enthalten können, dargestellt werden. Über die Eigenschaftenseite **Objekte** und die Schaltfläche **Boxen...** gelangen Sie zum Dialog **Boxen verwalten**, in dem Sie Boxen neu anlegen, kopieren, bearbeiten und löschen können.



Mithilfe der Eigenschaften **Ursprung**, **Referenzpunkt**, **X-Offset** und **Y-Offset** können Sie jede einzelne Box im Diagrammbereich positionieren, wobei die relative Position der Box zum Diagramm unabhängig von der Diagrammgröße ist.

Für jede Box können Sie hier folgendes festlegen:

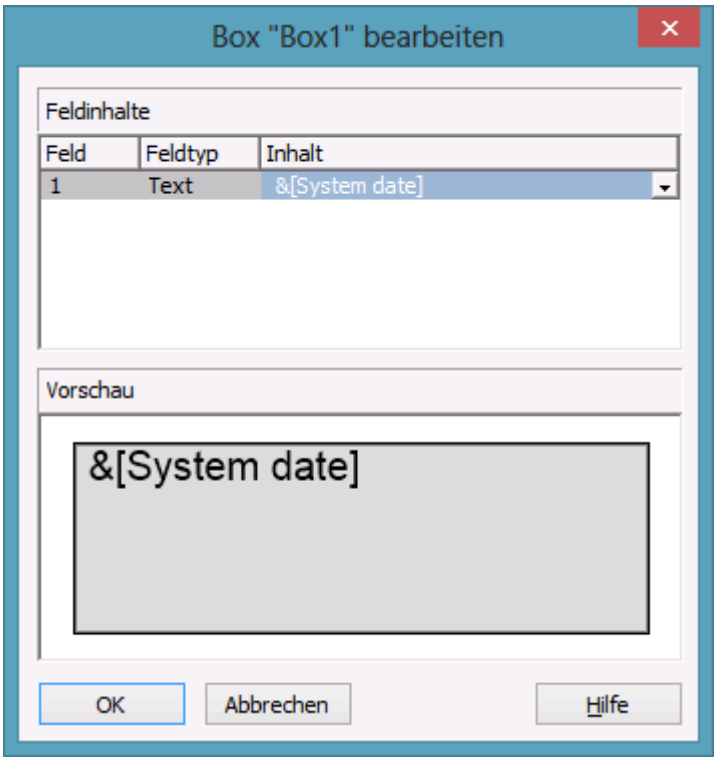
- ihre Bezeichnung
- ob die Box zur Laufzeit frei im Diagrammbereich verschiebbar sein soll
- ihren Ursprung (den Diagrammpunkt, von dem aus der Abstand zum Referenzpunkt der Box in x- bzw. y-Richtung angegeben wird)
- ihren Referenzpunkt (Punkt der Box, von dem aus der Abstand zum Ursprung in x- bzw. y-Richtung angegeben wird)
- ihren X- bzw. Y-Offset (Abstand zwischen Ursprung und Referenzpunkt in x- bzw. y-Richtung)

- Typ, Dicke und Farbe der Umrandungslinie der Box
- ihre Priorität gegenüber anderen Objekten im Diagramm
- ob die Box sichtbar ist
- das Boxformat

> Boxen bearbeiten

Im Dialogfeld **Box bearbeiten** können Sie den Inhalt der Felder festlegen.

Zur Designzeit erreichen Sie dieses Dialogfeld, indem Sie im Dialogfeld **Boxen verwalten** auf die **Box bearbeiten**-Schaltfläche klicken. Zur Laufzeit kann ein Benutzer durch Doppelklick mit der linken Maustaste auf die jeweilige Box in das Dialogfeld gelangen. Wenn die Option **In-Place-Editieren zulassen** auf der Eigenschaftsseite **Allgemeines** aktiviert wird, können Texte zur Laufzeit auch direkt bearbeitet werden.



In der Spalte **Feld** werden die Nummern aller Felder der Box aufgeführt. (Die Anzahl der Felder hängt vom gewählten Boxformat ab.)

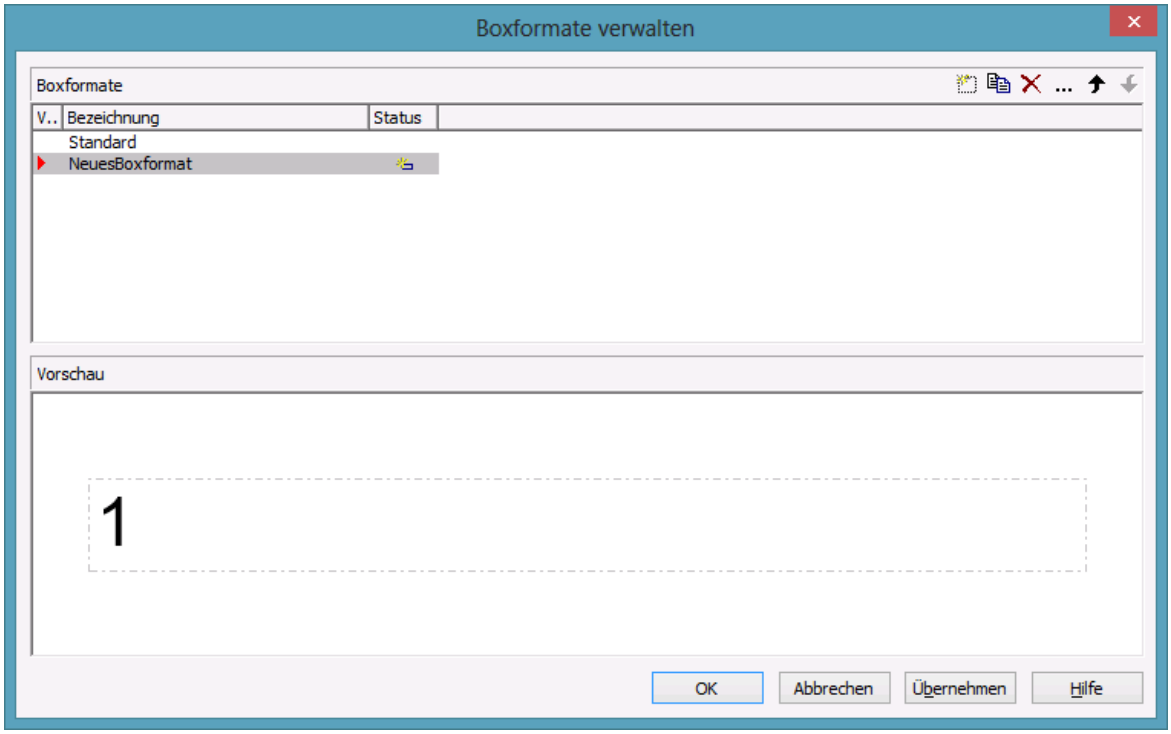
In der Spalte **Typ** wird der Feldtyp jedes Feldes angezeigt (Text oder Grafik).

In der Spalte **Inhalt** können Sie den Inhalt des Feldes bzw. den Namen einer Grafikdatei eingeben. Bei mehrzeiligen Textfeldern müssen für einen erzwungenen Umbruch die einzelnen Zeilen des Textfelds mit "\n" im String getrennt sein (Beispiel: "Zeile1\nZeile2"). Ohne erzwungenen Umbruch wird automatisch an Leerzeichen umgebrochen.

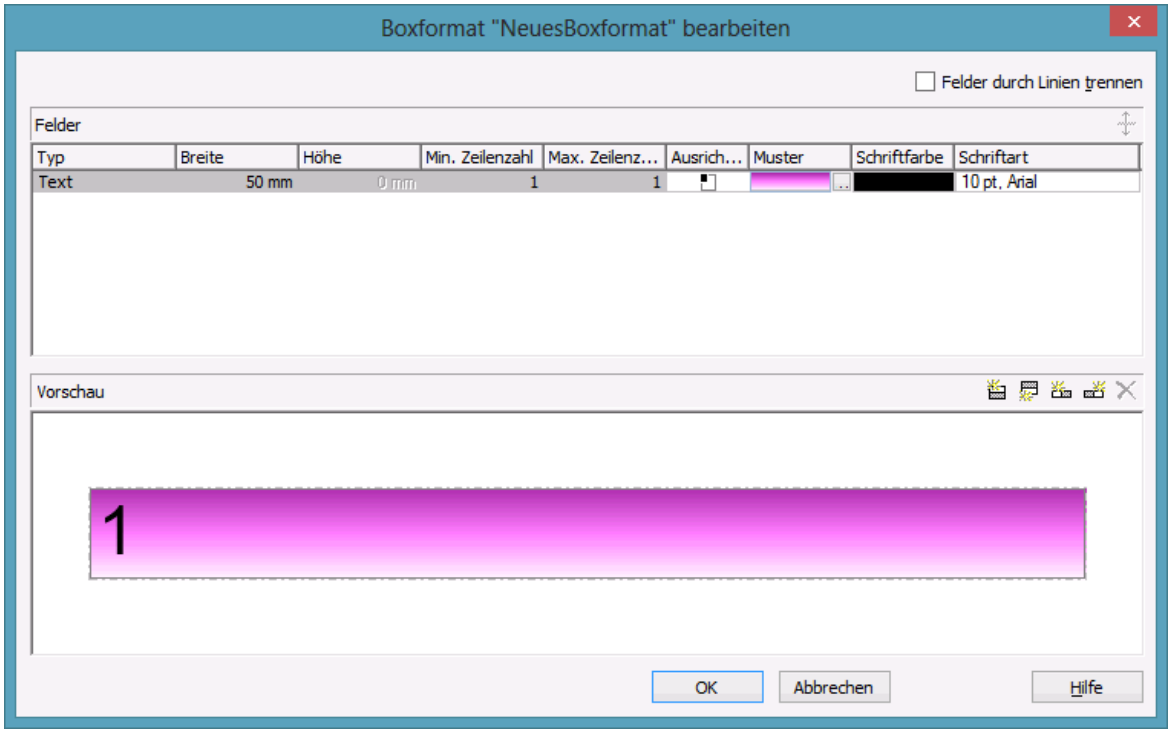
> Boxformate

Für jede Box können Sie ein Boxformat wählen. Die Boxformate können Sie selbst festlegen.

Im Dialog **Boxformate verwalten** können Sie Boxformate hinzufügen, kopieren, bearbeiten und löschen. Sie erreichen dieses Dialogfeld über die entsprechende Schaltfläche auf der Eigenschaftenseite **Objekte**.



Im Dialog **Boxformat bearbeiten** können Sie das Boxformat festlegen. Sie erreichen dieses Dialogfeld, indem Sie im Dialogfeld **Boxformate verwalten** auf die Schaltfläche ··· klicken.



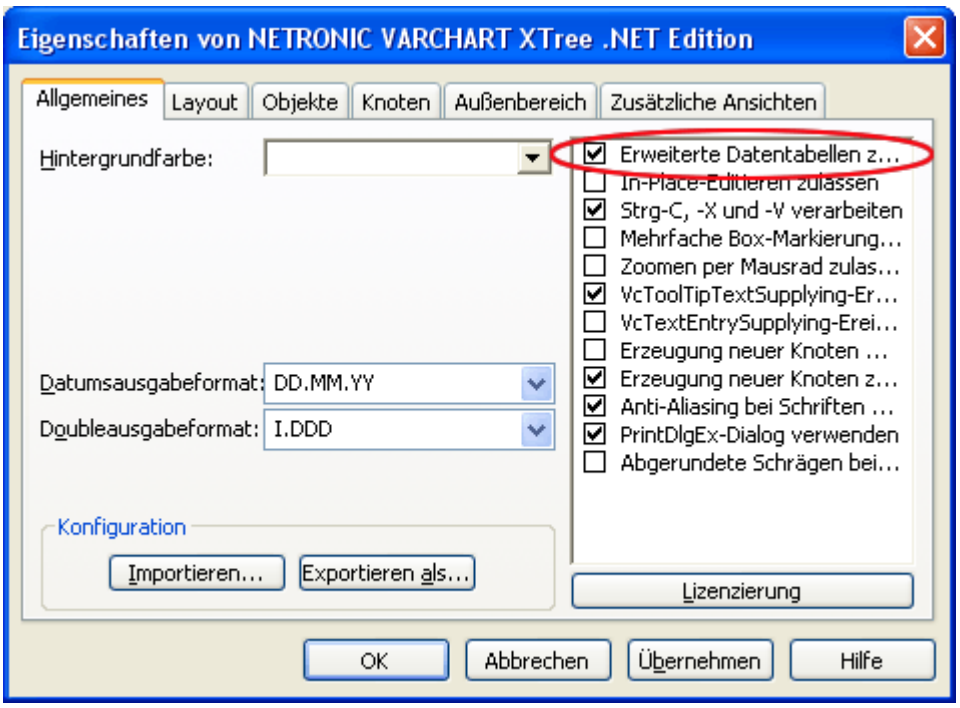
Sie können hier festlegen, ob die Felder der Box durch Linien getrennt werden sollen. Außerdem können Sie für jedes Feld der Box folgendes festlegen:

- Feldtyp (Text oder Grafik)
- Breite und Höhe
- Zeilenzahl
- Ausrichtung
- Füllfarbe und -muster
- Schriftattribute

3.2 Datentabellen

Zur Darstellung von Baum-Diagrammen verwendet VARCHART XTree eine Standard-Datentabelle für Knoten, deren Felder individuell festgelegt werden können. In VARCHART XTree 4.0 wurde dieses Konzept erweitert. Es können jetzt bis zu 90 Datentabellen vereinbart werden und zusätzlich Beziehungen in Form von 1:n Relationen zwischen diesen Tabellen definiert werden. Damit werden in bestimmten Situationen Redundanzen vermieden und der Zugriff auf die Felder des Hauptdatensatzes vom Detaildatensatz aus erleichtert.

Aus Kompatibilitätsgründen mit bestehenden Anwendungen arbeitet VARCHART XTree im bisherigen Modus. Erst durch Aktivieren der entsprechenden Option zur Design- oder Laufzeit können die neuen Möglichkeiten genutzt werden. Auf der Eigenschaftenseite **Allgemeines** finden Sie dazu die Option **Erweiterte Datentabellen zulassen**:



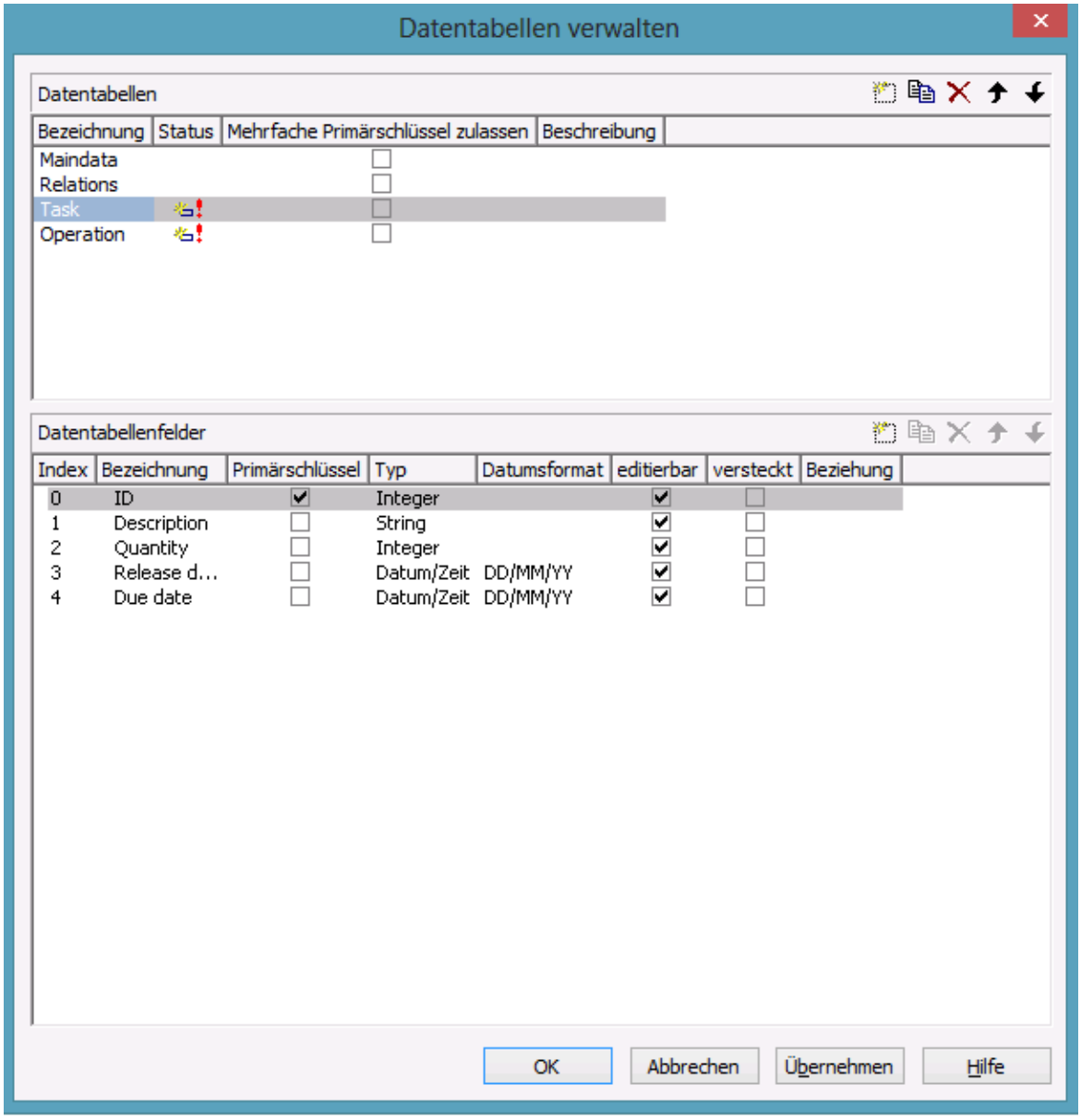
Alternativ können die erweiterten Datentabellen statt zur Designzeit auch zur Laufzeit über die API eingeschaltet werden, indem die Eigenschaft **ExtendendDataTablesEnabled** des Objekts **VcTree** auf **True** gesetzt wird.

> Datentabellen verwalten

Standardmäßig ist die Datentabelle **Maindata** vorhanden. Auf der Eigenschaftenseite **Objekte** gelangen Sie über die Schaltfläche **Datentabellen…** in

den Dialog **Datentabellen verwalten**. Nur im Modus **Erweiterte Datentabellen zulassen** ist es möglich, neue Datentabellen anzulegen.

Im Bereich **Datentabellenfelder** können Sie neue Felder anlegen , bestehende Felder löschen oder Felder kopieren .



Die Spalte **Index** besitzt eine zentrale Bedeutung, denn der Zugriff auf die Inhalte der Datenfelder innerhalb eines Datensatzes geschieht ausschließlich über den Index. Wenn Sie die Reihenfolge der Felder ändern, nachdem bereits Programmcode existiert, dann müssen Sie den Programmcode für den Datenfeldzugriff ebenso anpassen.

Die Änderung des Datentyps eines Feldes kann bewirken, dass bereits definierte Formate und Layer geändert werden müssen, um den Zugriff mit dem richtigen Datentyp sicherzustellen.

Mit der Primärschlüssel-Eigenschaft kennzeichnen Sie das Feld, dessen Werte die Datensätze einer Datentabelle eindeutig unterscheidbar machen. Der Primärschlüssel kann auch aus mehreren Feldern - *jedoch nicht mehr als 3* - bestehen. Eine ausführliche Beschreibung zur Handhabung von zusammengesetzen Primärschlüsseln finden sie im Kapitel **Datentabellen verwalten**.

Für jede Datentabelle, auf die in einer Verknüpfung verwiesen wird, ist die Festlegung eines Primärschlüsselfeldes zwingend erforderlich.

Die Verknüpfung zweier Tabellen ist sinnvoll, wenn eine inhaltliche 1:n Beziehung zwischen den Tabellen besteht und von jedem Detaildatensatz direkt auf ein Datenfeld im Hauptdatensatz Bezug genommen werden soll.

Zwischen zwei Tabellen A und B ist derzeit nur eine einzige 1:n-Beziehung möglich; es kann sich also kein zweites Feld der Tabelle B auf den Primärschlüssel in A beziehen. Allerdings kann sich ein Feld aus einer weiteren Tabellen C auf den Primärschlüssel in A beziehen.

Hinweis: Wird eine Datentabelle mit einem zusammengesetzten Primärschlüssel in einer Beziehung verwendet, muss auch die Beziehung entsprechend definiert werden. Andernfalls ist eine eindeutige Anbindung nicht möglich. Ist die Beziehung nicht vollständig definiert - was weder an der API noch im Dialog **Datentabellen verwalten** überprüft wird, findet **keine** Datensatzanbindung statt, was zum Ereignis **VcDataRecordNotFound** führt.

In der VARCHART XTree Version 4.0 sind aufgrund der erweiterten Datentabellen eine Reihe neuer Objekttypen entstanden, die bestehende Objekttypen ablösen werden. Aus Gründen der Kompatibilität sind in dieser Version die bisherigen Objekttypen noch enthalten. Für neue Anwendungen und beim Aktualisieren von bestehenden Anwendungen sollten nur noch die neuen Objekttypen eingesetzt werden.

Bisher	Ab Version 4.0
VcDataDefinition	VcDataTableCollectionVcDataTable
VcDataDefinitionTable	VcDataTableFieldCollection
VcDefinitionField	VcDataTableField
	VcDataRecordCollection
	VcDataRecord

> Datensätze anlegen und ändern

Nach der Definition der Datentabellenfelder können Sie einer Tabelle über die API Datensätze hinzufügen. Es gibt zwei Wege, auf denen die Datensätze mit Daten gefüllt werden können. Der normale und empfehlenswerte Weg besteht in der Vereinbarung eines Arrays vom Datentyp Object, wobei dessen Elementeanzahl auf die Anzahl der Datentabellenfelder gesetzt wird.

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection

Dim dataRecVal() As Object
Dim dataRec1 As VcDataRecord
Dim dataRec2 As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection

ReDim dataRecVal(dataTable.DataTableFieldCollection.Count)

dataRecVal(Main_ID) = "1"
dataRecVal(Main_Name) = "Node 1"
dataRecVal(Main_Start) = DateSerial(2013, 1, 8)
dataRecVal(Main_Duration) = 8
```

Code-Beispiel C#

```
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

Object [] dataRecVal = new
object[dataTable.DataTableFieldCollection.Count];

VcDataRecord dataRec1;
VcDataRecord dataRec2;

dataRecVal[Main_ID] = "1";
dataRecVal[Main_Name] = "Node 1";
dataRecVal[Main_Start] = "08.01.2013";
dataRecVal[Main_Duration] = 8;
```

Ein neuer Datensatz wird durch die Methode **Add()** des Objekts **DataRecordCollection** erzeugt, der als Parameter das Object-Array mitgegeben wird.

Code-Beispiel VB.NET

```
dataRec1 = dataRecCltn.Add(dataRecVal)
```

Code-Beispiel C#

```
dataRec1 = dataRecCltn.Add(dataRecVal);
```

Der zweite Weg besteht in der Verwendung einer Zeichenkette, bei der die Datenwerte durch Semikolon getrennt aneinander gereiht werden.

Code-Beispiel VB.NET

```
dataRecCltn.Add("2;Node 2;15.01.13;;9")
```

Code-Beispiel C#

```
dataRec2.AllData = "2;Activity Y;15.01.13;;9";
```

Wenn ein Semikolon selber im Datenwert enthalten ist, dann muss die Zeichenkette in doppelte Hochkommata eingeschlossen werden.

Code-Beispiel VB.NET

```
dataRec2 = dataRecCltn.Add("2;""Node 2;"";15.01.13;;9")
```

Code-Beispiel C#

```
dataRec2 = dataRecCltn.Add("2;\"Node 2;\";15.01.13;;9");
```

Der Verweis auf ein Datensatzobjekt kann mit Hilfe des Primärschlüssels über die Methode **DataRecordByID ()** schnell gefunden werden.

Code-Beispiel VB.NET

```
dataRec1 = dataRecCltn.DataRecordByID("1")
dataRec2 = dataRecCltn.DataRecordByID("2")
```

Code-Beispiel C#

```
dataRec1 = dataRecCltn.DataRecordByID(1);
dataRec2 = dataRecCltn.DataRecordByID(2);
```

Einzelne Datenfeldinhalte eines Datensatzes können Sie leicht über die indizierte Eigenschaft **DataField()** ändern. Mit der Eigenschaft **AllData** lassen sich alle Datenfeldinhalte eines Datensatzes auf einmal ersetzen.

Code-Beispiel VB.NET

```
dataRec1.DataField(Main_ID) = 1
dataRec1.DataField(Main_Name) = "Activity X"
dataRec1.DataField(Main_Start) = DateSerial(2013, 1, 4)
dataRec1.DataField(Main_Duration) = 12
```

```
dataRec1.Update()

dataRec2.AllData = "2;Activity Y;18.01.13;;5"
dataRec2.Update()
```

Code-Beispiel C#

```
dataRec1.set_DataField(Main_ID, 1);
dataRec1.set_DataField(Main_Name, "Activity X");
dataRec1.set_DataField(Main_Start, "04.01.2014");
dataRec1.set_DataField(Main_Duration, 12);
dataRec1.Update();

dataRec2.AllData = "2;Activity Y;18.01.14;;5";
dataRec2.Update();
```

Jede Änderung in einem Datensatz wird erst darstellungswirksam, wenn die Methode **Update()** des Objekts **DataRecord** aufgerufen wird.

Das Auslesen der Werte mit **Alldata** eignet sich für die schnelle Anzeige aller Datenwerte während der Programmentwicklung und zum leichten Übertragen des Datensatzinhalts in den Datensatz einer anderen Tabelle. Ebenso wird dieses Datenformat bei OLE Drag & Drop zum Informationsaustausch benutzt.

Code-Beispiel VB.NET

```
Dim content As String
content = dataRec1.AllData & vbCr & dataRec2.AllData & vbCr &
dataRec1.DataField(Main_Name)
MsgBox(content)
```

Code-Beispiel C#

```
content = dataRec1.AllData + "\r\n" + dataRec2.AllData + "\r\n" +
dataRec1.get_DataField(Main_Name);
MessageBox.Show(content);
```

Tipp:

Um die Lesbarkeit bei den Datenfeldzugriffen zu erhöhen, können Sie globale Konstanten definieren, deren Namen sprechender sind als die Indexzahlen.

Hier noch einmal das gesamte Programmfragment im Zusammenhang:

Code-Beispiel VB.NET

```
Const Main_ID = 0
Const Main_Name = 1
Const Main_Start = 2
Const Main_Duration = 4
```

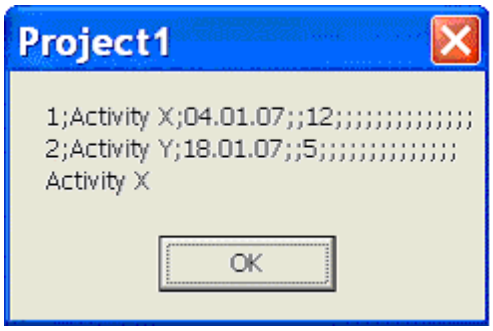
```
'...

Dim dataRec1 As VcDataRecord
Dim dataRec2 As VcDataRecord

Dim content As String

VcTree1.TimeScaleEnd = DateSerial(2014, 1, 1)
VcTree1.TimeScaleStart = DateSerial(2013, 1, 1)

VcTree1.ExtendedDataTablesEnabled = True
dataTable = VcNet1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection

ReDim dataRecVal(dataTable.DataTableFieldCollection.Count)

dataRecVal(Main_ID) = "1"
dataRecVal(Main_Name) = "Node 1"
dataRecVal(Main_Start) = DateSerial(2013, 1, 8)
dataRecVal(Main_Duration) = 8
dataRec1 = dataRecCltn.Add(dataRecVal)

dataRecCltn.Add("2;Node 2;15.01.13;;9")

VcTree1.EndLoading()

'...

dataRec1 = dataRecCltn.DataRecordByID("1")
dataRec2 = dataRecCltn.DataRecordByID("2")

dataRec1.DataField(Main_ID) = 1
dataRec1.DataField(Main_Name) = "Activity X"
dataRec1.DataField(Main_Start) = DateSerial(2013, 1, 4)
dataRec1.DataField(Main_Duration) = 12
dataRec1.Update()

dataRec2.AllData = "2;Activity Y;18.01.13;;5"
dataRec2.Update()

content = dataRec1.AllData & vbCr & dataRec2.AllData & vbCr &
dataRec1.DataField(Main_Name)
MsgBox(content)

'...


dataRec2.AllData = "2;""Activity Y;Z"";18.01.13;;5"
dataRec2.Update()
content = dataRec1.AllData & vbCr & dataRec2.AllData
MsgBox(content)
```

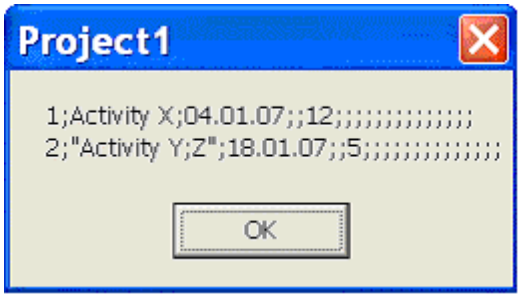
Code-Beispiel C#

```
const int Main_ID = 0;
const int Main_Name = 1;
const int Main_Start = 2;
const int Main_Duration = 4;
```

```
//...

VcDataRecord dataRec1;
VcDataRecord dataRec2;

string content;

vcTree1.TimeScaleEnd = Convert.ToDateTime("01.01.2014");
vcTree1.TimeScaleStart = Convert.ToDateTime("01.01.2013");

vcTree1.ExtendedDataTablesEnabled = true;
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;
Object [] dataRecVal = new
object[dataTable.DataTableFieldCollection.Count];)

dataRecVal[Main_ID] = "1";
dataRecVal[Main_Name] = "Node 1";
dataRecVal[Main_Start] = "08.01.2013";
dataRecVal[Main_Duration] = 8;
dataRec1 = dataRecCltn.Add(dataRecVal);

dataRecCltn.Add("2;Node 2;15.01.13;;9");

vcTree1.EndLoading();

//...

dataRec1 = dataRecCltn.DataRecordByID(1);
dataRec2 = dataRecCltn.DataRecordByID(2);

dataRec1.set_DataField(Main_ID, 1);
dataRec1.set_DataField(Main_Name, "Activity X");
dataRec1.set_DataField(Main_Start, "04.01.2013");
dataRec1.set_DataField(Main_Duration, 12);
dataRec1.Update();

dataRec2.AllData = "2;Activity Y;18.01.13;;5";
dataRec2.Update();

content = dataRec1.AllData + "\r\n" + dataRec2.AllData + "\r\n" +
dataRec1.get_DataField(Main_Name);
MessageBox.Show(content);

//...

dataRec2.AllData = "2;Activity Y;Z;18.01.13;;5";
dataRec2.Update();
content = dataRec1.AllData + "\r\n" + dataRec2.AllData;
MessageBox.Show(content);
```

Erzeugte Ausgabe:


Project1
1;Activity X;04.01.07;;12;;;;;;;;;;;;;;;
2;Activity Y;18.01.07;;5;;;;;;;;;;;;;;;
Activity X
OK


Project1
1;Activity X;04.01.07;;12;;;;;;;;;;;;;;;
2;"Activity Y;Z";18.01.07;;5;;;;;;;;;;;;;;;
OK

3.3 Datumsangaben und Zeitumstellung

Datumsangaben in VARCHART Komponenten sind immer bezogen auf die im System eingestellte Zeitzone. Es ist nicht möglich, Datumsangaben aus anderen Zeitzonen anzugeben, d.h. diese müssen vor der Übergabe an die Komponente für die eingestellte Zeitzone umgerechnet werden. Dabei beachtet die VARCHART-Komponente automatisch die im System vorhandenen Informationen zu Beginn und Ende der Sommerzeit.

Damit eine VARCHART-Komponente die Umschaltzeitpunkte vom System mitgeteilt bekommt, ist es wichtig, dass die Option **Uhr automatisch auf Sommer-/Winterzeit umstellen** gesetzt ist, wie im Bild gezeigt. Der Dialog ist im Windows-Betriebssystem unter "Start" > "Einstellungen" > "Systemsteuerung" > "Datum und Uhrzeit" über einen Doppelklick auf die Uhrzeit in der Task-Leiste erreichbar.

Eine VARCHART-Komponente nimmt bei der Umschaltung den Startzeitpunkt, den Endzeitpunkt inkl. Stunde, Monat und Tag, den das System als Regel mitteilt. Das heißt aber auch, dass die Umschaltzeiten für die Jahre vor und nach dem aktuellen Jahr extrapoliert werden, sodass etwaige Abweichungen, die für vorherige Jahre galten bzw. für kommende Jahre gelten werden, nicht berücksicktigt werden können, weil sie dem Betriebssystem ebenfalls nicht bekannt sind. Beispielsweise wurde die Sommerzeit in den USA vor wenigen Jahren um Wochen am Beginn und Ende verlängert. Da dem System hier nur die aktuelle Regelung bekannt ist,

werden Datumsangaben in der betroffenen Vergangenheit hier systembedingt falsch interpretiert.

Augenblicklich können VARCHART Komponenten bei der Sommerzeit nur eine Zeitdifferenz zur Winterzeit von exakt einer Stunde berücksichtigen. Außerdem darf die Umschaltung nur zur vollen Stunde erfolgen.

Da die VARCHART-Komponenten die Datumsangaben immer in lokaler Zeit entgegennimmt und ausgibt, gibt es am Beginn der Sommerzeit eine nicht erlaubte Stunde und am Ende der Sommerzeit zwei Stunden mit derselben Zahl, die derzeitig bei Übergabe, bei Rückgabe und bei Ausgabe nicht unterschieden werden kann.

3.4 Drag & Drop

Neben dem Verschieben oder Kopieren von Vorgängen innerhalb einer Instanz der VARCHART XTree Komponente kann ein Benutzer Vorgänge auch über die Grenzen einer Instanz (Quellkomponente) hinaus in eine andere Instanz (Zielkomponente) bewegen. In diesem Kapitel werden Themen behandelt, die vom Programmierer für diesen Interaktionstyp von Bedeutung sind.

Während sich bei einer Bewegung von Knoten innerhalb einer Instanz die Daten (Termine) der Knoten ändern, erfolgt beim Verschieben zwischen zwei Instanzen keine Datenänderung (welche aber dann in einem zweiten Bewegungsprozess innerhalb der zweiten Instanz möglich wäre).

Die Bewegung eines Knotens teilt sich in zwei Teilbewegungen: zum einen die Bewegung aus der Quellkomponente hinaus und zum anderen die Bewegung in die Zielkomponente hinein. Beide Teilbewegungen müssen von der jeweiligen Komponente erlaubt werden.

XTree ermöglicht es, einzelne oder mehrere Vorgänge gleichzeitig zu verschieben oder zu kopieren. Wird die linke Maustaste auf einem Vorgang gedrückt, wird intern ein Objekt des Typs **System.Windows.Forms.Data-Object** angelegt und mit den Vorgangsdaten im CSV-Format (d.h. Text bzw. Typ **System.String**) gefüllt. Anschließend wird direkt das Ereignis **VcDragStarting** ausgelöst. Damit kann die Anwendung die erlaubten Aktionen (Kopieren und/oder Verschieben) selbst bestimmen. Als Vorgabe ist beides, sowohl Kopieren als auch Verschieben, je nach Status der <Strg>-Taste möglich: Wird sie beim Loslassen der Maustaste gedrückt, wird kopiert, ansonsten verschoben.

Abschließend wird direkt das Ereignis VcDragCompleting ausgelöst, um der Anwendung zu erlauben, die durchgeführte Aktion (Kopieren, Verschieben oder Abbruch) zu erfahren und evtl. darauf zu reagieren.

Anschließend werden auf der Quellkomponente die Ereignisse **Control.-GiveFeedback** und **Control.QueryContinueDrag** ausgelöst. Auf der Zielkomponente werden die Ereignisse **Control.DragEnter**, **Control.DragOver** und **Control.DragLeave** ausgelöst.

Die Beschreibung des .NET-Mechanismus für Drag&Drop entnehmen Sie bitte der Beschreibung des .NET-Frameworks. Zusätzlich gibt es fünf Eigenschaften, die das Verhalten des Drag&Drop beeinflussen:

> Control.AllowDrop

Mit dieser Boole'schen Eigenschaft der Basisklasse **Control** bestimmen Sie, ob Objekte, die über das Steuerelement gezogen wurden, fallen gelassen werden dürfen. Vorgänge, die innerhalb eines VARCHART-Steuerelements verschoben oder kopiert werden, sind hiervon ausgenommen (d.h. sie dürfen immer fallen gelassen werden).

> VcTree.LeavingControlWhileDraggingAllowed

Mit dieser Boole'schen Eigenschaft des VcTree-Objektes können Sie bestimmen, ob ein Ziehen eines oder mehrerer Vorgänge über die Grenzen der Quellkomponente hinaus erlaubt sein soll. Damit können Sie z.B. einen Knoten von einem zu einem andern VARCHART-Steuerelement oder in andere Steuerelemente der gleichen Anwendung oder sogar in andere Anwendungen verschieben oder kopieren.

> VcTree.NodeCreationAtDroppingEnabled

Mit dieser Boole'schen Eigenschaft des VcTree-Objektes können Sie festlegen, ob beim Fallenlassen eines gezogenen Objekts in der Zielkomponente automatisch ein Vorgang im VARCHART-Steuerelement erzeugt werden soll.

> VcTree.PhantomDrawingWhileDraggingEnabled

Mit dieser Boole'schen Eigenschaft des VcTree-Objektes können Sie in der Zielkomponente festlegen, ob beim Ziehen das für das VARCHART-Steuerelement übliche Phantom angezeigt werden soll.

> VcTree.InbuiltMouseCursorWhileDraggingEnabled

Mit dieser Boole'schen Eigenschaft können Sie in der Zielkomponente festlegen, ob beim Ziehen die für das VARCHART-Steuerelement charakteristischen, eingebauten Mauszeiger angezeigt werden sollen oder der für das Drag&Drop übliche Mauszeiger (Pfeil mit Rechteck bzw. Verbotszeichen), oder gar anwendungseigene.

3.5 Ereignisse

Über Ereignisse werden die Interaktionen des Anwenders vom VARCHART Steuerelement an die Applikation weitergegeben. Immer wenn der Anwender mit dem VARCHART Steuerelement interagiert, indem er beispielsweise Daten ändert oder auf irgendeine Stelle des Steuerelements klickt, wird ein entsprechendes Ereignis gemeldet. Sie können in Ihrem Programmcode die Ereignisse auffangen und mit Ihrer Anwendung beliebig darauf reagieren.

In jeder Entwicklungsumgebung werden für die Ereignisse entsprechende Funktionen bereitgestellt, in denen die vom VARCHART-Windows-Forms-Steuerelement zur Verfügung gestellten Parameter schon eingetragen sind. Die einzelnen Ereignisse sind in der API-Referenz ausführlich beschrieben. Hier sei nur kurz darauf hingewiesen, dass Sie unter Verwendung des Parameters **returnStatus** in den Ereignissen alle in VARCHART Steuerelement verfügbaren Kontextmenüs abschalten (und ggf. durch eigene ersetzen) und alle Interaktionen kontrollieren und bei Bedarf unterbinden bzw. rückgängig machen können.

> Rückgabewerte

Die folgende Tabelle enthält die Rückgabewerte für VARCHART-Ereignisse:

Konstante	Wert	Beschreibung
vcRetStatDefault	2	standardmäßige Vorbesetzung
vcRetStatFalse	0	Abbrechen der Aktion
vcRetStatNoPopup	4	Kontextmenü erscheint nicht

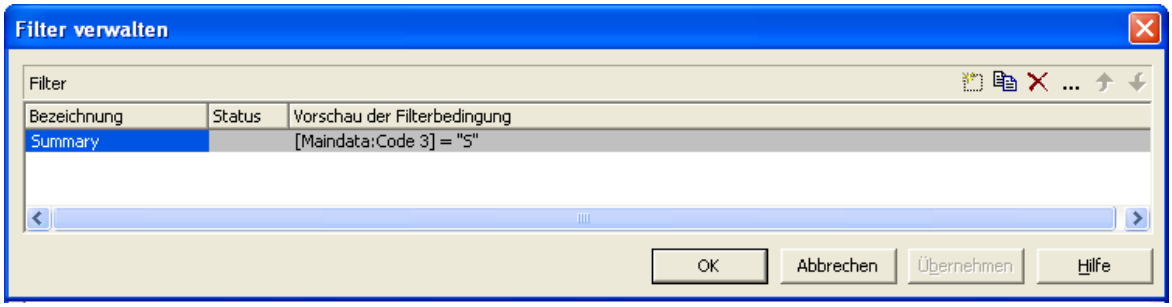
3.6 Filter

Ein Filter enthält Bedingungen für Knoten. Mit Hilfe eines Filters können Sie alle Knoten, die die Filterbedingungen erfüllen, auswählen, um sie beispielsweise grafisch hervorzuheben.

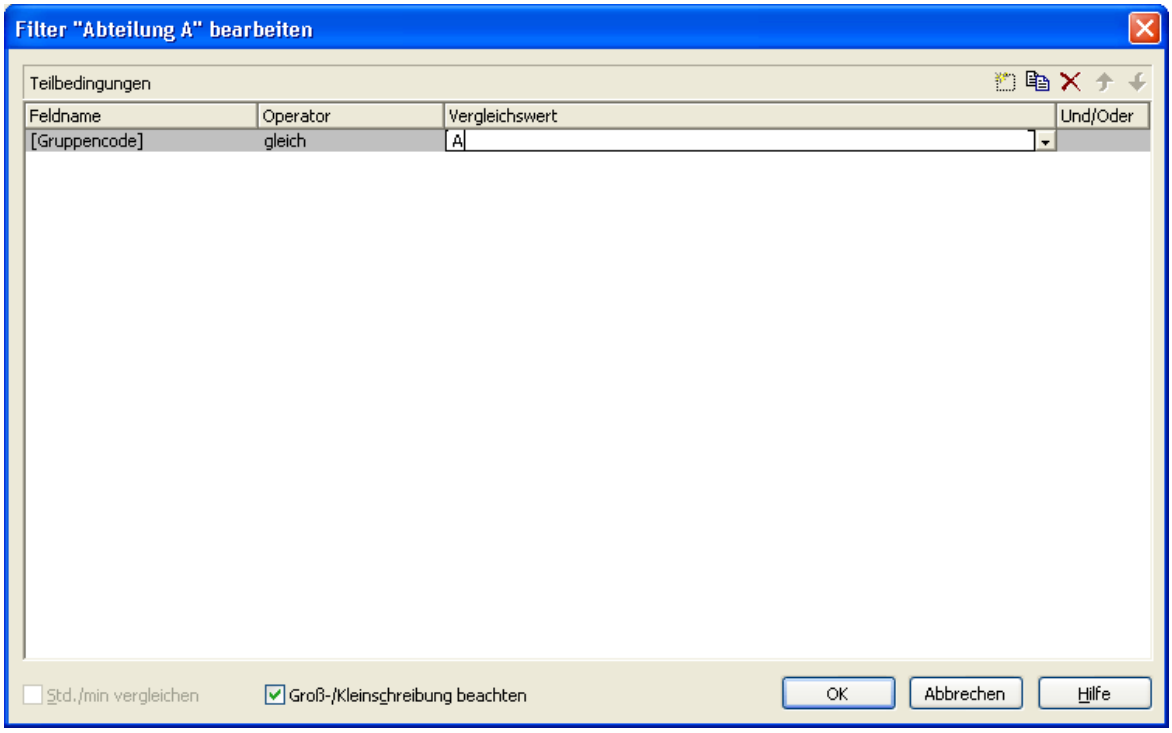
Wenn Sie einen Filter anwenden, werden die Informationen des Datensatzes eines Knotens mit den Kriterien aus dem Filter verglichen. Es werden die Knoten ausgewählt, die die Filterkriterien erfüllen. Beispielsweise können Sie den Filter "Alle Knoten von Abteilung A" definieren.

Filter können nur im Design-Modus erzeugt, bearbeitet und verwaltet werden.

Sie erreichen das Dialogfeld **Filter verwalten** über die Eigenschaftenseite **Objekte**. Im Dialogfeld **Filter verwalten** können Sie Filter umbenennen, kopieren, löschen, bearbeiten oder neu definieren.



Einen vorhandenen Filter bearbeiten können Sie im Dialogfeld **Filter bearbeiten**, das Sie vom Dialogfeld **Filter verwalten** erreichen.



3.7 Grafikformate

VARCHART unterstützt die folgenden Grafikformate, was für den Export von Grafiken für Bedeutung ist, vor allem für die Aufrufe **VcTree1.ShowGraphicsExportDialog** und **VcTree1.ExportGraphics**.

Das XTree-Steuerelement unterstützt sowohl den Import von Grafikdateien z.B. für die Darstellung in Knoten oder in Boxen wie auch den Export ganzer Charts in Grafikdateien. Dabei spielen die verschiedenen unterstützten Grafikformate eine nicht ganz unwichtige Rolle in Bezug auf die Qualität der Darstellung der Grafik im Steuerelement (nach dem Import) bzw. in einem externen Anzeigeprogramm (nach dem Export). Im folgenden werden die einzelnen Grafikformate mit ihren Vorzügen und Beschränkungen kurz vorgestellt. Dabei sind grundsätzlich zwei Typen zu unterscheiden:

Vektorgrafikformate speichern einzelne geometrische Figuren wie Linien, Ellipsen oder Rechtecke als Beschreibung der Figur mit darauf bezogenen Parametern wie Startkoordinaten, Ausdehnung und Farbe. Sie sind damit auflösungsunabhängig, d.h. bei stärkerem Hineinzoomen werden Linien weiterhin sauber dargestellt. Eine Einschränkung gibt es hier höchstens bei der Größe des zur Verfügung stehenden Koordinatenraums insbesondere beim WMF-Format. Allgemein haben Vektorgrafikformate also einen großen Vorteil durch die Auflösungsunabhängigkeit und oft auch bei der sich ergebenden Dateigröße. Leider hat sich kein plattformunabhängiges, genormtes Format durchgesetzt.

Bitmapgrafikformate speichern alle Bildpunkte mit ihrer Farbe in einer vorgegebenen Ausdehnung. Beim stärkeren Hineinzoomen werden die Grafiken dann automatisch "pixelig". Um einer ausufernden Dateigröße entgegenzuwirken, werden Bitmapgrafiken bei vielen Formaten verlustfrei oder gar verlustbehaftet komprimiert. Ein Verlust ist aber nur bei Fotos und nicht bei Diagrammen hinnehmbar. Ein Vorteil haben Bitmapgrafikformate nur dadurch, dass sie sich über Digitalkameras und das Internet durchgesetzt und plattformunabhängig weitverbreitet sind.

> WMF (Windows Metafile Format)

Dieses Vektorgrafik-Format existiert seit Windows 3.0 und besteht intern aus Befehlsdatensätzen, die den GDI-Befehlen der Windows-API entsprechen. Hiermit können GDI-Befehle sozusagen persistent gemacht werden. Dieses Format war aber schon zu Zeiten seiner Entwicklung nicht vollständig: So besaß und besitzt es bis heute nur einen eingeschränkten Koordinatenraum. Außerdem fehlt das Clipping, die Koordinatentransformation und das Füllen komplexer Polygone. Das Problem, nicht die tatsächliche Ausdehnung einer

WMF-Datei in "echten" Koordinaten wie cm oder Zoll festlegen zu können, wurde schon früh durch die Firma Aldus angegangen, die den sogenannten "Aldus Placeable Header" entwickelte, der seit langem in praktisch allen Programmen zur Anzeige von WMF-Dateien erkannt und genutzt wird mit Ausnahme der Windows-API selbst, die bis heute diesen Header nicht erzeugen oder verarbeiten kann, obwohl er in der Microsoft-Dokumentation erwähnt und erklärt wird.

Mit Windows NT und 95 wurde das Format von Microsoft eigentlich "in Rente" geschickt und sein Nachfolger namens EMF eingeführt. Trotzdem erfreut sich WMF bis heute einiger Beliebtheit vornehmlich im Bereich von ClipArt-Grafiken, wo die erweiterten Möglichkeiten des Nachfolgeformats nicht die Rolle spielen. Neuere Entwicklungen in Windows 95 und NT und deren Nachfolgern flossen nicht mehr in das Format ein; es ist seitdem unverändert geblieben.

WMF kennt auch einen Kommentardatensatz, der dazu genutzt werden kann, dort EMF-Befehle unterzubringen. Wenn ein Anzeigeprogramm solche Kommentare entdeckt, also auch EMF-Dateien anzeigen kann, dann verwirft es automatisch die WMF-Befehlsdatensätze und zeigt die EMF-Befehlsdatensätze. So kann eine einzige Datei WMF- wie auch EMF-Grafik enthalten. Dies ist wohl aus Kompatibilitätsgründen eingebaut worden, bläht aber die Dateigröße stark auf.

Formatbeschreibung siehe:

http://msdn.microsoft.com/en-us/library/cc215212.aspx

Beschränkungen des Formats siehe:

http://support.microsoft.com/kb/81497/en-us.

> EMF (Enhanced Metafile Format)

Dieses Vektorgrafik-Format wurde mit den 32bit-Betriebssystemen Windows NT und 95 eingeführt und behebt die Einschränkungen des WMF-Formats und besteht intern aus Grafikbefehlen, die den GDI32-Befehlen Windows-API entsprechen. Der Koordinatenraum ist nun also 32bittig, Transformationen und Clipping werden unterstützt. Die später ins GDI32 hinzu genommenen Befehle zum maskierten oder mit AlphaBlending versehenem Blitten von Speicher-Bitmaps werden aber nicht unterstützt.

Das Format ist bis heute trotz der Vorteile, die es gegenüber WMF bietet, recht unbekannt geblieben. Alle Anzeigeprogramme und Office-Pakete können allerdings mit EMF-Dateien umgehen.

Ein Nachteil bei der Verwendung von GDI+ ergibt sich dergestalt, dass die von GDI+ neu eingeführten Möglichkeiten der grafischen Gestaltung wie

Farbverläufen und Transparenzen nicht voll unterstützt werden und außerdem beim Export in eine EMF-Datei unterbrochene Linien (gestrichelt u.ä.) in einzelne kleine nicht unterbrochene Linien gespeichert werden, wodurch zum einen der Speicherbedarf stark ansteigt und zum anderen eine Anzeige einer so geschriebenen Datei sehr lange Zeit benötigt.

EMF kennt auch einen Kommentardatensatz, der dazu genutzt werden kann, dort EMF+-Befehle unterzubringen. Wenn ein Anzeigeprogramm solche Kommentare entdeckt, also auch EMF+-Dateien anzeigen kann, dann verwirft es automatisch die EMF-Befehlsdatensätze und zeigt die EMF+-Befehlsdatensätze. So kann eine einzige Datei EMF- wie auch EMF+-Grafik enthalten. Dies ist wohl aus Kompatibilitätsgründen eingebaut worden, bläht aber die Dateigröße stark auf.

Druckjobs werden Windows-intern übrigens auch als EMF-Datenstrom ggf. zwischengespeichert und an den Druckertreiber übergeben.

Formatbeschreibung siehe:

http://msdn.microsoft.com/ en-us/library/cc204166.aspx

> EMF+ (Enhanced Metafile Format Plus)

Obwohl der Name nahelegt, dass es sich um eine Erweiterung des EMF handelt, ist dies ein eigenes Vektorgrafikformat, das mit der Vorstellung der GDI+-Windows-API eingeführt wurde und intern aus Grafikbefehlsdatensätzen besteht, die den GDI+-Befehlen entspricht (GDI+ ist übrigens auch keine Erweiterung von der GDI-API, sondern eine eigene Grafikbibliothek). Zusätzlich zu EMF werden hier Transparenzen und Farbverläufe voll unterstützt.

Das Format ist bis heute recht unbekannt geblieben und wird auch von üblichen Anzeigeprogrammen oft nicht unterstützt, außer in Microsoft Office ab 2003. Microsoft hat erst 2003 den Aufbau des EMF+-Dateiformats veröffentlicht.

Formatbeschreibung siehe:

http://msdn.microsoft.com/ en-us/library/cc204376.aspx

> GIF (Graphics Interchange Format)

Dieses Bitmap-Format wurde von CompuServe in Zeiten vor dem Entstehen des World Wide Web zur verlustfreien, komprimierten Speicherung von Grafiken entwickelt und kann nur gleichzeitig 256 Farben darstellen. Daher kann es die heutigen Grafiken nicht zufriedenstellend speichern. Es wird nur noch aus Kompatibilitätsgründen angeboten.

Die Unterart "Animated GIF" wird überhaupt nicht unterstützt.

> JPEG (Joint Photographic Experts Group)

Dieses Bitmap-Format wurde von der JPEG zur verlustbehafteten, komprimierten Speicherung von Fotos entwickelt. Daher ist eine Speicherung von Diagrammen, bei denen es z.B. auf die saubere Speicherung von Linien ankommt, nicht sinnvoll. Es wird nur noch aus Kompatibilitätsgründen angeboten.

> BMP (Windows Bitmap)

Dieses Bitmap-Format wurde von Microsoft zur verlustfreien, unkomprimierten Speicherung von Grafiken entwickelt. Das Format wird auch intern im Speicher von der Windows-API GDI direkt verwendet. Eine einzige Einschränkung gibt es dadurch, dass es keinen Alphakanal unterstützt, d.h. es können maximal 24 Bits pro Pixel gespeichert werden. Aufgrund seines hohen Speicherbedarfs sollte auf dieses Format verzichtet werden. Es wird nur noch aus Kompatibilitätsgründen angeboten.

> TIFF (Tagged Image File Format)

Dieses Bitmap-Format wurde von Aldus (in Adobe aufgegangen) zur Speicherung von Grafiken entwickelt. Die Grafik kann darin verlustbehaftet oder verlustfrei gespeichert werden. Das Format wird seit längerem nicht mehr weiterentwickelt. Es wird nur noch aus Kompatibilitätsgründen angeboten.

> PNG (Portable Network Graphics)

Dieses Bitmap-Format wurde vom World Wide Web Consortium (W3C) zur verlustfreien, komprimierten Speicherung von Grafiken entwickelt, um das vormals Copyright-behaftete und beschränkte GIF Format abzulösen. PNG ist hervorragend geeignet zur Speicherung von VARCHART-Grafiken und beim Einlesen werden transparente Anteile auch transparent gezeichnet. Es wird auch durchgängig von praktisch allen Anzeigeprogrammen und Internetbrowsern unterstützt. Das Format selbst ist frei von Copyrights und vollständig dokumentiert.

Seit der Version 4.2 wird für den Export die frei verfügbare Bibliothek **libpng** verwendet, um durch die Vorgabe der Auflösung beliebig große Bitmaps speichern zu können. Hierbei muss aber darauf geachtet werden, dass sehr große PNG-Dateien zu Schwierigkeiten beim Einlesen führen können, denn üblicherweise wird die PNG-Datei im Speicher erst komplett entpackt und dann dargestellt.

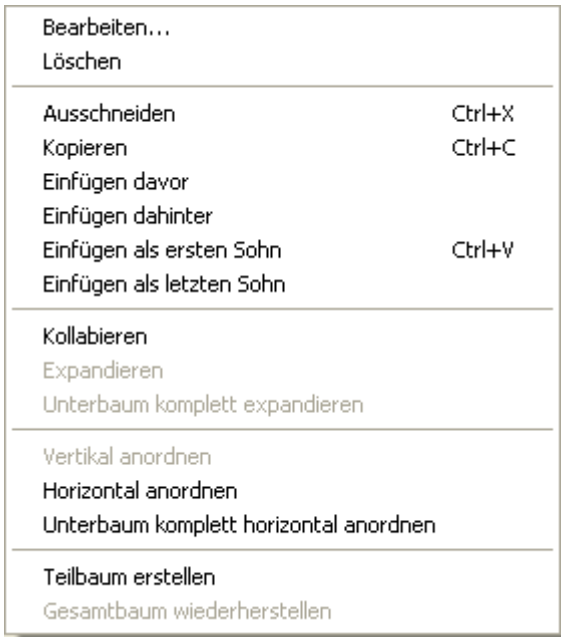
Formatbeschreibung siehe:
http://www.libpng.org/pub/png/spec/1.1/PNG-Contents.html

3.8 Horizontale/vertikale Anordnung

Wenn Sie Ihr Baum-Diagramm auf den Bildschirm holen, kann es sein, dass die Darstellung beim ersten Mal nicht optimal proportioniert wird. Durch eine geeignete Kombination horizontal und vertikal angeordneter Teilstrukturen lassen sich jedoch äußerst kompakte Baum-Diagramme erstellen.

- *Horizontale Anordnung:* Durch eine horizontale Anordnung lässt sich die Höhe eines Baum-Diagramms verringern. Alle Knoten einer Ebene werden dabei nebeneinander angeordnet. Die Ports (Anknüpfungspunkte) der Verbindungslinie liegen dann mittig am unteren Rand des Vaterknotens bzw. mittig am oberen Rand des Sohnknotens.
- *Vertikale Anordnung:* Durch eine vertikale Anordnung lässt sich die Breite des Baum-Diagramms verringern. Alle Knoten einer Ebene und deren Unterebenen werden dabei untereinander angeordnet. Die Ports der Verbindungslinie liegen dann in der unteren linken Ecke des Vaterknotens bzw. in der Mitte des linken Randes des Sohnknotens.

Um die Befehle zum horizontalen und vertikalen Anordnen von Bäumen zur Verfügung zu haben, markieren Sie einen Knoten und klicken Sie auf die rechte Maustaste. Es erscheint das folgende Kontextmenü, wobei jeweils nur die verfügbaren Befehle aktiviert sind.



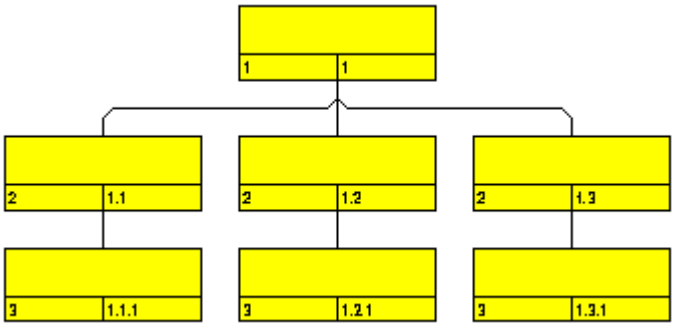
Um eine oder mehrere Teilstrukturen eines Baum-Diagramms horizontal anordnen zu lassen, markieren Sie den/die obersten Knoten jeder Teilstruktur und wählen Sie anschließend im Kontextmenü den Befehl **Horizontal anordnen**. Die Teilstrukturen der markierten Knoten werden dann horizontal

angeordnet, aber nur eine Ebene tief. Auf die Anordnungen in den nächst tieferen Ebenen hat der Befehl **Horizontal anordnen** keine Auswirkung.

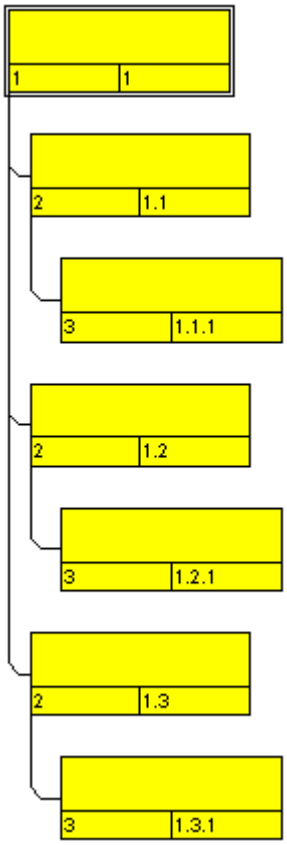
Mit dem Befehl **Unterbaum komplett horizontal anordnen** werden die Teilbäume unter den markierten Knoten vollständig, d. h. über alle Ebenen, horizontal angeordnet.

Mit dem Befehl **Vertikal anordnen** werden die Teilbäume unter den markierten Knoten vertikal angeordnet, und zwar beginnend beim höchsten ausgewählten Wurzelknoten.

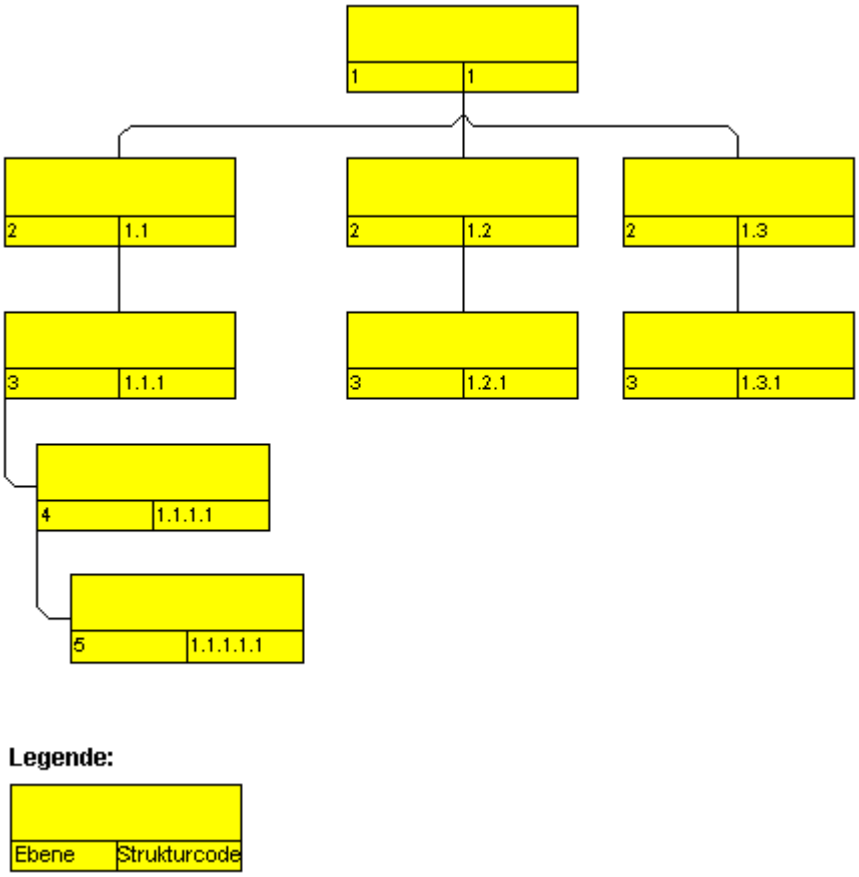
Hinweis: Falls ein Baum nicht vollständig vertikal angeordnet wird, prüfen Sie die vereinbarte maximale Baumhöhe auf der Eigenschaftenseite **Layout**. Die Anzahl der Ebenen bei der vertikalen Anordnung wird durch die Angabe unter **Max. Baumhöhe** begrenzt.



alle Ebenen horizontal angeordnet



alle Ebenen vertikal angeordnet



Beispiel für ein Baum-Diagramm mit horizontal und vertikal angeordneten Teilstrukturen

Hinweis: Änderungen in der Anordnung wirken sich auch auf kollabierte Teilbäume aus.

> Unterbaum-Anordnung in Datenfeld speichern

Aktivieren Sie auf der Eigenschaftenseite **Knoten** das Kontrollkästchen **Unterbaum-Anordnung in Feld**, um die Orientierung eines Teilbaums synchron in einem Datenfeld zu halten. Mögliche Inhalte des Datenfelds sind "0" (Teilbaum horizontal angeordnet) und "1" (Teilbaum vertikal angeordnet). Die horizontale Anordnung ist nur sichtbar, wenn der direkte oder indirekte Vaterknoten selbst eine vertikale Anordnung besitzt.

Alternativ können Sie die VcTree-Eigenschaft **ArrangementField** verwenden, um die Orientierung eines Teilbaums synchron in einem Datenfeld zu halten.

> Vertikal ab Ebene

Auf der Eigenschaftenseite **Layout** können Sie die Ebene angeben, ab der die Knoten vertikal angeordnet werden sollen. Aktivieren Sie dazu das Kontrollkästchen **Vertikal ab Ebene** und geben Sie an, ab welcher Ebene die

Knoten vertikal angeordnet werden sollen. Damit die Knoten tatsächlich vertikal angeordnet werden, muss die API-Methode **Arrange** aufgerufen werden.

Alternativ können Sie über die VcTree-Eigenschaft **FirstVerticalLevel** erfragen oder festlegen, ob ab einer bestimmten Ebene die Knoten vertikal angeordnet sind. Beim Wert "-1" ist diese Eigenschaft abgeschaltet.

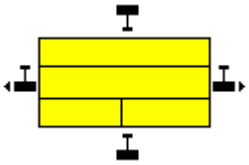
3.9 Knoten

Ein Knoten entspricht einem Datensatz aus der Maindata-Tabelle. Knoten können über die API geladen oder interaktiv vom Anwender erzeugt werden.

> Knoten erzeugen

Wenn auf der Eigenschaftenseite **Allgemeines** die Option **Erzeugung neuer Knoten zulassen** gewählt ist, kann der Anwender im Erzeugemodus neue Knoten per Mausklick erzeugen.

Nachdem Sie den ersten Knoten erzeugt haben, können Sie weitere Knoten nur durch Anlagern an einen bereits vorhandenen Knoten erzeugen. Wenn Sie sich im Erzeugemodus mit der Maus dem Knotens nähern, verändert der Mauszeiger seine Form und zeigt an, wo der neue Knoten angelegt würde (als Vater, Sohn oder Bruder des Bezugsknotens).



Ist auf der Eigenschaftenseite **Allgemeines** das Kontrollkästchen **Erzeugung neuer Knoten mit Dialog** aktiviert, öffnet sich das Dialogfeld **Vorgänge bearbeiten**, sobald ein Knoten durch Mausklick erzeugt wurde. Hier werden die Daten des interaktiv erzeugten neuen Knotens angezeigt, und Sie können sie nun bearbeiten.

Sie können Knoten auch über die API mit **InsertNodeRecord** anlegen.

Jedes interaktive Neuanlegen eines Knotens wird der Applikation mit dem Ereignis **VcNodeCreating** mitgeteilt.

> Markieren von Knoten

Auf der Eigenschaftenseite **Knoten** können Sie unter **Markierungstyp** festlegen, wie und in welcher Farbe markierte Knoten dargestellt werden sollen.

Mögliche Alternativen sind:

- Ohne
- Einrahmen
- Einrahmen innen
- Invertieren
- Pickmarks
- Pickmarks innen

Hinweis: Wenn Sie den Markierungstyp "Ohne" ausgewählt haben, können Knoten nicht markiert werden.

Jedes Markieren/Demarkieren von Knoten wird der Applikation mit dem Ereignis **VcNodesMarking** mitgeteilt. Das Ende einer Markier-/ Demarkier-operation wird durch das Ereignis **VcNodesMarked** mitgeteilt.

> Knoten löschen

Einen oder mehrere Knoten können Sie löschen, indem Sie sie bei gedrückter Umschalt- oder Strg-Taste mit der linken Maustaste markieren und dann die rechte Maustaste drücken. Wählen Sie dann im Kontextmenü für Knoten den Befehl **Löschen** bzw. **Ausschneiden**.

Markierte Knoten können auch mit der **Entf**-Taste gelöscht werden.

Das interaktive Löschen eines Knotens löst das Ereignis **VcNodeDeleting** aus.

Außerdem können Sie Knoten über die API mit der VARCHART-ActiveX-Methode **DeleteNodeRecord** oder mit der VcNode-Methode **DeleteNode** löschen.

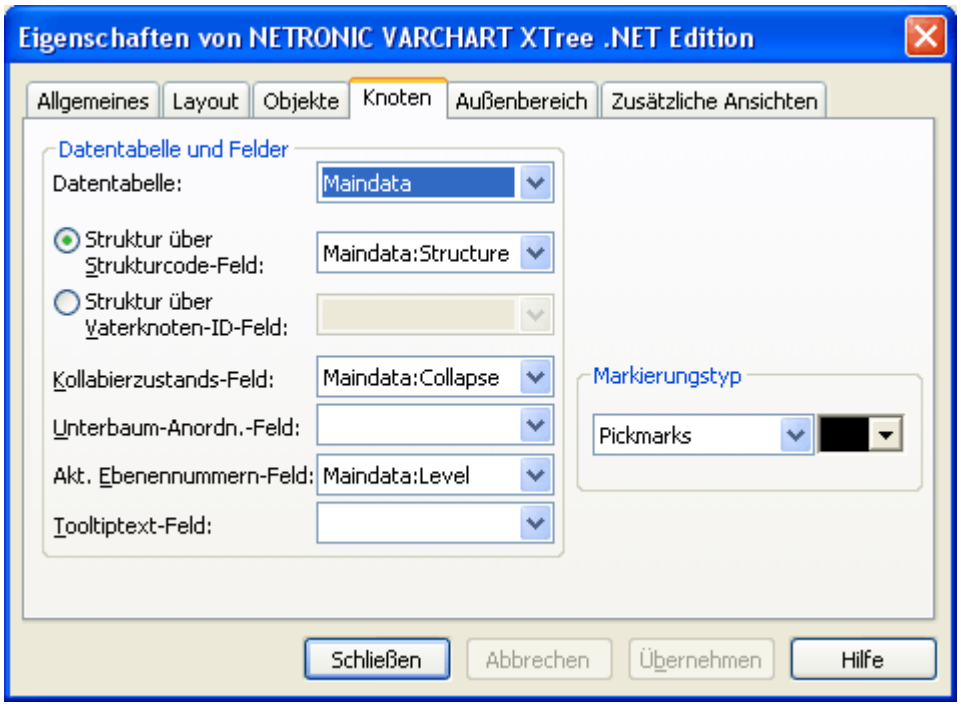
> Ereignisse

Auf folgende Ereignisse können Sie reagieren:

- **VcNodeCreating**
- **VcNodeCreated**
- **VcNodeDeleting**
- **VcNodeLeftClicking**
- **VcNodeLeftDoubleClicking**
- **VcNodeModified**
- **VcNodeModifiedEx**
- **VcNodeRightClicking**
- **VcNodesMarked**
- **VcNodesMarking**

> Datenfelder für Baumstrukturen festlegen

Auf der Eigenschaftenseite **Knoten** können Sie Datenfelder für die Baumstruktur festlegen.



Entscheiden Sie zuerst, ob die Baumstruktur über einen Strukturcode oder über die ID des Vaterknotens festgelegt werden soll (**Strukturcode über Strukturcode-Feld** oder **Struktur über Vaterknoten-ID-Feld**). Legen Sie dann die entsprechenden Datenfelder fest:

- **Struktur über Strukturcode-Feld**: Wenn diese Checkbox aktiviert ist, können Sie das Datenfeld auswählen, das den Strukturcode festlegen soll.
- **Struktur über Vaterknoten-ID- Feld**: Wenn diese Checkbox aktiviert ist, können Sie das Datenfeld auswählen, das für die ID des Vaterknotens verwendet werden soll.

Legen Sie weiterhin folgende Datenfelder fest:

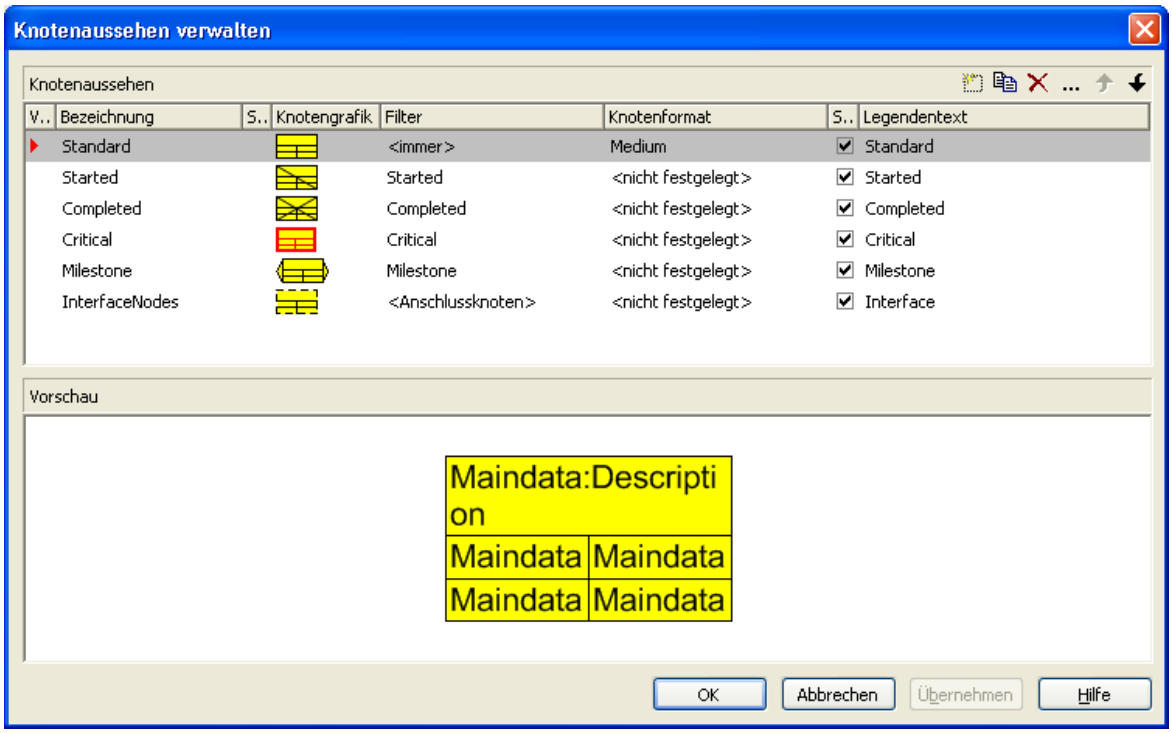
- **Kollabierzustands-Feld**: Aktivieren Sie dieses Kontrollkästchen, um den Kollabierstatus eines Knotens synchron in dem hier gewählten Datenfeld zu halten. Mögliche Inhalte des Datenfelds sind "0" (Knoten expandiert) und "1" (Knoten kollabiert).
- **Unterbaum-Anordn.-Feld**: Aktivieren Sie dieses Kontrollkästchen, um die Orientierung eines Teilbaums synchron in einem Datenfeld zu halten. Mögliche Inhalte des Datenfelds sind "0" (Teilbaum horizontal angeordnet) und "1" (Teilbaum vertikal angeordnet). Die horizontale Anordnung ist nur sichtbar, wenn der direkte oder indirekte Vaterknoten selbst eine vertikale Anordnung besitzt.
- **Akt. Ebenennummern-Feld:** Aktivieren Sie dieses Kontrollkästchen, um festzulegen, in welchem Datenfeld die Ebenennummer der Knoten abgelegt wird. Die Ebenennummern zählen von 0 an aufwärts. Zur

Laufzeit können Sie nicht die Ebene des Knotens verändern, indem Sie den Wert des Ebenennummer-Datenfeldes ändern.

3.10 Knotenaussehen

Das Aussehen von Knoten lässt sich in Abhängigkeit von deren Daten festlegen. Beispielsweise können alle Knoten der Abteilung A durch einen roten Hintergrund und eine doppelte schwarze Rahmenlinie gekennzeichnet werden, Knoten der Abteilung B durch einen blauen Hintergrund und eine einfache gelbe Rahmenlinie etc. Diese grafischen Attribute werden als Knotenaussehen bezeichnet.

Sie können Knotenaussehen festlegen, indem Sie auf der Eigenschaftenseite **Objekte** auf die **Knotenaussehen**-Schaltfläche klicken, um das Dialogfeld **Knotenaussehen verwalten** zu öffnen. Hier können Sie Knotenaussehen kopieren, bearbeiten, löschen und neu definieren oder deren Abarbeitungsreihenfolge verändern.



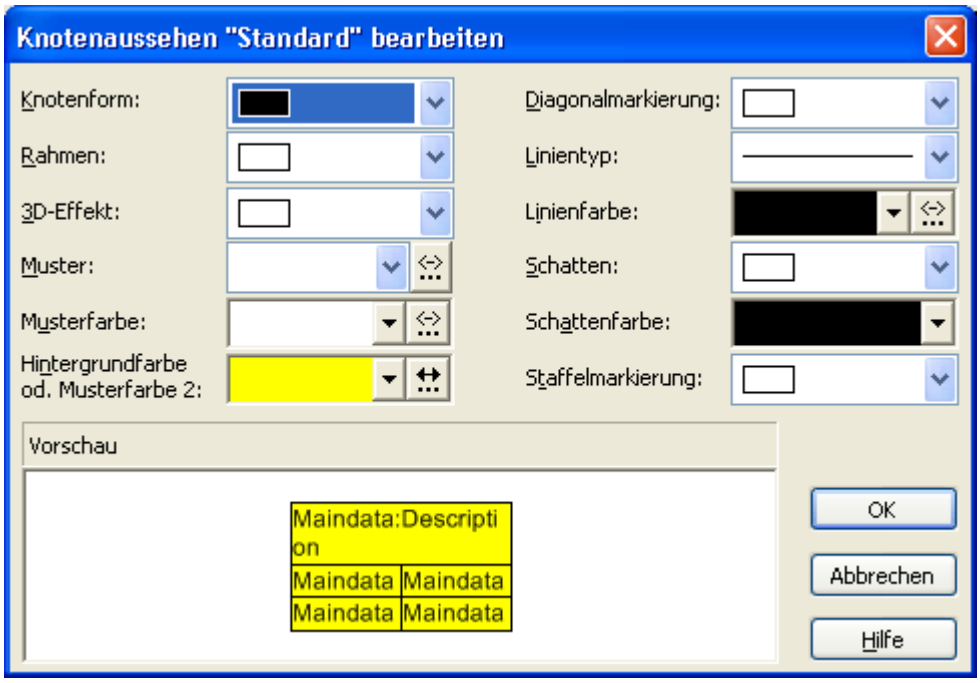
Jedes Knotenaussehen ist mit einem Filter und einem Knotenformat verbunden. Der Filter gibt die Bedingungen an, unter denen dieses Knotenaussehen auf einen Knoten angewandt wird. Beispielsweise ist das Knotenaussehen "Markiert" mit dem Filter "Markiert" verbunden, der alle markierten Knoten selektiert.

Erfüllt ein Knoten die Filterkriterien mehrerer Knotenaussehen, werden diese Knotenaussehen für den Knoten grafisch überlagert. Begonnen wird dabei jeweils mit dem Knotenaussehen, das in der Tabelle ganz oben steht. Das Knotenaussehen, das ganz unten steht, wird als letztes zugewiesen und überlagert daher alle anderen. Die niedrigste Position hat i. d. R. das

Knotenaussehen "Standard", das normalerweise ganz oben in der Tabelle steht. Es wird auf alle Knoten angewendet und kann nicht gelöscht werden.

Sie können die Abarbeitungsreihenfolge der Knotenaussehen mit Hilfe der Pfeil-Schaltflächen verändern.

Um ein Knotenaussehen zu bearbeiten, klicken Sie auf die **Knotenaussehen bearbeiten**-Schaltfläche oder doppelklicken Sie auf das **Knotengrafik**-Feld. Sie gelangen dann in das folgende Dialogfeld:

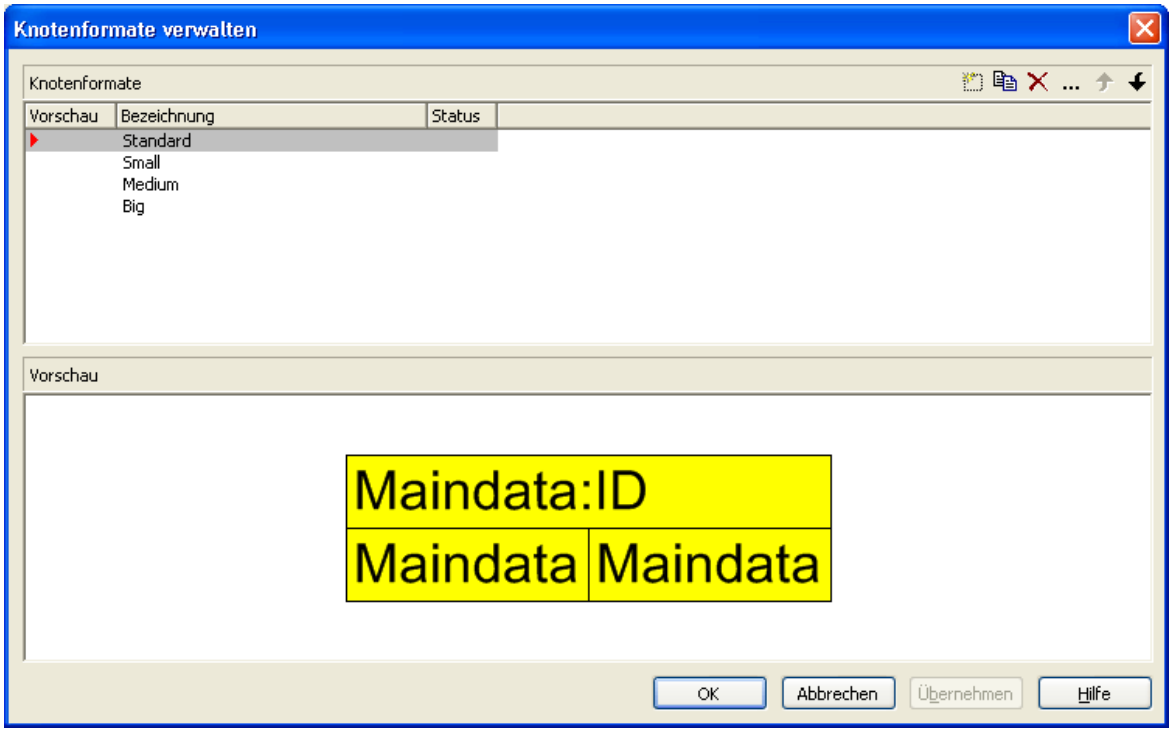


Die Hintergrundfarbe und die Linienfarbe eines Knotenaussehens können Sie mit Hilfe von Zuordnungstabellen in Abhängigkeit von den Knotendaten festlegen. Siehe hierzu das Kapitel "Wichtige Begriffe: Zuordnungstabellen".

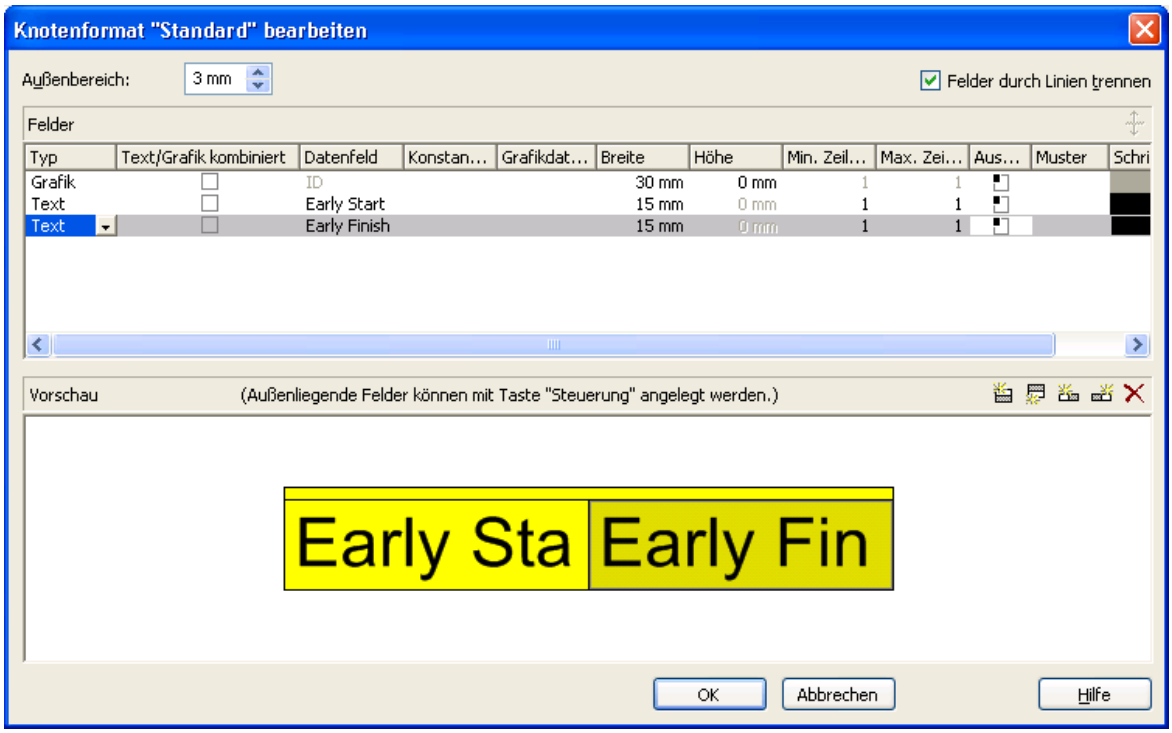
3.11 Knotenformat

Jedes Knotenaussehen ist mit einem Knotenformat verbunden. Sie können das Knotenformat für ein Knotenaussehen festlegen, indem Sie im Dialogfeld **Knotenaussehen verwalten** aus der Kombobox unter **Knotenformat** das gewünschte Knotenformat auswählen.

Knotenformate werden im Dialogfeld **Knotenformate verwalten** verwaltet. Dieses Dialogfeld erreichen Sie, indem Sie auf der Eigenschaftenseite **Objekte** auf die Schaltfläche **Knotenformate** klicken.



Um das aktuelle Knotenformat zu bearbeiten, klicken Sie auf die **Knotenformat bearbeiten**-Schaltfläche. Sie gelangen dann in das Dialogfeld **Knotenformat bearbeiten**, in dem Sie den Aufbau und das Aussehen des jeweiligen Knotens bearbeiten können.



In diesem Dialog können Sie für das gewählte Knotenformat Folgendes festlegen:

- ob die Knotenfelder durch Linien voneinander getrennt werden sollen
- den Außenbereich (Abstand in Millimetern, den Knoten mit diesem Knotenformat zu benachbarten Knoten und zum Rand der Darstellung halten)
- den Feldtyp des aktuellen Knotenfeldes (Text oder Grafik)
- für den Typ Text: das Datenfeld, dessen Inhalt in dem aktuellen Knotenfeldes ausgegeben werden soll, oder einen konstanten Text
- für den Typ Grafik: Name und Pfad der Grafikdatei, die in dem gewählten Knotenfeld dargestellt wird
- die Breite und Höhe des markierten Knotenfeldes
- die maximale Anzahl von Textzeilen im aktuellen Knotenfeld
- die Ausrichtung des Textes bzw. der Grafik des markierten Knotenfeldes
- das Füllmuster und die Musterfarben des Knotenfeldes
- die Schriftart und -farbe des Knotenfeldes

> Datumsformat der Terminfelder von Knoten

Das Datumsformat der Terminfelder von Knoten wird auf der Eigenschaftenseite **Allgemeines** festgelegt.

> Darstellung von Grafiken in Knotenfeldern

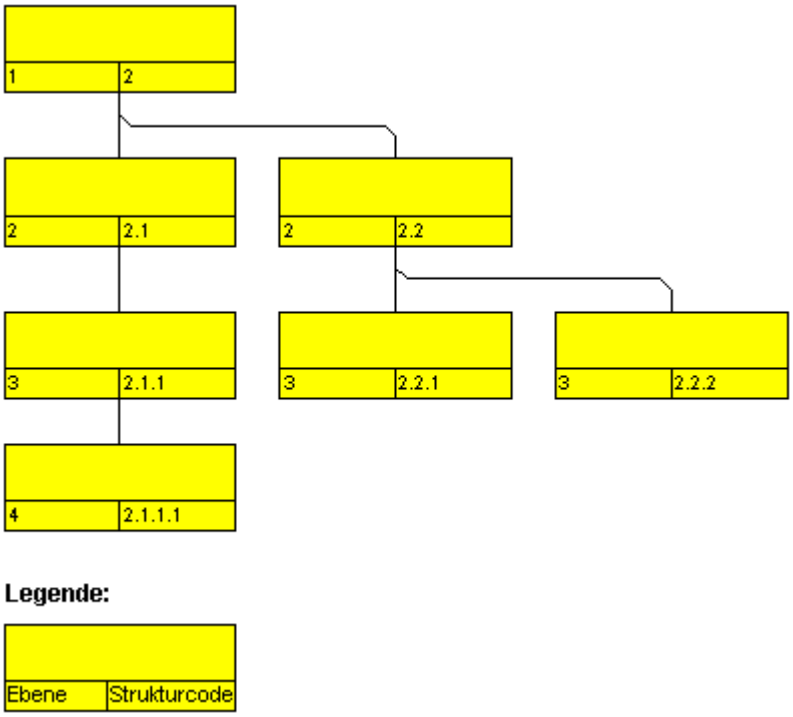
Sie können für ein Knotenfeld des Typs Grafik eine Grafikdatei wählen, indem Sie das Feld **Grafikdateiname** markieren, auf die dann erscheinende Schaltfläche **Grafikdatei auswählen** (...) klicken und anschließend im gleichnamigen Windows-Dialog eine Datei wählen.

Sie können aber auch eine Zuordnung zwischen den Einträgen eines Datenfeldes und Grafikdateien herstellen. Klicken Sie dazu auf die Schaltfläche **Zuordnungen einstellen** (), um den gleichnamigen Dialog zu öffnen. Wenn eine Zuordnung vorgenommen worden ist, wird das durch ein Symbol neben dem Grafikdateinamen dargestellt (). Einzelheiten hierzu finden Sie in den Kapiteln "Eigenschaftenseiten und Dialogfelder" und "Wichtige Begriffe: Zuordnungstabellen".

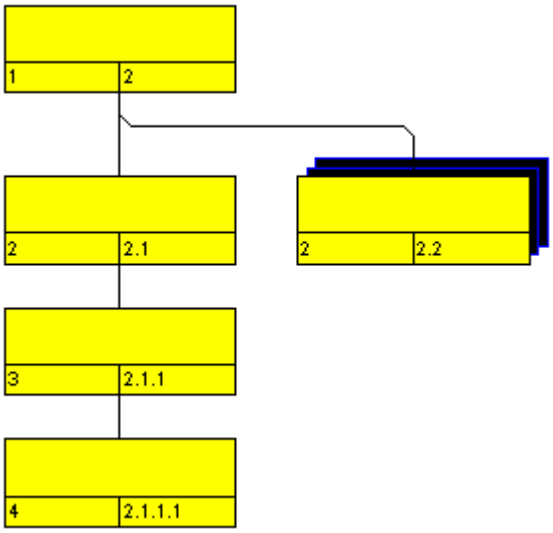
3.12 Kollabieren und Expandieren

Teilstrukturen von Baum-Diagrammen lassen sich kollabieren (wegklappen) und wieder expandieren (aufklappen). Jede beliebige Teilstruktur kann auf ihren obersten Knoten, den Gliederungsknoten, kollabiert werden. Als Gliederungsknoten wird jeweils der Wurzelknoten der Teilstruktur, die kollabiert werden soll, verwendet. Die kollabierte Teilstruktur kann ggf. wieder in ihre ursprüngliche Form expandiert werden.

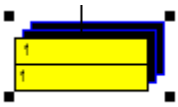
Durch das Kollabieren von Teilstrukturen lassen sich auch sehr komplexe Strukturen übersichtlich gestalten. Wenn Sie beispielsweise nur bestimmte Teilstrukturen präsentieren möchten, kollabieren Sie einfach alle anderen Teilstrukturen. Dadurch, dass die kollabierten Teilstrukturen erhalten bleiben, geht die Information über die Gesamtstruktur nicht verloren.



Expandierter Baum



In Form eines Gliederungsknotens kollabierte Teilstruktur



Vollständig kollabierter Baum

Mit dem Befehl **Kollabieren** aus dem Kontextmenü für Knoten können Sie die Teilstrukturen, die zu den markierten Wurzelknoten gehören, kollabieren. Die markierten Wurzelknoten verwandeln sich damit in Gliederungsknoten, die jeweils die nun verborgenen Teilstrukturen repräsentieren.

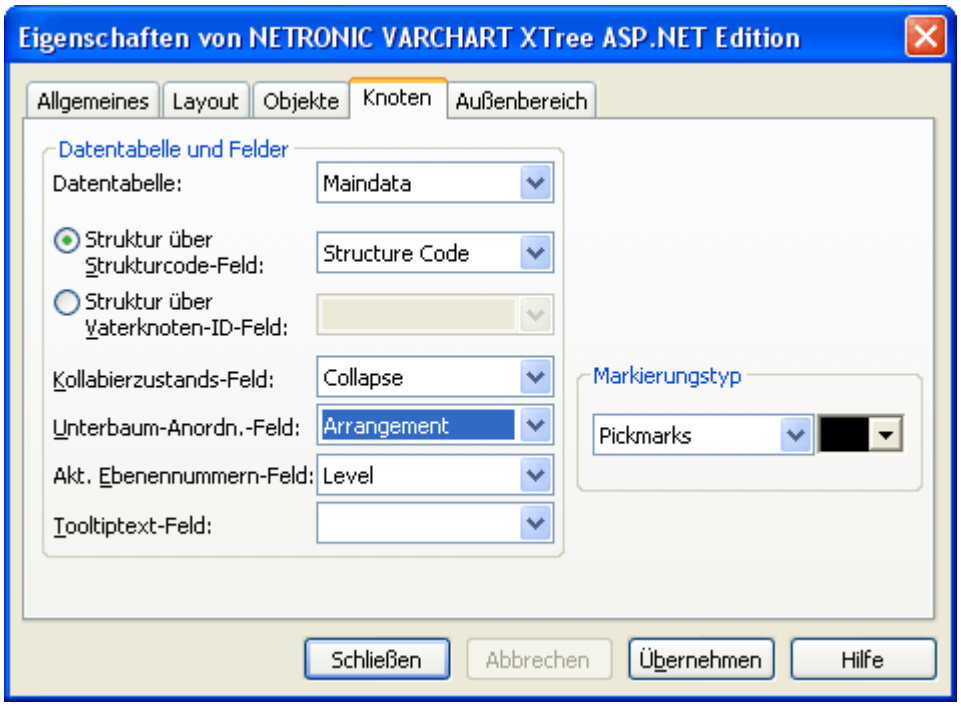
Mit dem Befehl **Expandieren** aus dem Kontextmenü für Knoten können Sie die Teilstrukturen wieder expandieren, die durch die markierten Gliederungsknoten repräsentiert werden. Es werden jeweils nur die kollabierten Knoten expandiert. Befinden sich in den Unterbäumen weitere kollabierte Knoten, bleiben diese kollabiert.

Mit dem Befehl **Unterbaum komplett expandieren** aus dem Kontextmenü für Knoten können Sie die kollabierten Teilstrukturen vollständig expandieren. Das heißt, dass hierbei auch alle kollabierten Knoten in den Unterbäumen expandiert werden.

> Kollabierzustand in Datenfeld speichern

Sie können den Kollabierzustand der Knoten in einem Datenfeld speichern.

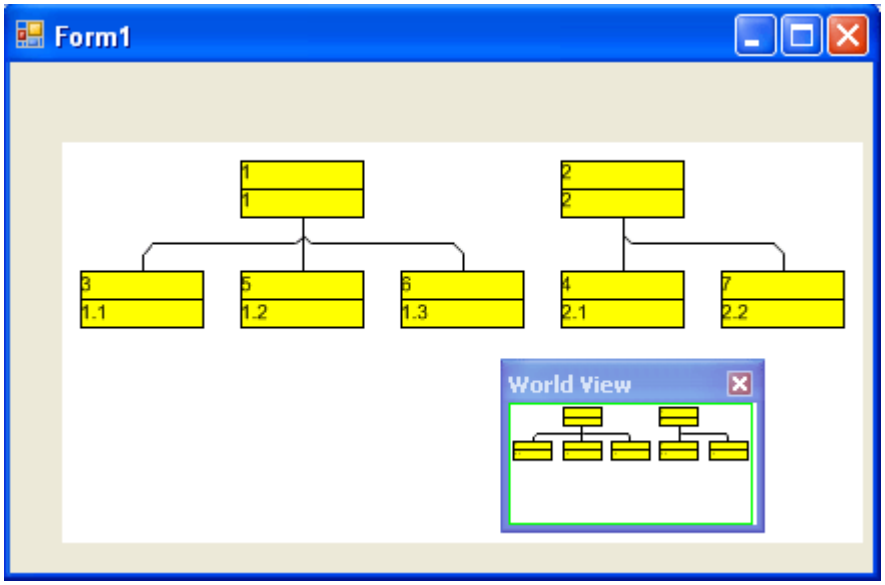
Aktivieren Sie dazu auf der Eigenschaftenseite **Knoten** das Kontrollkästchen **Kollabierzustand in Feld** und wählen Sie aus der Kombobox das Datenfeld "Kollabiert" aus.



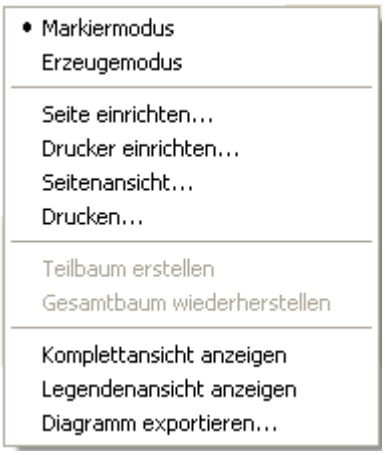
Dann wird der Kollabierstatus jedes Knotens synchron in dem Datenfeld "Kollabiert" gehalten. Mögliche Inhalte des Datenfelds sind "0" (Knoten expandiert) und "1" (Knoten kollabiert).

3.13 Komplettansicht (World View)

Die Komplettansicht ist ein zusätzliches Fenster, in dem das komplette Diagramm angezeigt wird. Ein Rahmen zeigt an, welchen Diagrammausschnitt das Hauptfenster gerade anzeigt. Wenn Sie mit der Maus diesen Rahmen verschieben, wird der angezeigte Ausschnitt des Hauptfensters beim Loslassen der Maustaste entsprechend verschoben. In ähnlicher Weise können Sie durch Größer- oder Kleinerziehen des Rahmens in der Komplettansicht den realen Bildausschnitt zoomen. Umgekehrt ändern sich Position bzw. Größe des Rechtecks, wenn der Ausschnitt im Hauptfenster gescrollt oder gezoomt wird.



Zur Laufzeit können Sie über den Menüpunkt **Komplettansicht anzeigen** des Standard-Kontextmenüs die Komplettansicht ein- bzw. ausschalten.



Auf der Eigenschaftenseite **Zusätzliche Ansichten** können Sie die Eigenschaften der Komplettansicht festlegen. Einzelheiten hierzu finden Sie im

Kapitel "Eigenschaftenseiten und Dialogfelder", Eigenschaftenseite "Zusätzliche Ansichten".

Alternativ können Sie die Eigenschaften der Komplettansicht auch über die API (**VcWorldView**) festlegen.

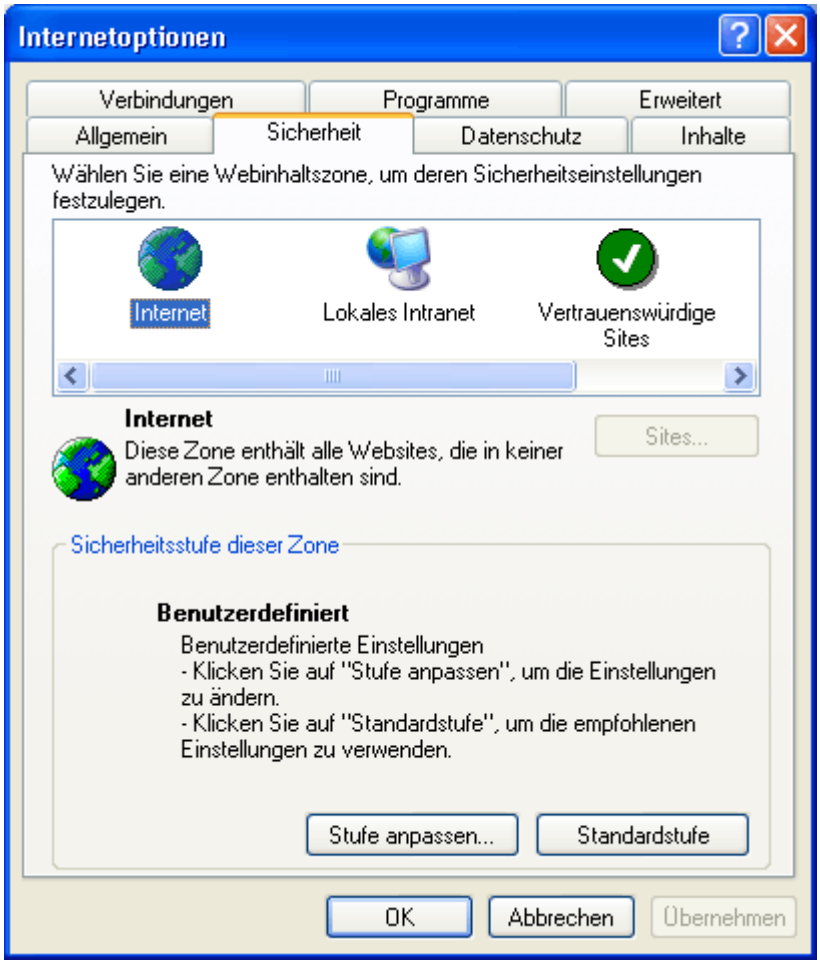
3.14 Laufzeitsicherheitsrichtlinien für den Einsatz im Internet Explorer

Die Laufzeitsicherheitsrichtlinien müssen verändert werden, wenn das VARCHART Tree Steuerelement im Internet Explorer innerhalb einer HTML-Seite verwendet wird.

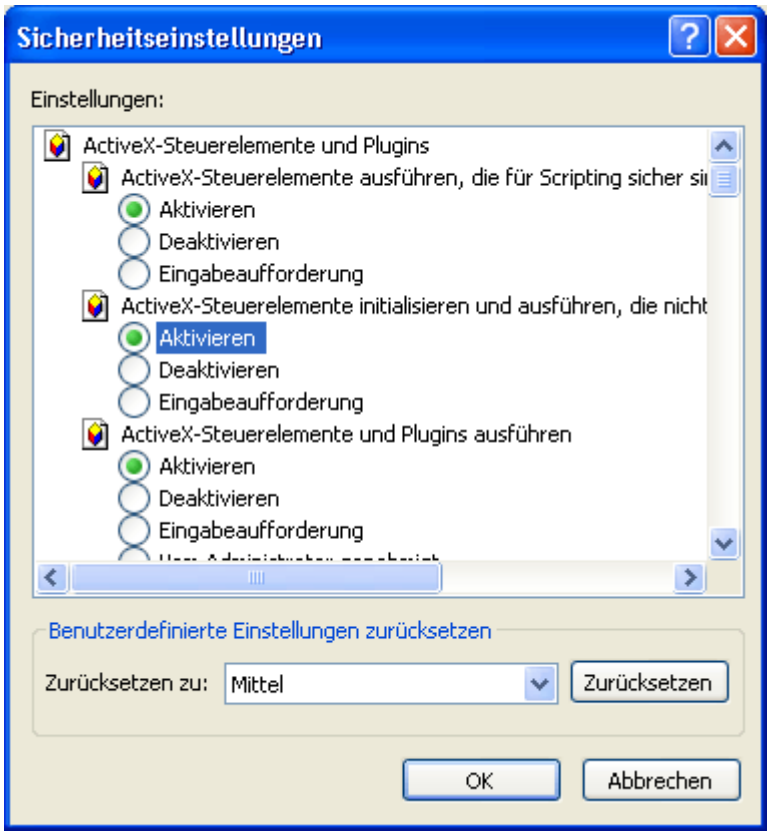
Sobald der Browser das Steuerelement von einem Webserver im Internet lädt, werden die **Laufzeitsicherheitsrichtlinien** der Internet_Zone wirksam. Die Standardeinstellungen verhindern die Ausführung des Steuerelements. Grundsätzlich muss der Internet Explorer das Ausführen von .NET Komponenten zulassen, damit sie überhaupt sichtbar werden können.

Im Dialog **Sicherheitseinstellungen** können die Richtlinien verändert werden. Sie finden den Dialog im Internet Explorer über

Extras > **Internetoptionen** > **Sicherheit** > **Internet** des Internet Explorer. Hier wählen Sie **Internet** oder **Vertrauenswürdige Sites** aus.



Für diese Zone aktivieren Sie bitte unter **Stufe anpassen...** sowohl die Option **ActiveX-Elemente ausführen, die für Scripting sicher sind** als auch **ActiveX-Elemente initialisieren und ausführen, die nicht sicher sind**.

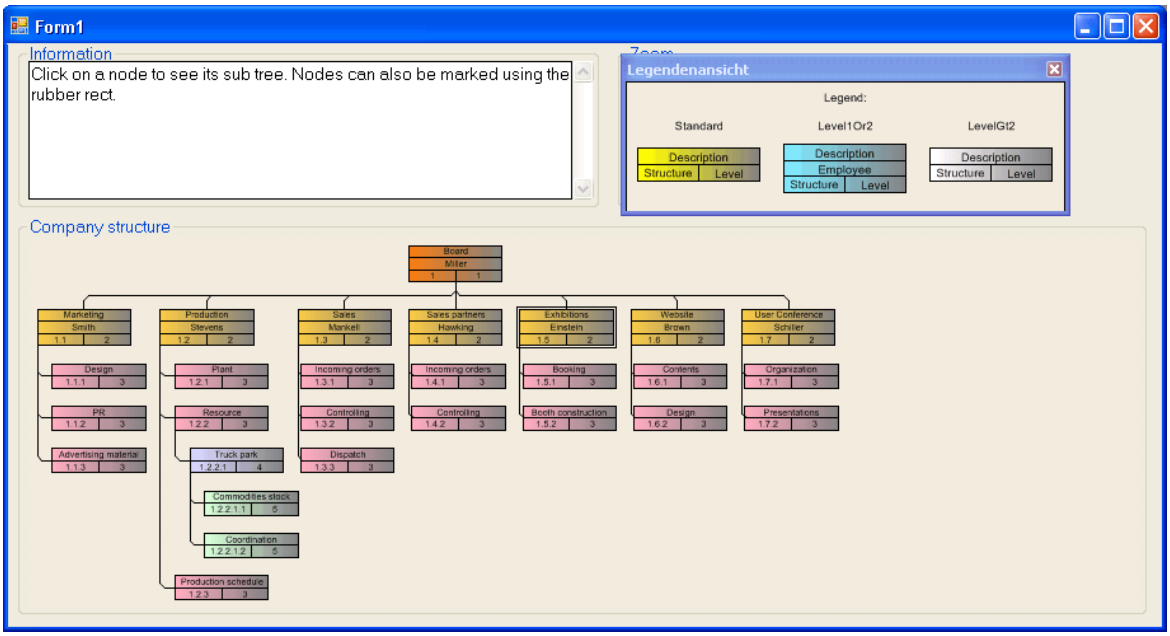


Zusätzlich müssen die Laufzeitsicherheitsrichtlinien auf dem lokalen Rechner eingestellt werden.

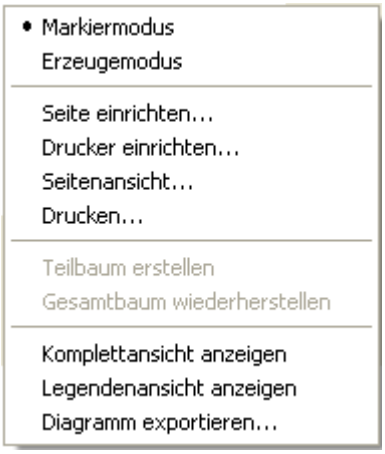
Im Unterverzeichnis **CAS** der VARCHART XGantt-Installation finden Sie zwei passende Batch-Dateien, die Sie von der Kommandozeile aus aufrufen können. Die erste ist **AddRights.bat**, mit der Sie einen Berechtigungssatz und eine Code-Gruppe für NETRONIC-Steuerelemente erzeugen können. Wenn Sie später eine Applikation bei Ihrem Endkunden ausliefern wollen, muss die Batch-Datei vorher auf jedem Client-System einmalig ausgeführt werden. Mit der zweiten Datei **RemoveRights.bat** können Sie Berechtigungen rückgängig machen. Auf diese Weise kann das VARCHART XGantt Steuerelement mit der erforderlichen Mindestmenge an Berechtigungssätzen auf einer HTML-Seite im Internet Explorer ausgeführt werden.

3.15 Legendenansicht (Legend View)

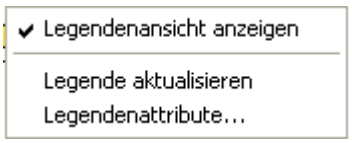
Die Legendenansicht ist ein zusätzliches Fenster zur Darstellung einer Legende auf dem Bildschirm. Das Aussehen der Legende wird festgelegt im Dialog **Legendenattribute**, der über die Eigenschaftenseite **Außenbereich** zu erreichen ist, oder über die Legendenattribute des Objektes **VcBorderBox**.



Zur Laufzeit können Sie über den Menüpunkt **Legendenansicht anzeigen** des Standard-Kontextmenüs die Legendenansicht ein- und ausschalten.



Die Legende verfügt über ein eigenes Kontextmenü, über das die Legendenansicht ebenfalls ein- und ausgeschaltet werden kann.



Außerdem können Sie über das Kontextmenü auch den Dialog **Legendenattribute** aufrufen sowie die Legende aktualisieren.

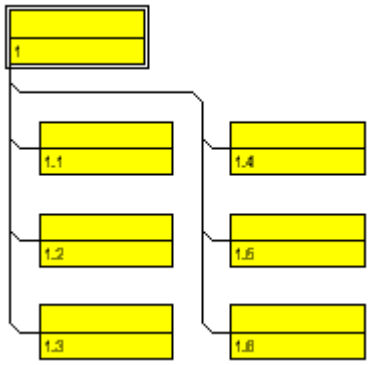
Eine Aktualisierung über das Menü kann notwendig sein, da nach Änderungen im Diagramm die Legende nicht automatisch aktualisiert wird. Werden also zum Beispiel Knoten hinzugefügt oder gelöscht, muss eine Aktualisierung entweder über das Kontextmenü oder durch Aus- und Einschalten der Legende durchgeführt werden. Dies gilt auch für das Laden von Knoten. Wenn für die Legendenansicht auf der Eigenschaftenseite **Zusätzliche Ansichten** die Option **Beim Start sichtbar** eingestellt wurde, aber zum Zeitpunkt des Aufbaus noch keine Knoten geladen waren, bleibt die Legende bis zur Aktualisierung leer.

Auf der Eigenschaftenseite **Zusätzliche Ansichten** können Sie die Eigenschaften der Legendenansicht festlegen. Einzelheiten hierzu finden Sie im Kapitel **Eigenschaftenseiten und Dialogfelder**, **Eigenschaftenseite Zusätzliche Ansichten**.

Alternativ können Sie die Optionen der Legendenansicht auch über die API mit der Eigenschaft **VcTree.VcLegendView** festlegen.

3.16 Maximale Höhe des Baum-Diagramms

Die Gesamthöhe eines Baum-Diagramms (in Zeilen) kann begrenzt werden. Aktivieren Sie dazu das Kontrollkästchen **Max. Baumhöhe** auf der Eigenschaftenseite **Layout** und geben Sie die maximale Baumhöhe in Zeilen an. Diese Einstellung wirkt sich nur bei vertikaler Anordnung aus. Falls in einem vertikal angeordneten Ast mehr Ebenen vorhanden sind, erfolgt ein Umbruch, und ein neuer Ast wird am Vaterknoten erzeugt.



Vertikal angeordnete Baumstruktur, maximale Höhe: 4 Zeilen

Sie können die maximale Höhe einer Baumstruktur auch über die VcTree-Eigenschaft **RowLimit** erfragen oder festlegen.

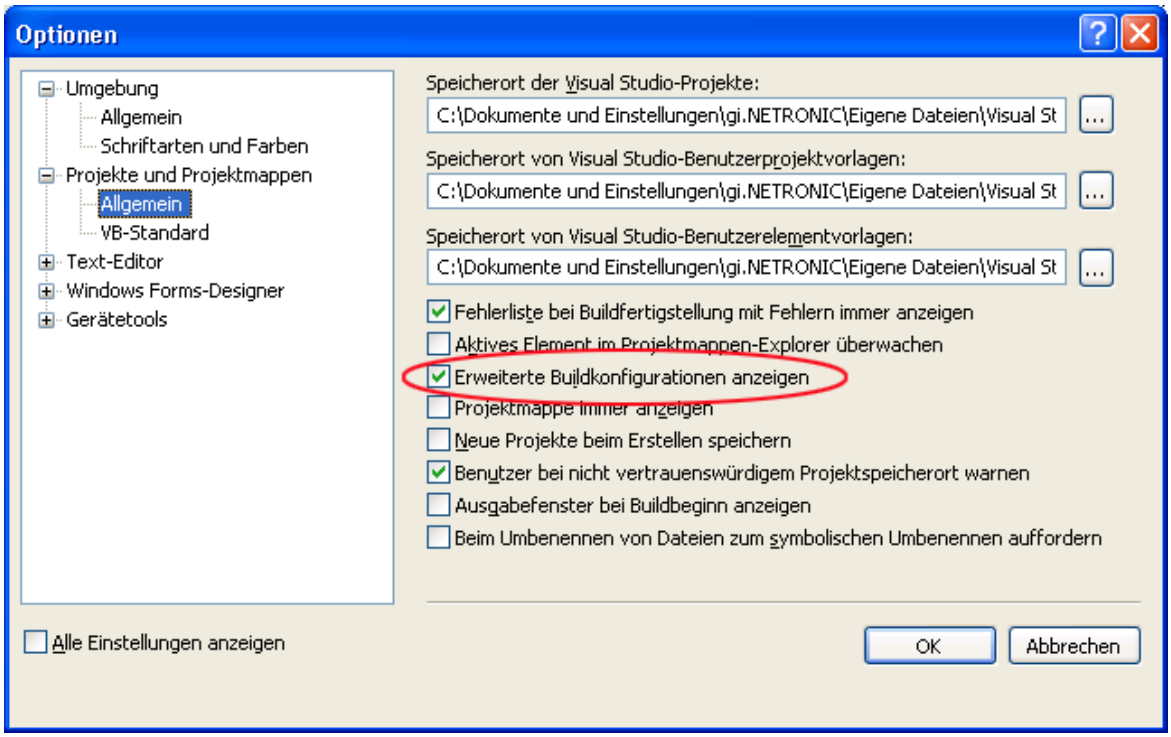
3.17 Plattformen x86 und x64

Applikationen, die mit dem .NET Framework geschrieben wurden, werden gewöhnlich über einen Compiler in MSIL, einen prozessorunabhängigen Bytecode, übersetzt. Beim Starten der Applikation wird MSIL in einen Maschinencode übersetzt, der vom jeweilig verwendeten Prozessor verstanden und in dessen voller Geschwindigkeit ausgeführt wird. Applikationen in MSIL sind also unter Windows auf jedem Prozessor ausführbar, wenn sie keine reinen Maschinencode-Komponenten (Assemblys oder DLLs) verwenden. Sie sind sogar auf anderen Betriebssystemen wie z.B. unter Mono mit Linux lauffähig, sofern keine betriebssystemabhängigen Komponenten verwendet werden. Wenn eine Anwendung die Voraussetzungen für die Prozessorunabhängigkeit nicht erfüllt, sollte sie entsprechend markiert werden, damit sie nicht irrtümlich auf einem nicht unterstützten Prozessor gestartet wird und es dann bei erstmaliger Verwendung einer prozessor- oder betriebssystemabhängigen Komponente zu Fehlermeldungen kommt.

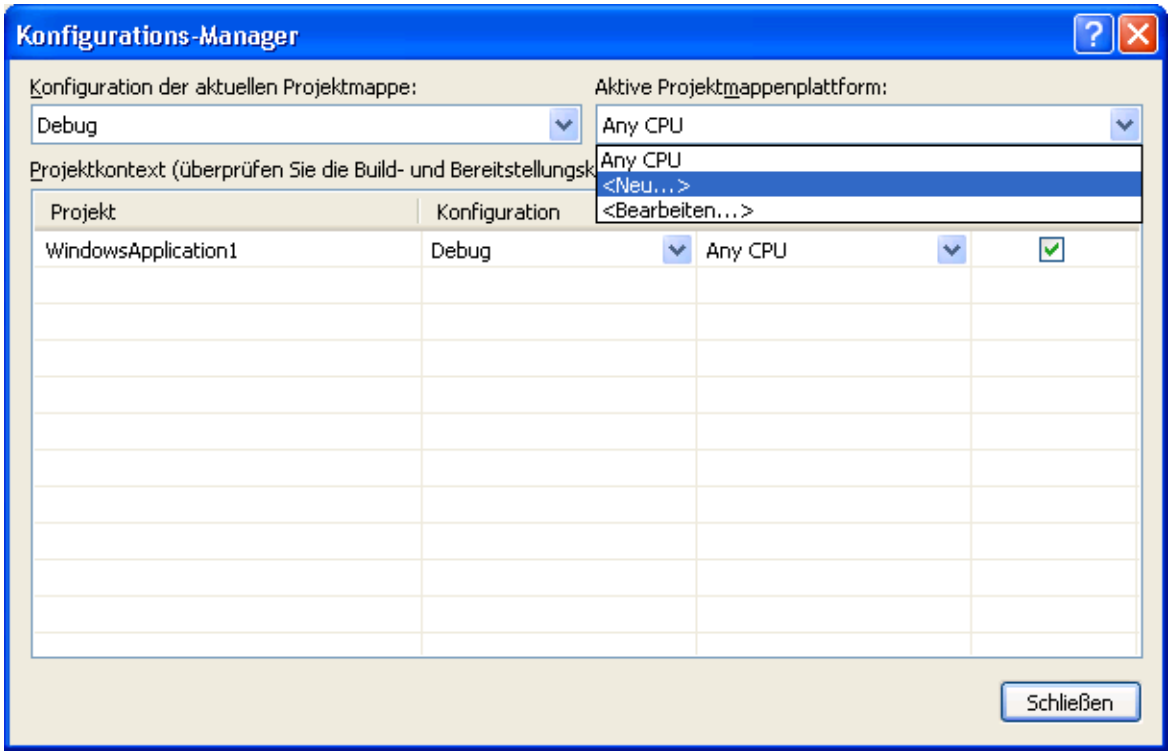
VARCHART XTree liegt intern teilweise in reinem Maschinencode vor, was unter .NET als **Mixed Mode** bezeichnet wird. Daher muss XTree für jeden Prozessor, mit dem es verwendet werden soll, neu übersetzt werden. Es sind Varianten für x86-Prozessoren und ab Version 4.3 auch für x64-Prozessore verfügbar.

Applikationen, in denen VARCHART XTree verwendet wird, sind also nicht prozessorunabhängig. Da dies von Visual Studio in den Versionen 2005 bis 2010 nicht automatisch erkannt wird, muss man den Prozessor in einem Projekt bzw. einer Projektmappe manuell einstellen. Dies geschieht über den Konfigurations-Manager, den Sie über **Erstellen/Konfigurations- Manager** erreichen.

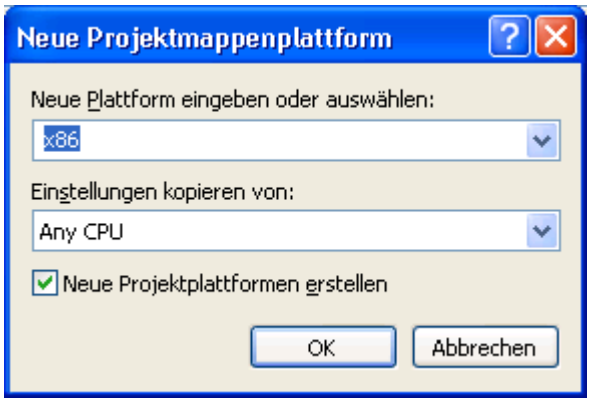
Sollte dieser Menüpunkt nicht zu sehen sein, muss erst die Einstellung **Erweiterte Buildkonfigurationen** im Dialog **Extras/Optionen/Projekte und Projektmappen/ Allgemein** aktiviert werden.



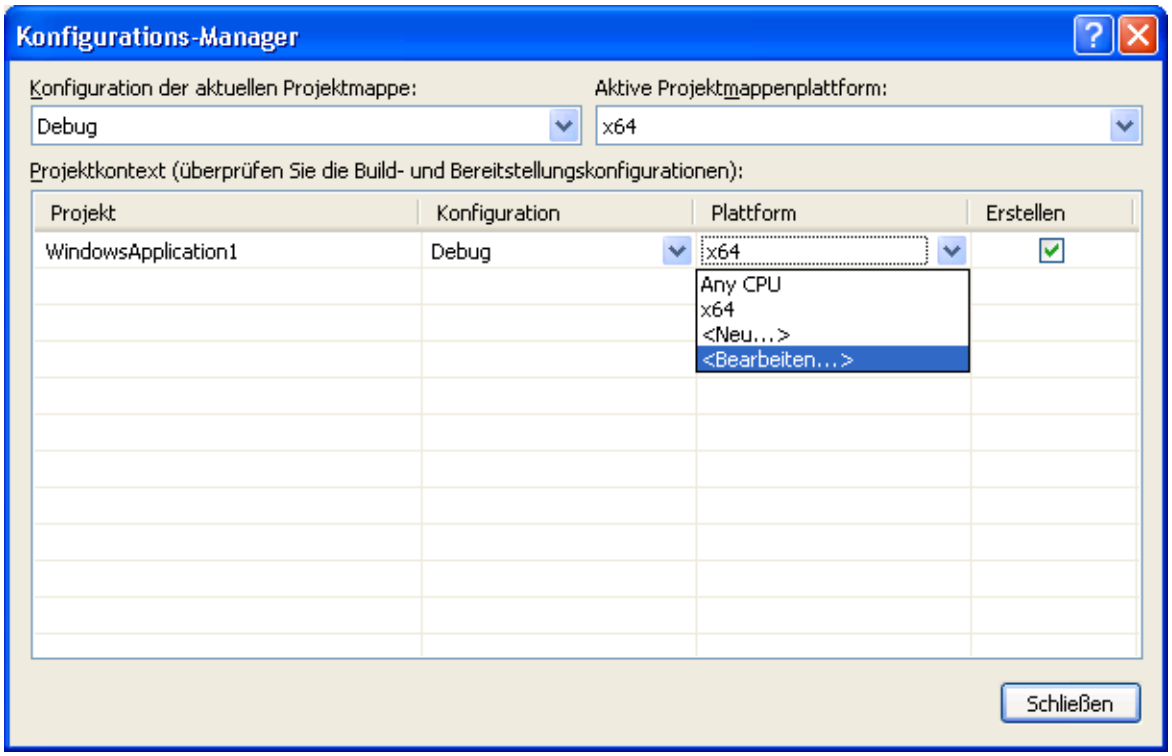
Zum Erstellen einer neuen Plattform wählen Sie bitte im Konfigurationsmanager den Eintrag <Neu> bei **Aktive Projektmappenplattform**.



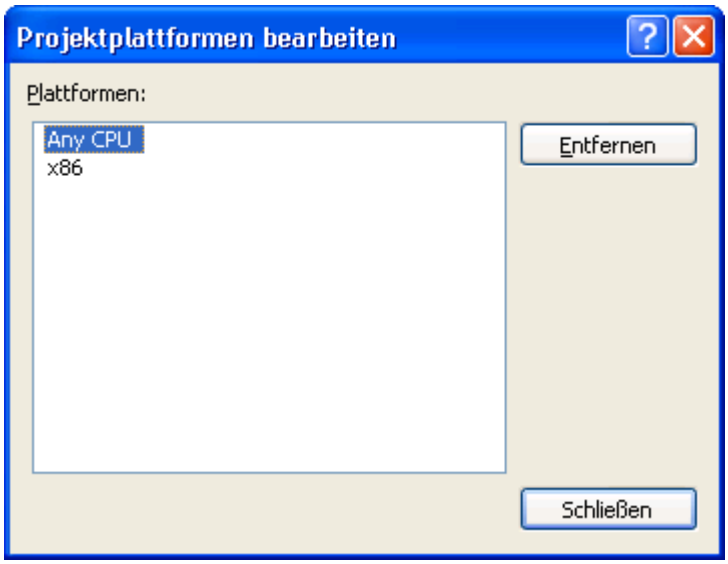
Folgender Dialog erscheint, in dem Sie nun die gewünschten Plattformen **x86** oder **x64** anlegen können:



Wenn Sie anschliessend die Plattform **Any CPU** löschen möchten, wählen Sie bitte zunächst im **Projektkontext** des Konfigurations-Managers in der Spalte **Plattform** den Menpünkt **Bearbeiten** aus der Drop-Down-Liste:



Im nun erscheinenden Dialog können Sie die gewünschte Plattform löschen, indem Sie sie markieren und auf **Entfernen** klicken:



Damit immer die richtige Variante von XTree im Visual Studio benutzt wird, müssen Sie bestimmte Prozeduren in den Pre-Build-Event und den Post-Build-Event Ihres Projektes einbauen. Diese Prozeduren befinden sich im Unterverzeichnis BuildSteps innerhalb des XTree-Installationsverzeichnisses (bei Zielframework .NET 2.0 bitte in beiden Build-Events die Zeile "set DOTNET=..." anpassen). Danach kompilieren Sie einmalig Ihr Projekt, was vom Visual Studio nicht unerwartet mit Fehlermeldungen quittiert wird. Im Anschluss fügen Sie eine Referenz auf die NETRONIC.XTree.dll im neu entstandenen Verzeichnis C:\XTreeReference ein (evtl. müssen Sie vorher eine bestehende Referenz auf das XTree-Installationsverzeichnis entfernen) und kompilieren Ihr Projekt noch einmal.

3.18 Schreiben von PDF-Dateien

Das Schreiben von PDF-Dateien ist nur möglich, wenn ein geeigneter PDF-Druckertreiber installiert ist. Die kostenlosen und kommerziell verfügbaren Treiber unterscheiden sich hinsichtlich der Funktionalität und Qualität der erzeugten PDF-Dateien.

Für die Ansteuerung der Treiber gibt es keinen einheitlichen Standard, sodass jeder Druckertreiber individuell konfiguriert werden muss. So ist beispielsweise bei vielen PDF-Druckertreibern der Zielpfad für die Ausgabedatei fest vorgegeben und kann nur durch Eingriffe in die Windows-Registry, durch Editieren von INI-Dateien oder durch Verwenden treiberspezifischer Funktions-APIs oder COM-Objekte geändert werden.

Ein PDF-Druckertreiber muss die folgenden Anforderungen hinsichtlich der Ansteuerung und Druckqualität erfüllen, damit er sich für den Einsatz eignet:

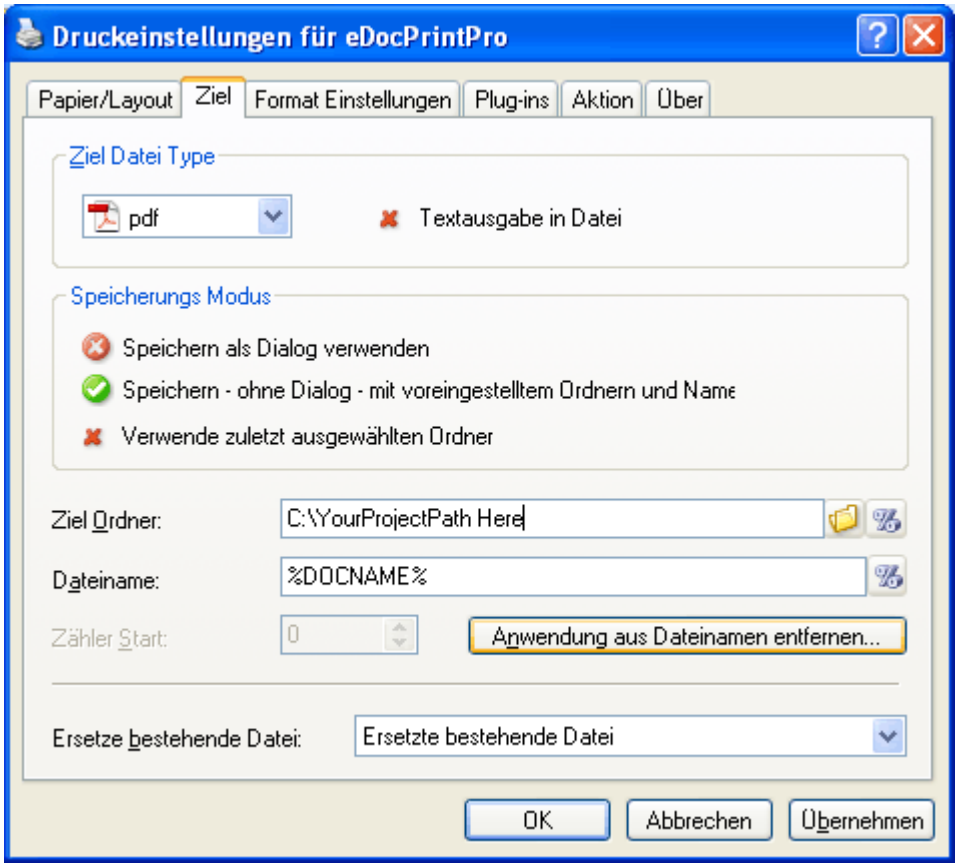
- Je nach Design der Anwendung kann es notwendig sein, den Treiber über die API so einzustellen, dass zur Laufzeit keine Dialoge und Messageboxen erscheinen. Dazu gehören insbesondere Dialoge zur Festlegung von Dateinamen und Pfaden.
- Soll das Setzen von Dateinamen und Pfad erst zur Laufzeit erfolgen, und ist dies nur über das Ändern von Windows-Registry-Einträgen möglich, dann müssen die Rechte des Benutzerkontos dies auch erlauben.
- Zur korrekten Ausgabe von Texten ist Unicode-Unterstützung erforderlich.
- Die Wiedergabe von Füllmustern muss in ausreichender Qualität erfolgen. Dabei ist zu beachten, dass Transparenzen mit Ausnahme bei Bitmaps grundsätzlich nicht dargestellt werden können. Dort können jedoch unerwünschte Artefakte auftreten.
- Der Treiber muss die Ausgabe vertikaler Texte unterstützen, sonst kann die vertikale Beschriftung von Datumslinien in VARCHART XGantt nicht genutzt werden.

Die vorgenannten Anforderungen erfüllen z.B. der in der **Adobe Acrobat Suite** ab Version 6 enthaltene Druckertreiber [www.adobe.com] sowie der kostenlose Treiber **eDocPrintPro** [www.pdfprinter.at].

Im Folgenden werden die Schritte skizziert, die zur Ansteuerung der Druckertreiber notwendig sind. Dies wird exemplarisch am Treiber **eDocPrintPro** verdeutlicht:

- In den **Druckeinstellungen** (erreichbar über die Einstellungen des Treibers in der Systemsteuerung oder über einen eigenen Eintrag des

Treibers unter Start/Programme oder über den normalen Druckdialog in einer Anwendung) kann gegebenenfalls spezifiziert werden, dass die PDF-Erzeugung ohne Dialog abläuft, und dass der Name der Zieldatei z.B. über den Dokumentennamen bestimmt wird. Bei **eDocPrintPro** sehen die notwendigen Einstellungen dann wie folgt aus:



- Im Programm wird dann das VcPrinter-Objekt von VARCHART XGantt folgendermaßen bestückt:

Code-Beispiel VB.NET

```
VcTree1.Printer.PrinterName = "eDocPrintPro"
VcTree1.Printer.DocumentName = "abc.pdf"
VcTree1.PrintEx
```

Code-Beispiel C#

```
vcTree1.Printer.PrinterName = "eDocPrintPro";
vcTree1.Printer.DocumentName = "abc.pdf";
vcTree1.PrintEx;
```

Ganz wenige Druckertreiber erfordern eine andere Code-Sequenz:

Code-Beispiel VB.NET

```
VcTree1.Printer.PrinterName = "Win2PDF"
VcTree1.PrintToFile "abc.pdf"
```

Code-Beispiel C#

```
vcTree1.Printer.PrinterName = "Win2PDF";
vcTree1.PrintToFile "abc.pdf";
```

Für weitere Fragen zur Konfiguration und Benutzung von **eDocPrintPro** bitten wir Sie, sich mit dem Hersteller in Verbindung zu setzen.

3.19 Sprachanpassung von Textausgaben

Sie können mit Hilfe des Ereignisses **VcTextEntrySupplying** die Texte aller zur Laufzeit erscheinenden Kontextmenüs, Dialogfelder, Infoboxen und Fehlermeldungen bearbeiten, z. B. um sie in unterschiedliche Sprachen zu übersetzen.

Aktivieren Sie dazu das Kontrollkästchen **VcTextEntrySupplying-Ereignisse** auf der Eigenschaftenseite **Allgemeines**.

Oder setzen Sie dazu die Eigenschaft **TextEntrySupplyingEventEnabled** auf den Wert **True**, um das Ereignis zu aktivieren.

Code-Beispiel VB.NET

```
VcTree1.TextEntrySupplyingEventEnabled = True
```

Code-Beispiel C#

```
vcTree1.TextEntrySupplyingEventEnabled = true;
```

Fangen Sie dann das Ereignis **VcTextEntrySupplyingEvent** ab und legen Sie fest, welcher Text erscheinen soll.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcTextEntrySupplying(ByVal sender As Object, ByVal e
As NETRONIC.XTree.VcTextEntrySupplyingEventArgs) Handles
VcTree1.VcTextEntrySupplying

   Select Case e.ControlIndex
      Case VcTextEntryIndex.vcTXERibCW
        e.Text = "KW"
      Case VcTextEntryIndex.vcTXERibDay0
         e.Text = "Mo"
      Case VcTextEntryIndex.vcTXERibMon8
         e.Text = "September"
      Case VcTextEntryIndex.vcTXERibQuar3
         e.Text = "3. Quartal"
   End Select
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcTextEntrySupplying(object sender,
NETRONIC.XTree.VcTextEntrySupplyingEventArgs e)
   {
   switch(e.ControlIndex)
      {
      case VcTextEntryIndex.vcTXERibCW:
         e.Text = "KW";
         break;
      case VcTextEntryIndex.vcTXERibDay0:
         e.Text = "Mo";
         break;
```

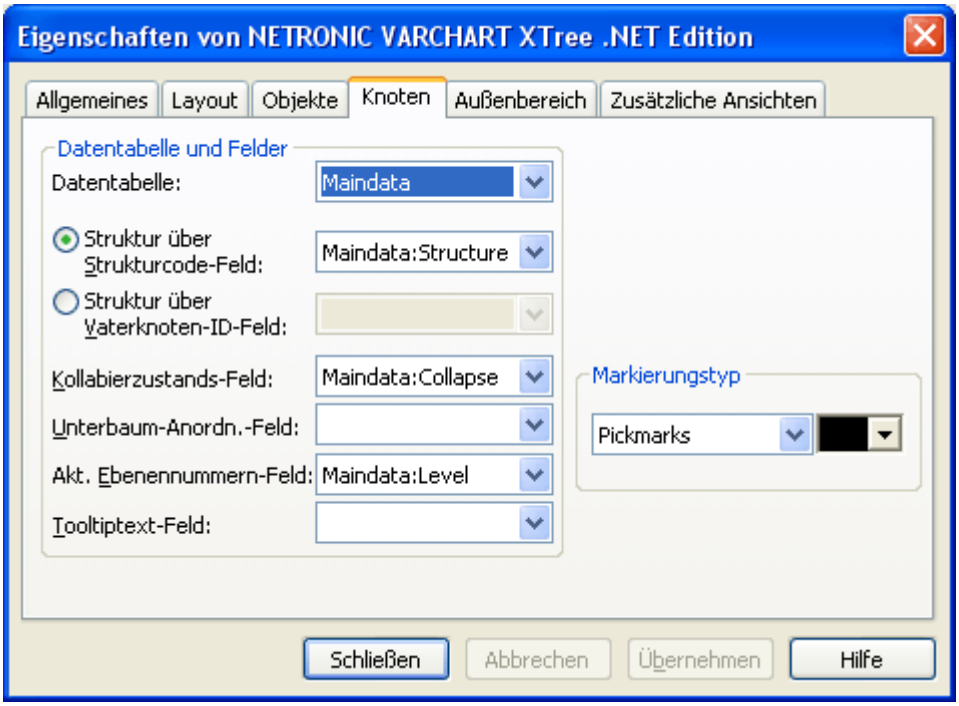
```
    case VcTextEntryIndex.vcTXERibMon8:
       e.Text = "September";
       break;
    case VcTextEntryIndex.vcTXERibQuar3:
       e.Text = "3. Quartal";
       break;
    }
}
```

3.20 Statuszeilentext

Sie können das Ereignis **VcStatusLineTextShowing** verwenden, um Informationen über den mit der Maus berührten Knoten in einer Statusleiste bereitzustellen.

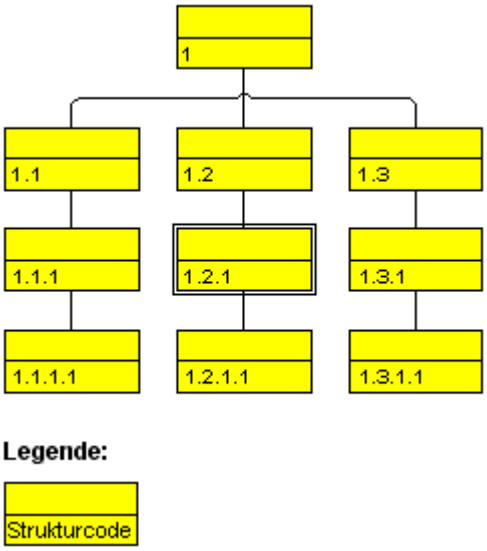
3.21 Struktur

Auf der Eigenschaftenseite **Knoten** können Sie die Struktur der Baum-Diagramme festlegen.

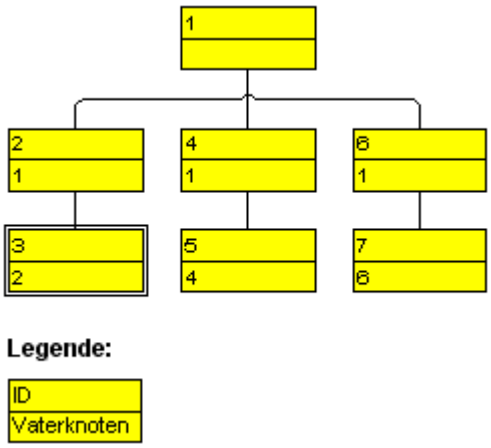


Dabei gibt es zwei grundsätzliche Alternativen:

1. **Struktur über Strukturcode-Feld:** Der Baum wird gemäß eines Strukturcodes aufgebaut. Sie können dann ein Datenfeld auswählen, das den Wert des Strukturcodes enthält. Als Trennzeichen wird ein Punkt verwendet.
2. **Struktur über Vaterknoten-ID- Feld:** Der Baum wird durch die ID des Vaterknotens jedes Knotens definiert. Sie können das Datenfeld wählen, das die ID des Vaterknotens enthält.



Beispiel für einen durch einen Strukturcode definierten Baum



Beispiel für einen durch die IDs der Vaterknoten definierten Baum

3.22 Tooltips zur Laufzeit

Sie können Tooltips verwenden, um Informationen über das mit der Maus berührte Objekt bereitzustellen. Mit Hilfe des Ereignisses **VcToolTipTextSupplying** können Sie die Texte aller zur Laufzeit erscheinenden Tooltips (Node, None) bearbeiten, z. B. um sie in unterschiedliche Sprachen zu übersetzen oder zu unterdrücken.

Setzen Sie dazu die VcTree-Eigenschaft **ToolTipTextSupplyingEventEnabled** auf den Wert **True**, um das Ereignis zu aktivieren.

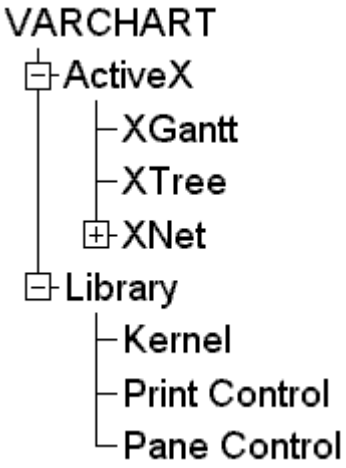
Alternativ können Sie auf der Eigenschaftenseite **Allgemeines** das Kontrollkästchen **VcToolTipTextSupplying-Ereignisse** aktivieren.

Fangen Sie dann das Ereignis **VcToolTipTextSupplying** ab und legen Sie fest, welcher Text erscheinen soll, oder ob an dieser Stelle kein Tooltip erscheinen soll.

3.23 TreeView-Stil

Knoten können im TreeView-Stil angeordnet werden. Bei dieser Ansicht werden vertikale Ebenen wie in TreeView-Controls (z. B. bekannt aus der Verzeichnisbaumansicht des Microsoft Explorers) mit Plus- oder Minus-Zeichen dargestellt. Dabei bedeutet ein Plus-Zeichen, dass der Knoten auf der gleichen Ebene kollabiert ist, und ein Minus-Zeichen, dass er expandiert ist. Die Zeichen werden nur bei Knoten angezeigt, die keine Blattknoten sind, d. h. Sohnknoten besitzen. Ein Mausklick auf eines der beiden Zeichen überführt den daneben stehenden Knoten in den jeweils anderen Kollabierzustand.

Ein Beispiel für den TreeView-Stil zeigt die folgende Abbildung:



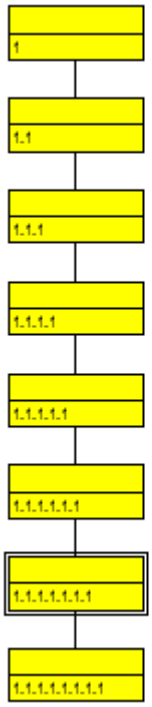
Um die Knoten im TreeView-Stil anordnen zu lassen, aktivieren Sie auf der Eigenschaftenseite **Layout** das Kontrollkästchen **TreeView-Stil**.

3.24 Vertikale Ebenen

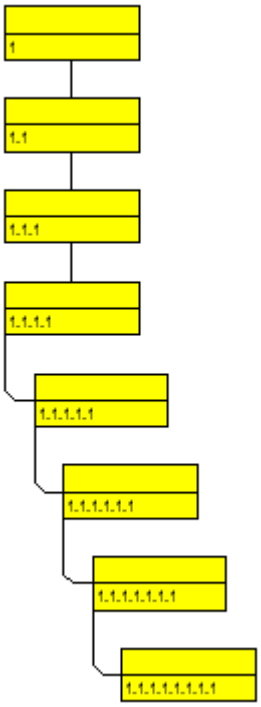
Bei der vertikalen Anordnung werden alle Knoten einer Ebene untereinander angeordnet. In vielen Fällen ist es sinnvoll, Knoten ab einer bestimmten Ebene vertikal anordnen zu lassen, um die Breite des Baum-Diagramms zu verringern.

Aktivieren Sie dazu auf der Eigenschaftenseite **Knoten** das Kontrollkästchen **Vertikal ab Ebene** und geben Sie an, ab welcher Ebene ggf. vertikal angeordnet werden soll.

Damit die Knoten tatsächlich vertikal angeordnet werden, muss die API-Methode **Arrange** aufgerufen werden.



vollständig horizontal angeordneter Baum



*nach dem Aufruf von **Arrange**: ab Ebene 4 vertikal angeordneter Baum*

3.25 Viewer Metafile (*.vmf)

Viewer Metafile (VMF) ist ein Grafikformat, das speziell für den von der NETRONIC Software GmbH entwickelten WebViewer (ein plattform- und browserunabhängiges Java Applet) entwickelt worden ist. Das VMF-Format ermöglicht es, Ihre Diagramme über das Intra- bzw. Internet in einem Browser zu betrachten, zu zoomen oder zu drucken.

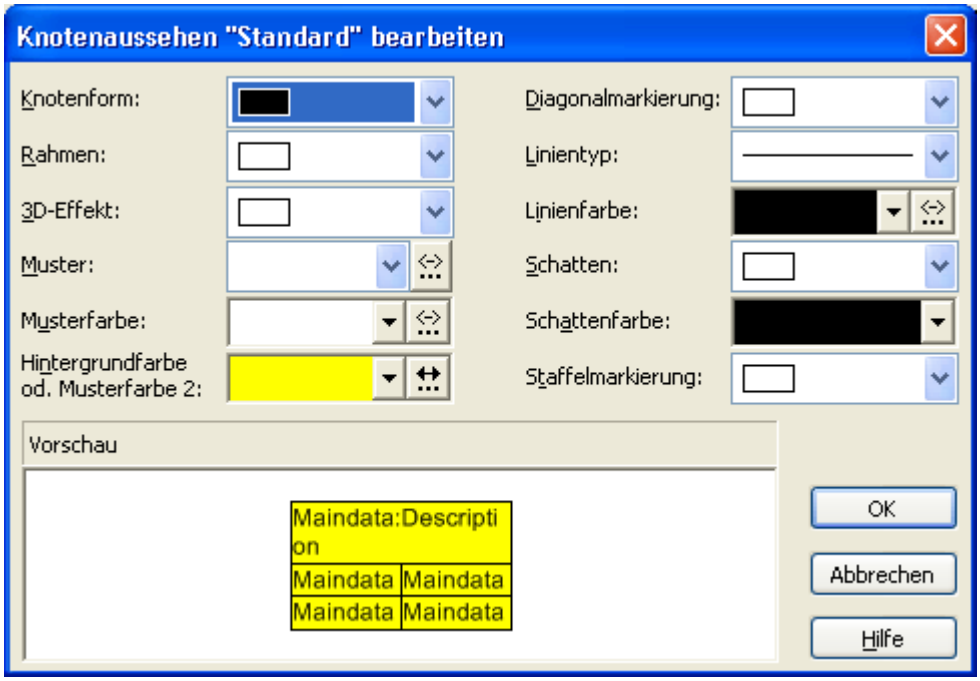
Mit der Methode **ExportGraphicsToFile** des Objektes VcTree oder mit dem Standard-Kontextmenü für das Diagramm können Sie ein Diagramm in einer Datei abspeichern.

3.26 Zuordnungstabellen

Das Knotenaussehen und das Knotenformat können über Zuordnungstabellen datenabhängig gestaltet werden.

> Knotenaussehen datenabhängig festlegen

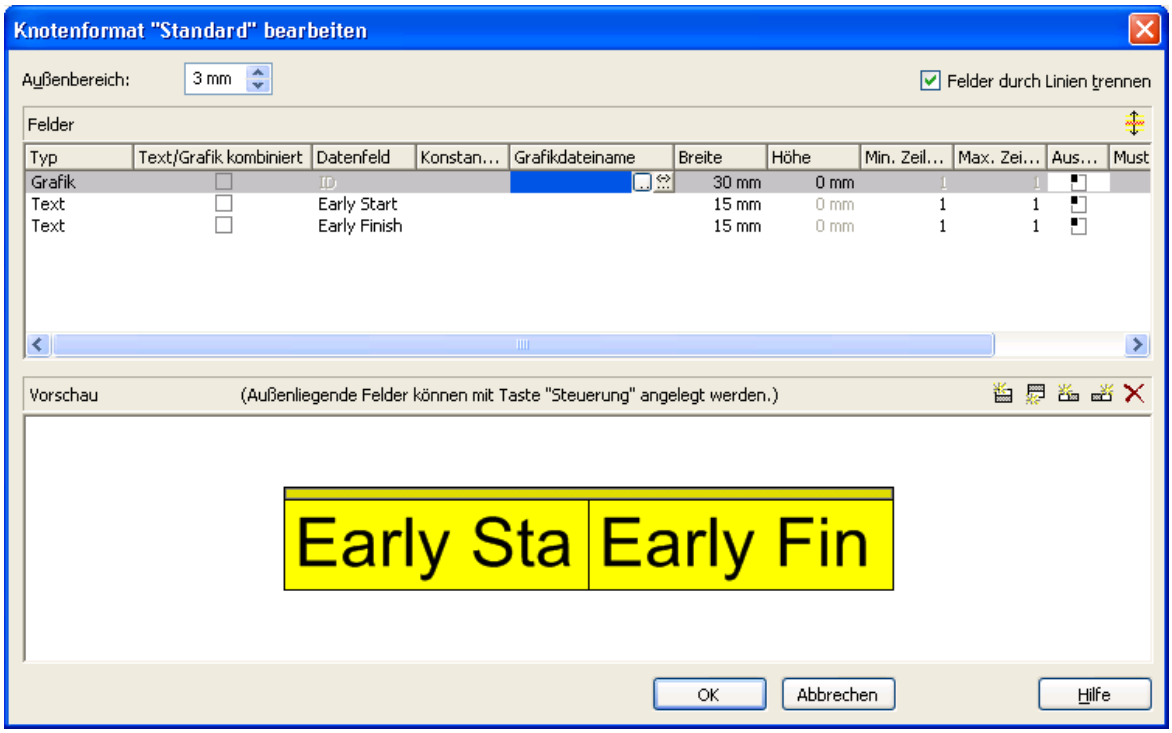
Für jedes Knotenaussehen können Muster, Musterfarbe, Hintergrund- oder 2. Musterfarbe sowie Linienfarbe über Zuordnungstabellen datenabhängig festgelegt werden. Klicken Sie dazu im Dialogfeld **Knotenaussehen bearbeiten** auf die zweite Schaltfläche für die **Hintergrundfarbe** bzw. **Linienfarbe** ( ).



Sie gelangen in den Dialog **Zuordnung einstellen**.

> Grafik eines Knotenformats datenabhängig festlegen

In den Feldern der Knotenformate können Grafiken über eine Zuordnungstabelle datenabhängig ausgegeben werden.

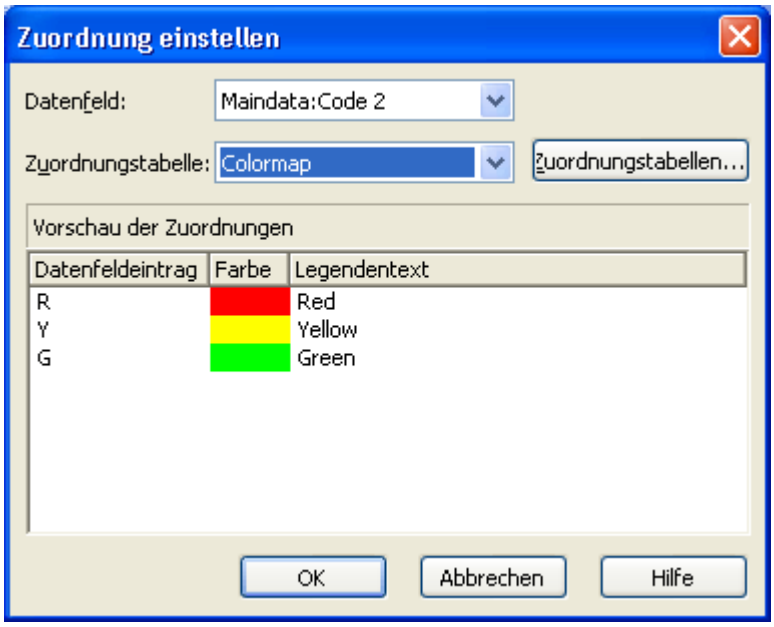


Um eine Zuordnung zwischen den Einträgen eines Datenfeldes vom Typ Grafik und Grafikdateien herzustellen, klicken Sie im Feld **Grafikdateiname** auf die 2. Schaltfläche (**Zuordnungen einstellen**). Der gleichnamige Dialog erscheint dann.

Wenn Sie dort eine Zuordnung vorgenommen haben, erscheint ein Symbol ( ) im Feld **Grafikdateiname**, sobald Sie die entsprechende Zeile verlassen.

> Zuordnung einstellen

Im Dialogfeld **Zuordnung einstellen** können Sie festlegen, dass in einem bestimmten Knotenformatfeld vom Typ Grafik datenabhängig Grafikdateien dargestellt werden sollen, bzw. dass die Hintergrundfarbe eines bestimmten Knotenaussehens datenabhängig sein soll.



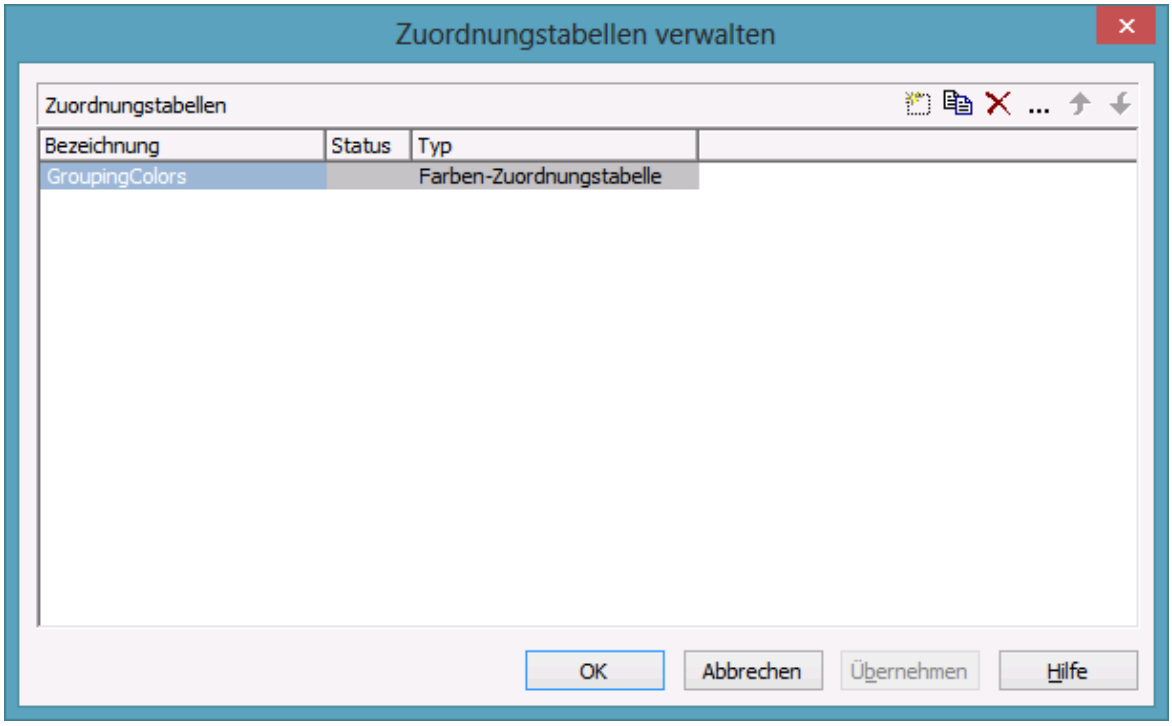
Wählen Sie dazu in der ersten Kombobox das **Datenfeld**, von dessen Einträgen die Grafikdatei des zu bearbeitenden Knotenformatfeldes bzw. die Hintergrundfarbe des zu bearbeitenden Knotenaussehens abhängen soll. Wählen Sie dann in der zweiten Kombobox die **Zuordnungstabelle**, die den einzelnen Datenfeldeinträgen eine Grafikdatei bzw. eine Farbe und einen Legendentext zuordnet.

In der **Vorschau der Zuordnungen** wird dargestellt, wie die gewählte Zuordnungstabelle den einzelnen Datenfeldeinträgen eine Grafikdatei bzw. eine Hintergrundfarbe und einen Legendentext zuordnet.

> Zuordnungstabellen verwalten

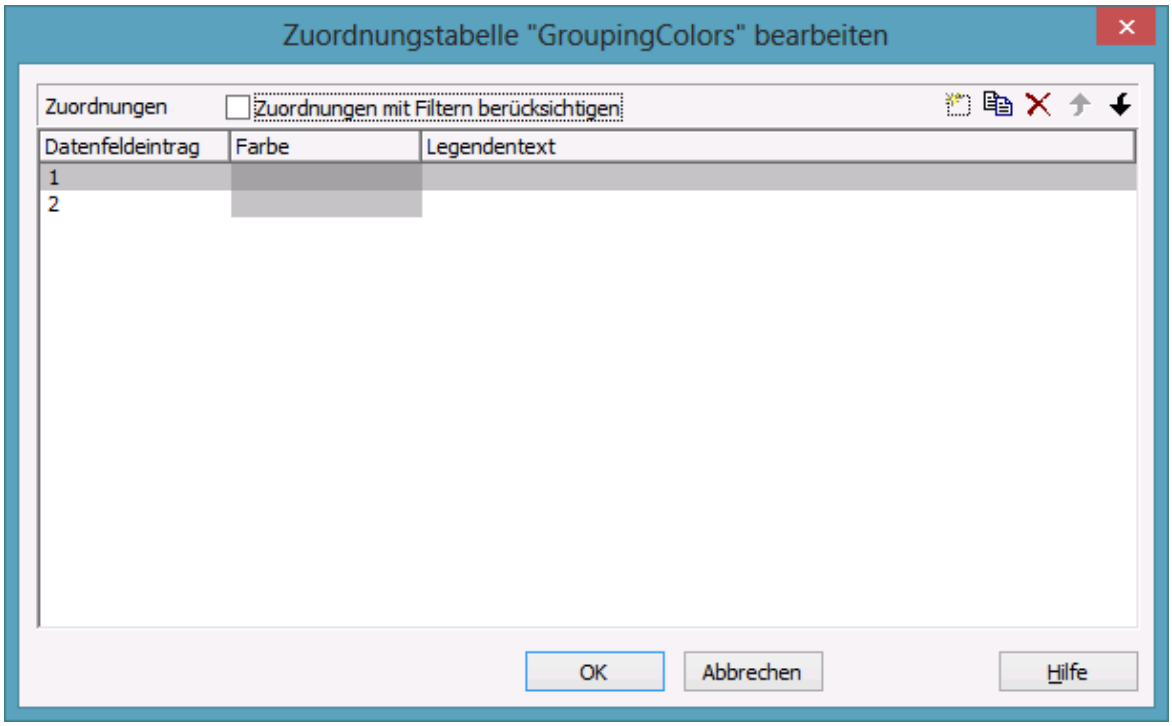
Im Dialogfeld **Zuordnungstabellen verwalten**, das Sie durch Klick auf die Schaltfläche **Zuordnungstabellen** oder über die Schaltfläche **Zuordnungstabellen** auf der Eigenschaftenseite **Objekte** erreichen, können Sie Namen und Typ einer Zuordnungstabelle durch direkte Eingabe verändern sowie über die entsprechenden Schaltflächen oben rechts im Fenster Zuordnungstabellen erstellen, kopieren, löschen oder bearbeiten.

Sie können aus verschiedenen Typen von Zuordnungstabellen auswählen, je nachdem, ob den Datenfeldinhalten Farben, Muster, Grafiken, Schrifttypen, Längen oder Nummern zugeordnet werden sollen.



> Zuordnungstabellen bearbeiten

Um eine Zuordnungstabelle zu bearbeiten, markieren Sie diese in der Tabelle und klicken Sie auf die Schaltfläche ... oberhalb der Tabelle. Es erscheint das Dialogfeld **Zuordnungstabelle bearbeiten**.



In der **Zuordnungen**-Tabelle werden für jeden Schlüssel die entsprechenden Werte aufgelistet, in unserem Beispiel sind dies die Hintergrundfarbe und der Legendentext.

Über die Schaltflächen oben rechts können Sie Schlüssel (Zuordnungen) hinzufügen, kopieren oder löschen oder deren Reihenfolge verändern.

Wenn die Option **Zuordnungen mit Filtern berücksichtigen** ausgewählt ist, werden nicht nur die in der Liste der Datenfeldeinträge angegebenen festen Werte als Schlüssel berücksichtigt, sondern auch Filter, die aus der Dropdown-Liste ausgewählt werden können. Dadurch hängen die Ausführungswerte nicht mehr nur von einem konkreten Wert , sondern von komplexeren Kriterien ab.

Sie können in einer Zuordnungstabelle maximal 150 Zuordnungen festlegen. Falls Sie weitere Zuordnungen benötigen, erstellen Sie einfach eine neue Zuordnungstabelle, z. B. als Kopie der bereits vorhandenen.

Einzelheiten zu den hier beschriebenen Dialogfeldern finden Sie im Kapitel "Eigenschaftenseiten und Dialogfelder".

> Anpassung der Zuordnungstabelle zur Laufzeit

Sie können die Zuordnungstabellen auch zur Laufzeit noch mit Hilfe der VcMap-Methoden anpassen. Damit geben Sie dem Anwender die Möglichkeit, Ihre Voreinstellungen über einen von Ihnen erstellten Dialog zu verändern.

4 Eigenschaftenseiten und Dialogfelder

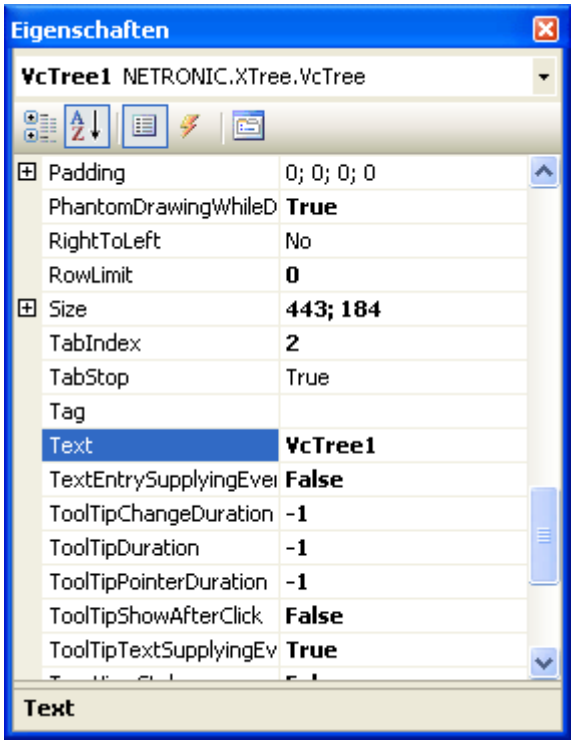
4.1 Allgemeines

Über die Eigenschaftenseiten kann VARCHART XTree bereits zur Entwurfszeit konfiguriert werden. Es gibt zwei Möglichkeiten, zu den Eigenschaftenseiten zu gelangen:

- Drücken Sie die rechte Maustaste, wenn der Mauszeiger sich innerhalb des Steuerelements befindet, und wählen Sie im Kontextmenü den Befehl **Eigenschaften** aus.

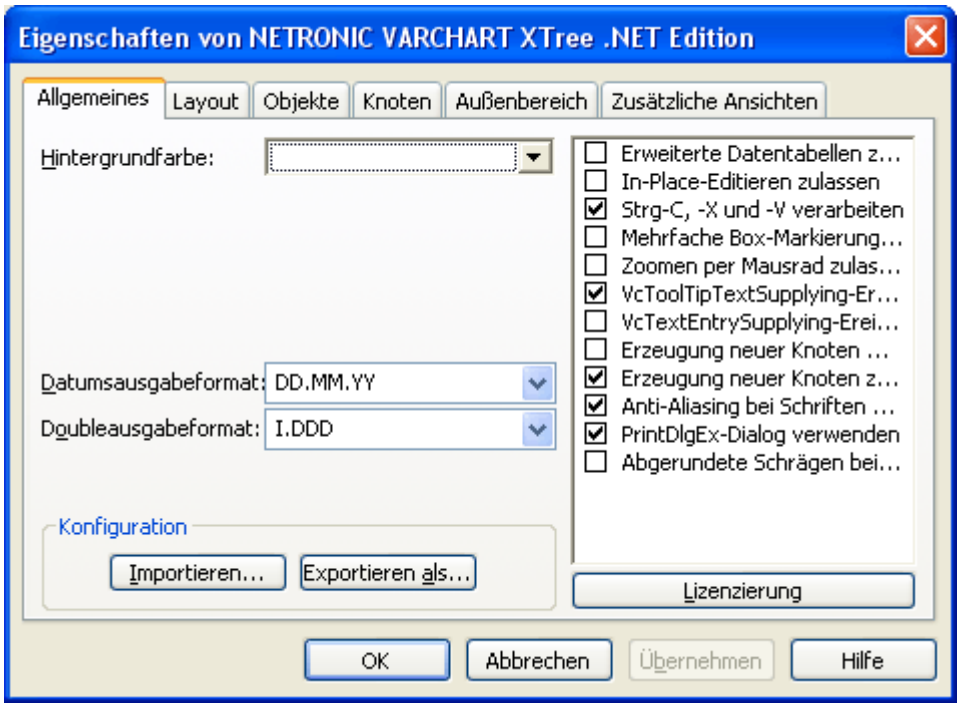
oder

- Im Eigenschaftenfenster (kann mit F4 geöffnet werden) klicken Sie in der Symbolleiste auf das ganz rechts stehende Symbol .



Nähere Informationen zu jeder Eigenschaftenseite bzw. jedem Dialogfeld erhalten Sie, indem Sie auf die **Hilfe**-Schaltfläche klicken oder die F1-Taste drücken. Sie erhalten dann direkt die Online-Hilfe zu der Eigenschaftenseite bzw. dem Dialogfeld.

4.2 Eigenschaftenseite "Allgemeines"



Auf dieser Eigenschaftenseite können Sie allgemeine Einstellungen für VARCHART XTree vornehmen.

Hintergrundfarbe

Wählen Sie hier die Hintergrundfarbe für das Baum-Diagramm aus. Die Standardfarbe ist weiß.

Datumsausgabeformat

Wählen Sie aus der Kombobox das Datumsformat aus, in dem Ihre Daten ausgegeben werden sollen, oder definieren Sie Ihr eigenes Datumsformat. Dieses Format gilt auch für alle im Steuerelement integrierten Dialogfelder, die zur Laufzeit verfügbar sind.

Diese Option kann auch über die Eigenschaft **VcTree.DateOutputFormat** festgelegt werden.

Für das Datum stehen folgende Kürzel zur Verfügung:

D: Wochentagsname erster Buchstabe (nicht anpassbar)

TD: Wochentagsname (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

DD: Tagesnummer zweistellig: 01-31 (nicht anpassbar)

DDD: die ersten drei Buchstaben des Wochentagsnamens (nicht anpassbar)

M: erster Buchstabe des Monatsnamens (nicht anpassbar)

TM: Monatsname (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

MM: zweistellige Monatsnummer: 01-12

MMM: erste drei Buchstaben des Monatsnamens (nicht anpassbar)

YY: zweistellige Jahreszahl

YYYY: vierstellige Jahreszahl

WW: zweistellige Nummer der Kalenderwoche: 01-53

TW: Text für "Kalenderwoche" (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

Q: einstellige Quartalsnummer: 1-4

TQ: Quartalsname (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

hh: Stunde zweistellig im 24-Stunden-Format: 00-23

HH: Stunde zweistellig im 12-Stunden-Format: 01-12

Th: Text für "o' clock" (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

TH: "am" oder "pm" (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

mm: Minute zweistellig: 00-59

ss: Sekunde zweistellig: 00-59

TS: kurzes Datumsformat, wie in der Windows-Systemsteuerung über die Regions- und Sprachoptionen definiert

TL: langes Datumsformat, wie in der Windows-Systemsteuerung über die Regions- und Sprachoptionen definiert

TT: Zeitformat, wie in der Windows-Systemsteuerung über die Regions- und Sprachoptionen definiert

Hinweis: Zeichen, die nicht als Teil des Datums interpretiert werden sollen, sollten mit einem vorangehenden rückwärtigen Schrägstrich '\' gekennzeichnet werden. '\\' zum Beispiel ergibt '\'. Die Sonderzeichen **':, /, -'** und **Leerzeichen** benötigen keinen '\' als Präfix.

Double-Ausgabeformat

Markieren Sie in der Kombobox das Format, in dem Ihre Daten vom Typ **Double** ausgegeben werden sollen. Sie können wählen zwischen **I** (ganze Zahl), **I,DDD, I,DDDDDD bzw. I.DDD, I.DDDDDD** (3 oder 6 Dezimalstellen) und **$ I,III.DD** bzw. **I.III,DD €** (Währung mit 2 Dezimalstellen).

Diese Option kann auch über die Eigenschaft **VcTree.DoubleOutputFormat** festgelegt werden.

Konfiguration

Alle Einstellungen der Eigenschaftenseiten können Sie auch jederzeit in Form einer Konfiguration außerhalb Ihres Projektes speichern und nach Bedarf wieder einlesen. Dies ist sehr praktisch, wenn Sie zu einem früheren Stand der Einstellungen zurückkehren oder die gleichen Einstellungen für andere Projekte verwenden möchten.

Eine gespeicherte Konfiguration besteht aus zwei Dateien mit gleichem Namen aber unterschiedlichen Dateiendungen. Zu einer Konfiguration gehört jeweils eine INI- und eine IFD-Datei, die beide zwingend benötigt werden.

Als Konfigurationsdatei können Sie eine lokale Datei mit Pfad oder eine URL angeben.

Die Angabe einer URL als Konfigurationsdatei ist nur sinnvoll, wenn die Konfiguration über die API zur Laufzeit festgelegt wird, da dann die INI- und IFD-Dateien von der angegebenen URL heruntergeladen werden. (Gibt man schon zur Designzeit eine URL als Konfigurationsdatei an, werden die INI- und IFD-Dateien zwar ebenfalls heruntergeladen, aber als Ressource dem Projekt hinzugefügt und zur Laufzeit verwendet, statt die Dateien direkt herunterzuladen.)

So speichern Sie Ihre aktuelle Konfiguration:

Klicken Sie auf die Schaltfläche **Exportieren als...** und vergeben im folgenden Dialog den gewünschten Dateinamen für die INI-Datei. Die IFD-Datei wird automatisch erzeugt.

So lesen Sie eine bereits gespeicherte Konfiguration wieder ein:

Klicken Sie auf die Schaltfläche **Importieren...** und wählen die gewünschte Datei aus.

Erweiterte Datentabellen zulassen

Wenn Sie dieses Kontrollkästchen aktivieren, können Sie mehr (bis zu 99) Datentabellen anlegen und nutzen, als die standardmäßig vorhandenen **Main data** und **Relations**. Diese Option kann auch über die Eigenschaft **VcTree.ExtendedDataTablesEnabled** gesetzt werden.

In-Place-Editieren zulassen

Aktivieren Sie dieses Kontrollkästchen, wenn das direkte Editieren in Knotenfeldern und in Boxen möglich sein soll. Diese Option kann auch über die Eigenschaft **VcTree.InPlaceEditingAllowed** gesetzt werden.

Wenn einzelne Datenfelder nicht editierbar sein sollen, so darf in der Datendefinition die Option **editierbar** nicht ausgewählt werden.

Strg-C, -X und -V verarbeiten

Aktivieren Sie dieses Kontrollkästchen, damit die Tastenkombinationen Strg+C, Strg+X und Strg+V automatisch in die Zwischenablage-Operationen **CopyNodesToClipboard**, **CutNodesToClipboard** bzw. **PasteNodes-FromClipboard** übersetzt werden. Dieses Verhalten kann abgeschaltet werden, damit in Visual Basic kein Konflikt mit Bearbeitungsmethoden für Menüpunkte entsteht, die dieselben Tastaturkombinationen benutzen. Diese Option kann auch über die Eigenschaft **VcTree.CtrlCXVProcessing-Enabled** gesetzt werden.

Mehrfache Box-Markierung zulassen

Wenn Sie dieses Kontrollkästchen aktivieren, können zur Laufzeit mehrere Boxen gleichzeitig durch Anklicken markiert werden, ohne gleichzeitig die STRG-Taste gedrückt halten zu müssen. Diese Option ist standardmäßig deaktiviert.

Diese Option kann auch über die Eigenschaft **VcTree.MultipleBox-MarkingAllowed** gesetzt werden

Zoomen per Mausrad zulassen

Aktivieren Sie dieses Kontrollkästchen, um das Zoomen per Mausrad zuzulassen. Um zu zoomen, muss der Anwender die Strg-Taste festhalten und das Mausrad drehen.

Diese Option kann auch über die Eigenschaft **VcTree.ZoomingPerMouse-WheelAllowed** gesetzt werden.

VcToolTipTextSupplying-Ereignisse

Aktivieren Sie dieses Kontrollkästchen, um das Ereignis **VcToolTipText-Supplying** freizuschalten. Dies kann auch durch die Eigenschaft **ToolTip-TextSupplyingEventEnabled** geschehen. Mit Hilfe dieses Ereignisses können Sie die Texte der Tooltips, die bei Objekten angezeigt werden sollen, festlegen.

VcTextEntrySupplying-Ereignisse

Aktivieren Sie dieses Kontrollkästchen, um das Ereignis **VcTextEntry-Supplying** freizuschalten. Mit Hilfe dieses Ereignisses können Sie die Texte aller Kontextmenüs, Dialogfelder, Infoboxen und Fehlermeldungen, die zur Laufzeit erscheinen, verändern, beispielsweise um sie in unterschiedliche Sprachen zu übersetzen.

Diese Option kann auch über die Eigenschaft **VcTree.TextEntrySupplying-EventEnabled** gesetzt werden.

Erzeugung neuer Knoten mit Dialog

Wenn Sie diese Option wählen, öffnet sich das Dialogfeld **Vorgänge bearbeiten** automatisch, wenn der Anwender interaktiv einen neuen Knoten erzeugt hat. Das Dialogfeld **Vorgänge bearbeiten** kann - auch wenn diese Option deaktiviert ist - nach dem Erzeugen eines Knotens durch einen Doppelklick auf diesen geöffnet werden.

Diese Option kann auch über die Eigenschaft **VcTree.NodeCreationWith-Dialog** eingestellt werden.

Erzeugung neuer Knoten zulassen

Nur wenn Sie diese Option wählen, lassen Sie zu, dass der Anwender in einem geöffneten Projekt interaktiv neue Knoten erzeugen kann.

Diese Option kann auch über die Eigenschaft **VcTree.NodeCreation-Allowed** eingestellt werden.

Anti-Aliasing bei Schriften

Mit dieser Option können Sie das Glätten von Schriftzeichen erlauben oder unterdrücken. Wenn dies bei bestimmten Schriftarten, insbesondere bei nichtlateinischen Zeichen, zu verminderter Lesefähigkeit führt, sollte die Eigenschaft abgeschaltet werden.

Die Glättung per GDI+ bewirkt zusätzlich, dass die Texte bei jeder Zoomstufe die gleiche relative Ausdehnung haben, so dass immer dieselbe Anzahl Zeichen z.B. in ein Tabellenfeld passt. Wenn diese Eigenschaft aber auf **False** steht, dann wird stattdessen die Einstellung des Betriebssystems übernommen (einstellbar in der **Systemsteuerung**, Dialogfeld **Anzeige**, Reiter **Darstellung: Effekte**). Wenn dort eine Kantenglättung eingeschaltet ist, dann werden also Texte weiterhin geglättet. Es kann dann aber sein, dass bei manchen Zoomstufen mehr Text sichtbar ist als bei anderen, weil die systemeigene Kantenglättung dies nicht garantiert.

Diese Option kann auch über die Eigenschaft **VcTree.FontAntiAliasing-Enabled** gesetzt werden

Anti-Aliasing bei Schriften

Mit dieser Option können Sie das Glätten von Schriftzeichen erlauben oder unterdrücken. Wenn dies bei bestimmten Schriftarten, insbesondere bei nichtlateinischen Zeichen, zu verminderter Lesefähigkeit führt, sollte die Eigenschaft abgeschaltet werden.

Die Glättung per GDI+ bewirkt zusätzlich, dass die Texte bei jeder Zoomstufe die gleiche relative Ausdehnung haben, so dass immer dieselbe Anzahl Zeichen z.B. in ein Tabellenfeld passt. Wenn diese Eigenschaft aber auf **False** steht, dann wird stattdessen die Einstellung des Betriebssystems übernommen (einstellbar in der **Systemsteuerung**, Dialogfeld **Anzeige**, Reiter **Darstellung: Effekte**). Wenn dort eine Kantenglättung eingeschaltet ist, dann werden also Texte weiterhin geglättet. Es kann dann aber sein, dass bei manchen Zoomstufen mehr Text sichtbar ist als bei anderen, weil die systemeigene Kantenglättung dies nicht garantiert.

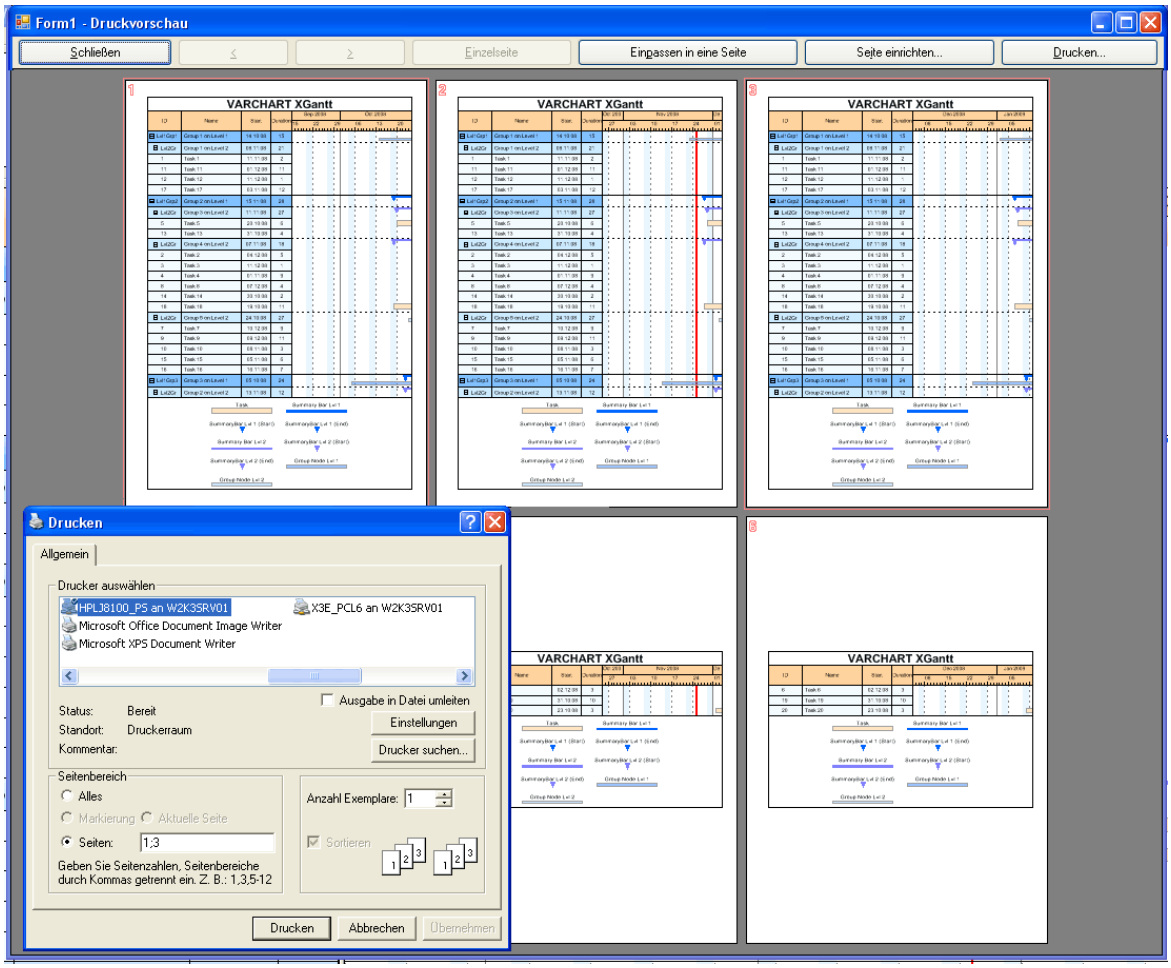
Diese Option kann auch über die Eigenschaft **VcTree.FontAntiAliasing-Enabled** gesetzt werden

PrintDlgEx Dialog verwenden

Wenn diese Option ausgewählt ist, kann zur Laufzeit der Dialog **Drucker einrichten** weder aus dem Kontextmenü noch aus der Druckvorschau aufgerufen werden, da er sich nun im (erweiterten) **Drucken**-Dialog befindet. Bei neuen Projekten ist die Option standardmäßig aktiviert, bei bestehenden Projekten ist sie aus Gründen der Kompatibilität ausgeschaltet.

In der Druckvorschau können nun einzelne oder mehrere Seiten durch Klick bzw. STRG+Klick ausgewählt werden. Diese Seiten sind dann im Dialog **Drucken** unter **Seitenbereich** bereits voreingestellt.

Wenn man aus der Druckvorschau den **Drucken**-Dialog aufruft, sind alle Blätter mit einer Seitennummer versehen, um so die Seitenauswahl zu erleichtern.



Diese Option kann **nicht** an der API gesetzt werden.

Wartecursor bei zeitkritischen Operationen eingeschaltet

Wählen Sie diese Option, wenn bei zeitkritischen Aufrufen (wie ScheduleProject) von uns intern ein Wartecursor gesetzt werden soll.

Die Option kann auch über die Eigenschaft **VcTree.WaitCursorEnabled** gesetzt werden.

Panning-Modus zulassen

Diese Option muss aktiviert werden, damit Sie zur Laufzeit einen bestimmten Bildschirmausschnitt verschieben können. Im Kontextmenü des Diagramms erscheint dann der zusätzliche Eintrag **Verschiebe-Modus**.

Der Verschiebemodus gilt standardmäßig für **alle** grafischen Grundelemente. Mithilfe der Eigenschaft **VcGantt.VcViewComponent** kann die Verwendung auf einzelne Elemente beschränkt werden.

Die Option kann auch über die Eigenschaft **VcTree.PanningModeAllowed** gesetzt werden.

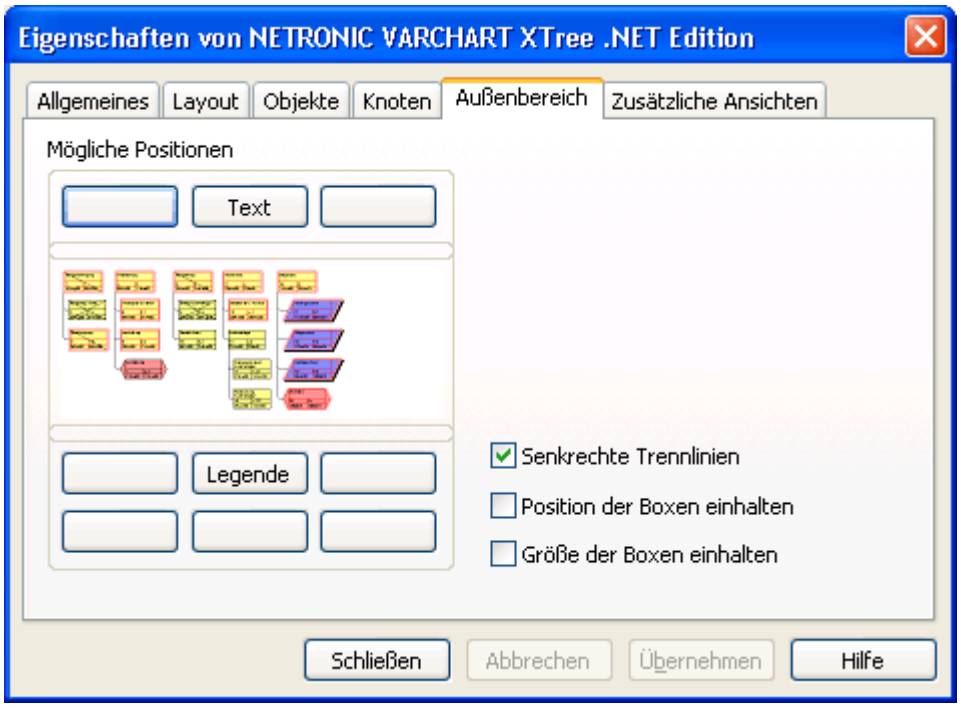
Abgerundete Schrägen bei Verbindungen

Aktivieren Sie dieses Kontrollkästchen, damit Schrägen bei Verbindungen als Viertelkreise statt als gerade Linien dargestellt werden. Diese Option kann auch über die VcTree-Eigenschaft **RoundedLinkSlantsEnabled** gesetzt werden.

Lizenzierung

Über diese Schaltfläche gelangen Sie in den Dialog **Lizenzierung**. Für weitere Informationen s. Kap. **Lizenzierung**

4.3 Eigenschaftenseite "Außenbereich"



Mögliche Positionen

Oberhalb der Grafik stehen Ihnen drei und unterhalb der Grafik sechs Bereiche zur Verfügung, in denen Sie Texte, Grafiken oder eine Legende plazieren können. Jeder dieser Bereiche wird in diesem Dialog durch je eine Schaltfläche repräsentiert. Alle diese Bereiche werden nur in der Seitenansicht und im Ausdruck angezeigt. Klicken Sie auf eine der Schaltflächen ober- bzw. unterhalb der Grafik, um den Dialog **Texte, Grafiken und Legende festlegen** zu öffnen.

Senkrechte Trennlinien

Aktivieren Sie dieses Kontrollkästchen, wenn die Bereiche für Texte, Grafiken oder Legende durch senkrechte Trennlinien getrennt werden sollen.

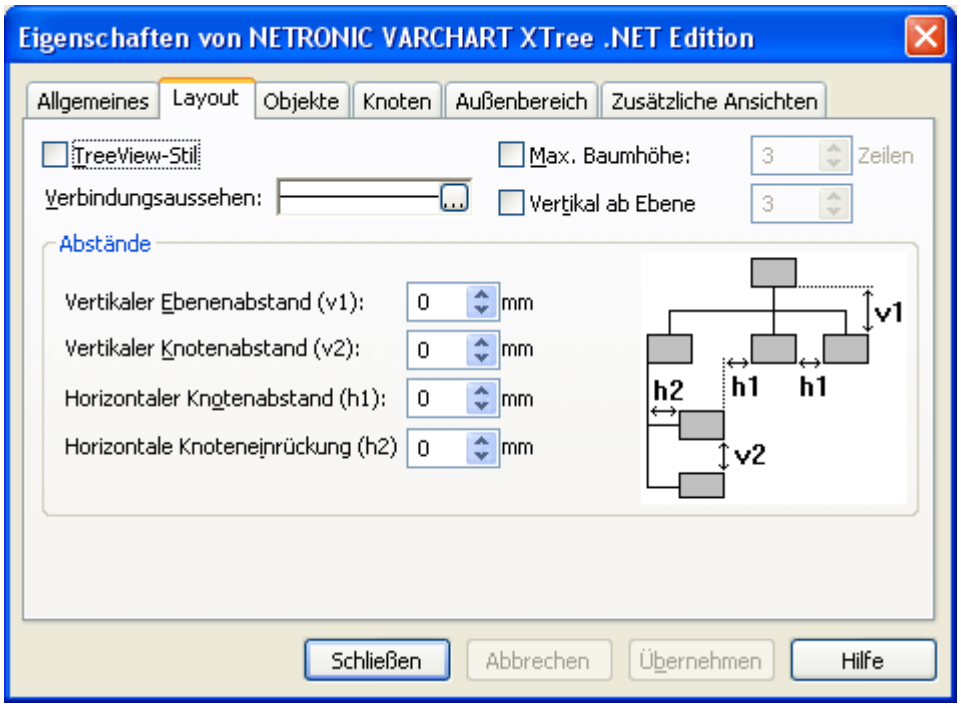
Position der Boxen einhalten

Aktivieren Sie dieses Kontrollkästchen, wenn die Position der Boxen möglichst genau eingehalten werden soll. Andernfalls wird der vorhandene Platz proportional auf die in der jeweiligen Zeile vorhandenen Elemente verteilt.

Größe der Boxen einhalten

Aktivieren Sie dieses Kontrollkästchen, wenn die Größe der Boxen möglichst genau eingehalten werden soll. Gegebenenfalls wird das Diagramm vergrößert und/oder die Texte in den Boxen abgeschnitten.

4.4 Eigenschaftenseite "Layout"



Auf dieser Eigenschaftenseite können Sie das Layout Ihres Diagramms festlegen.

TreeView-Stil

Aktivieren Sie dieses Kontrollkästchen, damit Knoten im TreeView-Stil angeordnet werden. Bei dieser Ansicht werden vertikale Ebenen wie in TreeView-Controls (z. B. bekannt aus der Verzeichnisbaumansicht von Microsoft Explorer) mit Plus- oder Minus-Zeichen dargestellt. Dabei bedeutet ein Plus-Zeichen, dass der Knoten auf der gleichen Ebene kollabiert ist, und ein Minus-Zeichen, dass er expandiert ist. Die Zeichen werden nur bei Knoten angezeigt, die keine Blattknoten sind, d. h. Sohnknoten besitzen. Ein Mausklick auf eines der beiden Zeichen überführt den daneben stehenden Knoten in den jeweils anderen Kollabierzustand.

Verbindungsaussehen

Hier wird Ihnen das aktuelle Verbindungsaussehen angezeigt. Um es zu ändern, klicken Sie auf die **Bearbeiten**-Schaltfläche. Sie gelangen dann in das Dialogfeld **Linienattribute**, in dem Sie Typ, Dicke und Farbe der Verbindungslinien festlegen können.

Max. Baumhöhe

Die Gesamthöhe eines Baum-Diagramms (in Zeilen) kann begrenzt werden. Wenn Sie dieses Kontrollkästchen aktivieren, können Sie hier die maximale Höhe der Baumstruktur vorgeben. Diese Einstellung wirkt sich nur bei vertikaler Anordnung aus. Falls in einem vertikal angeordneten Ast mehr Ebenen vorhanden sind, erfolgt ein Umbruch, und ein neuer Ast wird am Vaterknoten erzeugt.

Vertikal ab Ebene

Wenn dieses Kontrollkästchen aktiviert ist, werden die Knoten ab der Ebene, die Sie hier angegeben haben, vertikal angeordnet. Damit die Knoten tatsächlich vertikal angeordnet werden, muss die API-Methode **Arrange** aufgerufen werden.

Vertikaler Ebenenabstand (v1)

Hier können Sie den vertikalen Abstand zwischen zwei horizontal angeordneten Knotenebenen in Millimetern festlegen.

Vertikaler Knotenabstand (v2)

Hier können Sie den vertikalen Abstand zwischen zwei vertikal angeordneten Knoten in Millimetern festlegen.

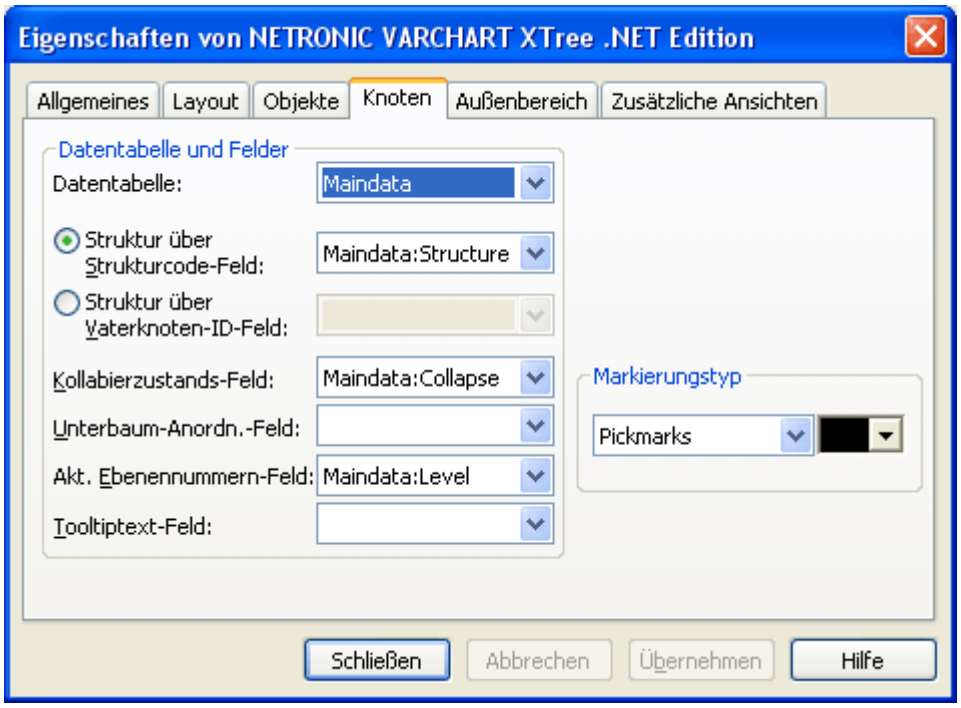
Horizontaler Knotenabstand (h1)

Hier können Sie den horizontalen Abstand zwischen zwei horizontal angeordneten Knoten in Millimetern festlegen.

Horizontale Knoteneinrückung (h2)

Hier können Sie die horizontale Einrückung vertikal angeordneter Knoten in Millimetern festlegen.

4.5 Eigenschaftenseite "Knoten"



Datentabelle

Wählen Sie hier die Datentabelle aus, die für die Darstellung der Knoten herangezogen werden soll.

Diese Option kann auch über die Eigenschaft **VcTree.NodesDataTable-Name** festgelegt werden.

Struktur über Struktur-Code-Feld

Wenn Sie diese Checkbox aktiviert haben, können Sie hier das Datenfeld auswählen, das den Strukturcode bestimmen soll.

Struktur über Vaterknoten-ID-Feld

Wenn Sie diese Checkbox aktiviert haben, können Sie hier das Datenfeld auswählen, das für die ID des Vaterknotens verwendet werden soll.

Kollabierzustands-Feld

Aktivieren Sie dieses Kontrollkästchen, um den Kollabierstatus eines Knotens synchron in dem hier gewählten Datenfeld zu halten. Mögliche

Inhalte des Datenfelds sind "0" (Knoten expandiert) und "1" (Knoten kollabiert).

Unterbaum-Anordn.-Feld

Aktivieren Sie dieses Kontrollkästchen, um die Orientierung eines Teilbaums synchron in einem Datenfeld zu halten. Mögliche Inhalte des Datenfelds sind "0" (Teilbaum horizontal angeordnet) und "1" (Teilbaum vertikal angeordnet). Die horizontale Anordnung ist nur sichtbar, wenn der direkte und alle indirekten Vaterknoten horizontal angeordnet sind.

Akt. Ebenennummern-Feld

Aktivieren Sie dieses Kontrollkästchen, um festzulegen, in welchem Datenfeld die Ebenennummer der Knoten abgelegt wird. Die Ebenennummern zählen von 1 an aufwärts. Diese Eigenschaft kann auch über die VcTree-Eigenschaft **LevelDataFieldIndex** festgelegt werden.

Hinweis: Zur Laufzeit können Sie die Ebene des Knotens nicht verändern indem Sie den Wert des Ebenennummer-Datenfeldes ändern.

Tooltiptext-Feld

Das Datenfeld, das Sie hier auswählen, wird als Tooltip angezeigt, wenn Sie eine VMF-Datei mit dem WebViewer ansehen und dort auf einen Knoten rechtsklicken. Es sind keine weiteren Einstellungen notwendig.

Das VMF- (Viewer Metafile) Format ist ein Vektorformat, in denen ein Diagramm auflösungsunabhängig gespeichert werden kann. Es kann mittels des als Java Applet ausgeführten GRANEDA WebViewers plattformunabhängig mit jedem Java-fähigen Internet-Browser angesehen werden.

Um in Ihrer Applikation in VARCHART XTree Tooltips anzuzeigen, müssen Sie auf der Eigenschaftenseite **Allgemeines** das Kontrollkästchen **VcToolTipTextSupplying-Ereignisse** aktivieren bzw. die Eigenschaft **ToolTipTextSupplyingEventEnabled** auf **True** setzen und im **VcToolTipTextSupplying**-Ereignis programmieren, welche Datenfelder angezeigt werden sollen.

Markierungstyp

Legen Sie hier den Markierungstyp für Knoten fest.

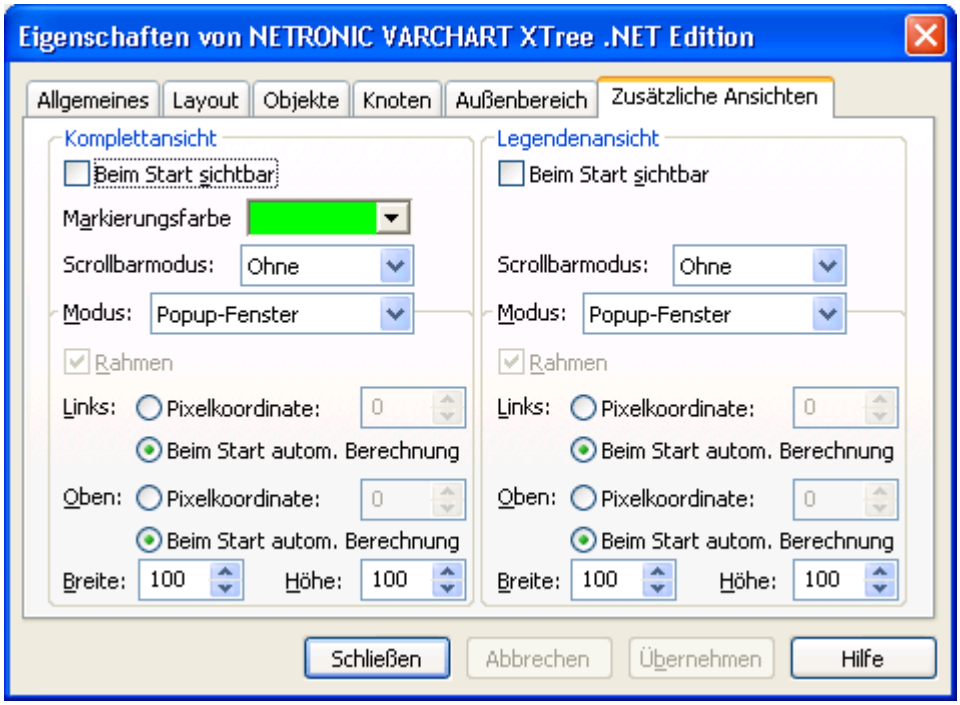
Mögliche Alternativen sind:

- Ohne
- Einrahmen
- Einrahmen innen
- Invertieren
- Pickmarks
- Pickmarks innen

Wenn Sie den Markierungstyp "Ohne" ausgewählt haben, können Knoten nicht markiert werden.

Sie können zudem der gewählten Markierung eine eigene Farbe zuordnen.

4.6 Eigenschaftenseite "Zusätzliche Ansichten"



Auf dieser Eigenschaftenseite können Sie die Eigenschaften der Komplettansicht (World View) sowie der Legendenansicht (Legend View) festlegen. Die Komplettansicht ist ein zusätzliches Fenster, in dem das komplette Diagramm angezeigt wird. Ein Rahmen darin zeigt an, welchen Ausschnitt des Diagramms das Hauptfenster gerade anzeigt.

Mithilfe der Legendenansicht lässt sich, ebenfalls in einem zusätzlichen Fenster, eine Legende auf dem Bildschirm darstellen.

Um die Ansichten anzeigen zu lassen, wählen Sie zur Laufzeit im Standard-Kontextmenü **Komplettansicht anzeigen** bzw. für die Legende **Legendenansicht anzeigen**. Über diese Menüpunkte können die Ansichten auch wieder ausgeschaltet werden (alternativ über die **Schließen**-Schaltfläche in der Titelleiste des jeweiligen Fensters).

im Folgenden sind die möglichen Einstellungen für beide Ansichten gleichzeitig beschrieben. Sollte ein Kriterium nicht für beide Ansichten gelten, so wird gesondert darauf hingewiesen.

Beim Start sichtbar

Aktivieren Sie dieses Kontrollkästchen, damit die Ansicht beim Start des Programms sichtbar ist.

Diese Eigenschaft kann auch über die Aufrufe **VcWorldView.Visible** bzw. **VcLegendView.Visible** der Programmierschnittstelle gesetzt werden.

Markierungsfarbe (nur für Komplettansicht)

Wählen Sie hier die Farbe der Linie des Rechtecks aus, das in der Komplettansicht den aktuell gewählten Ausschnitt anzeigt.

Diese Eigenschaft kann auch über die Aufrufe **VcWorld-View.MarkingColor** bzw. **VcLegendView.MarkingColor** der Programmierschnittstelle gesetzt werden.

Scrollbarmodus

Wählen Sie hier, ob und welche Bildlaufleisten in der Ansicht dargestellt werden sollen. Durch die Verwendung von Bildlaufleisten werden Leerbereiche vermieden und das Diagramm bzw. die Legende ist besser zu erkennen, weil es bzw. sie größer dargestellt wird. Folgende Möglichkeiten stehen zur Verfügung:

- **Ohne:** In der Ansicht wird immer alles vollständig dargestellt. Dadurch können Leerbereiche entstehen, wenn die Ansicht in ihren Proportionen nicht denen des Charts(oder der Legende) entspricht.
- **Horizontal:** Es wird, wenn notwendig, eine horizontale Bildlaufleiste dargestellt.
- **Vertikal:** Es wird, wenn notwendig, eine vertikale Bildlaufleiste dargestellt.
- **Automatisch:** Es wird, wenn notwendig, ein horizontaler oder eine vertikale Bildlaufleiste dargestellt.

Diese Eigenschaft kann auch über die Aufrufe **VcWorld-View.ScrollBarMode** bzw. **VcLegendView.ScrollBarMode** der Programmierschnittstelle gesetzt werden.

Modus

Wählen Sie hier den Modus für die Gesamtansicht aus. Es gibt folgende Möglichkeiten:

- **fest an linker Seite:** Die Ansicht wird links im Fenster des Steuerelementes angezeigt. Dann kann nur die Breite festgelegt werden, während Position und Höhe vorgegeben sind.

- **fest an rechter Seite:** Die Ansicht wird rechts im Fenster des Steuerelementes angezeigt. Dann kann nur die Breite festgelegt werden, während Position und Höhe vorgegeben sind.
- **fest an oberer Seite:** Die Ansicht wird oben im Fenster des Steuerelementes angezeigt. Dann kann nur die Höhe festgelegt werden, während Position und Breite vorgegeben sind.
- **fest an unterer Seite:** Die Ansicht wird unten im Fenster des Steuerelementes angezeigt.
- **nicht fest positioniert:** Die Ansicht ist ein untergeordnetes Kindfenster des aktuellen Vaterfensters des Steuerelementes und kann an beliebiger Position mit beliebiger Ausdehnung angeordnet werden. Das Vaterfenster kann bei Bedarf über die Eigenschaft **VcWorldView.ParentHWnd** geändert werden.
- **Popup-Fenster:** Die Ansicht ist ein Popup-Fenster, das einen eigenen Rahmen besitzt und vom Benutzer in Position und Größe verändert werden kann. Es kann über das Standard-Kontextmenü ein- bzw. ausgeschaltet oder über die **Schließen**-Schaltfläche in der Titelleiste ausgeschaltet werden.

Diese Eigenschaft kann auch über die Aufrufe **VcWorldView.Mode** bzw. **VcLegendView.Mode** der Programmierschnittstelle gesetzt werden.

Rahmen

*Nicht aktiviert, wenn der Modus **Popup-Fenster** gewählt wurde.* Aktivieren Sie dieses Kontrollkästchen, wenn die Ansicht einen Rahmen erhalten soll. Die Rahmenfarbe kann aus der Drop-Down-Liste gewählt werden.

Diese OPpionen können auch über die Aufrufe **VcWorldView.Border** und **VcWorldView.Border.Color** bzw. **VcLegendView.Border** und **VcLegend-View.Border.Color**der Programmierschnittstelle gesetzt werden.

Links

*Nur aktiviert, wenn der Modus **nicht fest positioniert** oder **Popup-Fenster** gewählt wurde.* Legen Sie hier die linke Position der Ansicht fest. Dabei gibt es zwei Möglichkeiten:

1. Geben Sie unter **Pixelkoordinate** einen Wert an. Dabei handelt es sich um Gerätekoordinaten.
2. Wählen Sie die Option **Beim Start automat. Berechnung**, damit die Position der Ansicht beim Programmstart automatisch berechnet wird.

Diese Eigenschaft kann auch über die Aufrufe **VcWorldView.Left** bzw. **VcLegendView.Left** der Programmierschnittstelle gesetzt werden.

Oben

*Nur aktiviert, wenn der Modus **nicht fest positioniert** oder **Popup-Fenster** gewählt wurde.* Legen Sie hier die obere Position der Ansicht fest. Dabei gibt es zwei Möglichkeiten:

1. Geben Sie unter **Pixelkoordinate** einen Wert an. Dabei handelt es sich um Gerätekoordinaten.
2. Wählen Sie die Option **Beim Start automat. Berechnung**, damit die Position der Ansicht beim Programmstart automatisch berechnet wird.

Diese Eigenschaft kann auch über die Aufrufe **VcWorldView.Left** bzw. **VcLegendView.Left** der Programmierschnittstelle gesetzt werden.

Breite

*Nicht aktiviert, wenn der Modus **fest an oberer/unterer Seite** gewählt wurde.* Legen Sie hier die horizontale Ausdehnung der Ansicht fest. Der Wert wird in Pixelkoordinaten (Gerätekoordinaten) angegeben.

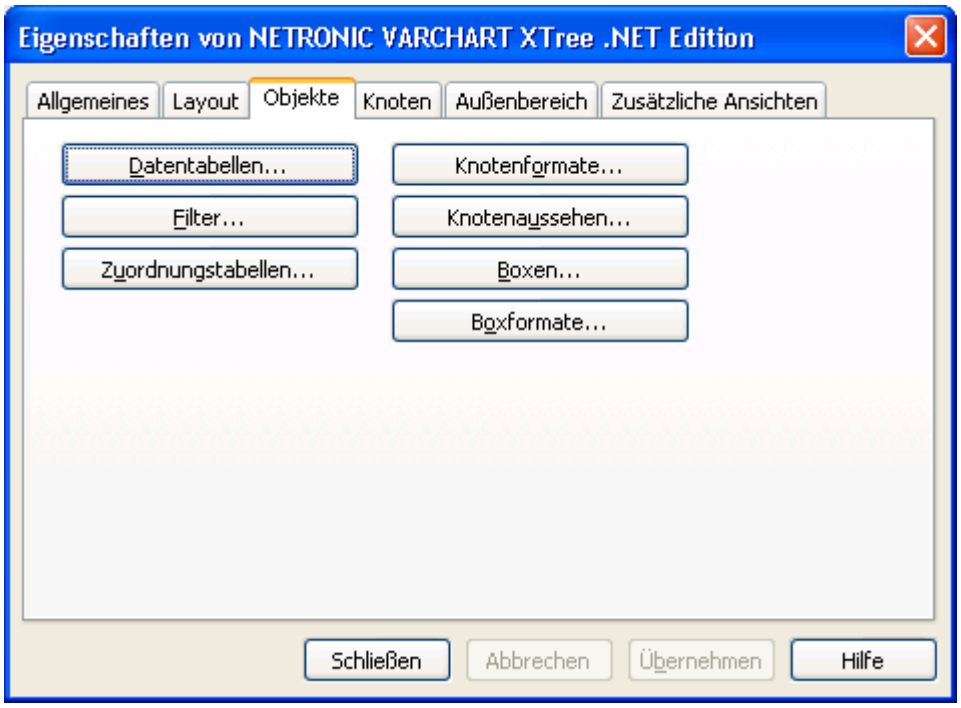
Diese Eigenschaft kann auch über die Aufrufe **VcWorldView.Width** bzw. **VcLegendView.Width** der Programmierschnittstelle gesetzt werden.

Höhe

*Nicht aktiviert, wenn der Modus **fest an linker/rechter Seite** gewählt wurde.* Legen Sie hier die vertikale Ausdehnung der Ansicht fest. Der Wert wird in Pixelkoordinaten (Gerätekoordinaten) angegeben.

Diese Eigenschaft kann auch über die Aufrufe **VcWorldView.Height** bzw. **VcLegendView.Height** der Programmierschnittstelle gesetzt werden.

4.7 Eigenschaftenseite "Objekte"



p

Datentabellen

Über diese Schaltfläche öffnen Sie das Dialogfeld **Datentabellen verwalten**.

Filter

Über diese Schaltfläche öffnen Sie das Dialogfeld **Filter verwalten**.

Zuordnungstabellen

Über diese Schaltfläche öffnen Sie das Dialogfeld **Zuordnungstabellen verwalten**.

Knotenformate

Über diese Schaltfläche öffnen Sie das Dialogfeld **Knotenformate verwalten**.

Knotenaussehen

Über diese Schaltfläche öffnen Sie das Dialogfeld **Knotenaussehen verwalten**.

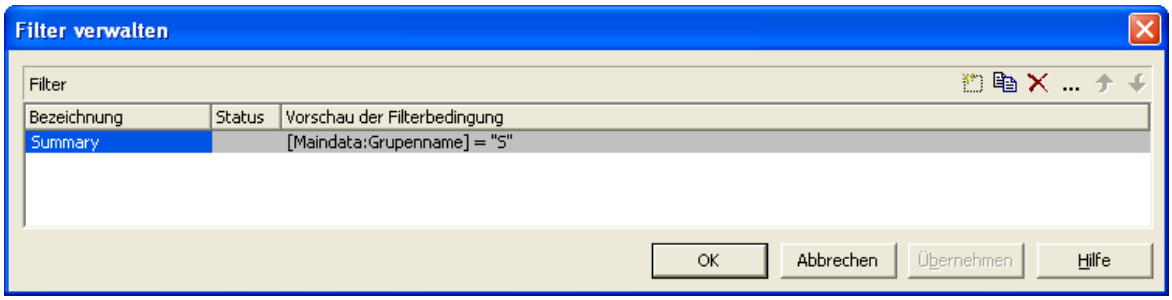
Boxen

Über diese Schaltfläche öffnen Sie das Dialogfeld **Boxen verwalten**.

Boxformate

Über diese Schaltfläche öffnen Sie das Dialogfeld **Boxformate verwalten**.

4.8 Dialogfeld "Filter verwalten"



Sie gelangen in dieses Dialogfeld über die Eigenschaftenseite **Objekte**.

Bezeichnung

In dieser Spalte stehen die Namen aller vorhandenen Filter. Die Namen sind editierbar.

Status

In der Spalte **Status** wird jeder Filter gekennzeichnet, der seit dem Aufruf des Dialogs hinzugefügt ( ) und/oder geändert ( ) worden ist.

Datendefinitionstabelle

Hier wird die Datendefinitionstabelle angezeigt, die dem jeweiligen Filter zu Grunde liegt. Diese Spalte wird nur angezeigt, wenn auf der Eigenschaftenseite **Allgemeines** die Option **Erweiterte Datentabellen zulassen** nicht ausgewählt ist.

Vorschau der Filterbedingung

In dieser Spalte wird die Bedingung jedes Filters angezeigt. Sie kann hier nicht editiert werden. Um die Filterbedingung zu bearbeiten, klicken Sie auf die **Filter bearbeiten**-Schaltfläche.

Filter hinzufügen

Ein neuer Filter mit einem Standardnamen wird angelegt. Diesen Namen können Sie editieren, indem Sie darauf doppelklicken und ihn dann verändern.

Neue Filter werden kontextabhängig angelegt, d. h. es wird immer die jeweils passende Datendefinitionstabelle (**Maindata** bzw. **Relations**) verwendet.

Filter kopieren

Der markierte Filter wird kopiert.

Filter löschen

Der Filter, den Sie in der Liste markiert haben, wird gelöscht. Es können nur Filter gelöscht werden, die zur Zeit nicht benutzt werden.

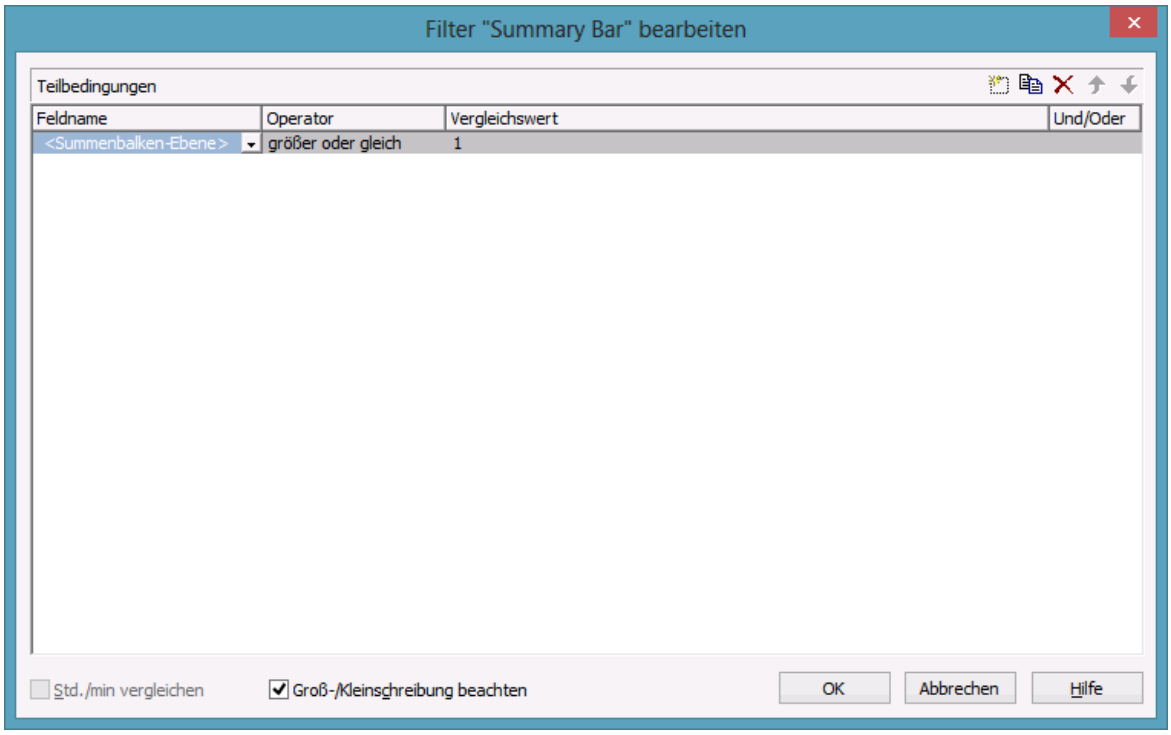
Filter bearbeiten

Um die Bedingungen eines Filters anzusehen oder zu ändern, klicken Sie auf die Schaltfläche **Filter bearbeiten**. Es erscheint das Dialogfeld **Filter bearbeiten**. Nun können Sie die für den Filter formulierten Bedingungen bearbeiten.

Filter eine Zeile nach oben/unten

Mit Hilfe dieser Schaltflächen können Sie den markierten Filter eine Zeile nach oben/unten schieben.

4.9 Dialogfeld "Filter bearbeiten"



Sie erreichen dieses Dialogfeld in dem Sie entweder

- von der Eigenschaftenseite **Objekte**
- aus dem Dialog **Knotenaussehen verwalten oder**
- aus dem Dialog **Verbindungsaussehen verwalten**

den **Filter verwalten**-Dialog aktivieren und anschließend die Schaltfläche **Filter bearbeiten** anklicken. In der Kopfzeile dieses Dialogfeldes steht der Name des aktuellen Filters.

Teilbedingung hinzufügen

Vor der markierten Zeile wird eine neue Zeile für eine Teilbedingung eingefügt.

Teilbedingung kopieren

Die markierte Teilbedingung wird kopiert.

Teilbedingung löschen

Die markierte Teilbedingung wird gelöscht.

Teilbedingung früher/später abarbeiten

Wenn ein Filter mehrere Teilbedingungen enthält, werden die einzelnen Teilbedingungen in der Reihenfolge, in der sie in der **Teilbedingungen**-Tabelle stehen, abgearbeitet.

Klicken Sie die Schaltfläche **Teilbedingung früher/später abarbeiten** an, um eine markierte Teilbedingung in der Tabelle eine Position höher/tiefer zu setzen, damit sie früher/später abgearbeitet wird.

Feldname

Hier können Sie das Datenfeld auswählen, das mit dem Vergleichswert verglichen werden soll. Die Liste enthält alle verfügbaren Datenfelder.

Operator

Wählen Sie hier den Vergleichsoperator für den Vergleich des unter **Feldname** gewählten Datenfeldes mit dem unter **Vergleichswert** angegebenen Wertes bzw. Datenfeldes aus.

Vergleichswert

Hier wird der aktuelle Vergleichswert angezeigt. Im Feld **Vergleichswert** werden in eckigen Klammern alle Felder des passenden Datentyps angeboten, die jeweils als Vergleichswert eingesetzt werden können. Wurde unter **Feldname** beispielsweise das Feld "Frühester Anfang" eingetragen, werden als Vergleichswerte nur Terminfelder (z. B. "Frühestes Ende") und die Optionen <heute> und <Eingabe> angeboten.

Mit Hilfe des Vergleichswertes <Eingabe> können Sie einen variablen Filter definieren. In variablen Filtern sind nur der Feldname und der Operator, aber nicht der Vergleichswert definiert. Diesen können Sie bei Bedarf angeben. Sie können einen variablen Filter verwenden, wenn Sie ein Projekt öffnen und die darzustellenden Vorgänge begrenzen möchten.

Termine werden in dem Format, das auf der Eigenschaftenseite **Allgemeines** unter **Datumsausgabeformat** festgelegt wurde, eingegeben. Wenn Sie unter **Feldname** ein Terminfeld gewählt haben, erscheinen im Feld **Vergleichswert** zwei Pfeil-Schaltflächen, sobald Sie dieses Feld anklicken. Über die erste Pfeil-Schaltfläche öffnen Sie eine Kombobox, die Ihnen alle verfügbaren Termin-Datenfelder anzeigt. Über die zweite Pfeil-Schaltfläche öffnen Sie den Datumsdialog, aus dem Sie das gewünschte Datum per Mausklick auswählen können. Sie können das Datum aber auch direkt editieren.

Zahlenwerte oder Texte müssen direkt eingegeben werden.

Bei den Operatoren "gleich" und "ungleich" können Sie für Textfelder auch Wildcards verwenden:

*: kein oder mehrere beliebige Zeichen

?: genau ein beliebiges Zeichen

Wenn Sie die Zeichen * oder ? nicht als Wildcards verwenden, sondern auf diese Zeichen abfragen wollen, müssen Sie dies durch einen vorangestellten Backslash kenntlich machen:

*: *

\?: ?

Wenn dem Backslash kein * oder ? folgt, wird nach dem Vorhandensein von \ gefragt.

Beispiele:

Vorgang 1 : Name = "Baugenehmigung"

Vorgang 2 : Name = "*Baugenehmigung"

Mögliche Filter für Vorgang 1:

[Name] = B*

[Name] = B?ugenehmigung

Mögliche Filter für Vorgang 2:

[Name] = *B*

[Name] = **

[Name] = ?B*

Und/Oder

Hier wird die logische Verknüpfung der aktuellen mit der jeweils nächsten Teilbedingung in der Tabelle angezeigt.

Wenn Sie den UND-Operator wählen, werden nur die Objekte ausgewählt, die beide Teilbedingungen erfüllen. Wählen Sie den ODER-Operator, werden die Objekte ausgewählt, die mindestens eine der verknüpften Teilbedingungen erfüllen.

Haben Sie mehrere Teilbedingungen formuliert und diese zum Teil mit UND, zum Teil mit ODER verknüpft, werden erst die UND-Verknüpfungen abgearbeitet. (UND bindet stärker als ODER.)

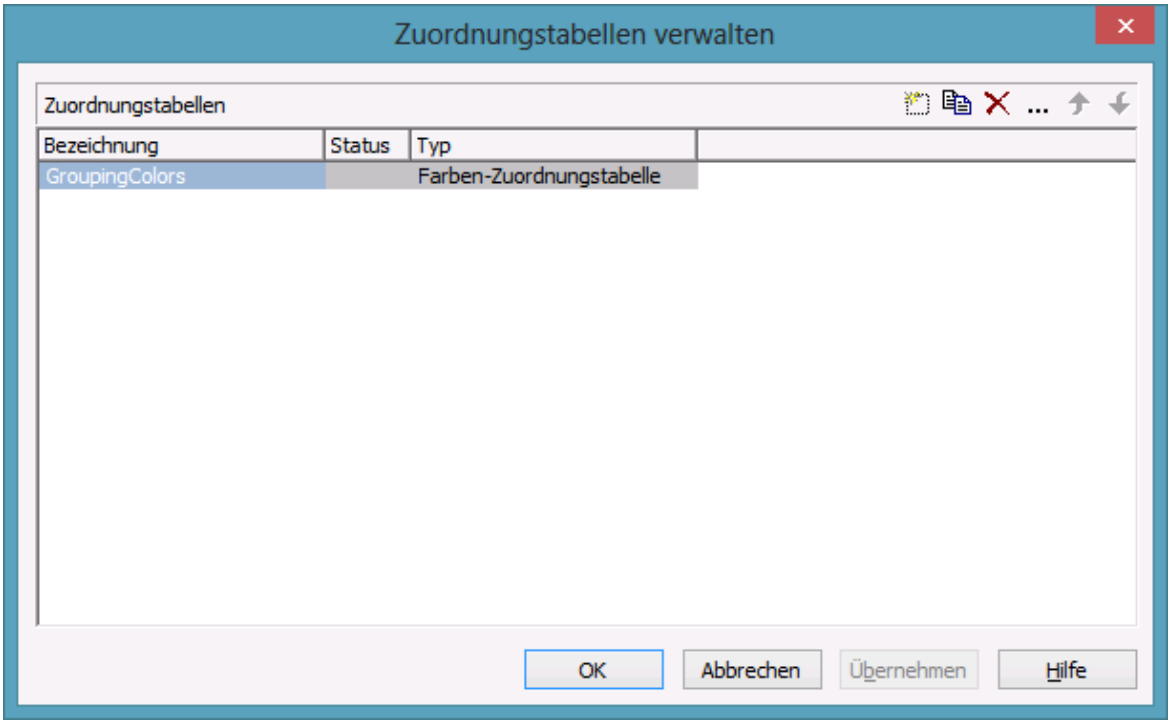
Std./min vergleichen

Aktivieren Sie dieses Kontrollkästchen, wenn beim Datumsvergleich auch Stunde und Minute berücksichtigt werden sollen.

Groß-/Kleinschreibung beachten

Aktivieren Sie dieses Kontrollkästchen, wenn beim Vergleich die Groß-/Kleinschreibung beachtet werden soll.

4.10 Dialogfeld "Zuordnungstabellen verwalten"



Sie gelangen über die Eigenschaftenseite **Objekte** in dieses Dialogfeld oder indem Sie im Dialogfeld **Zuordnung einstellen** auf die Schaltfläche **Zuordnungstabellen** klicken.

Bezeichnung

In dieser Spalte stehen die Namen aller vorhandenen Zuordnungstabellen. Die Namen sind editierbar.

Status

In der Spalte **Status** wird jede Zuordnungstabelle gekennzeichnet, die seit dem Aufruf des Dialogs hinzugefügt ( ) und/oder geändert ( ) worden ist.

Typ

Wählen Sie hier den Typ der Zuordnungstabelle aus:

- Farben-Zuordnungstabelle
- Schraffuren-Zuordnungstabelle (für spätere Anwendungen)

- Grafikdateien-Zuordnungstabelle

Zuordnungstabelle hinzufügen

Eine neue Zuordnungstabelle mit einem Standardnamen wird angelegt. Diesen Namen können Sie editieren, indem Sie darauf doppelklicken und ihn dann verändern.

Zuordnungstabelle kopieren

Die markierte Zuordnungstabelle wird kopiert.

Zuordnungstabelle löschen

Die Zuordnungstabelle, die Sie in der Liste markiert haben, wird gelöscht. Es können nur die Zuordnungstabellen gelöscht werden, die zur Zeit nicht benutzt werden.

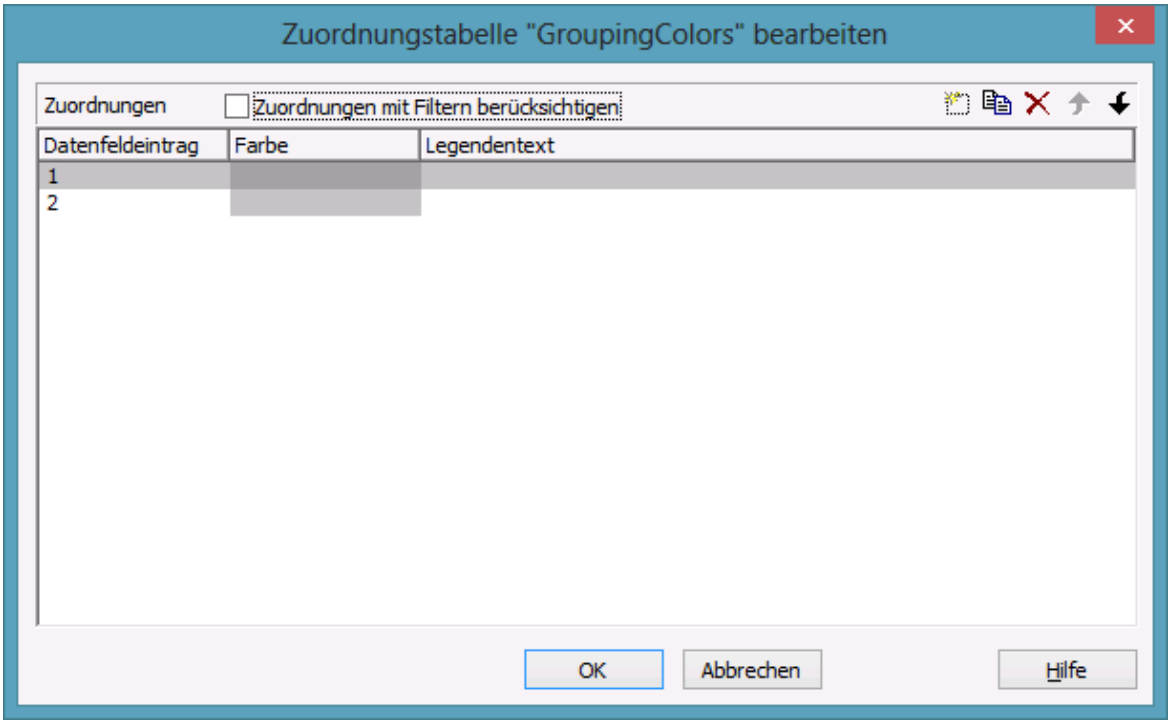
Zuordnungstabelle bearbeiten

Sie gelangen in das Dialogfeld **Zuordnungstabelle bearbeiten**.

Zuordnungstabelle eine Zeile nach oben/unten

Mit Hilfe dieser Schaltfläche können Sie die markierte Zuordnungstabelle in der Tabelle eine Zeile nach oben/unten schieben.

4.11 Dialogfeld "Zuordnungstabelle bearbeiten"



Sie gelangen in dieses Dialogfeld, indem Sie im Dialogfeld **Zuordnungstabellen verwalten** auf die Schaltfläche **Zuordnungstabelle bearbeiten** (...) klicken.

Sie können in einer Zuordnungstabelle maximal 150 Zuordnungen festlegen. Falls Sie noch mehr Zuordnungen benötigen, erstellen Sie einfach eine neue Zuordnungstabelle, z. B. als Kopie der bereits vorhandenen.

Zuordnungen mit Filtern berücksichtigen

Wenn die Option **Zuordnungen mit Filtern berücksichtigen** ausgewählt ist, werden nicht nur die in der Liste der Datenfeldeinträge angegebenen festen Werte als Schlüsselwerte berücksichtigt, sondern auch Filter, die aus der Dropdown-Liste ausgewählt werden können. Dadurch hängen die Ausführungswerte nicht mehr nur von einem konkreten Wert ab, sondern von einem Wertebereich.

Datenfeldeintrag

In dieser Spalte werden die Einträge des gewählten Datenfeldes aufgeführt, für die Sie eine Farbe bzw. eine Grafikdatei und den Legendentext vereinbaren können.

Farbe/Grafikdateiname

Um die Farbe bzw. die Grafikdatei für einen Datenfeldeintrag festzulegen, klicken Sie auf das entsprechende Feld. Dann erscheint eine Schaltfläche, über die Sie in den Dialog zur Auswahl der Farbe bzw. der Grafikdatei gelangen.

Wird ein relativer Dateiname angegeben, so wird die Datei zur Laufzeit zuerst in dem Verzeichnis gesucht, das in der VARCHART-ActiveX-Eigenschaft **FilePath** gesetzt ist. Wird sie dort nicht gefunden, wird sie zuerst im gerade aktiven Arbeitsverzeichnis der Applikation und dann im Installationsverzeichnis des Steuerelements gesucht.

Legendentext

(nur für Farb-Zuordnungstabellen) Hier können Sie für jeden Datenfeldeintrag einen Legendentext festlegen.

Zuordnung hinzufügen

Eine neue Zuordnung mit einem Standardnamen wird angelegt. Diesen Namen können Sie editieren, indem Sie darauf doppelklicken und ihn dann verändern.

Zuordnung kopieren

Die markierte Zuordnung wird kopiert.

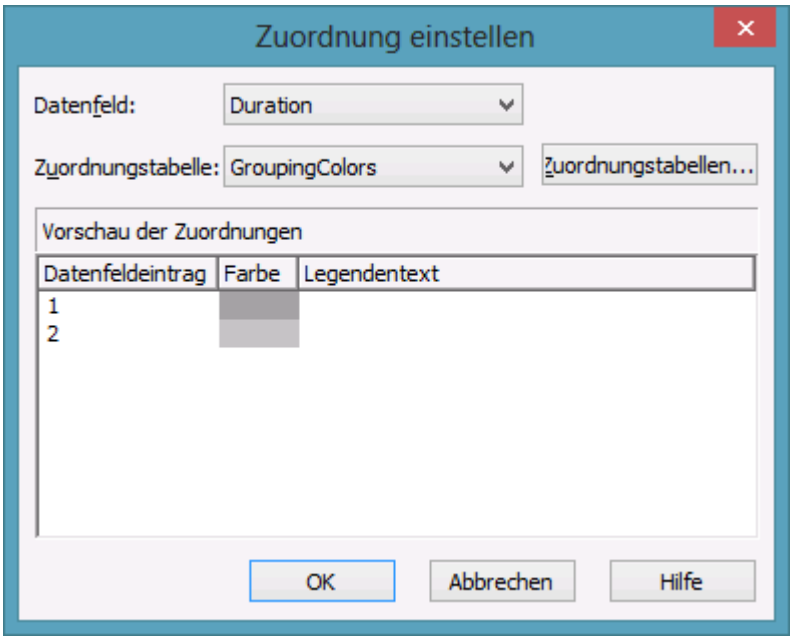
Zuordnung löschen

Die Zuordnung, die Sie in der Liste markiert haben, wird gelöscht. Es können nur die Zuordnungen gelöscht werden, die zur Zeit nicht benutzt werden.

Zuordnung eine Zeile nach oben/unten

Die markierte Zuordnung wird in der Tabelle eine Zeile nach oben/unten geschoben.

4.12 Dialogfeld "Zuordnung einstellen"



Verbinden Sie in diesem Dialogfeld das Datenfeld eines Knotens mit einer Zuordnungstabelle. Sie erreichen es über verschiedene Dialoge z.B. den Dialog **Layer bearbeiten**. Klicken Sie dort beim gewünschten Attribut auf die Schaltfläche .

Datenfeld

Wählen Sie hier das Datenfeld aus, von dessen Einträgen die gewünschten Attribute des zu bearbeitenden Objekts abhängen soll.

Zuordnungstabelle

(nur aktiv, wenn ein Datenfeld gewählt wurde) Wählen Sie hier die Zuordnungstabelle aus, die den einzelnen Datenfeldeinträgen eine Farbe bzw. eine Grafikdatei und einen Legendentext zuordnet.

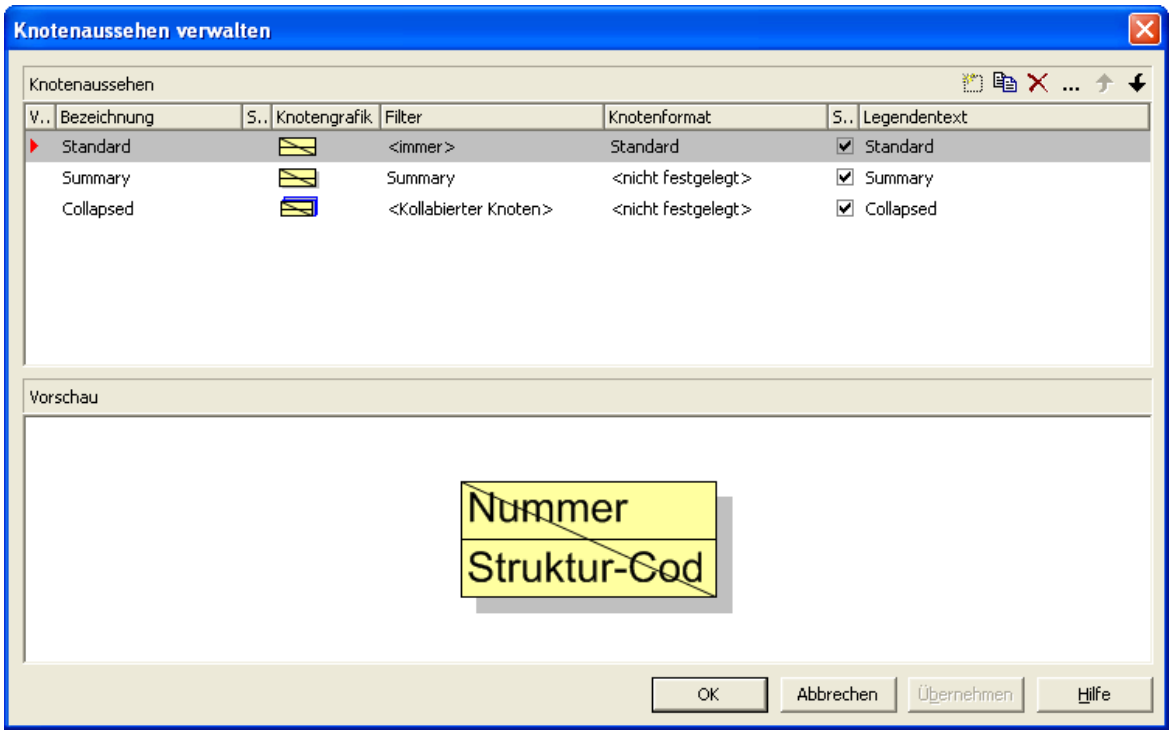
Zuordnungstabellen

Klicken Sie auf diese Schaltfläche, um das Dialogfeld **Zuordnungstabellen verwalten** aufzurufen. Hier können Sie Zuordnungstabellen erstellen, bearbeiten, kopieren oder löschen.

Vorschau der Zuordnungen

Hier wird die gewählte Zuordnungstabelle dargestellt: alle Datenfelder, die diesen zugeordneten Farben und Legendentexte bzw. die Grafikdateinamen.

4.13 Dialogfeld "Knotenaussehen verwalten"



Sie gelangen über die Eigenschaftenseite **Objekte** in dieses Dialogfeld.

Die Darstellung von Knoten wird festgelegt, indem diesen ein oder mehrere Knotenaussehen dynamisch über Filter zugewiesen werden. Aus der Überlagerung der Attribute aller Knotenaussehen, die auf einen Knoten zutreffen, ergibt sich das gesamte Knotenaussehen.

Vorschau

Alle Knotenaussehen, die in der **Vorschau**-Spalte durch eine kleine rote Pfeilspitze markiert sind, werden im Vorschaufenster überlagert und mit den durch die Abarbeitungsreihenfolge bestimmten Überlagerungen angezeigt.

Das Knotenaussehen, auf dem der Cursor gerade steht, wird durch eine grüne Pfeilspitze gekennzeichnet.

Bezeichnung

Diese Spalte enthält die Namen aller vorhandenen Knotenaussehen. Die Namen sind editierbar.

Status

In dieser Spalte wird jedes Knotenaussehen, das seit dem Aufruf des Dialogs hinzugefügt ( ) und/oder geändert ( ) worden ist.

Knotengrafik

Hier wird jedes Knotenaussehen dargestellt. Um die Knotengrafik, d. h. die grafischen Attribute des Knotenaussehens, zu verändern, klicken Sie auf die **Knotenaussehen bearbeiten**-Schaltfläche oberhalb der Tabelle oder doppelklicken Sie auf den Eintrag **Knotengrafik**. Sie gelangen dann in den Dialog **Knotenaussehen bearbeiten**.

Filter

Der Filter, der mit einem Knotenaussehen verbunden ist, bestimmt, welche Knoten mit dem betreffenden Knotenaussehen versehen werden.

Sie können für die meisten Knotenaussehen den Filter auswählen. Nur für die Knotenaussehen "Standard" und "Kollabierter Knoten" ist der Filter festgelegt ("<immer>" bzw. "<Kollabierter Knoten>").

Um einem Knotenaussehen einen Filter zuzuweisen, markieren Sie das **Filter**-Feld. Dann erscheinen eine Schaltfläche für eine Kombobox mit allen verfügbaren Filtern sowie eine **Bearbeiten**-Schaltfläche. Wählen Sie aus der Kombobox einen Filter für das Knotenaussehen aus oder klicken Sie auf die **Bearbeiten**-Schaltfläche des **Filter**-Feldes, um den Dialog **Filter verwalten** aufzurufen. Dort können Sie Filter bearbeiten, kopieren, neu definieren oder löschen.

Knotenformat

Ein Knotenformat definiert die Anzahl, die Anordnung und das Aussehen der Felder, die zur Beschriftung eines Knotens verwendet werden. Wählen Sie hier das Knotenformat für das Knotenaussehen aus. Markieren Sie dazu das **Knotenformat**-Feld. Dann erscheinen eine Schaltfläche für eine Kombobox mit allen verfügbaren Knotenformaten sowie eine **Bearbeiten**-Schaltfläche. Wählen Sie aus der Kombobox eine Knotenformat für die Knotenaussehen aus. Oder klicken Sie auf die **Bearbeiten**-Schaltfläche des **Knotenformat**-Feldes, um den Dialog **Knotenformate verwalten** aufzurufen. Dort können Sie Knotenformate bearbeiten, kopieren, hinzufügen oder löschen.

Sichtbar in Legende

Aktivieren Sie dieses Kontrollkästchen für alle Knotenaussehen, die in der Legende dargestellt werden sollen.

Legendentext

In dieser Spalte können Sie für jedes Knotenaussehen einen Legendentext eintragen.

Knotenaussehen hinzufügen

Ein neues Knotenaussehen wird am Ende der Liste hinzugefügt.

Knotenaussehen kopieren

Das markierte Knotenaussehen wird kopiert.

Knotenaussehen löschen

Knotenaussehen, die Sie nicht mehr verwenden, können Sie über diese Schaltfläche löschen. Es erfolgt noch eine Abfrage, ob Sie das markierte Knotenaussehen wirklich löschen wollen. Das Standard-Knotenaussehen kann nicht gelöscht werden.

Knotenaussehen bearbeiten

Über diese Schaltfläche gelangen Sie in das Dialogfeld **Knotenaussehen bearbeiten**.

Knotenaussehen früher/später abarbeiten

Wenn einem Knoten mehrere Knotenaussehen zugewiesen sind, werden die einzelnen Knotenaussehen nacheinander abgearbeitet. In der Tabelle sind die Knotenaussehen nach ihrer Abarbeitungsreihenfolge sortiert. Das Standard-Knotenaussehen steht immer ganz oben in der Tabelle, da es immer zugewiesen ist und immer zuerst abgearbeitet wird. Das Knotenaussehen, das zuletzt abgearbeitet wird, steht ganz unten in der Tabelle.

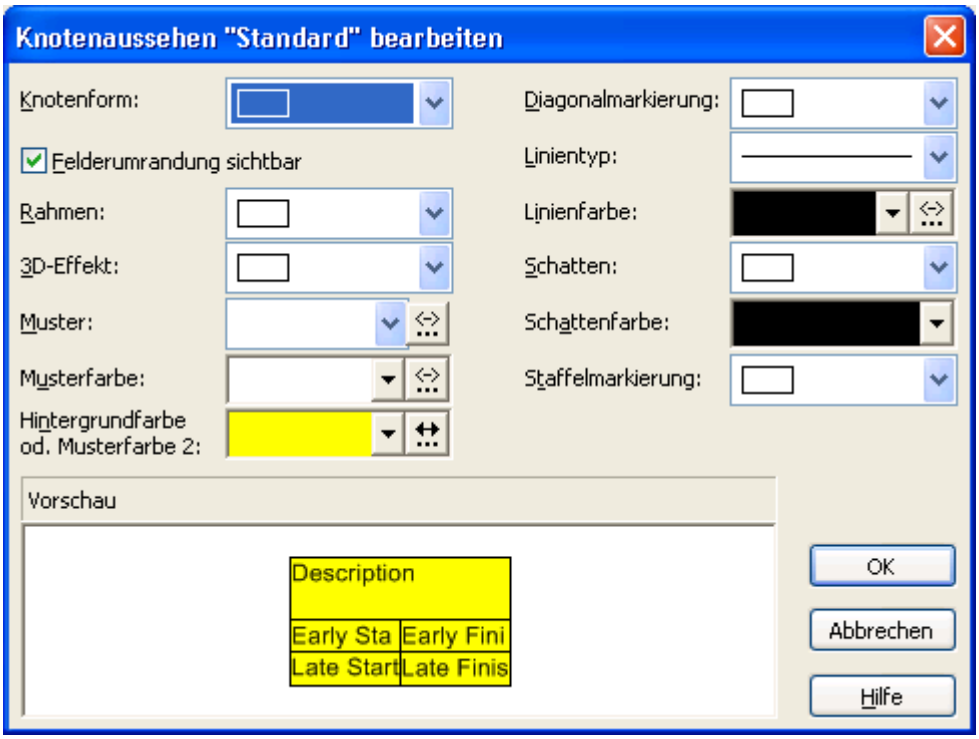
Wenn mehrere Knotenaussehen auf einen Knoten zutreffen, werden die Attribute jedes Knotenaussehens durch die Attribute aller Knotenaussehen, die später abgearbeitet werden, ersetzt. Nur die Attribute, deren Wert "nicht festgelegt" ist, ersetzen nicht die Attribute ihrer Vorgänger.

Mit Hilfe dieser Schaltflächen können Sie die Position eines Knotenaussehens in der Abarbeitungsreihenfolge verändern:

Wenn Sie die Schaltfläche **Knotenaussehen früher abarbeiten** anklicken, steigt das markierte Knotenaussehen um eine Position in der Tabelle und wird entsprechend früher abgearbeitet.

Wenn Sie die Schaltfläche **Knotenaussehen später abarbeiten** anklicken, fällt das markierte Knotenaussehen um eine Position in der Tabelle und wird entsprechend später abgearbeitet.

4.14 Dialogfeld "Knotenaussehen bearbeiten"



In der Titelzeile wird der Name des Knotenaussehens angezeigt, das Sie in diesem Dialogfeld bearbeiten können.

Wenn einem Knoten mehrere Knotenaussehen zugewiesen sind, werden die Attribute jedes Knotenaussehens durch die Attribute aller Knotenaussehen, die eine höhere Priorität haben, ersetzt. Nur die Attribute, deren Wert <nicht festgelegt> ist, ersetzen nicht die Attribute ihrer Vorgänger.

Knotenform

Wählen Sie hier die Knotenform für das aktuelle Knotenaussehen aus. Zur Auswahl stehen verschiedene Knotenformen sowie die Einträge <nicht festgelegt> und <ohne Rahmen>.

Felderumrandung sichtbar

Mit dieser Eigenschaft kann festgelegt werden, ob der Rahmen um die innenliegenden Felder sichtbar ist oder nicht. Die Außenrandlinie der Form ist davon nicht betroffen, daher wirkt sich diese Eigenschaft bei den möglichen Rahmenformen unterschiedlich aus und hat z.B. beim Typ **vcRectangle** keine Auswirkung.

Diese Option kann auch über die Eigenschaft **VcNodeAppearance.FrameAroundFieldsVisible** gesetzt werden.

Rahmen

Über dieses Feld legen Sie fest, ob die Knoten einen einfachen Rahmen oder einen Doppelrahmen erhalten.

3D-Effekt

Über dieses Feld bestimmen Sie, ob die Knoten einen 3D-Effekt erhalten oder nicht.

Muster

Wählen Sie hier ein Muster für das aktuell ausgewählte Knotenaussehen aus.

Sie können über die Pfeil-Schaltfläche die Farbzuordnungstabelle öffnen, um daraus eine Farbe auszuwählen. Auch transparente Farben stehen zur Verfügung.

Über die zweite Schaltfläche gelangen Sie in den Dialog **Zuordnung einstellen**. Hier können Sie eine datenabhängige Zuordnung der Hintergrundfarbe vereinbaren.

Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der Schaltfläche gefüllt dargestellt.

Musterfarbe

Wählen Sie hier die Musterfarbe für die Knoten aus.

Sie können über die Pfeil-Schaltfläche die Farbzuordnungstabelle öffnen, um daraus eine Farbe auszuwählen.

Über die zweite Schaltfläche gelangen Sie in den Dialog **Zuordnung einstellen**. Hier können Sie eine datenabhängige Zuordnung der Linienfarbe vereinbaren.

Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der Schaltfläche gefüllt dargestellt.

Hintergrundfarbe od. Musterfarbe 2

Wählen Sie hier die Hintergrundfarbe für das aktuell ausgewählte Knotenaussehen aus.

Sie können über die Pfeil-Schaltfläche die Farbzuordnungstabelle öffnen, um daraus eine Farbe auszuwählen. Auch transparente Farben stehen zur Verfügung.

Über die zweite Schaltfläche gelangen Sie in den Dialog **Zuordnung einstellen**. Hier können Sie eine datenabhängige Zuordnung der Hintergrundfarbe vereinbaren.

Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der Schaltfläche gefüllt dargestellt.

Diagonalmarkierung

Über dieses Feld bestimmen Sie, ob die Knoten eine Diagonalmarkierung erhalten und ggf. welche.

Linientyp

Wählen Sie hier die Linienart für die Umrandung der Knoten.

Linienfarbe

Wählen Sie hier die Linienfarbe für die Umrandung der Knoten aus.

Sie können über die Pfeil-Schaltfläche die Farbzuordnungstabelle öffnen, um daraus eine Farbe auszuwählen.

Über die zweite Schaltfläche gelangen Sie in den Dialog **Zuordnung einstellen**. Hier können Sie eine datenabhängige Zuordnung der Linienfarbe vereinbaren.

Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der Schaltfläche gefüllt dargestellt.

Schatten

Über dieses Feld steuern Sie, ob die Knoten einen Schatten erhalten oder nicht.

Schattenfarbe

Legen Sie hier die Farbe fest, die ggf. der Schatten bzw. die Staffelmarkierung erhalten soll.

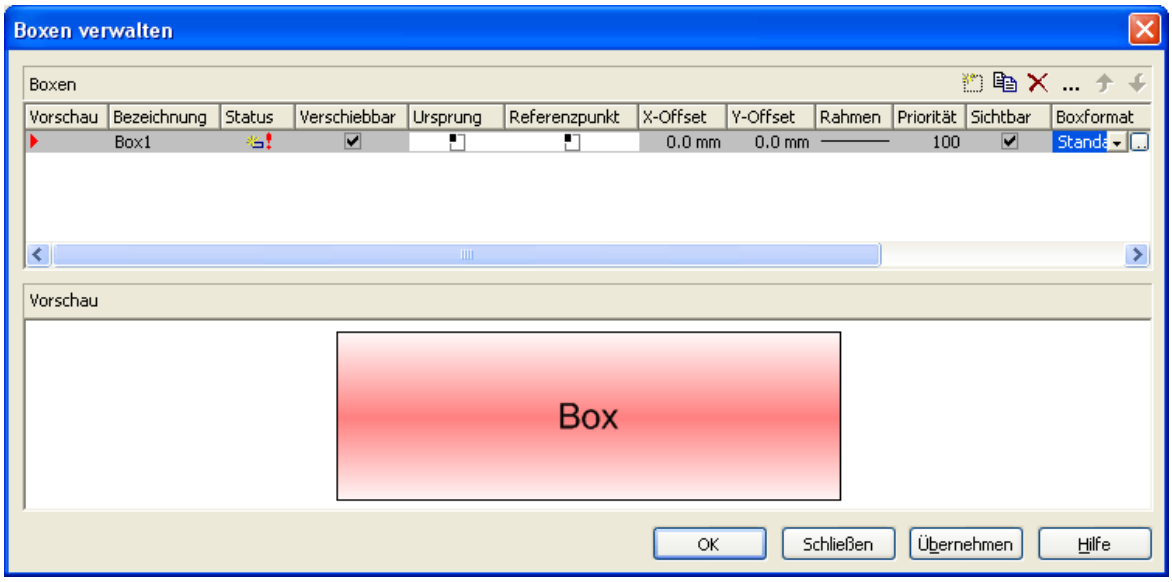
Staffelmarkierung

Über dieses Feld bestimmen Sie, ob die Knoten gestaffelt dargestellt werden sollen (max. 8fach) oder nicht.

Vorschau

In dem Fenster wird das Erscheinungsbild des aktuell festgelegten Knotenaussehens dargestellt.

4.15 Dialogfeld "Boxen verwalten"



Sie erreichen dieses Dialogfeld über die Eigenschaftenseite **Objekte**. Im Diagrammbereich können beliebig viele Boxen dargestellt werden, die Sie in diesem Dialog verwalten können.

Vorschau

Die in dieser Spalte markierte Box wird im Vorschaufenster angezeigt.

Bezeichnung

In dieser Spalte stehen die Namen aller vorhandenen Boxen. Die Namen sind editierbar.

Status

In der Spalte **Status** wird jede Box gekennzeichnet, die seit dem Aufruf des Dialogs hinzugefügt ( ) und/oder geändert ( ) worden ist.

Aktualisierungsverhalten

Wählen Sie hier das gewünschte Aktualisierungsverhalten aus. Bei der Einstellung <nicht festgelegt>, gilt das Aktualisierungsverhalten, das im Dialog **Aktualisierungsverhalten bearbeiten** für Boxen festgelegt wurde.

Verschiebbar

Durch das Verschieben einer Box wird ihr Offset verändert. Legen Sie hier fest, ob die Box zur Laufzeit frei im Diagrammbereich verschiebbar sein soll. Wenn die Box nicht mehr verschiebbar sein soll, nachdem Sie sie positioniert haben, schalten Sie diese Option danach ab.

Ursprung

Mithilfe der Eigenschaften **Ursprung**, **Referenzpunkt**, **X-Offset** und **Y-Offset** können Sie jede einzelne Box im Diagrammbereich positionieren, wobei die relative Position der Box zum Diagramm unabhängig von der Diagrammgröße ist.

Legen Sie hier den Ursprung fest, d. h. den Diagrammpunkt, von dem aus der Abstand zum Referenzpunkt der Box in x- bzw. y-Richtung angegeben wird. Mögliche Werte des Ursprungs: links oben, mittig oben, rechts oben, links mittig, mittig mittig, rechts mittig, links unten, mittig unten, rechts unten.

Referenzpunkt

Legen Sie hier den Referenzpunkt der Box fest, d. h. den Punkt der Box, von dem aus der Abstand zum Ursprung in x- bzw. y-Richtung angegeben wird. Mögliche Werte des Referenzpunkts: oben links, oben mittig, oben rechts, mittig links, mittig mittig, mittig rechts, unten links, unten mittig, unten rechts.

X-Offset

Legen Sie hier den Abstand zwischen Ursprung und Referenzpunkt in x-Richtung fest.

Y-Offset

Legen Sie hier den Abstand zwischen Ursprung und Referenzpunkt in y-Richtung fest.

Rahmen

Wenn Sie auf dieses Feld klicken, erscheint eine **Bearbeiten**-Schaltfläche, über die Sie in den Dialog **Linie bearbeiten** gelangen. Hier können Sie den Typ, die Dicke und die Farbe der Umrandungslinie der Box festlegen.

Priorität

Legen Sie hier die relative Zeichnungspriorität der Box zu anderen Objekten im Diagramm (Knoten, Liniengitter etc.) fest. Die Priorität von Knoten ist 0. Falls die Boxen eine höhere Priorität als die Knoten haben, überdecken sie die Knoten so, dass darauf interaktiv nicht mehr zugegriffen werden kann.

Sichtbar

Legen Sie fest, ob die Box zur Laufzeit sichtbar sein soll.

Boxformat

Hier wird das Boxformat der Box angezeigt. Wenn Sie auf dieses Feld klicken, erscheinen zwei Schaltflächen:

Aus der Kombobox können Sie ein vorhandenes Boxformat wählen.

Über die **Bearbeiten**-Schaltfläche gelangen Sie in den Dialog **Boxformate verwalten**.

Box hinzufügen

Eine neue Box mit einem Standardnamen wird angelegt. Diesen Namen können Sie editieren, indem Sie darauf doppelklicken und ihn dann verändern.

Box kopieren

Die markierte Box wird kopiert.

Box löschen

Die Box, die Sie in der Liste markiert haben, wird gelöscht.

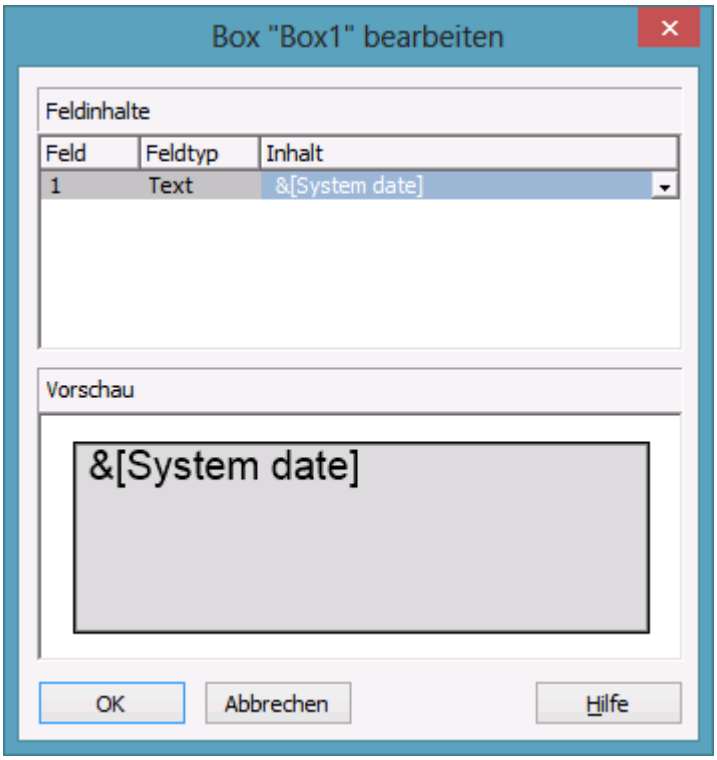
Box bearbeiten

Sie gelangen in das Dialogfeld **Box bearbeiten**.

Box eine Zeile nach oben/unten

Mit Hilfe dieser Schaltflächen können Sie die markierte Box in der Tabelle eine Zeile nach oben bzw. unten schieben.

4.16 Dialogfeld "Box bearbeiten"



Sie erreichen dieses Dialogfeld über die Eigenschaftenseite **Objekte** und das Dialogfeld **Boxen verwalten**, indem Sie dort auf die **Box bearbeiten**-Schaltfläche klicken. Dieser Dialog erscheint auch zur Laufzeit, wenn sie auf eine Box doppelklicken.

Feld

In dieser Spalte werden die Nummern aller Felder der Box aufgeführt. (Die Anzahl der Felder hängt vom gewählten Boxformat ab.)

Feldtyp

Hier wird der Feldtyp jedes Feldes angezeigt (Text oder Grafik).

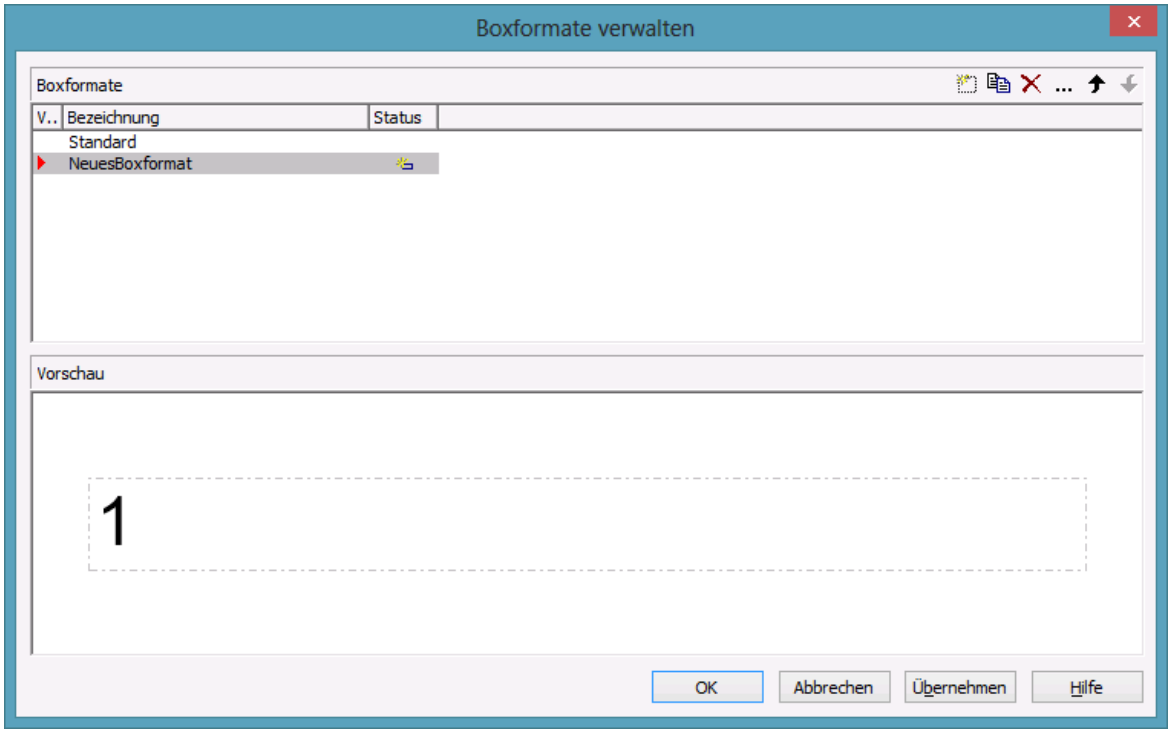
Inhalt

Hier können Sie den Inhalt des Feldes bzw. den Namen einer Grafikdatei eingeben.

Bei mehrzeiligen Textfeldern müssen für einen erzwungenen Umbruch die einzelnen Zeilen des Textfelds mit "\n" im String getrennt sein (Beispiel: "Zeile1\nZeile2"). Ohne erzwungenen Umbruch wird automatisch an Leerzeichen umgebrochen.

Mögliche Grafikformate: WMF, JPG, BMP, GIF, PCX, PNG, TIF.

4.17 Dialogfeld "Boxformate/Knotenformate verwalten"



Sie gelangen in dieses Dialogfeld über die Eigenschaftenseite **Objekte**.

Vorschau

Das in der Vorschauspalte markierte Format wird im Vorschaufenster angezeigt.

Bezeichnung

In dieser Spalte stehen die Namen aller vorhandenen Formate. Die Namen sind editierbar.

Status

In der Spalte **Status** wird jedes Format gekennzeichnet, das seit dem Aufruf des Dialogs hinzugefügt ( ) und/oder geändert ( ) worden ist.

Box/Knotenformat hinzufügen

Ein neues Format mit einem Standardnamen wird angelegt. Diesen Namen können Sie editieren, indem Sie darauf doppelklicken und ihn dann verändern.

Box/Knotenformat kopieren

Das markierte Format wird kopiert.

Box/Knotenformat löschen

Das Format, das Sie in der Liste markiert haben, wird gelöscht. Es können nur die Formate gelöscht werden, die zur Zeit nicht benutzt werden.

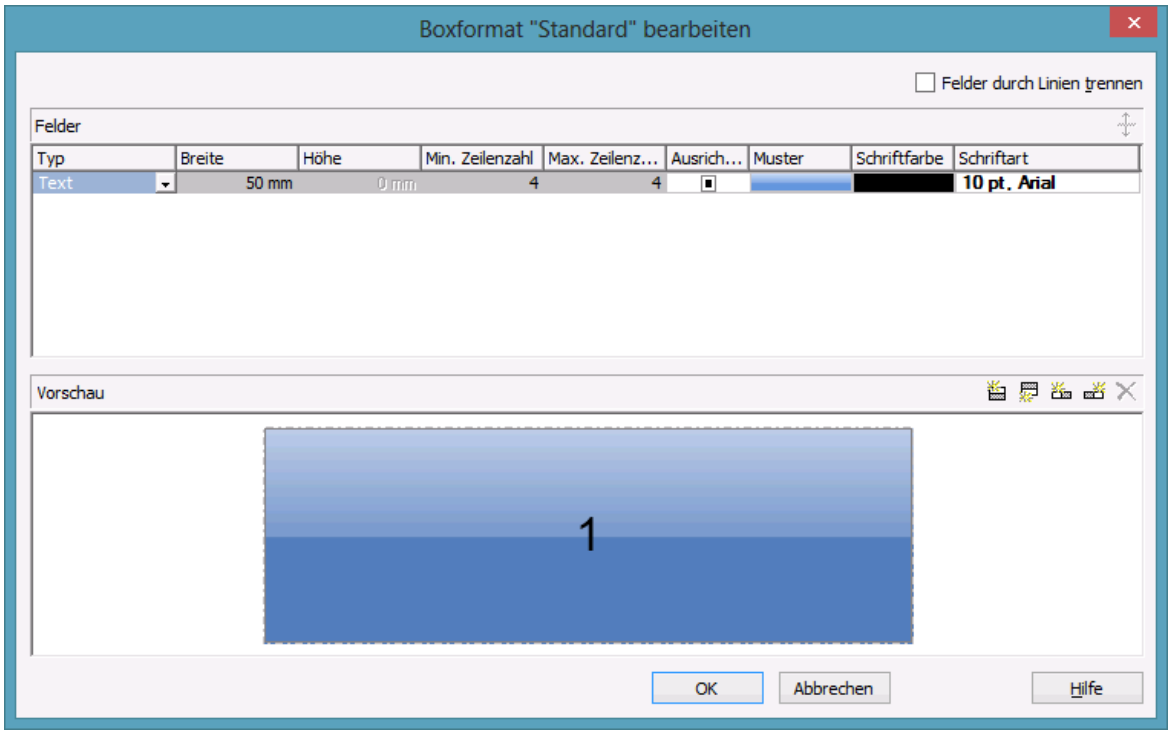
Box/Knotenformat bearbeiten

Sie gelangen in das Dialogfeld **Boxformat bearbeiten** bzw. **Knotenformat bearbeiten**.

Box / Knotenformat eine Zeile nach oben / unten

Mit Hilfe dieser Schaltflächen können Sie das markierte Format in der Tabelle eine Zeile nach oben bzw. unten schieben.

4.18 Dialogfeld "Boxformat bearbeiten"



Sie erreichen dieses Dialogfeld, indem Sie auf der Eigenschaftenseite **Objekte** das Dialogfeld **Boxformate verwalten** aktivieren und dort auf die **Boxformat bearbeiten**-Schaltfläche klicken.

Felder durch Linien trennen

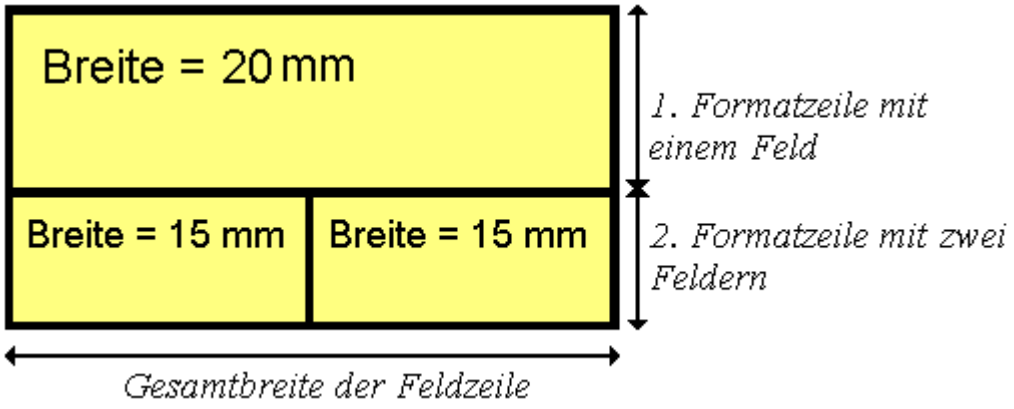
Aktivieren Sie dieses Kontrollkästchen, wenn die Felder der Box durch Linien getrennt werden sollen.

Typ

Wählen Sie hier den Feldtyp (Text oder Grafik).

Breite

Legen Sie hier die Breite in Millimetern für das markierte Feld fest. Die maximale Breite eines Feldes beträgt 200 mm. Ist eine Feldzeile in zwei oder mehr Felder unterteilt und die Gesamtbreite der einzelnen Feldzeilen unterschiedlich groß, so richtet sich die Gesamtbreite nach der insgesamt breitesten Feldzeile.



Höhe

(nur für den Typ Grafik) Legen Sie hier die minimale Höhe in Millimetern für das markierte Feld fest. Die maximale Höhe eines Feldes beträgt 200 mm.

Min./Max. Zeilenzahl

(nur für den Typ Text) Bestimmen Sie die minimale bzw. die maximale Anzahl der Textzeilen des aktuellen Feldes. Die maximale Zeilenzahl eines Feldes beträgt neun.

Ausrichtung

Wählen Sie hier die Ausrichtung des Inhalts in dem markierten Feld (9 Möglichkeiten).

Muster

Legen Sie hier Hintergrundfarbe und -muster des Feldes fest. Durch Klick auf ... gelangen Sie in den Dialog **Musterattribute bearbeiten**, in dem Sie ein Muster, eine Hintergrundfarbe sowie eine 2. Musterfarbe auswählen können. Zusätzlich zu der vorgegebenen Farbpalette können Sie beliebige Farben definieren. Es stehen außerdem transparente Farben zur Verfügung.

Schriftfarbe

(nur für den Typ Text) Die Schriftfarbe des Feldes wird hier angezeigt.

Die Farbzuordnungstabelle öffnet sich, und Sie können daraus eine Schriftfarbe auswählen.

Schriftart

(nur für den Typ Text) Die Schriftart des Feldes wird hier angezeigt.

... Über diese Schaltfläche öffnen Sie den Windows-Dialog **Schriftart**.

selektierte Eigenschaft in alle Felder übernehmen

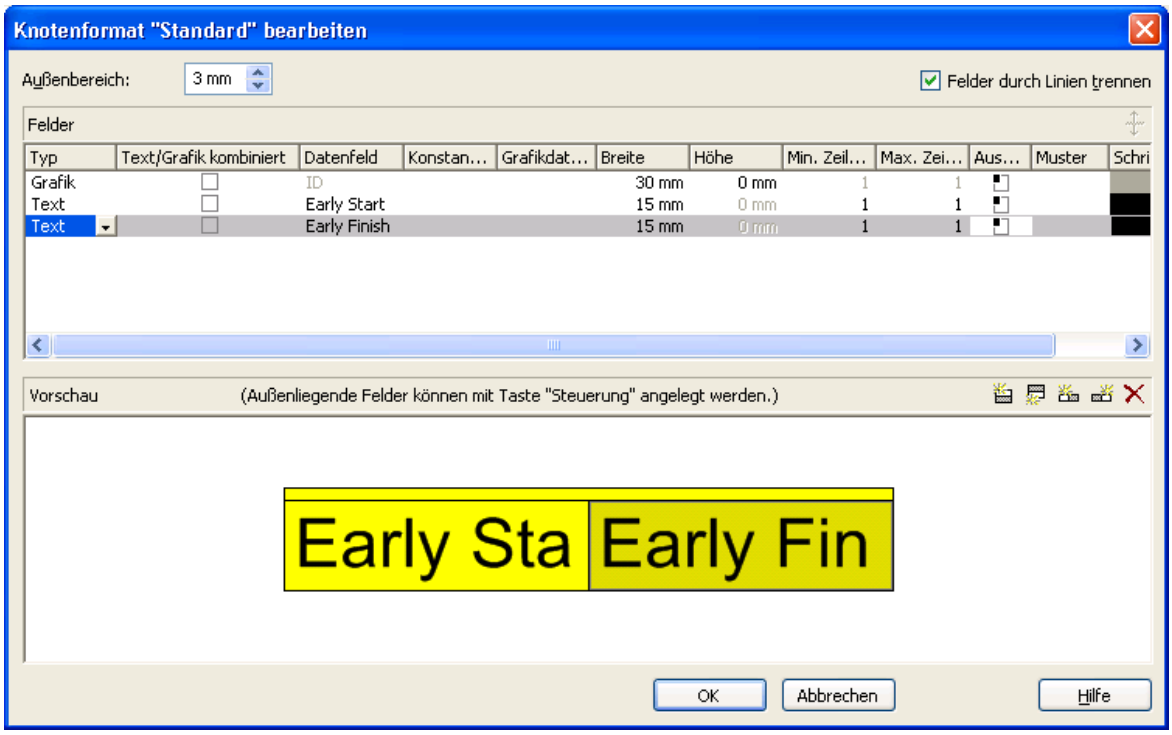
Klicken Sie auf diese Schaltfläche, damit die selektierte Eigenschaft in alle Felder übernommen wird.

Vorschau

Hier werden die aktuellen Felder des Boxformats dargestellt. Wenn Sie im Vorschaufenster ein Feld anklicken, können Sie dessen Attribute in der **Felder**-Tabelle bearbeiten.

Mit Hilfe der Schaltflächen oberhalb der Vorschau können Sie neue Felder anlegen oder das markierte Feld löschen. Zum Löschen von Feldern können Sie auch die Entf-Taste benutzen.

4.19 Dialogfeld "Knotenformat bearbeiten"



Sie gelangen in dieses Dialogfeld, indem Sie im Dialogfeld **Knotenformate verwalten** auf die **Knotenformat bearbeiten**-Schaltfläche klicken.

Außenbereich

Legen Sie hier den Abstand in Millimetern fest, den Knoten mit diesem Knotenformat zu benachbarten Knoten und zum Rand der Darstellung halten sollen. Standardmäßig beträgt der Außenbereich 3 mm. Bei kleineren Werten kann es gelegentlich zu Überlagerungen von grafischen Elementen kommen. Daher sollten Sie den Standardwert nur in begründeten Fällen unterschreiten.

Felder durch Linien trennen

Aktivieren Sie dieses Kontrollkästchen, wenn die einzelnen Felder durch Linien getrennt werden sollen.

Typ

Wählen Sie hier den Feldtyp (Text oder Grafik).

Text/Grafik kombiniert

Wenn dieses Kontrollkästchen aktiviert ist, können in dem Knotenfeld ein Text und eine Grafik kombiniert werden, und zwar folgendermaßen:

- **Typ**: Text, **Text/Grafik kombiniert**: nein: nur Text wird ausgegeben (wie unter **Datenfeld** oder unter **Konstanter Text** angegeben)
- **Typ**: Grafik, **Text/Grafik kombiniert**: nein: nur eine Grafik wird ausgegeben (wie unter **Grafikdateiname** angegeben)
- **Typ**: Text, **Text/Grafik kombiniert**: ja: Text (wie unter **Datenfeld** oder unter **Konstanter Text** angegeben) und Grafik (wie unter **Grafikdateiname** angegeben) werden ausgegeben
- **Typ**: Grafik, **Text/Grafik kombiniert**: ja: nur eine Grafik wird ausgegeben (wie unter **Grafikdateiname** angegeben). Text (unter **Datenfeld** angegeben) ist nur im Tooltip sichtbar; er kann aber, falls geeignet, als Hyperlink genutzt werden.)

Datenfeld

Wählen Sie hier das Datenfeld, dessen Inhalt in dem aktuellen Feld ausgegeben werden soll. Passt der Inhalt des Datenfeldes nicht in das Feld, wird der Überhang bei der Ausgabe abgeschnitten.

Konstanter Text

(nur wenn kein Datenfeld gewählt wurde) Sie können hier einen konstanten Text eingeben, der in dem Knotenfeld ausgeben werden soll.

Grafikdateiname

Hier werden Name und Pfad der Grafikdatei angezeigt, die in dem gewählten Knotenfeld dargestellt werden soll.

Wenn Sie auf ein **Grafikdateiname**-Feld klicken, erscheinen zwei Schaltflächen:

Der Windows-Dialog **Grafikdatei auswählen** erscheint. Hier können Sie eine Grafikdatei auswählen, die in dem aktuellen Formatfeld erscheinen soll.

Wird ein relativer Dateiname angegeben, so wird die Datei zur Laufzeit zuerst in dem Verzeichnis gesucht, das in der VARCHART-Windows-Forms-Eigenschaft **FilePath** gesetzt ist. Wird sie dort nicht gefunden, wird sie zuerst im gerade aktiven Arbeitsverzeichnis der Applikation und dann im

Installationsverzeichnis des VARCHART Windows-Forms-Steuerelement gesucht.

Klicken Sie auf diese Schaltfläche (**Zuordnungen einstellen**), um eine Zuordnungstabelle zu verwenden, um Grafiken abhängig vom Inhalt eines Datenfelds in einem Knotenfeld erscheinen zu lassen.

Wird im Dialog **Zuordnung einstellen** nur ein Datenfeld, aber keine Zuordnungstabelle ausgewählt, so wird der Inhalt des eingestellten Datenfelds direkt als Name einer Grafikdatei interpretiert. Findet sich in dem Datenfeld bzw. in der Zuordnungstabelle kein gültiger Name einer Grafikdatei, dann wird der direkt in Spalte angegebene Dateiname verwendet.

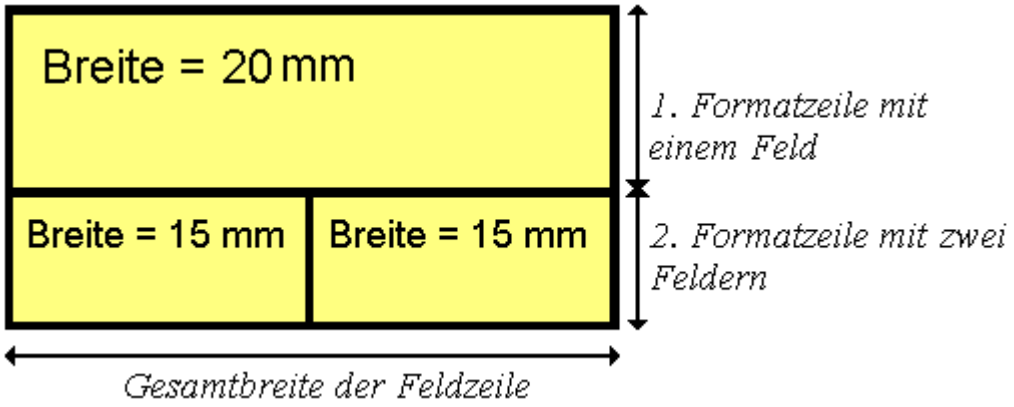
Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der 2. Schaltfläche fett dargestellt ( ).

Sobald Sie die entsprechende Zeile verlassen, zeigt ein Symbol im Feld **Grafikdateiname** an, dass eine Zuordnung vorgenommen worden ist.

Bei der Darstellung der Grafik wird die Farbe des Pixels von ihrer Ecke links oben durch die Diagramm-Hintergrundfarbe ersetzt, d. h. alle Pixel der Grafik in dieser Farbe werden transparent dargestellt.

Breite

Legen Sie hier die Breite in Millimetern für das markierte Feld fest. Die maximale Breite eines Feldes beträgt 99 mm. Ist eine Feldzeile in zwei oder mehr Felder unterteilt und die Gesamtbreite der einzelnen Feldzeilen unterschiedlich groß, so richtet sich die Gesamtbreite nach der insgesamt breitesten Feldzeile.



Höhe

(nur für den Typ Grafik) Legen Sie hier die minimale Höhe in Millimetern für das markierte Feld fest. Die maximale Höhe beträgt 99 mm.

Min./Max. Zeilenzahl

(nur für den Typ Text) Bestimmen Sie die minimale und die maximale Anzahl der Textzeilen des aktuellen Feldes. Die maximale Zeilenzahl eines Feldes beträgt neun.

Ausrichtung

Hier bestimmen Sie die Ausrichtung des Textes bzw. der Grafik des markierten Feldes.

Muster

Legen Sie hier Hintergrundfarbe und -muster des Feldes fest. Durch Klick auf ... gelangen Sie in den Dialog **Musterattribute bearbeiten**, in dem Sie ein Muster, eine Hintergrundfarbe sowie eine 2. Musterfarbe auswählen können. Zusätzlich zu der vorgegebenen Farbpalette können Sie beliebige Farben definieren. Es stehen außerdem transparente Farben zur Verfügung.

: Über diese Schaltfläche im Dialog **Musterattribute bearbeiten** gelangen Sie in den Dialog **Zuordnung einstellen**. Hier können Sie für das jeweilige Attribut eine datenabhängige Zuordnung vereinbaren. Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der Schaltfläche fett dargestellt ( ).

Wird hier nichts fesgelegt, wird automatisch das Attribut benutzt, das für das Knotenaussehen vereinbart wurde.

Schriftfarbe

(nur für den Typ Text) Die Schriftfarbe des Feldes wird angezeigt, und Sie können sie bearbeiten. Wenn Sie auf dieses Feld klicken, erscheinen zwei Schaltflächen:

Die Farbzuordnungstabelle öffnet sich, und Sie können daraus eine Schriftfarbe auswählen.

Über die zweite Schaltfläche gelangen Sie in den Dialog **Zuordnung einstellen**. Hier können Sie eine datenabhängige Zuordnung der Farbe vereinbaren. Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der Schaltfläche fett dargestellt ( ).

Schriftart

Die Schriftart des Feldes wird angezeigt, und Sie können sie bearbeiten. Wenn Sie auf dieses Feld klicken, erscheint eine Schaltfläche (...), über die Sie den Windows-Dialog **Schriftart** öffnen können.

selektierte Eigenschaft in alle Felder übernehmen

Klicken Sie auf diese Schaltfläche, damit die selektierte Eigenschaft in alle Felder übernommen wird.

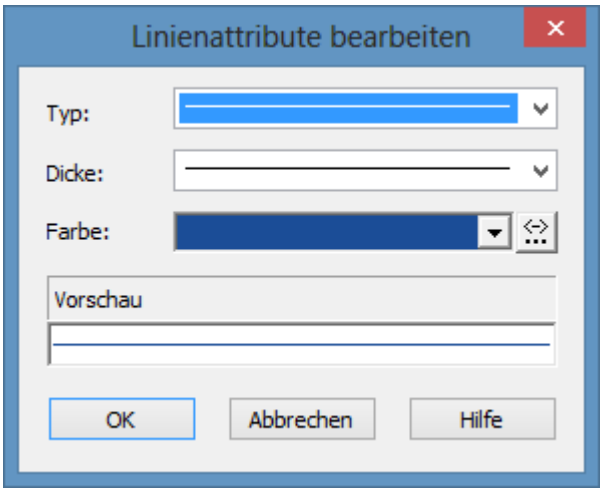
Vorschau

Hier wird das aktuelle Knotenformat dargestellt. Wenn Sie im Vorschaufenster ein Feld anklicken, können Sie dessen Attribute in der **Felder**-Tabelle bearbeiten.

Mit Hilfe der Schaltflächen oberhalb der Vorschau können Sie neue Felder anlegen oder das markierte Feld löschen. Sie können auch die Entf-Tast benutzen, um Felder zu löschen.

Wenn Sie Felder außerhalb des Knotens anlegen möchten, so ist zusätzlich die Strg-Taste zu drücken.

4.20 Dialogfeld "Linienattribute bearbeiten"



Der Dialog zum Bearbeiten der Linienattribute, der jeweils über die Schaltfläche ... aufgerufen werden kann, steht zur Verfügung für das Verbindungsaussehen, Layer, sowie für den Rahmen von Boxen.

Typ

Wählen Sie hier den Typ der Linie aus (durchgezogen, gestrichelt, etc.).

Dicke

Wählen Sie hier die Linienstärke aus.

Farbe

Wählen Sie hier die Farbe der Linien aus.

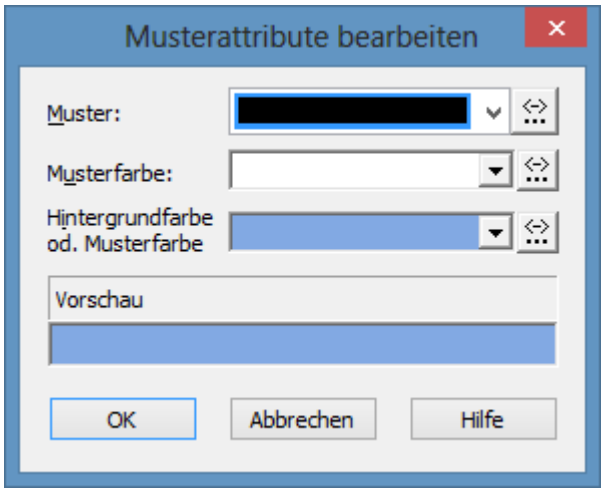
Über diese Schaltfläche gelangen Sie in den Dialog **Zuordnung einstellen**, in dem Sie eine datenabhängige Zuordnung der Linienfarbe vereinbaren können.

Wenn eine Zuordnung vorgenommen worden ist, wird der Pfeil auf der Schaltfläche fett dargestellt.

Vorschau

Hier wird das Aussehen der Linie mit den gewählten Einstellungen angezeigt.

4.21 Dialogfeld "Musterattribute bearbeiten"



Der Muster-Dialog, der jeweils über die Schaltfläche ··· aufgerufen werden kann, steht zur Verfügung für Box- und Knotenformate.

Muster

Hier können Sie ein Hintergrundmuster auswählen.

Musterfarbe

Wählen Sie hier die Vordergrund-Farbe des Musters aus.

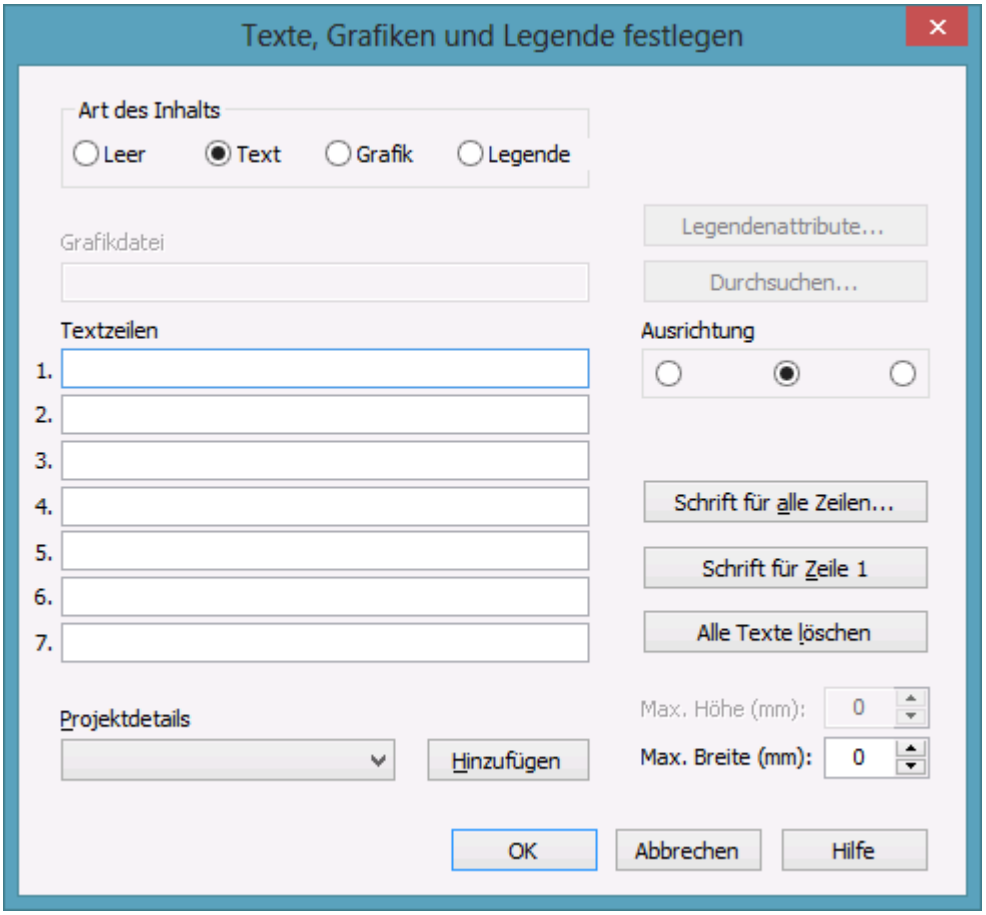
Hintergrundfarbe oder Musterfarbe 2

Wählen Sie hier die Hintergrund-Farbe oder eine zweite Musterfarbe aus.

Vorschau

Hier wird das Aussehen des Musters mit den gewählten Einstellungen angezeigt.

4.22 Dialogfeld "Texte, Grafiken und Legende festlegen"



Sie erreichen dieses Dialogfeld, indem Sie auf der Eigenschaftenseite **Außenbereich** auf eine der neun Schaltflächen ober- bzw. unterhalb der Grafik klicken.

Art des Inhalts

Wählen Sie hier die Art des Inhalts, der in dem zuvor gewählten Bereich der Darstellung ausgegeben werden soll:

- **Leer:** Der gewählte Diagrammbereich bleibt leer.
- **Text:** In dem gewählten Diagrammbereich wird der Text der sechs Textzeilen dargestellt.
- **Grafik:** Sie können in dem gewählten Bereich eine Grafik (z.B. Ihr Firmenlogo) plazieren. Grafiken werden immer mittig ausgerichtet.

- **Legende:** Eine Legende wird in dem gewählten Diagrammbereich ausgeben. Sie dokumentiert die Layer, die in der aktuellen Darstellung auftreten.

Die jeweils nicht benötigten Bereiche des Dialogs werden abhängig von Ihrer Auswahl deaktiviert. Dabei bleiben alle Angaben erhalten.

Legendenattribute

*Nur aktiv, wenn das Kontrollkästchen **Legende** angeklickt wurde.* Sie gelangen in den gleichnamigen Dialog, der für die Legende weitere Gestaltungsmöglichkeiten bietet.

Grafikdatei

*Nur aktiv, wenn das Kontrollkästchen **Grafik** angeklickt wurde.* Tragen Sie hier den Namen der Grafikdatei ein. Falls sich die gewünschte Grafikdatei nicht im Installationsverzeichnis von VARCHART befindet, müssen Sie das Laufwerk und den Pfad auch angeben.

Durchsuchen

*Nur aktiv, wenn das Kontrollkästchen **Grafik** angeklickt wurde.* Wenn Sie auf diese Schaltfläche klicken, erscheint der Windows-Dialog **Grafikdatei auswählen**, mit dessen Hilfe Sie das Laufwerk, das Verzeichnis und den Dateinamen der Grafik festlegen können.

Textzeilen

*Nur aktiv, wenn das Kontrollkästchen **Text** angeklickt wurde.* Vereinbaren Sie den Text (max. 6 Zeilen), mit denen der gewählte Diagrammbereich beschriftet werden soll. Sie können einen beliebigen Text eintragen, aber auch Platzhalter (z.B. &[System-Datum]) für Projektdetails einsetzen. Sind alle sechs Textzeilen leer, wird dieser Bereich nicht dargestellt.

Projektdetails

*Nur aktiv, wenn das Kontrollkästchen **Text** angeklickt wurde.* Hier können Sie dem Diagramm verschiedene Informationen (Anzahl der Seiten, Seitennummer, Systemdatum) hinzufügen, indem Sie aus der Kombobox den gewünschten Platzhalter auswählen und auf die Schaltfläche **Hinzufügen** klicken. Die Platzhalter werden in der Druckvorschau oder beim Ausdruck

durch die entsprechenden Daten ersetzt und stets auf dem aktuellen Stand gehalten.

Hinzufügen

*Nur aktiv, wenn das Kontrollkästchen **Text** angeklickt wurde.* Nachdem Sie aus der Liste ein Projektdetail ausgewählt haben, bestätigen Sie Ihre Wahl mit der Schaltfläche **Hinzufügen**. Die Projektdetails werden in die Zeile geschrieben, in der sich der Cursor gerade befindet.

Textausrichtung

*Nur aktiv, wenn das Kontrollkästchen **Text** angeklickt wurde.* Über die Kontrollkästchen können Sie die Textzeilen linksbündig, zentriert oder rechtsbündig ausrichten.

Schrift für alle Zeilen

*Nur aktiv, wenn das Kontrollkästchen **Text** angeklickt wurde.* Über diese Schaltfläche öffnen Sie den Windows-Dialog **Schriftart**. Hier können Sie die Schriftart, Schriftgröße usw. für alle sechs Zeilen festlegen. Beim Ausführen dieser Aktion werden die Einstellungen für die Schrift der einzelnen Zeilen überschrieben.

Schrift für Zeile 1...6

*Nur aktiv, wenn das Kontrollkästchen **Text** angeklickt wurde.* Über diese Schaltfläche öffnen Sie den Windows-Dialog **Schriftart**. Hier können Sie die Schriftart, Schriftgröße usw. für die Zeile festlegen, in der sich der Cursor befindet.

Alle Texte löschen

*Nur aktiv, wenn das Kontrollkästchen **Text** angeklickt wurde.* Wenn Sie diese Schaltfläche anklicken, werden die Texte aller sechs Textzeilen gelöscht.

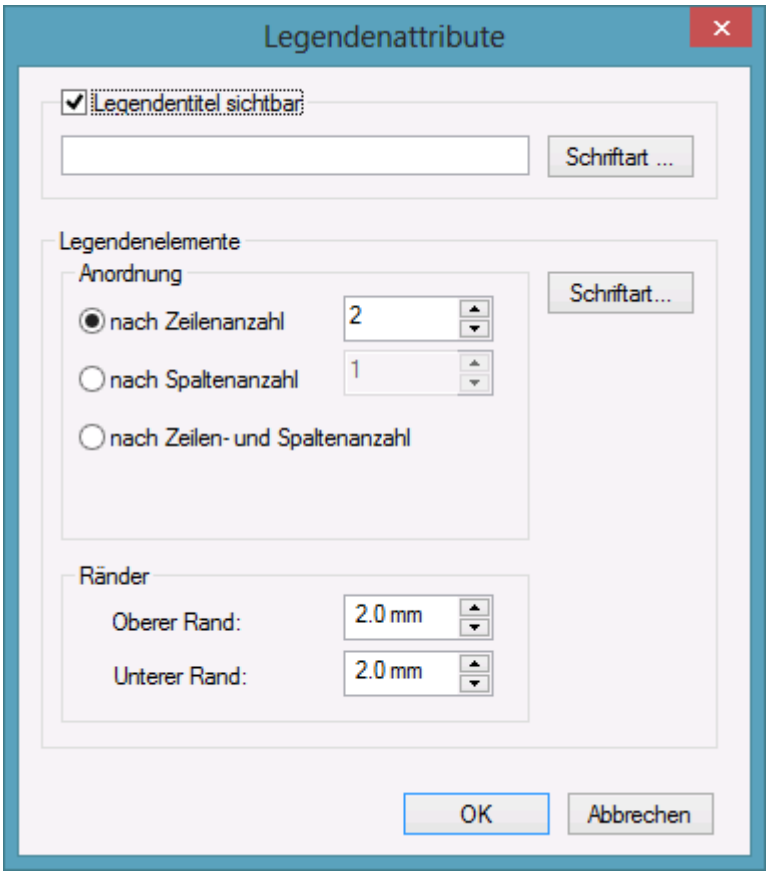
Max. Höhe (mm)

*Nur aktiv, wenn das Kontrollkästchen **Grafik** angeklickt wurde.* Falls mehrere Felder für Text, Grafik oder Legende vereinbart worden sind, können Sie hier die maximale Höhe des aktuellen Feldes festlegen. So können Sie verhindern, dass Feldinhalte abgeschnitten werden.

Max. Breite (mm)

*Nur aktiv, wenn das Kontrollkästchen **Text** oder **Grafik** angeklickt wurde.* Falls mehrere Felder für Text, Grafik oder Legende vereinbart worden sind, können Sie hier die max. Breite des aktuellen Feldes festlegen. So können Sie verhindern, dass Feldinhalte abgeschnitten werden.

4.23 Dialogfeld "Legendenattribute"



Sie erreichen dieses Dialogfeld zur Laufzeit über das Kontextmenü der Legende oder zur Designzeit über den Dialog **Texte, Grafiken und Legende festlegen** durch Klick auf die entsprechende Schaltfläche. Diese wird erst wählbar, nachdem Sie bei **Art des Inhalts Legende** ausgewählt haben.

Legendentitel sichtbar

Legen Sie hier fest, ob ein Legendentitel angezeigt werden soll und geben Sie einen Text ein. Durch Klick auf **Schriftart** öffnen Sie den gleichnamigen Windows-Dialog, in dem Sie die Schriftattribute für den Legendentitel festlegen können.

Anordnung

- nach Zeilenanzahl: Geben Sie hier an, ob und mit wie vielen Zeilen die Legende dargestellt werden soll.
- nach Spaltenanzahl: Geben Sie hier an, ob und mit wie vielen Spalten die Legende dargestellt werden soll.

- Nach Zeilen- und Spaltenanzahl: Die Legende wird in Spalten und Zeilen dargestellt. Ist die hier eingegebene Zahl niedriger als die vorhandenen Layern, so werden die überzähligen Layer nicht dargestellt.

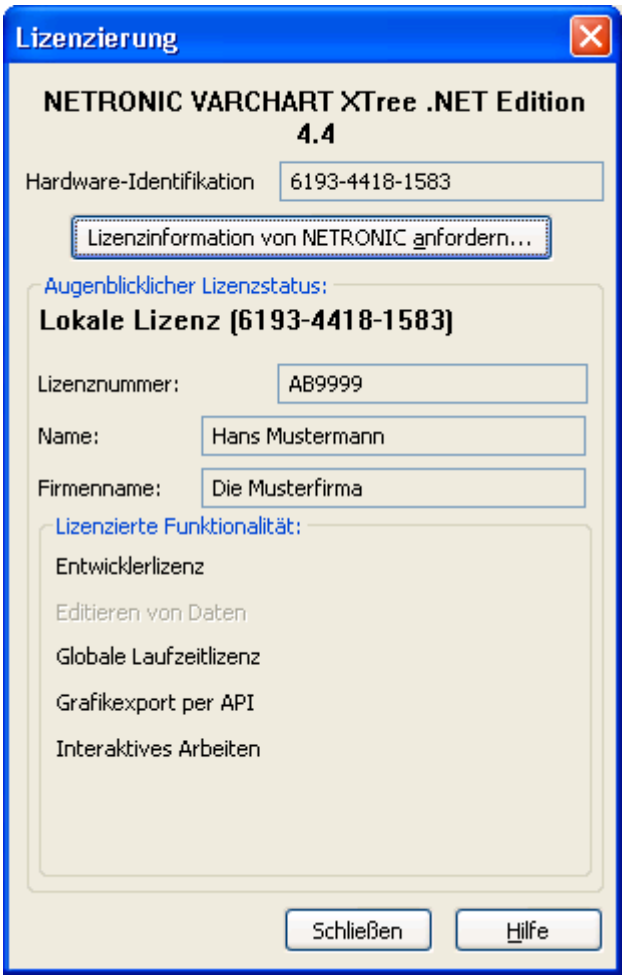
Ränder

- oberer Rand: Geben Sie ein Maß für den oberen Rand des Legendenelements an.
- unterer Rand: Geben Sie ein Maß für den unteren Rand des Legendenelements an.

Schrift

Über diese Schaltfläche öffnen Sie den Windows-Dialog **Schriftart**, in dem Sie die Schriftattribute für die Legende festlegen können.

4.24 Dialogfeld "Lizenzierung"



Sie erreichen dieses Dialogfeld über die Eigenschaftenseite **Allgemeines**.

Vor der Lizenzierung ist das Programm nur als Demoversion lizenziert. Folgende Einschränkungen gelten gegenüber der Vollversion: Die Nutzungsdauer zum Testen des Produkts ist auf 30 Tage begrenzt. Nach Ablauf dieses Zeitraums erscheint das Wort "Demo" im Diagramm.

Hardware-Identifikation

(nicht editierbar) Die Nummer, die in diesem Feld angezeigt wird, wird aus der Hardware-Konfiguration Ihres Rechners berechnet. Sie wird von NETRONIC Software GmbH für die Lizenzierung benötigt.

Bei Änderungen an Ihrer Hardware ist eine neue Lizenzierung erforderlich. Der Kundendienst von NETRONIC Software GmbH hilft Ihnen dann gern weiter.

Anfordern

Um die Lizenzierung vorzunehmen, klicken Sie auf diese Schaltfläche. Dann erscheint der Dialog **Lizenzinformationen anfordern**.

Lizenznummer/Name/Firmenname

(nicht editierbar) Hier werden Ihre Lizenznummer, Ihr Name und der Name Ihrer Firma angezeigt.

Lizenzierte Funktionalität

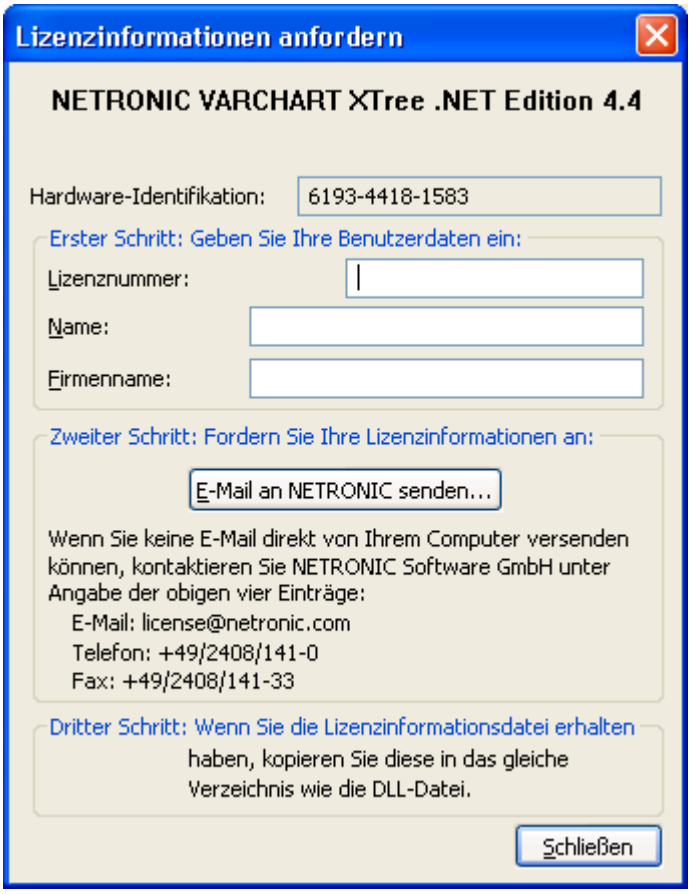
Hier wird angezeigt, welche Module für Sie freigegeben wurden. Wenn Sie die Lizenzierung vorgenommen haben, sind die freigegebenen Module aktiviert.

- **Entwicklerlizenz**
- **Globale Laufzeitlizenz** (Das VARCHART ActiveX läuft im Runtime-Modus auf jedem anderen Rechner.)
- **Einzelplatz-Laufzeitlizenzen** (Das VARCHART ActiveX muss auf jedem Rechner, auf dem es im Runtime-Modus laufen soll, einzeln lizenziert werden.)
- **Grafikexport per API**
- **Interaktives Arbeiten**

Schließen

Sie verlassen den Dialog.

4.25 Dialogfeld "Lizenzinformationen anfordern"



Geben Sie Ihre Lizenznummer, Ihren Namen und den Namen Ihrer Firma an und klicken Sie auf **E-Mail an NETRONIC senden**. Damit wird automatisch eine E-Mail generiert, die Sie nur noch absenden müssen. Sobald wir Ihre E-Mail erhalten haben, werden wir unverzüglich eine Datei mit Ihren Lizenzinformationen (vctree.lic) generieren und sie Ihnen zusenden. Bitte kopieren Sie dann diese Datei in das Verzeichnis, in dem die Datei vctree.ocx steht.

Wenn Sie die neue Lizenzierung vorgenommen haben, müssen Sie diese in jedem Ihrer Projekte aktivieren. Öffnen Sie dazu in jedem Ihrer Projekte eine beliebige Eigenschaftenseite, nehmen Sie dort eine beliebige Änderung vor und speichern Sie diese. Nun ist die neue Lizenzierung aktiviert.

5 Benutzerschnittstelle

5.1 Übersicht

Die folgende Liste gibt einen Überblick über die Interaktionsmöglichkeiten für Anwender.

- Navigation im Diagramm
- Zoomen
- Knoten erzeugen
- Knoten markieren
- Knoten löschen, ausschneiden, kopieren und einfügen
- Knoten bearbeiten
- Verbindungsaussehen festlegen
- Knoten mit seinem Teilbaum umhängen
- Teilstrukturen horizontal und vertikal anordnen
- Teilstrukturen kollabieren und expandieren
- Legende bearbeiten
- Seite einrichten
- Druckvorschau verwenden

Kontextmenüs (rechte Maustaste):

- für das Diagramm
- für Knoten
- für die Legende

Bei allen Interaktionen wird ein Ereignis ausgelöst, sodass Sie im Programm darüber informiert werden und ggf. darauf reagieren können.

5.2 Navigation im Diagramm

Sie können mit Hilfe der Pfeil-Tasten mit der Markierung von einem Knoten zum anderen springen.

Sie können bei gedrückter Strg-Taste mit Hilfe der Pfeil-Tasten im Diagramm scrollen.

Weitere Tastenkombinationen für die Navigation im Diagramm:

- **Strg** + **Pos1:** an den linken oberen Diagrammrand scrollen
- **Strg** + **Ende:** an den rechten unteren Diagrammrand scrollen
- **Strg** + **Bild rauf/runter:** an den oberen/ unteren Diagrammrand scrollen
- **Strg** + **Num +** (Nummernblock): Zoom in
- **Strg** + **Num -** (Nummernblock): Zoom out
- **Strg** + **Num *** (Nummernblock): zum nächsten Knoten scrollen
- **Strg** + **Num /** (Nummernblock): Komplettansicht

Mit Hilfe von **Strg + C**, **Strg + X** bzw. **Strg + V** können Sie markierte Knoten kopieren, ausschneiden bzw. einfügen. Mit Hilfe der **Entf**-Taste können Sie markierte Knoten löschen.

5.3 Zoomen

Mithilfe der folgenden Tastenkombinationen lässt sich die Darstellung vergrößern bzw. verkleinern:

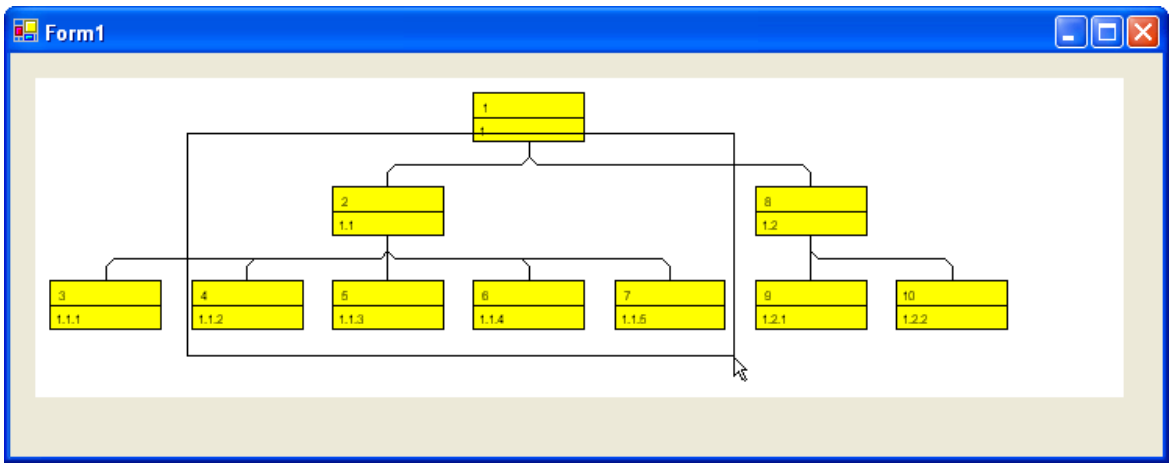
- **Strg** + **-** (Nummernblock): Verkleinern
- **Strg** + **+** (Nummernblock): Vergrößern

Auch die Maus kann zum Zoomen genutzt werden:

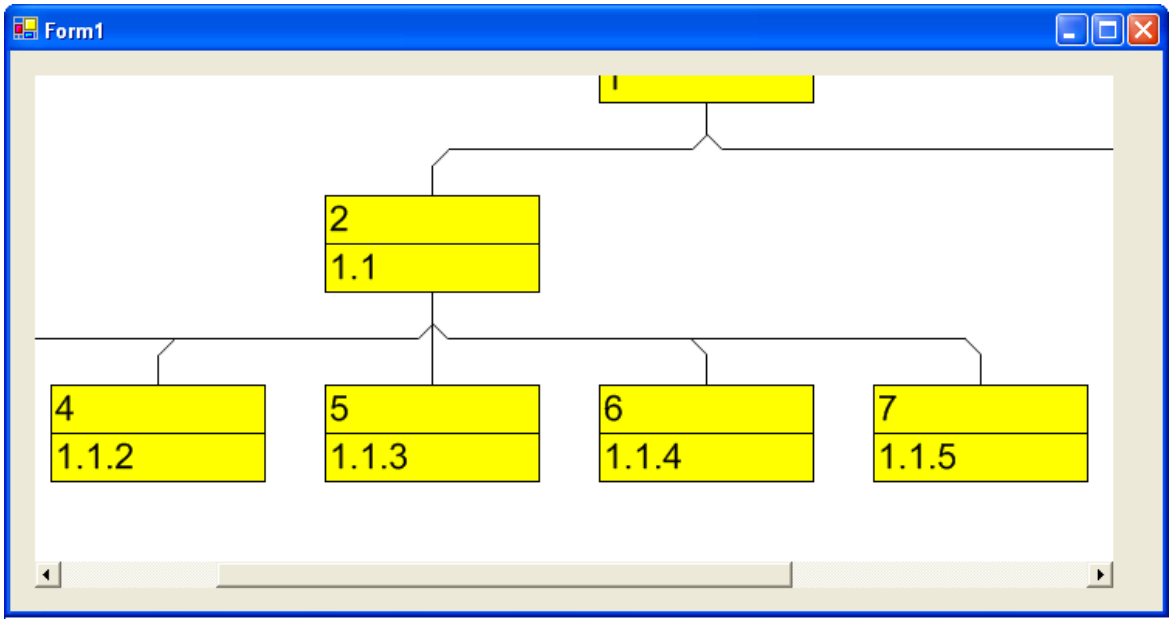
- Drehen Sie das Mausrad während Sie die Strg-Taste gedrückt halten. Dazu muss das Zoomen per Mausrad zugelassen sein. Dies geschieht entweder über die Option **Zoomen per Mausrad zulassen** auf der Eigenschaftenseite **Allgemeines** oder über die API-Eigenschaft **VcTree1.-ZoomingPerMouseWheelAllowed**. (Diese Eigenschaft st standardmäßig deaktiviert.)
- Sie können einen Ausschnitt Ihres Diagramms bildschirmfüllend darstellen lassen, indem Sie mit gedrückter linker Maustaste ein Rechteck um den zu vergrößernden Ausschnitt aufziehen und dann (bei noch gedrückter linker Maustaste) die rechte Maustaste drücken. Mit Hilfe der Bildlaufleisten können Sie dann das Fenster wie eine Lupe über der Darstellung verschieben und so auch die anderen Bereiche der Darstellung in derselben Vergrößerung betrachten.

Mit der API-Methode **ShowAlwaysCompleteView** können Sie die Darstellung so einstellen, dass stets das komplette Diagramm angezeigt wird. Der Zoomfaktor passt sich bei jeder Änderung des Diagramms automatisch an. Der maximale Zoomfaktor von 100% wird nicht überschritten, die Knoten werden also höchstens in Originalgröße dargestellt.

Weitere Information zu den Zoommöglicheiten für den Druck finden Sie in Kapitel 5.21 "Seite einrichten".



vor dem Zoomen

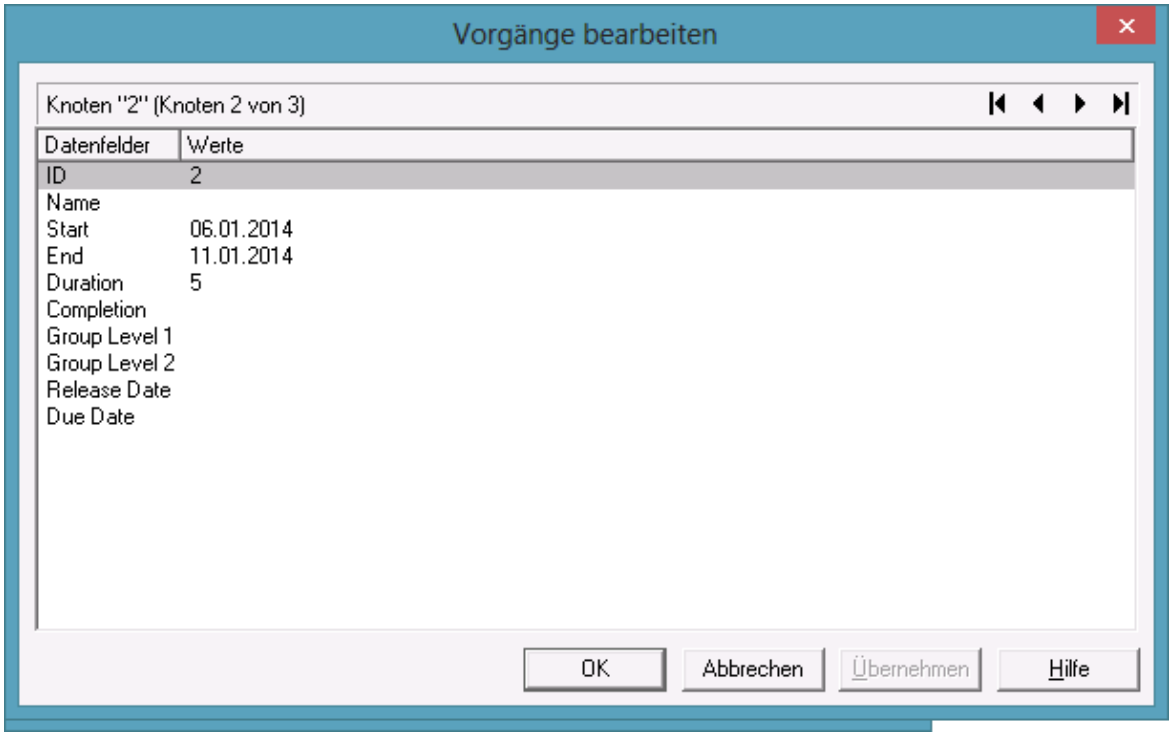


nach dem Zoomen

5.4 Knotendaten bearbeiten

Alle Daten eines Knotens können Sie im Dialogfeld **Vorgänge bearbeiten** bearbeiten. Sie erreichen das Dialogfeld durch einen Doppelklick auf einen Knoten oder über den Befehl <Bearbeiten> seines Kontextmenüs.

Um die Daten mehrerer Knoten zu bearbeiten, markieren Sie die gewünschten Knoten und wählen Sie aus dem Kontextmenü eines der markierten Knoten ebenfalls den Befehl **Bearbeiten** um das Dialogfeld **Vorgänge bearbeiten** zu öffnen. Jetzt können Sie nacheinander die Daten aller markierten Knoten bearbeiten



Durch einen Doppelklick auf einen Knoten wird das Ereignis **VcNodeLeftDoubleClicking** ausgelöst.

Wenn ein Knoten interaktiv verändert worden ist (hier durch die Veränderung eines Wertes im Dialog **Vorgänge bearbeiten**), wird das Ereignis **VcNodeModifying** ausgelöst. Über den Parameter **modificationType** erhalten Sie nähere Informationen über die Art der Veränderung. Durch Setzen des Rückgabestatus auf **vcRetStatFalse** wird die Veränderung unterdrückt.

Der Dialog "Vorgänge bearbeiten"

Oberhalb der Tabelle wird der Name des aktuellen Knotens angezeigt und ggf. um den wievielten Knoten der markierten Knoten es sich handelt.

Sie können hier alle Werte dieses Knotens bearbeiten und ggf. mit Hilfe der **Übernehmen**-Schaltfläche übernehmen. Mit Hilfe der Pfeil-Schaltflächen können Sie zwischen den Knoten navigieren.

Datenfelder

In dieser Spalte werden alle Datenfelder angezeigt, durch die der markierte Knoten beschrieben wird und die **nicht** im Dialog **Datentabellen verwalten** als **versteckt** definiert wurden. Welche Datenfelder verfügbar sind, hängt von Ihrer Datendefinition ab.

Werte

Sie können in dieser Spalte alle Werte des markierten Knotens direkt bearbeiten, sofern sie im Dialog **Datentabellen verwalten** als **editierbar** definiert worden sind.

Wenn Sie hier ein Datenfeld vom Typ **Datum/Zeit** bearbeiten, erscheint ein Datumsdialog, in dem Sie das gewünschte Datum anklicken können. Fehler durch die Eingabe eines falschen Datumsformats werden so vermieden.

Das **Datumsausgabeformat** wird auf der Eigenschaftenseite **Allgemeines** festgelegt.

Wenn Sie ein Datenfeld vom Typ **Integer** bearbeiten, erscheint ein Spincontrol, mit dem Sie den gewünschten Wert einstellen können.

5.5 Navigation mit Hilfe der Tastatur

Sie können mit Hilfe der Pfeil-Tasten mit der Markierung von einem Knoten zum anderen springen.

Sie können bei gedrückter Strg-Taste mit Hilfe der Pfeil-Tasten im Diagramm scrollen.

Weitere Tastenkombinationen für die Navigation im Diagramm:

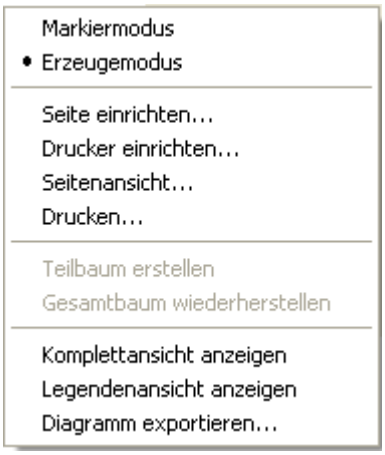
- **Strg** + **Pos1:** an den linken oberen Diagrammrand scrollen
- **Strg** + **Ende:** an den rechten unteren Diagrammrand scrollen
- **Strg** + **Bild rauf/runter:** an den oberen/ unteren Diagrammrand scrollen
- **Strg** + **Num +** (Nummernblock): Zoom in
- **Strg** + **Num -** (Nummernblock): Zoom out
- **Strg** + **Num *** (Nummernblock): zum nächsten Knoten scrollen
- **Strg** + **Num /** (Nummernblock): Komplettansicht

Mit Hilfe von **Strg + C**, **Strg + X** bzw. **Strg + V** können Sie markierte Knoten kopieren, ausschneiden bzw. einfügen. Mit Hilfe der **Entf**-Taste können Sie markierte Knoten löschen.

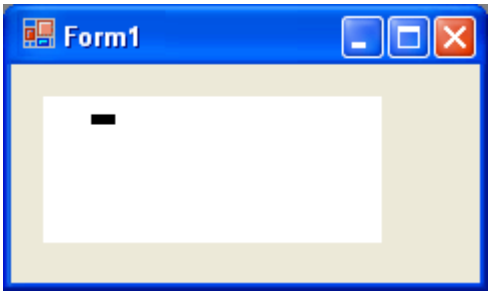
5.6 Knoten erzeugen

Varchart XTree besitzt zur Laufzeit zwei Modi: den Markiermodus und den Erzeugemodus. Knoten können Sie nur im Erzeugemodus anlegen. Um in den Erzeugemodus zu wechseln, klicken Sie mit der rechten Maustaste in den freien Diagrammbereich und wählen Sie im Kontextmenü den Befehl **Erzeugemodus**.

(Damit Sie Knoten interaktiv erzeugen können, muss auf der Eigenschaftenseite **Allgemeines** die Option **Erzeugung neuer Knoten zulassen** aktiviert sein.)

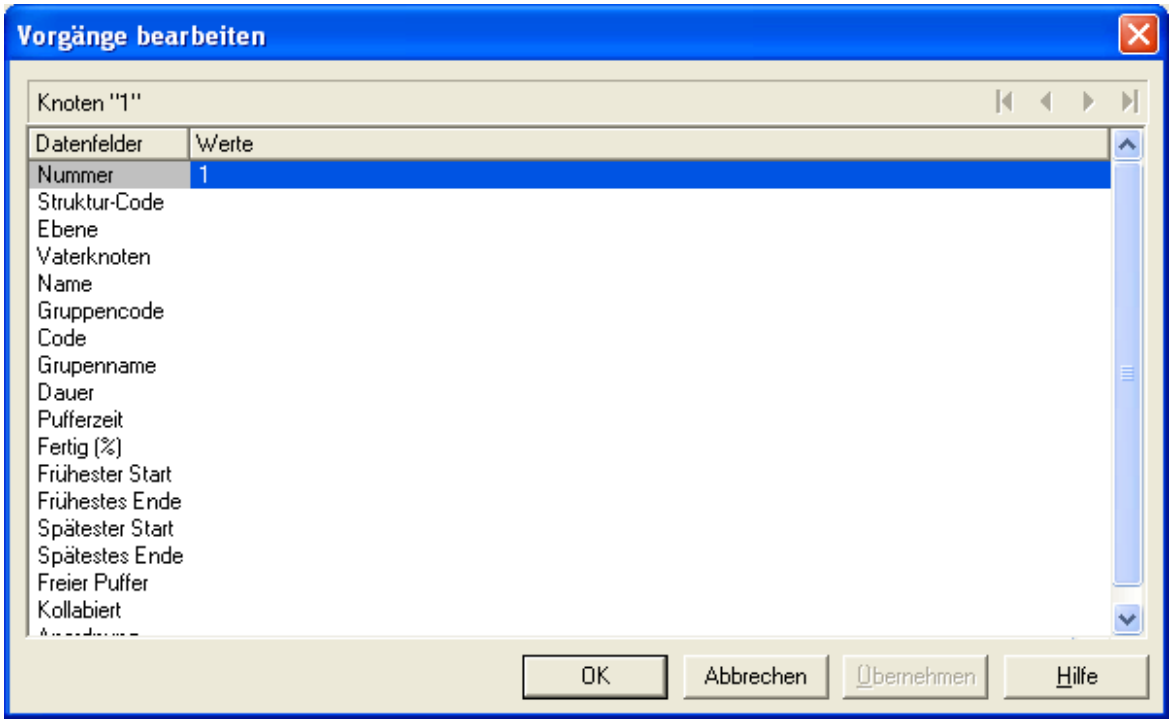


Im Erzeugemodus wird der Mauszeiger im leeren Diagrammbereich zu einem Knotenphantom in der Form eines kleinen schwarzen Rechtecks.

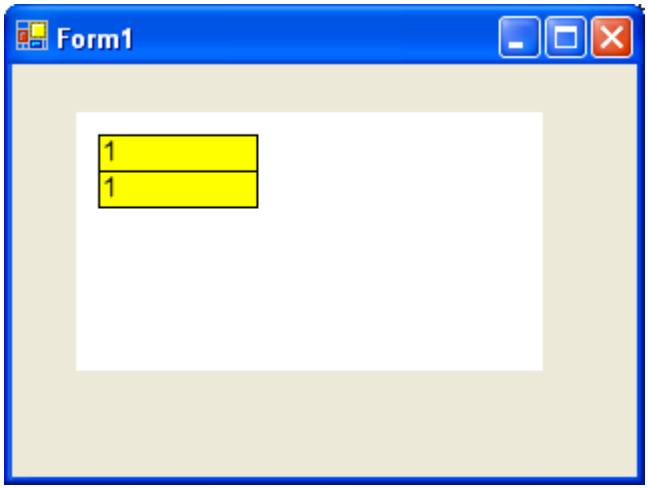


Klicken Sie nun einmal auf die linke Maustaste.

Wenn auf der Eigenschaftenseite **Allgemeines** die Option **Erzeugung neuer Knoten mit Dialog** aktiviert ist, erscheint zunächst das Dialogfeld **Vorgänge bearbeiten**, in dem die Daten dieses Knotens angezeigt werden.

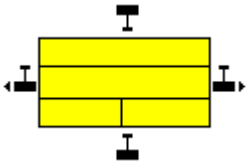


Zunächst ist nur das Datenfeld "Nummer" vorbesetzt (beim ersten Knoten mit dem Wert "1"). Sie können ggf. weitere Daten ergänzen, z. B. die Termindaten und eine Beschreibung. Sobald Sie die Daten mit **OK** bestätigen, wird der erste Knoten erzeugt.



War auf der Eigenschaftenseite **Allgemeines** die Option **Erzeugung neuer Knoten mit Dialog** nicht aktiviert, wird der erste Knoten angelegt, sobald Sie im Erzeugemodus mit der linken Maustaste in den Diagrammbereich klicken. Das Dialogfeld **Vorgänge bearbeiten** erscheint dann nicht.

Weitere Knoten können Sie durch Anlagern an einen bereits vorhandenen Knoten erzeugen. Wenn Sie den Mauszeiger im Erzeugemodus in die Nähe eines Knotens führen, verändert der Mauszeiger seine Form und zeigt an, wo der neue Knoten angelegt würde (als Vater, Sohn oder als rechter bzw. linker Bruder des Bezugsknotens).



5.7 Knoten markieren

Einzelnen Knoten markieren

Einen einzelnen Knoten markieren Sie, indem Sie ihn mit der linken Maustaste anklicken.

Knoten sammeln und toggeln

Sie können Knoten sammeln und toggeln, indem Sie sie bei gedrückter Strg-Taste mit der linken Maustaste anklicken. Jeder Klick auf einen markierten Knoten demarkiert diesen, jeder Klick auf einen nicht markierten Knoten markiert diesen.

Teilbaum markieren

Ein Teilbaum wird markiert, indem Sie bei gedrückter Umschalt-Taste den Vaterknoten des Teilbaums anklicken.

Alle Knoten demarkieren

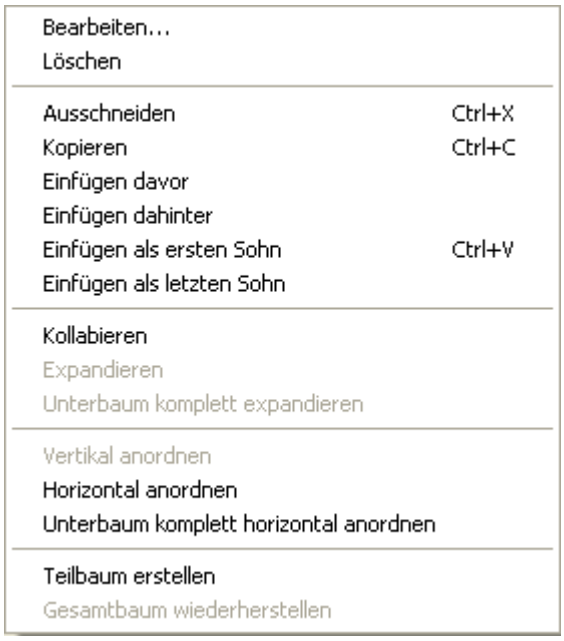
Sie können alle markierten Knoten demarkieren, indem Sie mit der linken Maustaste in den leeren Diagrammbereich klicken.

Markierungstyp festlegen

Auf der Eigenschaftenseite **Knoten** können Sie aus der Kombobox **Markierungstyp** auswählen, in welcher Weise Knoten markiert werden sollen.

5.8 Knoten löschen, ausschneiden, kopieren und einfügen

Mit Hilfe der entsprechenden Befehle des Kontextmenüs für Knoten können Sie die jeweils markierten Knoten löschen, ausschneiden, kopieren und die kopierten bzw. ausgeschnittenen Knoten einfügen.



Kontextmenü für Knoten

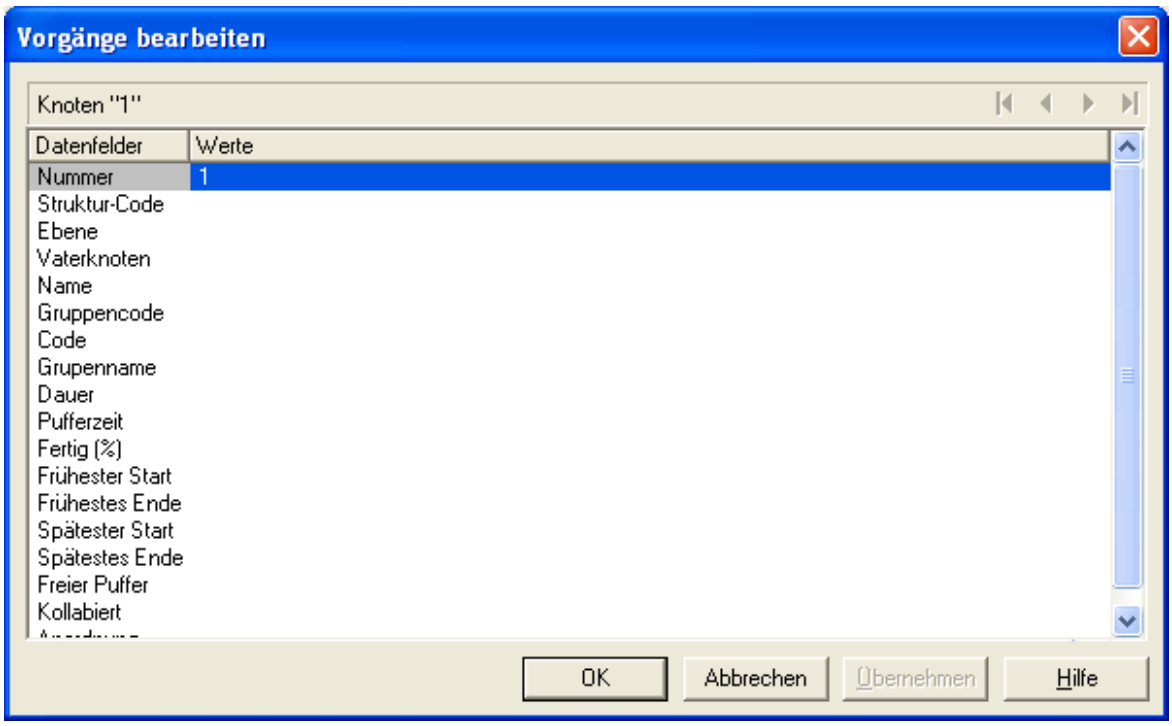
Damit ein Knoten eingefügt werden kann, muss ein Knoten markiert sein. Sie können dann wählen, ob der kopierte bzw. ausgeschnittene Knoten vor oder hinter dem markierten Knoten oder als erster oder letzter Sohn des markierten Knotens eingefügt werden soll.

Sie können markierte Knoten auch mit Hilfe der Entf-Taste löschen.

5.9 Knoten bearbeiten

Einen Knoten können Sie bearbeiten, indem Sie mit einem Doppelklick darauf das Dialogfeld **Vorgänge bearbeiten** öffnen. Hier können Sie alle Daten dieses Knotens bearbeiten.

Mehrere Knoten können Sie bearbeiten, indem Sie diese markieren und auf die rechte Maustaste klicken, während der Mauszeiger auf einem der Knoten steht. Dann öffnet sich das Kontextmenü für Knoten. Wählen Sie hier die Option **Bearbeiten**. Das Dialogfeld **Vorgänge bearbeiten** erscheint.



In diesem Dialogfeld können Sie nacheinander die Daten aller markierten Knoten bearbeiten.

Oberhalb der Tabelle wird die Identifikation (ID) des aktuellen Knotens angezeigt und ggf. um den wievielten Knoten der markierten Knoten es sich handelt.

Sie können hier alle Werte dieses Knotens bearbeiten und ggf. mit Hilfe der **Übernehmen**-Schaltfläche übernehmen. Mit Hilfe der Pfeil-Schaltflächen können Sie zwischen den Knoten navigieren.

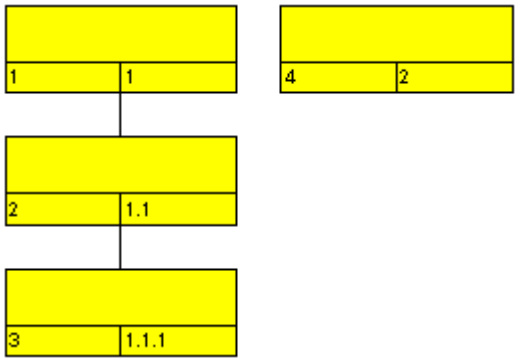
5.10 Knoten mit seinem Teilbaum umhängen

Sie können im Markiermodus einen Knoten mit dem gesamten darunter hängenden Teilbaum auf einmal per Drag & Drop umhängen. Es lässt sich immer nur ein einzelner Knoten mit seinem Teilbaum umhängen, auch wenn mehrere Knoten markiert sind.

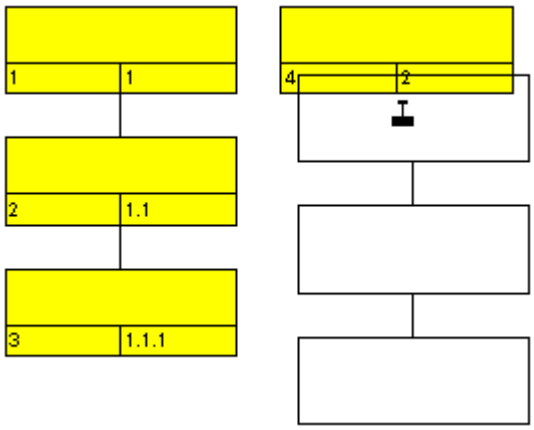
Markieren Sie den Knoten, den Sie zusammen mit seinen Sohnknoten umhängen möchten, und schieben Sie ihn mit gedrückter Maustaste an die gewünschte Position. Während Sie den Knoten verschieben, erscheint ein Phantom für den markierten Knoten mit seinem Teilbaum. Ziehen Sie das Phantom auf den Knoten, an dem Sie den verschobenen Knoten mit seinem Teilbaum anhängen möchten. Das Phantom des umgehängten Knotens muss dabei den Zielknoten teilweise überdecken. Sobald Sie die Maustaste loslassen, wird der Teilbaum umgehängt.

Beim Drag & Drop in Baum-Diagrammen wird der umgehängte Knoten stets dem Zielknoten als letzter Sohnknoten angehängt.

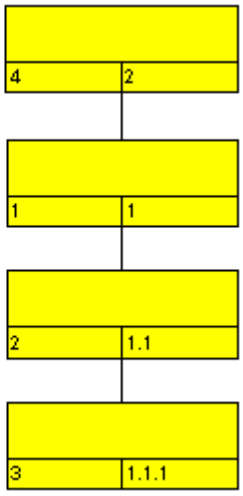
Beim Umhängen eines Knotens werden seine Verbindungen und Knotendaten automatisch angepasst.



Der linke Vaterknoten soll mit seinen Söhnen unter den rechten Knoten gehängt werden.



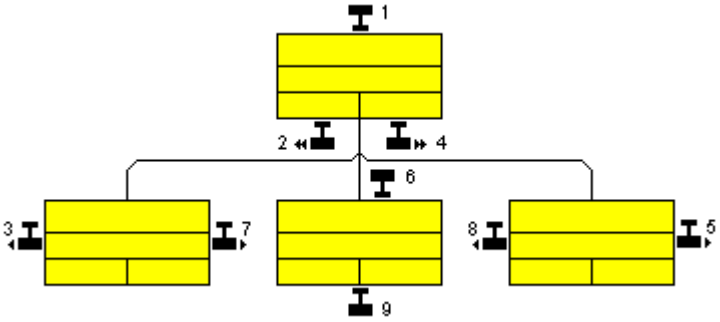
Während des Verschiebens erscheint ein Phantom, und der Mauszeiger zeigt an, wo der verschobene Knoten mit seinem Teilbaum beim Loslassen der Maustaste angehängt würde.



Der verschobene Teilbaum wurde unter dem zuvor rechten Knoten angehängt.

Hinweis: Die Reihenfolge von Sohnknoten lässt sich nicht direkt, sondern nur indirekt durch gezieltes Umhängen der Sohnknoten unter denselben Vaterknoten verändern.

In der folgenden Abbildung sind die verschiedenen Anlagerungsmöglichkeiten bei horizontaler Anordnung dargestellt:



1: Neuer Vaterknoten für den gesamten Baum

2: Neuer Sohnknoten ganz links außen

4: Neuer Sohnknoten ganz rechts außen

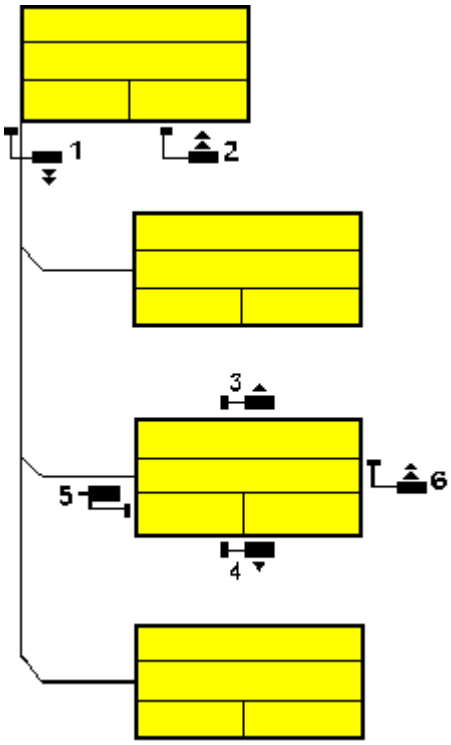
6: Neuer Vater zum Knoten

5, 7: Neuer Bruder rechts zum Knoten

3, 8: Neuer Bruder links zum Knoten

9: Neuer Sohn zum Knoten

Bei vertikaler Anordnung gibt es folgende Anlagerungsmöglichkeiten:



1: Neuer Sohnknoten ganz unten

2: Neuer Sohnknoten ganz oben

3: Neuer Bruder oberhalb

4: Neuer Bruder unterhalb

5: Neuer Vater zum Knoten

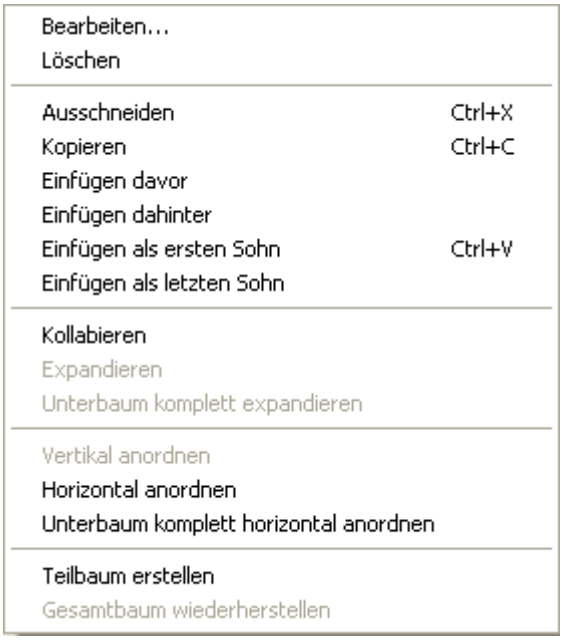
6: Neuer Sohn zum Knoten

5.11 Teilstrukturen horizontal und vertikal anordnen

Teilstrukturen von Baum-Diagrammen können ganz oder teilweise horizontal oder vertikal angeordnet werden.

- *Horizontale Anordnung:* Alle Knoten einer Ebene werden nebeneinander angeordnet. Der Port (Anknüpfungspunkt) der Verbindungslinie liegt mittig am unteren Rand des Vaterknotens und mittig am oberen Rand des Sohnknotens. Durch eine horizontale Anordnung beliebiger Teilstrukturen lässt sich die Höhe eines Baum-Diagramms verringern.
- *Vertikale Anordnung:* Alle Knoten einer Ebene werden untereinander angeordnet. Der Port der Verbindungslinie liegt in der unteren linken Ecke des Vaterknotens und in der Mitte des linken Randes des Sohnknotens. Durch eine vertikale Anordnung beliebiger Teilstrukturen lässt sich die Breite des Baum-Diagramms verringern.

Um die Befehle zum horizontalen und vertikalen Anordnen von Bäumen zur Verfügung zu haben, markieren Sie einen Knoten und klicken Sie auf die rechte Maustaste. Es erscheint das folgende Kontextmenü, wobei jeweils nur die verfügbaren Befehle aktiviert sind.

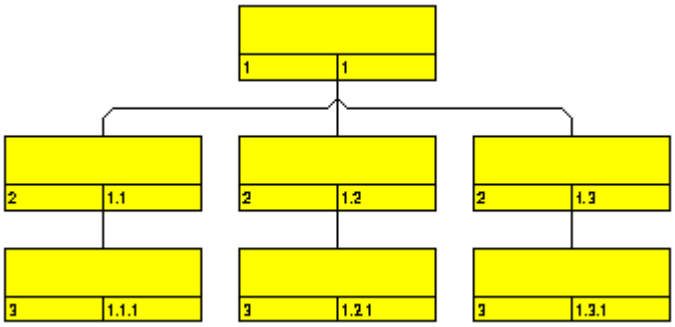


Um eine Teilstruktur eines Baum-Diagramms horizontal anordnen zu lassen, markieren Sie den obersten Knoten dieser Teilstruktur und wählen anschließend im Kontextmenü den Befehl **Horizontal anordnen**. Die Teilstrukturen des markierten Knotens werden dann horizontal angeordnet, aber nur eine Ebene tief. Auf die Anordnungen in den nächst tieferen Ebenen hat der Befehl **Horizontal anordnen** keine Auswirkung.

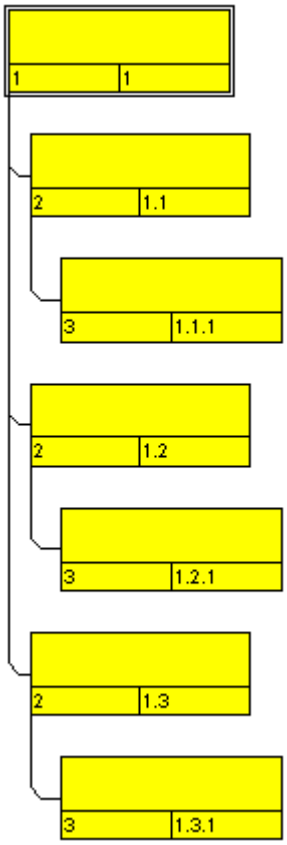
Mit dem Befehl **Unterbaum komplett horizontal anordnen** werden die Teilbäume unter den markierten Knoten vollständig, d. h. über alle Ebenen, horizontal angeordnet.

Mit dem Befehl **Vertikal anordnen** werden die Teilbäume unter den markierten Knoten vertikal angeordnet, und zwar beginnend beim höchsten ausgewählten Wurzelknoten.

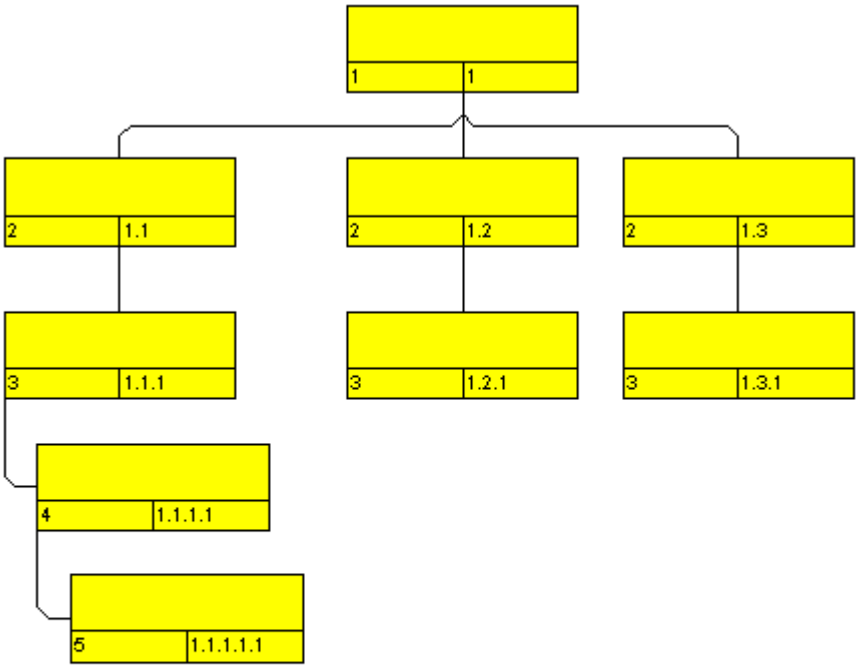
Hinweis: Falls ein Baum nicht vollständig vertikal angeordnet wird, prüfen Sie die vereinbarte maximale Baumhöhe auf der Eigenschaftenseite **Layout**. Die Anzahl der Ebenen bei der vertikalen Anordnung wird durch die Angabe unter **Max. Baumhöhe** begrenzt.

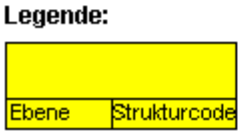


alle Ebenen horizontal angeordnet



alle Ebenen vertikal angeordnet





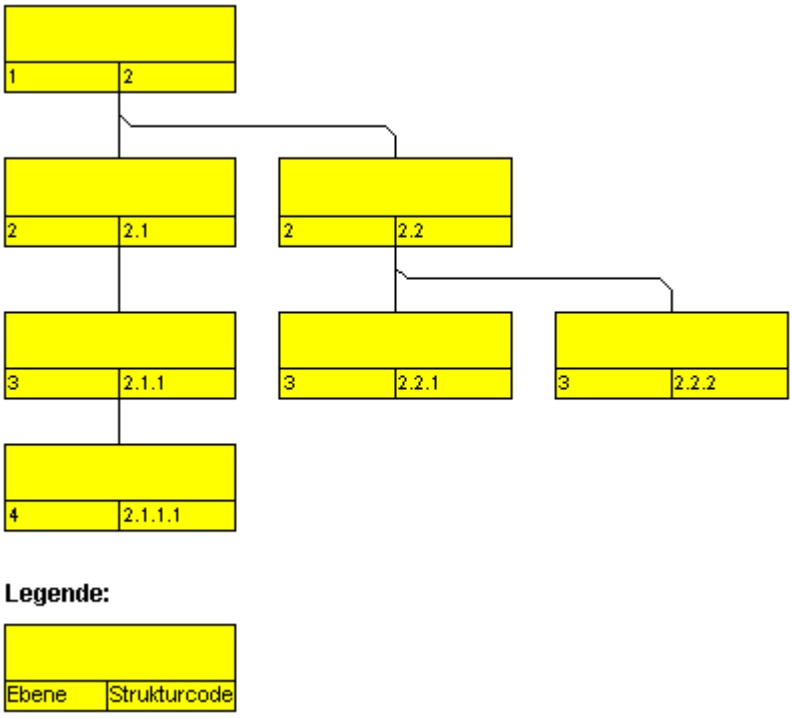
Beispiel für ein Baumdiagramm mit horizontal und vertikal angeordneten Teilstrukturen

Hinweis: Änderungen in der Anordnung wirken sich auch auf kollabierte Teilbäume aus.

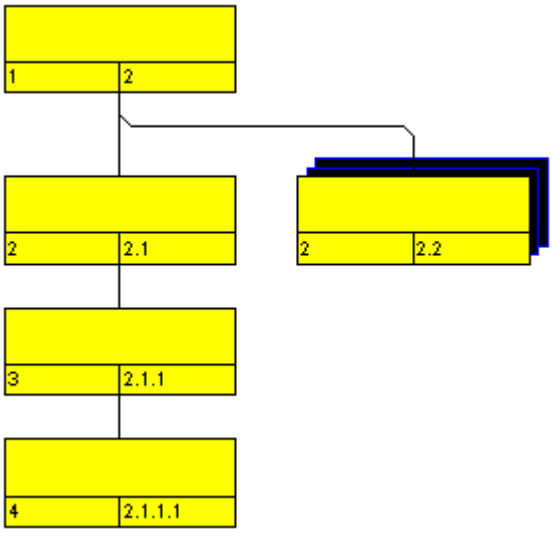
5.12 Teilstrukturen kollabieren und expandieren

Durch das Kollabieren von Teilstrukturen auf Gliederungsknoten lassen sich auch sehr komplexe Strukturen übersichtlich gestalten. Wenn Sie beispielsweise nur bestimmte Teilstrukturen präsentieren möchten, kollabieren Sie einfach alle anderen Teilstrukturen. Dadurch, dass die kollabierten Teilstrukturen erhalten bleiben, geht die Information über die Gesamtstruktur nicht verloren.

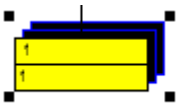
In Baum-Diagrammen lässt sich jede beliebige Teilstruktur auf einen einzigen Knoten, den Gliederungsknoten, kollabieren und auf Wunsch wieder in ihre ursprüngliche Form expandieren. Als Gliederungsknoten wird jeweils der Wurzelknoten der Teilstruktur, die kollabiert werden soll, verwendet.



Expandierter Baum



In Form eines Gliederungsknotens kollabierte Teilstruktur



Vollständig kollabierter Baum

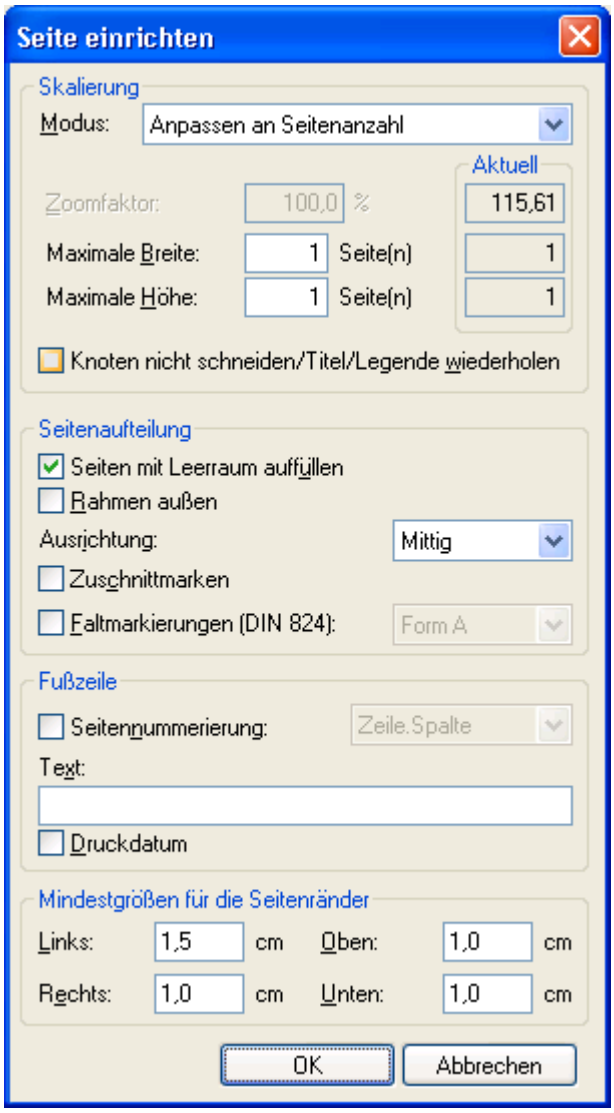
Mit dem Befehl **Kollabieren** aus dem Kontextmenü für Knoten können Sie die Teilstrukturen, die zu den markierten Wurzelknoten gehören, kollabieren. Die markierten Wurzelknoten verwandeln sich damit in Gliederungsknoten, die jeweils die nun verborgenen Teilstrukturen repräsentieren. Kollabierte Knoten werden automatisch mit dem Knotenaussehen "Kollabiert" versehen.

Mit dem Befehl **Expandieren** aus dem Kontextmenü für Knoten können Sie die Teilstrukturen wieder expandieren, die durch die markierten Gliederungsknoten repräsentiert werden. Es wird jeweils nur die nächsttieferliegende Ebene expandiert. Befinden sich in dieser Ebene weitere Gliederungsknoten, bleiben diese erhalten.

Mit dem Befehl **Unterbaum komplett expandieren** aus dem Kontextmenü für Knoten können Sie die markierten Gliederungsknoten vollständig expandieren.

5.13 Seite einrichten

Alle Einstellungen zum Seitenlayout können Sie im Dialog "Seite einrichten" vornehmen. Sie gelangen in diesen Dialog entweder über den entsprechenden Befehl im Diagramm-Kontextmenü über aus der Druckvorschau durch Klick auf die gleichnamige Schaltfläche.



Modus

Durch Auswahl einer Skalierungsart aus der Drop-Down-Liste und der entsprechenden Werte bei **Zoomfaktor** bzw. **Maximale Breite/Höhe** bestimmen Sie den Maßstab der Darstellung bei der Ausgabe. Nach Klick auf **Übernehmen** werden unter **Aktuell** die Werte angezeigt, die sich aus Ihren Einstellungen ergeben.

Zoomfaktor

Ein Skalierungsfaktor von 100 % entspricht der Originalgröße, ein kleinerer Wert bewirkt eine entsprechende Verkleinerung, ein größerer Wert eine Vergrößerung.

Anpassen an Seitenzahl

Durch Auswahl dieser Option können Sie die Anzahl der Seiten in Höhe und Breite vorgeben, auf die das Diagramm bei der Ausgabe maximal aufgeteilt werden soll (**max. Höhe, max. Breite**). Die Diagramme werden so groß wie möglich, aber ohne Verzerrungen dargestellt.

Knoten nicht schneiden/Titel/Legende wiederholen

Aktivieren Sie dieses Kontrollkästchen, damit bei der Ausgabe des Diagramms auf mehreren Seiten Knoten nicht durchtrennt werden, und damit Titel und Legende - sofern vorhanden - auf jeder Seite ausgegeben werden.

Zeitskala an Breite der Seiten anpassen

Diese Option führt dazu, dass der Platz auf den Druckseiten besser ausgenutzt wird:

- Bei einer Skalierung über eine feste Anzahl Seiten: Der Zoomfaktor wird so berechnet, dass die eingestellte Anzahl Seiten in der Höhe voll bedruckt wird. Gleichzeitig wird die Zeiskala so gestaucht oder gestreckt, dass die eingestellte Anzahl Seiten in der Breite voll ausgenutzt wird.
- Bei einer Skalierung über einen Zoomfaktor: Die Zeitskala wird so gestaucht oder gestreckt, dass die eingestellte Anzahl Seiten in der Breite voll ausgenutzt wird.

Seiten mit Leerraum auffüllen

Mithilfe dieser Option können Sie festlegen, ob zwischen dem Diagramm und den Boxen für Titel und Legende so viel Platz gelassen wird, dass die Boxen auf jeder Druckseite immer in voller Breite gedruckt werden können und fest am Blattrand positioniert sind. Wenn diese Option nicht ausgewählt ist, werden die Boxen ohne Zwischenraum am Diagramm gedruckt und können dann je nach Diagramm auf den verschiedenen Druckseiten in der Breite variieren.

Rahmen außen

Aktivieren Sie dieses Kontrollkästchen, damit ein Rahmen um das Diagramm herum ausgegeben wird. Wenn die Option **Knoten nicht durchtrennen/ Titel wiederholen** ausgewählt ist, erhält jede Seite einen Rahmen, andernfalls wird ein Rahmen um das gesamte Diagramm gezogen.

Ausrichtung

Bestimmen Sie die Ausrichtung des Diagramms auf dem Blatt.

Zuschnittmarken

Wurde dieses Kontrollfeld aktiviert, wird das Diagramm mit Zuschnittmarken versehen, die das Zusammenkleben der ausgedruckten Einzelseiten zu einer Gesamtgrafik erleichtern.

Faltmarkierungen (DIN 824)

In der DIN Norm 824 ist für Bauzeichnungen eine ganz bestimmte Art der Faltung vorgeschrieben, mit der man die Zeichnung auf DIN A4-Größe zusammenfalten kann. Die Ausgabe von entsprechenden Faltmarkierungen auf Ihrem Diagramm erleichtert Ihnen die Faltung. Folgende Möglichkeiten stehen zur Verfügung:

- **Form A:** mit Heftrand auf der linken Seite, damit die Zeichnung gelocht und in einem Ordner abgeheftet werden kann.
- **Form B:** insgesamt etwas schmaler, damit ein Heftstreifen angebracht werden kann, der dann gemeinsam mit der Zeichnung die Breite von DIN A4 erreicht.
- **Form C:** die gefaltete Zeichnung die gefaltete Zeichnung wird nicht gelocht, sondern in eine Sichthülle gelegt.

Die vorliegenden Faltmarkierungen können für jedwedes Zielformat ausgegeben werden, während die DIN 824 explizit nur die Formate DIN A0 bis A3 kennt.

Seitennummerierung

Ist dieses Kontrollkästchen aktiviert, wird auf jeder Seite unten links die Seitennummer ausgegeben. Folgende Möglichkeiten stehen dabei zur Auswahl:

- **Zeile.Spalte**: Sinnvoll, wenn das Diagramm sich auf mehr als eine Seite in der Länge als auch in der Breite erstreckt. Vor dem Punkt wird die Position der Seite in der vertikalen, dann die in der horizontalen Reihenfolge ausgegeben.
- **Spalte.Zeile**: Sinnvoll, wenn das Diagramm sich auf mehr als eine Seite in der Länge als auch in der Breite erstreckt. Vor dem Punkt wird die Position der Seite in der horizontalen, dann die in der vertikalen Reihenfolge ausgegeben.
- **Seite/Anzahl**: Zuerst erscheint die aktuelle Seitenzahl, danach die Anzahl der Gesamtseiten: 1/6, 2/6 etc.

Text

Aktivieren Sie dieses Kontrollkästchen, um jede Seite unten links mit einem beliebigen Text zu versehen. Dieser Zusatztext wird ggf. rechts von der Seitennummer ausgegeben.

Für die Seitennummerierung können Sie in die Zeile **Zusatztext** folgende Platzhalter eingeben, die dann beim Ausdruck durch die entsprechenden Inhalte ersetzt werden:

{PAGE}	= fortlaufende Seitennummer
{NUMPAGES}	= Gesamtanzahl der Seiten
{ROW}	= Zeilenposition des Ausschnitts im Gesamtdiagramm
{COLUMN}	= Spaltenposition des Ausschnitts im Gesamtchart

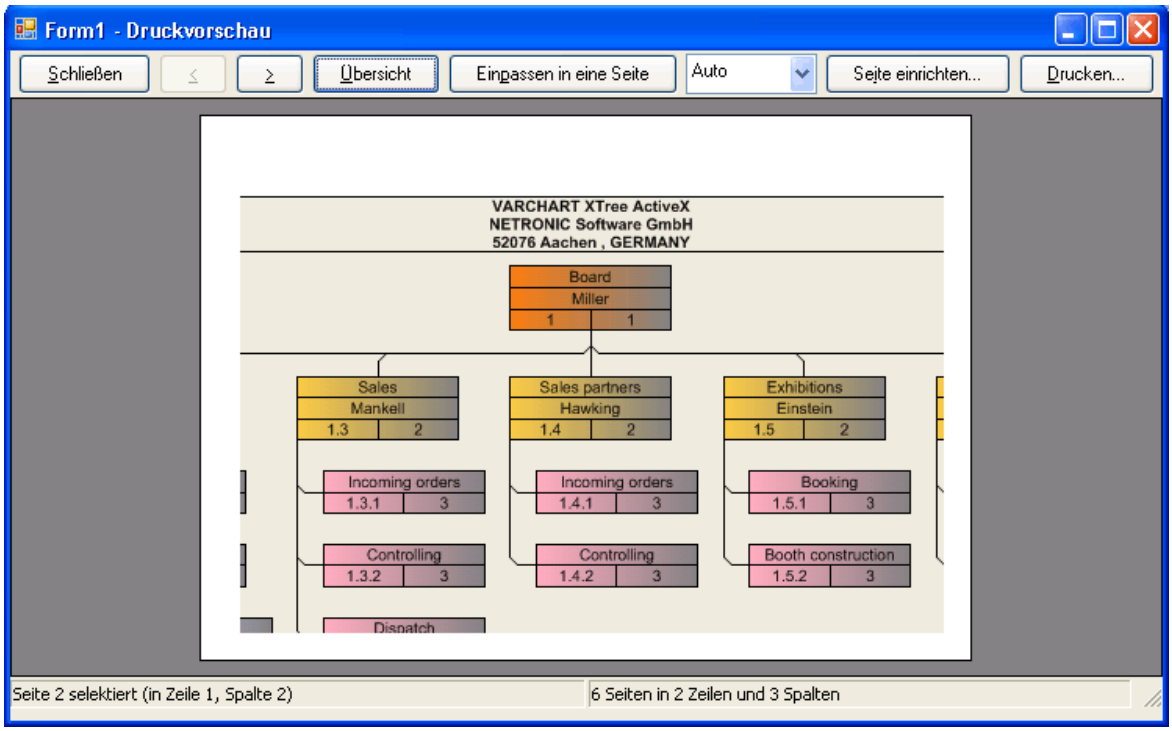
Druckdatum

Ist dieses Kontrollkästchen aktiviert, wird auf jeder Seite unten links das Druckdatum ausgegeben. Das Druckdatum wird ggf. rechts von der Seitennummer und dem Zusatztext ausgegeben.

Seitenränder

Über die Felder **Oben**, **Unten**, **Links** und **Rechts** legen Sie den Raum zwischen Papierrand und dem Diagramm in cm fest. Mindeströnder, die aus technischen Gründen bei den Druckern entstehen, können nicht unterschritten werden. Bei Druckern mit Mindestrandvorgaben gelten die hier eingetragenen Werte für die Seitenränder zusätzlich zu den Mindestwerten, sodass die tatsächlich ausgegebenen Seitenränder größer sind als hier festgelegt.

5.14 Druckvorschau



Vor dem Drucken können Sie das Diagramm in der Druckvorschau prüfen. Es wird so auf dem Bildschirm dargestellt, wie es im Dialogfeld **Seite einrichten** festgelegt ist und wie es gedruckt wird.

Sie können hier einzelne Seiten oder die Gesamtübersicht über alle Seiten Ihrer Darstellung ansehen oder einen Ausschnitt Ihres Diagramms interaktiv vergrößern und anschliessend drucken.

Die Statuszeile informiert über die Gesamtseitenzahl und die horizontale und vertikale Aufteilung der Seiten. In der Ansicht **Einzelseite** wird zusätzlich noch die aktuelle Seitenzahl angezeigt.

Schließen

Sie verlassen die Druckvorschau und gelangen zurück in die Darstellung.

<

*Nur aktiv, wenn die Schaltfläche **Einzel** gedrückt wurde*. Wenn das Diagramm sich über mehrere Seiten erstreckt, können Sie sich die Seiten einzeln ansehen. Mit Hilfe dieser Schaltfläche gelangen Sie zur vorangehenden Seite. Sie bewegen sich über die Seiten von rechts nach links in aufsteigenden Zeilen.

>

*Nur aktiv, wenn die Schaltfläche **Einzel** gedrückt wurde*. Wenn das Diagramm sich über mehrere Seiten erstreckt, können Sie sich die Seiten einzeln ansehen. Mit Hilfe dieser Schaltfläche gelangen Sie zur nächsten Seite. Sie bewegen sich über die Seiten von links nach rechts in absteigenden Zeilen.

Einzelseite/Übersicht

Wenn das Diagramm sich über mehrere Seiten erstreckt, können Sie sich die Seiten entweder einzeln oder in der Übersicht ansehen. In der **Übersicht** sehen Sie alle Seiten - je nach Seitenanzahl mehr oder weniger stark verkleinert, im Darstellungsmodus **Einzelseite**, wird zunächst die erste Seite des Diagramms einzeln und vergrößert dargestellt. Mit ≥ oder ≤ können Sie dann durch die Seiten blättern. Mit einem Doppelklick auf eine Seite wechseln Sie bequem zwischen den beiden Darstellungsarten **Einzelseite** und **Übersicht**.

Ein Ausschnitt Ihres Diagramms lässt sich im Darstellungsmodus **Einzelseite** interaktiv vergrößern, indem Sie bei gedrückter linker Maustaste ein Rechteck um den zu vergrößernden Bildausschnitt ziehen. Sobald Sie die linke Maustaste loslassen, wird der eingerahmte Bildausschnitt entsprechend vergrößert und statt der Schaltfläche **Drucken** erscheint die Schaltfläche Ausschnitt drucken... über die Sie den Bildausschnitt dann in der aktuellen Vergrößerung drucken können. Beachten Sie bitte, dass der in der Druckvorschau interaktiv gewählte Vergrößerungsfaktor nicht den Skalierungsfaktor im Dialogfeld **Seite einrichten** verändert.

Einpassen in eine Seite

Mit dieser Schaltfläche läßt sich ein mehrseitiges Diagramm auf eine Seite verkleinern. Auch hier können Teile des Diagramms, wie unter **Einzelseite/Übersicht** beschrieben, interaktiv vergrößert und anschließend gedruckt werden.

Zoomfaktor

Wählen Sie einen Zoomfaktor aus der Liste oder definieren einen individuellen, um die Darstellungsgröße ihres Diagramms für die Druckvorschau zu verändern. Dies ist nur im Modus "Einzelseite" möglich. Der Zoomfaktor lässt sich auch durch Drehen des Mausrades bei gedrückter <STRG>-Taste verändern. Er hat keinen Einfluss auf den späteren Druck. Je

nach gewählter Größe werden vertikale und/oder horizontale Bildlaufleisten angezeigt. Auch das Mausrad kann zum Bewegen des Bildes verwendet werden (ohne Umschalttaste vertikal, mit Umschalttaste horizontal).

Der Zoomfaktor **Auto** ist voreingestellt und verkleinert bzw. vergrößert das Blatt immer so, dass es bildschirmfüllend dargestellt wird.

Seite einrichten

Sobald Sie auf diese Schaltfläche klicken, gelangen Sie in das Dialogfeld **Seite einrichten** und können dort Änderungen am Seitenlayout vornehmen.

Drucker einrichten

*Nur sichtbar, wenn auf der Eigenschaftenseite **Allgemeines** die Option **PrintDlgEx Dialog verwenden** nicht gewählt wurde.*

Sobald Sie auf diese Schaltfläche klicken, gelangen Sie in das Windows-Dialogfeld **Drucker einrichten** und können dort Änderungen an den Druckereinstellungen vornehmen.

Drucken/Ausschnitt drucken

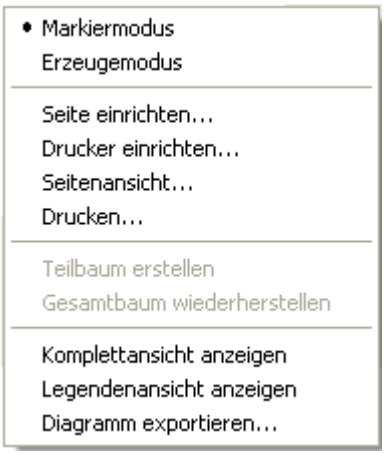
Über diese Schaltfläche gelangen Sie in das Windows-Dialogfeld **Drucken** und können von dort aus den Druckvorgang einleiten.

Wenn Sie in der Druckvorschau einen Ausschnitt interaktiv gezoomt haben, ändert die Schaltfläche ihre Beschriftung zu **Ausschnitt drucken**. Wenn Sie sie anklicken, ist im Windows-Dialogfeld **Drucken** die Option **Markierung** bereits ausgewählt. Durch Klick auf **OK** wird der am Bildschirm dargestellte Ausschnitt gedruckt.

Beachten Sie bitte, dass der in der Druckvorschau interaktiv gewählte Vergrößerungsfaktor nicht den Skalierungsfaktor im Dialogfeld **Seite einrichten** verändert.

5.15 Kontextmenü für das Diagramm

Wenn Sie die rechte Maustaste drücken, wenn der Mauszeiger im Diagrammbereich (nicht auf einem Knoten) steht, öffnet sich das folgende Kontextmenü:



Markiermodus

Der Markiermodus ist der Standardmodus.

Erzeugemodus

Dieser Modus kann nur eingeschaltet werden, wenn auf der Eigenschaftenseite **Allgemeines** die Option **Neue Knoten zulassen** aktiviert ist.

Der Mauszeiger wird im leeren Diagrammbereich zu einem Knotenphantom, das anzeigt, wo Sie durch einen Klick auf die linke Maustaste einen neuen Knoten erzeugen könnten.

Wenn auf der Eigenschaftenseite **Allgemeines** außerdem die Option **Erzeugung neuer Knoten mit Dialog> aktiviert ist, öffnet sich beim Loslassen der Maustaste das Dialogfeld Vorgänge bearbeiten**. Hier können Sie alle Daten des neu angelegten Knotens bearbeiten.

Der Erzeugemodus lässt sich auf zwei Arten aktivieren:

1. über das standardmäßige Kontextmenü für das Diagramm
2. über das Setzen der Eigenschaft **InteractionMode** auf den Wert **vcCreateNodes**.

Seite einrichten

Sie gelangen in das Dialogfeld **Seite einrichten**.

Drucker einrichten

*Nur aktiv, wenn auf der Eigenschaftenseite **Allgemeines** die Option **PrintDlgEx Dialog verwenden** nicht gewählt wurde.*

Sie gelangen in das Windows-Dialogfeld **Drucker einrichten**.

Druckvorschau

Sie gelangen in das Dialogfeld **Druckvorschau**.

Drucken

Wenn Sie diese Option wählen, gelangen Sie in das Windows-Dialogfeld **Drucken**.

Teilbaum erstellen

(nur aktiv, wenn Knoten markiert sind) Wählen Sie diese Option, um einen Teilbaum aus den markierten Knoten darzustellen.

Gesamtbaum wiederherstellen

*(nur aktiv, wenn zuvor die Option **Teilbaum erstellen** gewählt wurde)* Wählen Sie diese Option, um den Gesamtbaum wiederherzustellen.

Komplettansicht anzeigen

Über diesen Menüpunkt können Sie die Komplettansicht ein- oder ausschalten. Die Komplettansicht ist ein zusätzliches Fenster, in dem das komplette Diagramm angezeigt wird. Ein farbiger Rahmen markiert den aktuell im Hauptfenster dargestellten Diagrammausschnitt. Wenn Sie diesen Rahmen mit der Maus verschieben, wird auch im Hauptfenster der entsprechende Diagrammausschnitt angezeigt.

Die Komplettansicht lässt sich auch über die Eigenschaft **VcWorldView.Visible** an- oder abschalten.

Legendenansicht anzeigen

Über diesen Menüpunkt können Sie die Legendenansicht ein- oder ausschalten. Die Legende erscheint in einem zusätzlichen Fenster.

Sie können die Legendenansicht auch über die Eigenschaft **VcLegendView.Visible** an- oder abschalten.

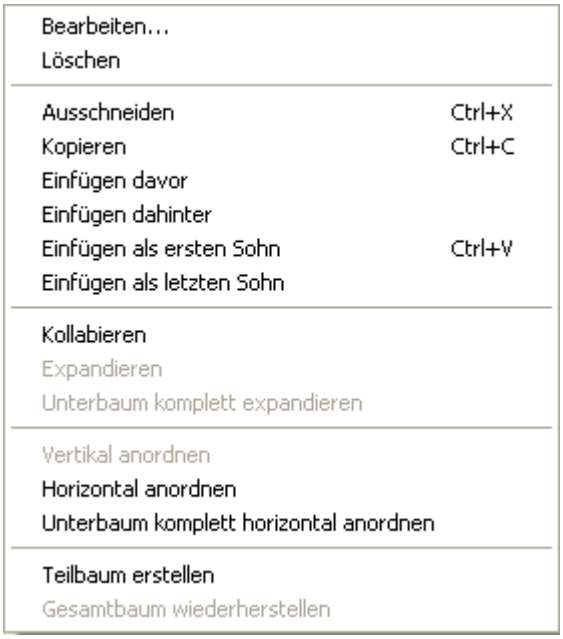
Diagramm exportieren

Wenn Sie diese Option wählen, gelangen Sie in das Windows-Dialogfeld **Speichern unter**, in dem Sie das dargestellte Diagramm als Grafikdatei speichern können.

Sie können diesen Dialog auch mit der VcTree-Methode **ShowExport-GraphicsDialog** aufrufen.

5.16 Kontextmenü für Knoten

Wenn ein oder mehrere Knoten markiert sind und Sie die rechte Maustaste drücken, öffnet sich das folgende Kontextmenü:



Bearbeiten

Wenn Sie diese Option wählen, öffnet sich das Dialogfeld **Vorgänge bearbeiten**.

Löschen

Wenn Sie diese Option wählen, werden die markierten Knoten gelöscht.

Ausschneiden

Mit diesem Befehl können Sie die markierten Knoten ausschneiden.

Kopieren

Mit diesem Befehl können Sie die markierten Knoten kopieren.

Einfügen davor

Damit ein Knoten eingefügt werden kann, muss ein Knoten markiert sein. Wählen Sie diesen Befehl, um den kopierten bzw. ausgeschnittenen Knoten vor dem markierten Knoten einzufügen.

Einfügen dahinter

Wählen Sie diesen Befehl, um den kopierten bzw. ausgeschnittenen Knoten hinter dem markierten Knoten einzufügen.

Einfügen als ersten Sohn

Wählen Sie diesen Befehl, um den kopierten bzw. ausgeschnittenen Knoten als ersten Sohn des markierten Knotens einzufügen.

Einfügen als letzten Sohn

Wählen Sie diesen Befehl, um den kopierten bzw. ausgeschnittenen Knoten als letzten Sohn des markierten Knotens einzufügen.

Kollabieren

Mit diesem Befehl können Sie die Teilstrukturen, die zu den markierten Wurzelknoten gehören, wegklappen (kollabieren). Die markierten Wurzelknoten verwandeln sich damit in Gliederungsknoten, die jeweils die nun verborgenen Teilstrukturen repräsentieren.

Dadurch, dass die kollabierten Teilstrukturen erhalten bleiben, geht die Information über die Gesamtstruktur nicht verloren.

Expandieren

Mit diesem Befehl können Sie die Teilstrukturen wieder aufklappen (expandieren), die durch die markierten Gliederungsknoten repräsentiert werden. Es wird jeweils nur die nächsttieferliegende Ebene expandiert. Befinden sich in dieser Ebene weitere Gliederungsknoten, bleiben diese erhalten.

Unterbaum komplett expandieren

Mit diesem Befehl können Sie die markierten Gliederungsknoten vollständig expandieren.

Vertikal anordnen

Um die Breite des Baum-Diagramms zu verringern, können Sie beliebige Teilstrukturen vertikal anordnen. Bei vertikaler Anordnung werden alle Knoten einer Ebene untereinander angeordnet. Der Port der Verbindungslinie

liegt dann in der unteren linken Ecke des Vaterknotens und in der Mitte des rechten Randes des Sohnknotens.

Mit dem Befehl **Vertikal anordnen** werden die Teilbäume unter den markierten Knoten vertikal angeordnet, und zwar beginnend beim höchsten ausgewählten Wurzelknoten.

Hinweis: Falls ein Baum nicht vollständig vertikal angeordnet wird, prüfen Sie die vereinbarte maximale Baumhöhe auf der Eigenschaftenseite **Layout**. Die Anzahl der Ebenen bei der vertikalen Anordnung wird durch die Angabe unter **Max. Baumhöhe** begrenzt.

Horizontal anordnen

Um die Höhe eines Baum-Diagramms zu verringern, können Sie beliebige Teilstrukturen horizontal anordnen. Bei horizontaler Anordnung werden alle Knoten einer Ebene nebeneinander angeordnet. Die Verbindungslinie beginnt dann mittig am unteren Rand des Vaterknotens und endet mittig am oberen Rand des Sohnknotens.

Um eine Teilstruktur eines Baum-Diagramms horizontal anordnen zu lassen, markieren Sie den obersten Knoten dieser Teilstruktur und wählen Sie anschließend den Befehl **Horizontal anordnen**. Die Teilstrukturen des selektierten Knotens werden dann horizontal angeordnet, aber nur eine Ebene tief. Auf die Anordnungen in den nächst tieferen Ebenen hat der Befehl **Horizontal anordnen** keine Auswirkung.

Unterbaum komplett horizontal anordnen

Mit diesem Befehl werden die Teilbäume unter den markierten Knoten vollständig, d. h. über alle Ebenen, horizontal angeordnet.

Teilbaum erstellen

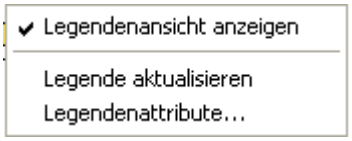
Wählen Sie diese Option, um einen Teilbaum aus den markierten Knoten darzustellen.

Gesamtbaum wiederherstellen

*(nur aktiv, wenn zuvor die Option **Teilbaum erstellen** gewählt wurde)*
Wählen Sie diese Option, um den Gesamtbaum wiederherzustellen.

5.17 Kontextmenü für die Legende

Wenn Sie mit der rechten Maustaste auf die Legende klicken, erscheint das folgende Menü:



Legendenansicht anzeigen

Über diesen Menüpunkt können Sie die Legendenansicht ein- oder ausschalten.

Legende aktualisieren

Über diesen Menüpunkt können Sie die Legendenansicht aktualisieren.

Eine Aktualisierung über das Menü kann nötig sein, da nach Änderungen im Diagramm die Legende nicht automatisch aktualisiert wird. Werden also zum Beispiel Knoten hinzugefügt oder gelöscht, muss eine Aktualisierung entweder über das Kontextmenü oder durch Aus- und Einschalten der Legende durchgeführt werden. Dies gilt auch für das Laden von Knoten. Wenn für die Legendenansicht auf der Eigenschaftenseite **Zusätzliche Ansichten** die Option **Beim Start sichtbar** eingestellt wurde, aber zum Zeitpunkt des Aufbaus noch keine Knoten geladen waren, bleibt die Legende bis zur Aktualisierung leer.

Legendenattribute

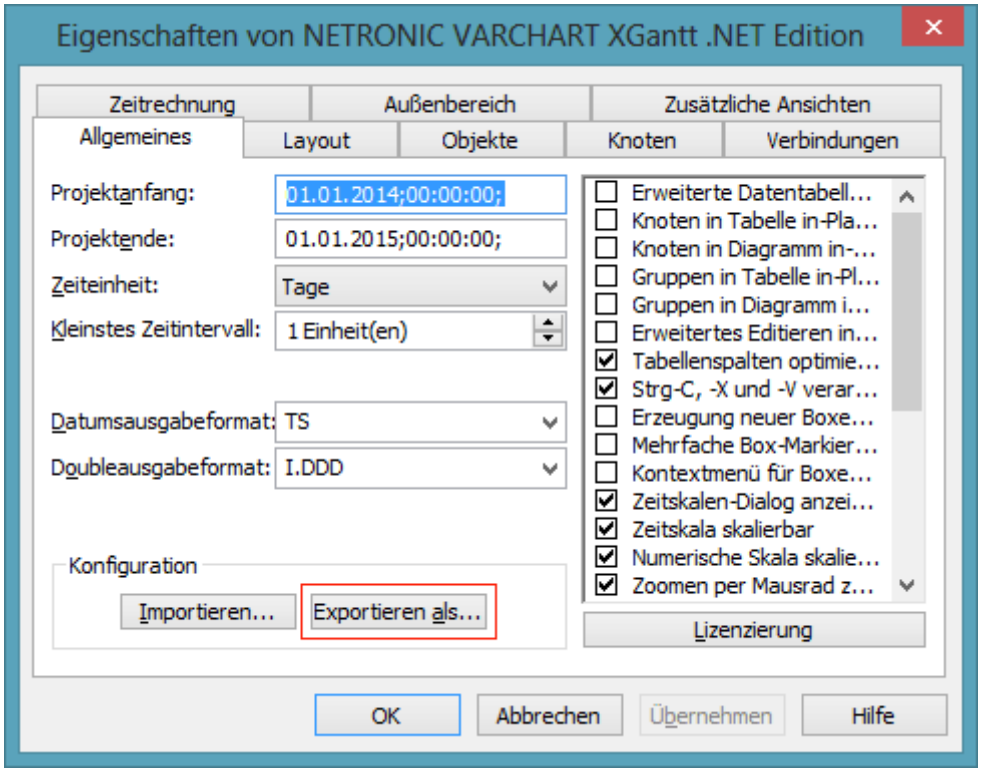
Über diesen Menüpunkt öffnen Sie den gleichnamigen Dialog, in dem Sie Einstellungen zum Legendentitel, den Legendenelementen und den Rändern der Legende vornehmen können. Weitere Information zu diesem Dialog finden Sie in Kapitel 4.44 "Dialogfeld Legendenattribute".

6 Häufig gestellte Fragen

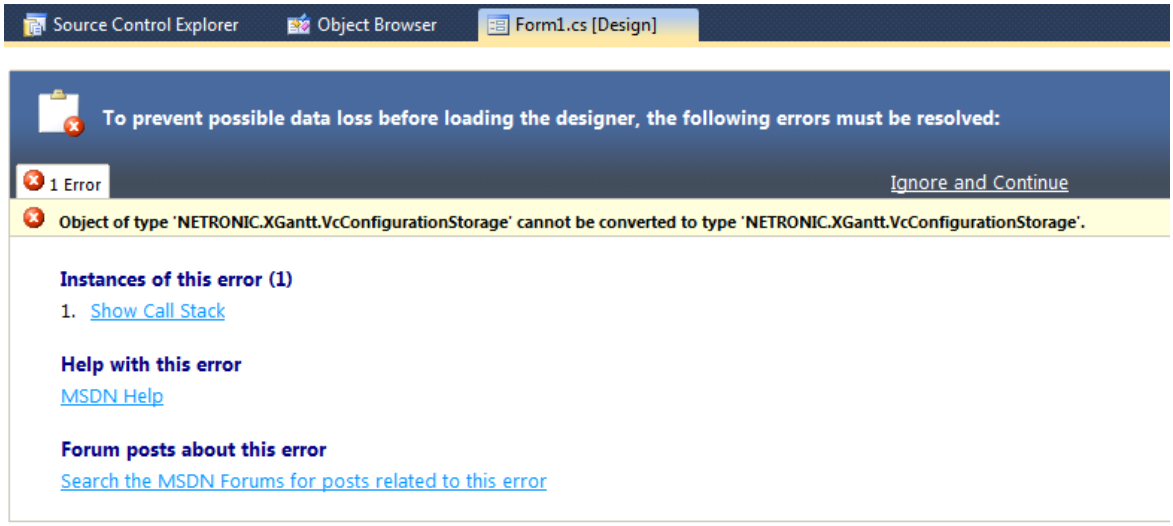
6.1 Was muss ich tun, wenn ich von VARCHART XGantt 4.4 für .NET nach XGantt 5.0 umsteigen möchte?

6.2 Was muss ich tun, wenn ich im Rahmen eines Service Releases auf einen neuen Build von VARCHART XGantt umsteigen möchte?

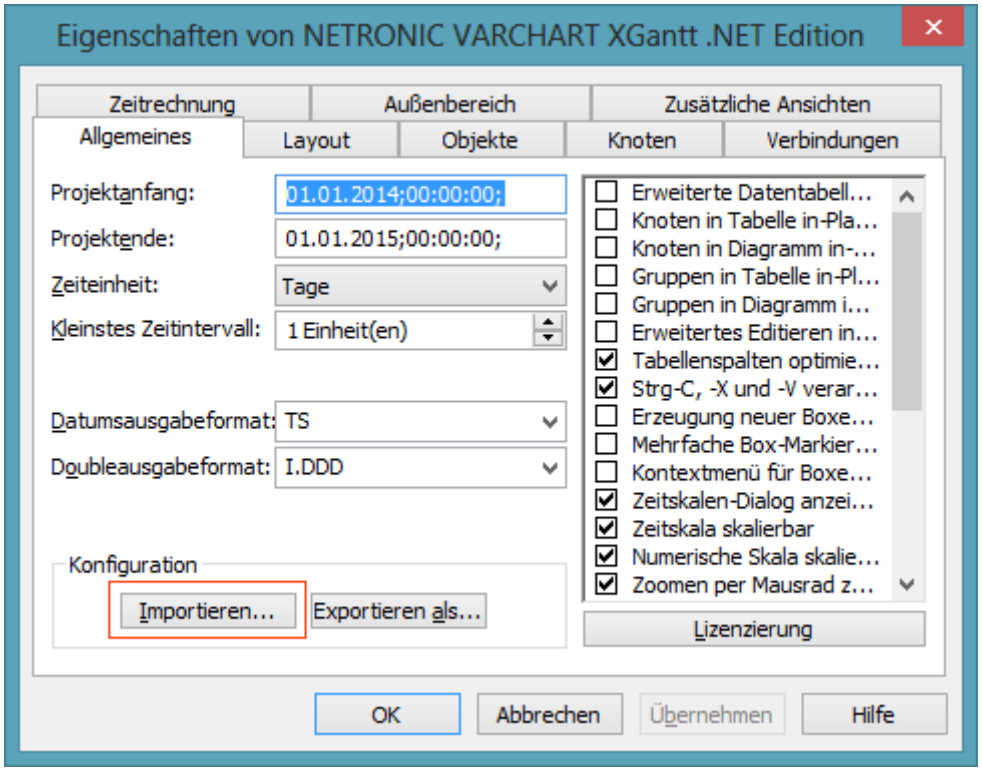
1. Bevor Sie VARCHART XGantt 5.0 für .NET installieren, öffnen Sie im Formdesigner von Visual Studio das Formular, das XGantt 4.4 verwendet, und sichern Sie die aktuelle Konfiguration von XGantt über die Schaltfläche **Exportieren** auf der Eigenschaftenseite **Allgemeines**:



2. Schließen Sie zuerst das Formular und dann Visual Studio.
3. nstallieren Sie den neuen Build von VARCHART XGantt für .NET in denselben Ordner, in dem der alte Build vorhanden ist.
4. Öffnen Sie im Formdesigner das Formular, das XGantt enthält. Folgende Fehlermeldung erscheint:



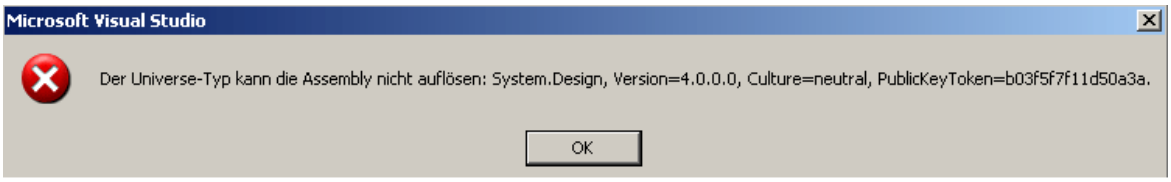
5. Klicken Sie auf **Ignore and Continue**. Damit wird das Formular im Formdesigner wieder korrekt angezeigt, XGantt wurde jedoch auf seine Standardkonfiguration zurückgesetzt.
6. Importieren Sie Ihre zuvor gesicherte Konfiguration über die Schaltfläche **Importieren** auf der Eigenschaftenseite **Allgemeines**:



7.

Damit hat VARCHART XGantt dann wieder die von Ihnen eingestellte Konfiguration.

6.3 Warum erscheint eine Fehlermeldung, wenn man bei Visual Studio 2010 ein neues Projekt erzeugt und versucht, das Steuerelement auf die Form zu ziehen?



Diese Fehlermeldung erscheint, da bei Visual Studio 2010 das **.NET Framework 4 Client Profile** als Standard voreingestellt ist, das VARCHART-Steuerelement von NETRONIC jedoch das Zielframework **.NET Framework 4** benötigt, da ersteres nicht die von den Eigenschaftenseiten zur Designzeit benötigte System.Design.dll enthält. Sie müssen daher **bevor** Sie das Steuerelement auf die Form ziehen, unter **Projekteigenschaften/Anwendung** (C#) bzw. **Erweiterte Kompilereinstellungen** (VB) das Zielframework von **.NET Framework 4 Client Profile** auf **.NET Framework 4** umstellen.

6.4 Wie kann das VARCHART Windows Forms neu lizenziert werden?

1. Entwicklungsumgebung schließen
2. Lizenzdatei NETRONIC.XTree.VcTree.lic in das Installationsverzeichnis von VARCHART XTree.NET kopieren
3. Umgebung wieder starten und Projekt erneut übersetzen

6.5 Wieso können Knoten u. U. nicht interaktiv erzeugt werden?

Wenn Sie zur Laufzeit keine Knoten mit der Maus anlegen können, stellen Sie sicher, dass auf der Eigenschaftenseite **Allgemeines** das Kontrollkästchen **Erzeugung neuer Knoten zulassen** aktiviert ist.

Kontrollieren Sie außerdem, ob in Ihrem Programmcode die VARCHART-VcTree-Eigenschaft **NodeCreationAllowed** auf **False** gesetzt wurde; dann können ebenfalls keine neuen Knoten angelegt werden.

6.6 Wie verhindert man das interaktive Erzeugen von Knoten?

Sie können auf verschiedene Weisen verhindern, dass Knoten interaktiv angelegt werden können:

1. Deaktivieren Sie auf der Eigenschaftenseite **Knoten** das Kontrollkästchen **Neue Knoten zulassen**.
2. Setzen Sie Rückgabestatus von **OnNodeCreate** auf **vcRetStatFalse**, damit interaktiv erzeugte Knoten wieder gelöscht werden.
3. Ergänzen Sie die folgenden Codezeilen:

Code-Beispiel

```
Sub Form_Load
   VcTree1.NodeCreationAllowed = False
End Sub
```

6.7 Wie lassen sich die Standard-Kontextmenüs abschalten?

Die vordefinierten Kontextmenüs können Sie durch Setzen des Rückgabestatus auf **vcRetStatNoPopup** unterdrücken.

Code-Beispiel

```
' Kontextmenü für das Diagramm abschalten
Private Sub VcTree1_VcDiagramRightClicking(ByVal x As Long, ByVal y As
Long, _
                                returnStatus As Variant)
    returnStatus = vcRetStatNoPopup
End Sub

' Kontextmenü für Knoten abschalten
Private Sub VcTree1_VcNodeRightClicking(ByVal node As VcTreeLib.VcNode,
_
                                ByVal location As VcTreeLib.VcLocation,
_
                                ByVal x As Long, ByVal y As Long, _
                                returnStatus As Variant)
   returnStatus = vcRetStatNoPopup
End Sub
```

6.8 Wie lässt sich die Performance verbessern?

> Suspend update

Bei größeren Datenmengen kann es unter Umständen sehr lange dauern, wenn bei einer großen Anzahl von Vorgängen dieselbe Aktion durchgeführt wird. Nicht jeder automatische Aktualisierungsvorgang im Diagramm ist notwendig; in einem solchen Fall können Sie Aktualisierungen für den Einzelfall unterdrücken und nach der Abarbeitung einer Code-Sequenz final einmal durchführen. Unterdrückung und Wiedereinsatz der Aktualisierung erfolgen mit der Methode **SuspendUpdate**, die zu Beginn der Code-Sequenz auf **True** bzw. am Ende auf **False** gesetzt wird. Auf diese Weise können Sie die Performance insgesamt beträchtlich erhöhen.

Code-Beispiel

```
VcTree1.SuspendUpdate (True)

   If updateFlag Then
      For Each node In nodeCltn
         If node.DataField(2) < "07.09.98" Then
            node.DataField(13) = "X"
            node.Update
            counter = counter + 1
         End If
      Next node
   Else
      For Each node In nodeCltn
         If node.DataField(2) < "07.09.98" Then
            node.DataField(13) = ""
            node.Update
            counter = counter + 1
         End If
      Next node
   End If

VcTree1.SuspendUpdate (False)
```

> Grafiken

Eine Ursache für eine zu geringe Performance können Grafiken beispielsweise in Tabellen-, Knoten- oder Boxfeldern sein, die zu groß sind oder eine zu große Pixelzahl haben.

6.9 Fehlermeldungen

> Fehlermeldungen zur Laufzeit, vom Entwickler verursacht

Wird noch bearbeitet.

6.10 Was tun, wenn das Steuerelement unerwartet nicht in allen Benutzerkonten eines Rechners funktioniert?

Wenn Sie feststellen, dass das Steuerelement nicht ansprechbar ist, wenn ein anderer Benutzer die gleiche, das Steuerelement nutzende Applikation aufruft, dann liegt es daran, dass das Steuerelement nicht für alle Benutzer installiert wurde. Bei der Erstellung des Setup-Programms, mit dem Sie das Steuerelement auf dem Kundenrechner installieren, sollte die Option "Für alle Benutzer installieren" ausgewählt sein.

Nachträglich oder von Hand lässt sich eine Installation für mehrere Benutzerkonten freischalten, indem Sie die Sicherheitseinstellungen der zum Steuerelement gehörenden Dateien so erweitern, dass andere Benutzerkonten als das, unter dem die Installation stattfand, auf die Dateien zugreifen dürfen. Die Sicherheitseinstellungen können Sie einerseits über den Menüpunkt "Eigenschaften" des Kontextmenüs zur jeweiligen Datei ändern oder per Kommandozeile über den Befehl 'cacls'. Die zum Steuerelement gehörenden Dateien finden Sie im Kapitel "Auslieferung" am Anfang dieses Buches.

6.11 Sind alle Fonts verwendbar?

Bedingt durch die Umstellung auf GDI+ gibt es Einschränkungen bzgl. der Darstellung von Schriftarten. So ist GDI+ nicht in der Lage, Fonts darzustellen, die als Postscript- oder gar Bitmap-Fonts vorliegen. Zu ersteren gehören auch Fonts, die zwar vom Typ **OpenType** sind, aber als "klassische Fonts" intern eine Art Postscript-Struktur besitzen (wie z.B. "Warnock Pro"). Zu den letzteren gehören z.B. die "alten" Windows-Fonts "Courier" und "Times", aber auch "System" und "MS Sans Serif".

Diese Fonts werden von den Fontauswahldialogen im VARCHART-Steuerelement daher nicht angeboten. Werden sie dennoch über die API gesetzt, so wird ein Ersatzfont angezeigt. Bei den "alten" Windows-Fonts sind von NETRONIC Ersetzungsregeln eingebaut worden, die einen ähnlichen "neuen" Font auswählen, bei externen Fonts wird dann "Arial" als Ersatz genommen, damit überhaupt Text auf dem Bildschirm erscheint.

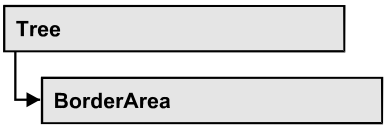
Es ist zwar unwahrscheinlich, aber nicht auszuschließen, dass in zukünftigen Versionen von GDI+ Unterstützung für die bisher nicht unterstützten Fonts eingebaut wird. Informationen sind bei Microsoft leider nur in Blogs und News-Gruppen zu erhalten, nicht jedoch bei MSDN.

7 API Referenz

7.1 Objekttypen

- VcBorderArea
- VcBorderBox
- VcBox
- VcBoxCollection
- VcBoxFormat
- VcBoxFormatCollection
- VcBoxFormatField
- VcDataRecord
- VcDataRecordCollection
- VcDataTable
- VcDataTableCollection
- VcDataTableField
- VcDataTableFieldCollection
- VcFilter
- VcFilterCollection
- VcFilterSubCondition
- VcLegendView
- VcMap
- VcMapCollection
- VcMapEntry
- VcNode
- VcNodeAppearance
- VcNodeAppearanceCollection
- VcNodeCollection
- VcNodeFormat
- VcNodeFormatCollection
- VcNodeFormatField
- VcPrinter
- VcRect
- VcTree
- VcWorldView

7.2 VcBorderArea



Ein Objekt vom Typ **VcBorderArea** bezeichnet den Titel- bzw. Legendenbereich der Grafik.

Methoden

- BorderBox

Methoden

BorderBox

Methode von VcBorderArea

Diese Methode ermöglicht den Zugriff auf ein BorderBox-Objekt.

	Datentyp	Beschreibung
Parameter:		
boxPosition	VcBorderBoxPosition	Position der Box
	Mögliche Werte:	
	.vcBBXPBottomBottomCentered 8	zweite Zeile im unteren Bereich, mittig
	.vcBBXPBottomBottomLeft 7	zweite Zeile im unteren Bereich, links
	.vcBBXPBottomBottomRight 9	zweite Zeile im unteren Bereich, rechts
	.vcBBXPBottomTopCentered 5	erste Zeile im unteren Bereich, mittig
	.vcBBXPBottomTopLeft 4	erste Zeile im unteren Bereich, links
	.vcBBXPBottomTopRight 6	erste Zeile im unteren Bereich, rechts
	.vcBBXPLegend 51	Legende
	.vcBBXPTopCentered 2	oben mittig
	.vcBBXPTopLeft 1	oben links
	.vcBBXPTopRight 3	oben rechts
Rückgabewert	VcBorderBox	Box des Titel- und Legendenbereichs

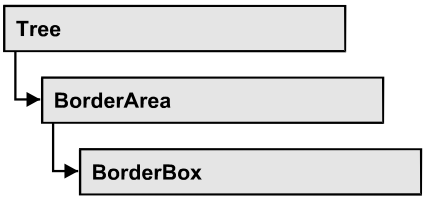
Code-Beispiel VB.NET

```
Dim boardArea As VcBorderArea
Dim bBoxBBL As VcBorderBox

boardArea = VcTree1.BorderArea
bBoxBBL = boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
bBoxBBL.LegendTitle = "Explanation"
```

Code-Beispiel C#

```
VcBorderArea boardArea = vcTree1.BorderArea;

VcBorderBox bBoxBBL =
boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
bBoxBBL.LegendTitle = "Explanation";
```

7.3 VcBorderBox



Ein Objekt vom Typ **VcBorderBox** bezeichnet eine der Boxen des Titel- bzw. Legendenbereichs der Grafik.

Eigenschaften

- Alignment
- GraphicsFileName
- LegendElementsArrangement
- LegendElementsBottomMargin
- LegendElementsMaximumColumnCount
- LegendElementsMaximumRowCount
- LegendElementsTopMargin
- LegendFont
- LegendTitle
- LegendTitleFont
- LegendTitleVisible
- Text
- TextFont
- Type

Eigenschaften

Alignment

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie die Ausrichtung dieses BorderBox-Objektes festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcBorderBoxAlignment **Mögliche Werte:** .vcBBXACentered -1 .vcBBXALeft -3	Ausrichtung der BorderBox unten mittig links

.vcBBXARight -2	rechts

GraphicsFileName

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie den Namen der Grafikdatei angeben oder erfragen, die in dem aktuellen VcBorderBox-Objekt verwendet wird. Mögliche Formate:

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile oder Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)
- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, ggf. mit eingebauten EMF)

Nur EMF, EMF+, VMF und WMF sind Vektorformate, in denen das Diagramm auflösungsunabhängig gespeichert werden kann. Die übrigen Formate sind pixelorientiert und bieten damit nicht beliebige Auflösungen.

Das VMF-Format wird in der Zukunft nicht mehr weiterentwickelt, aus Kompatibilitätsgründen für bestehende Anwendungen aber zunächst noch weiter unterstützt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Grafikdatei

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcTree1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomTopRight)
borderBox.Type = VcBorderBoxType.vcBBXTGraphics
borderBox.GraphicsFileName = "C:\Asterix.jpg"
```

Code-Beispiel C#

```
VcBorderArea borderArea = vcTree1.BorderArea;
VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomTopRight);
borderBox.Type = VcBorderBoxType.vcBBXTGraphics;
borderBox.GraphicsFileName = @"C:\Asterix.jpg";
```

LegendElementsArrangement

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie die Anordnung der Elemente der Legende setzen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcLegendElementsArrangement	Typ der Anordnung der Legendenelemente
	Mögliche Werte:	
	.vcLEAFixedToColumns 0	Die Legendenelemente werden nur in Spalten ausgerichtet.
	.vcLEAFixedToRows 1	Die Legendenelemente werden nur in Zeilen ausgerichtet.
	.vcLEAFixedToRowsAndColumns 2	Die Legendenelemente werden in Zeilen und Spalten ausgerichtet.

LegendElementsBottomMargin

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie den Abstand der Elemente zum unteren Rand festlegen oder erfragen (Einheit: mm).

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Breite des unteren Randes

LegendElementsMaximumColumnCount

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie die Anzahl der Spalten setzen oder erfragen, über die sich die Elemente der Legende verteilen sollen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Anzahl der Spalten

LegendElementsMaximumRowCount

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie die Anzahl der Zeilen setzen oder erfragen, über die sich die Elemente der Legende verteilen sollen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Anzahl der Zeilen

LegendElementsTopMargin

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie den Abstand der Elemente zum oberen Rand festlegen oder erfragen (Einheit: mm).

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Breite des oberen Randes

LegendFont

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie die Schriftattribute der Legende angeben bzw. erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.DrawingFont	Schriftattribute der Legende

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcTree1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.Type = VcBorderBoxType.vcBBXTLegend
MsgBox(borderBox.LegendFont.Name)
```

Code-Beispiel C#

```
VcBorderArea borderArea = vcTree1.BorderArea;
VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.Type = VcBorderBoxType.vcBBXTLegend;
MessageBox.Show(borderBox.LegendFont.Name);
```

LegendTitle

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie den Legendentitel angeben bzw. erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Legendentitel

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcTree1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.LegendTitle = "Explanation"
```

Code-Beispiel C#

```
VcBorderArea borderArea = vcTree1.BorderArea;

VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.LegendTitle = "Explanation";
```

LegendTitleFont

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie die Schriftattribute des Legendentitels angeben bzw. erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.DrawingFont	Schriftattribute des Legendentitels

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcTree1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.Type = VcBorderBoxType.vcBBXTLegend
MsgBox(borderBox.LegendTitleFont.Name)
```

Code-Beispiel C#

```
VcBorderArea borderArea = vcTree1.BorderArea;
VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.Type = VcBorderBoxType.vcBBXTLegend;
MessageBox.Show(borderBox.LegendTitleFont.Name);
```

LegendTitleVisible

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft legen Sie fest, ob der Legendentitel sichtbar ist oder nicht.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Legendentitel sichtbar (True)/nicht sichtbar (False)

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcTree1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.LegendTitleVisible = False
```

Code-Beispiel C#

```
VcBorderArea borderArea = vcTree1.BorderArea;

VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.LegendTitleVisible = false;
```

Text

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie den Text einer Titelzeile (oberhalb oder unterhalb des Diagramms) angeben oder erfragen. Für die Seitennummerierung oder die Ausgabe des Systemdatums können Sie folgende Platzhalter angeben, die dann beim Ausdruck durch die entsprechenden Inhalte ersetzt werden:

{COLUMN} = Seitennummer in der Breite (einer zweidimensionalen Seitenanordnung)

{NUMPAGES} = Gesamtanzahl der Seiten

{PAGE} = fortlaufende Seitennummer

{ROW} = Seitennummer in der Höhe (einer zweidimensionalen Seitenanordnung)

{SYSTEMDATE} = Systemdatum

Die Eigenschaft Text ist eine indizierte Eigenschaft, die in C# über die beiden Methoden set_Text (rowIndex, pvn) und get_Text (rowIndex) angesprochen wird.

	Datentyp	Beschreibung
Parameter:		
rowIndex	System.Int16	Zeilenindex {0...6}
Eigenschaftswert	System.String	Text in Textfeldern

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcTree1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.Type = VcBorderBoxType.vcBBXTText
borderBox.Text(index) = "Department A"
```

Code-Beispiel C#

```
VcBorderArea borderArea = vcTree1.BorderArea;

VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.Type = VcBorderBoxType.vcBBXTText;
borderBox.set_Text(index, "DepartmentA");
```

TextFont

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie die Schriftattribute einer Titelzeile (oberhalb oder unterhalb des Diagramms) angeben bzw. erfragen.

Dies ist eine indizierte Eigenschaft, die in C# über die beiden Methoden **set_TextFont (rowIndex, pvn)** und **get_TextFont (row-Index)** angesprochen wird.

Die Eigenschaft TextFont ist eine indizierte Eigenschaft, die in C# über die beiden Methoden set_TextFont (rowIndex, pvn) und get_FieldText (row-Index) angesprochen wird.

	Datentyp	Beschreibung
Parameter:		
rowIndex	System.Int16	Zeilenindex {0...6}
Eigenschaftswert	System.DrawingFont	Schriftattribute des Textes

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea
Dim bBoxTL As VcBorderBox

Set borderArea = VcTree1.BorderArea
Set bBoxBBL = borderArea.BorderBox(vcBBXPBottomBottomLeft)

bBoxTL.TextFont(i).Bold = False
bBoxTL.TextFont(i).Italic = False
bBoxTL.TextFont(i).Name = "Symbol"
```

Code-Beispiel C#

```
// Text for Title
VcBorderBox borderBox =
VcTree1.BorderArea.BorderBox(VcBorderBoxPosition.vcBBXPTopCentered);
borderBox.Type = VcBorderBoxType.vcBBXTText;

Font titleFont1 = new Font("Arial", 20, FontStyle.Bold);

borderBox.set_Text(1, "Time Scheduler");
borderBox.set_TextFont(1, titleFont1);
```

Type

Eigenschaft von VcBorderBox

Mit dieser Eigenschaft können Sie den Typ des BorderBox-Objekts angeben bzw. erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcBorderBoxType	Typ der Box
	Mögliche Werte:	
	.vcBBXTGraphics 3	Grafik
	.vcBBXTLegend 4	Legende
	.vcBBXTNothing 0	Leer
	.vcBBXTText 1	Text
	.vcBBXTTextWithGraphics 2	Text und Grafik

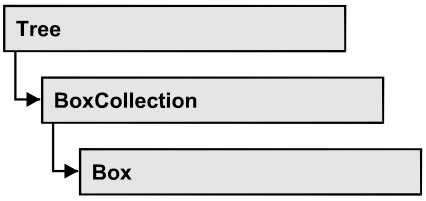
Code-Beispiel VB.NET

```
Dim bBoxBBL As VcBorderBox

bBoxBBL = boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomLeft)
bBoxBBL.Type = VcBorderBoxType.vcBBXTGraphics
```

Code-Beispiel C#

```
VcBorderArea boardArea = vcTree1.BorderArea;

VcBorderBox bBoxBBL =
boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
bBoxBBL.Type = VcBorderBoxType.vcBBXTGraphics;
```

7.4 VcBox



Ein Objekt vom Typ **VcBox** beschreibt eine Box, in der Texte und Grafiken ausgegeben werden können.

Eigenschaften

- FieldText
- FormatName
- LineColor
- LineThickness
- LineType
- Marked
- Moveable
- Name
- Origin
- Priority
- ReferencePoint
- UpdateBehaviorName
- Visible

Methoden

- GetActualExtent
- GetTopLeftPixel
- GetXYOffset
- IdentifyFormatField
- SetXYOffset
- SetXYOffsetByTopLeftPixel

Eigenschaften

FieldText

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie den Inhalt eines Feldes einer Box erfragen oder festlegen. Diese Eigenschaft können Sie auch im Dialog **Box bearbeiten** festlegen.

Bei mehrzeiligen Textfeldern müssen für einen erzwungenen Umbruch die einzelnen Zeilen des Textfelds mit "\n" im String getrennt sein (Beispiel: "Zeile1\nZeile2"). Ohne erzwungenen Umbruch wird automatisch an Leerzeichen umgebrochen.

Die Eigenschaft FieldText ist eine indizierte Eigenschaft, die in C# über die beiden Methoden set_FieldText (fieldIndex, pvn) und get_FieldText (field-Index) angesprochen wird.

	Datentyp	Beschreibung
Parameter:		
⇨ fieldIndex	System.Int16	Feldindex
Eigenschaftswert	System.String	Feldinhalt

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.FieldText(0) = "User: "
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.set_FieldText(0, "User: ");
```

FormatName

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie den Namen des Boxformats erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	VcBoxFormat	BoxFormat-Objekt oder **Nothing**

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.FormatName = "Standard"
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.FormatName = "Standard";
```

LineColor

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie die Farbe der Randlinie der Box festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	RGB-Farbwerte ({0...255},{0...255},{0...255})

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.LineColor = System.Drawing.Color.Blue
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.LineColor = System.Drawing.Color.Blue;
```

LineThickness

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie die Stärke der Randlinie der Box erfragen oder festlegen.

Wenn Sie diese Eigenschaft auf Werte zwischen 1 und 4 setzen, wird damit eine absolute Liniendicke in Pixel definiert, d.h. die Linien haben unabhängig vom Zoomfaktor immer die gleiche feste Linienstärke in Pixeln. Dies wird jedoch aufgrund der besseren Lesbarkeit beim Drucken in eine vom Zoomfaktor abhängige Liniendicke umgewandelt:

Wert	Punkte	mm
1	1/2 Punkt	0,09 mm
2	1 Punkt	0,18 mm
3	3/2 Punkt	0,26 mm
4	2 Punkt	0,35 mm

Ein Punkt ist 1/72 Zoll groß und stellt die Maßeinheit für Schriftgrößen dar.

Wenn Sie diese Eigenschaft auf Werte zwischen 5 und 1.000 setzen, wird damit eine Linienstärke in 1/100 mm definiert, d.h. die Linien bekommen eine tatsächliche Dicke in Pixeln, die abhängig vom Zoomfaktor ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Linienstärke LineType {1...4}: Werte in Pixeln LineType {5...1000}: Werte in 1/100 mm **Standardwert:** As defined in the dialog

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.LineThickness = 2
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.LineThickness = 2;
```

LineType

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie den Typ der Randlinie der Box festlegen oder erfragen.

	Datentyp	**Beschreibung**
Eigenschaftswert	VcLineType	Linientyp **Standardwert:** vcSolid
	Mögliche Werte:	
	.vcDashed 4	Linientyp **gestrichelt**
	.vcDashed 4	Linientyp **gestrichelt**
	.vcDashedDotted 5	Linientyp **gestrichelt-gepunktet**
	.vcDashedDotted 5	Linientyp **gestrichelt-gepunktet**
	.vcDotted 3	Linientyp **gepunktet**
	.vcDotted 3	Linientyp **gepunktet**
	.vcLineType0 100	Linientyp 0
	.vcLineType1 101	Linientyp 1
	.vcLineType10 110	Linientyp 10
	.vcLineType11 111	Linientyp 11
	.vcLineType12 112	Linientyp 12
	.vcLineType13 113	Linientyp 13
	.vcLineType14 114	Linientyp 14
	.vcLineType15 115	Linientyp 15
	.vcLineType16 116	Linientyp 16
	.vcLineType17 117	Linientyp 17
	.vcLineType18 118	Linientyp 18
	.vcLineType2 102	Linientyp 2
	.vcLineType3 103	Linientyp 3
	.vcLineType4 104	Linientyp 4
	.vcLineType5 105	Linientyp 5
	.vcLineType6 106	Linientyp 6
	.vcLineType7 107	Linientyp 7
	.vcLineType8 108	Linientyp 8
	.vcLineType9 109	Linientyp 9
	.vcNone 1	Kein Linientyp zugewiesen
	.vcNone 1	Kein Linientyp
	.vcNotSet -1	Kein Linientyp zugewiesen
	.vcSolid 2	Linientyp **durchgezogen**
	.vcSolid 2	Linientyp **durchgezogen**

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.LineType = VcLineType.vcDotted
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.LineType = VcLineType.vcDotted;
```

Marked

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie festlegen oder abfragen, ob eine Textbox markiert ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	**True**: Box markiert; **false**: Box nicht markiert

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.Marked = True
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Marked = true;
```

Moveable

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie erfragen oder festlegen, ob die Box interaktiv verschiebbar sein soll.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Verschiebbar (True)/ nicht verschiebbar (False) **Standardwert:** True

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.Moveable = False
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Moveable = false;
```

Name

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie den Namen einer Box erfragen oder setzen. Diese Eigenschaft können Sie im Dialog **Boxen verwalten** festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Box

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox
Dim boxName As String

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
MsgBox(box.Name)
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();

MessageBox.Show(box.Name);
```

Origin

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie den Ursprungspunkt der Box festlegen oder erfragen, d. h. den Diagrammpunkt, von dem aus der Abstand zum Referenzpunkt der Box in x- bzw. y-Richtung angegeben wird.

Mithilfe der Eigenschaften **Origin** und **ReferencePoint** sowie der Methode **GetXYOffset** können Sie jede einzelne Box im Diagrammbereich positionieren, wobei die relative Position der Box zum Diagramm unabhängig von der Diagrammgröße ist.

	Datentyp	Beschreibung
Eigenschaftswert	VcBoxOrigin **Mögliche Werte:** .vcBOBottomCenter 28 .vcBOBottomLeft 27 .vcBOBottomRight 29 .vcBOCenterCenter 25 .vcBOCenterLeft 24 .vcBOCenterRight 26 .vcBOTopCenter 22 .vcBOTopLeft 21 .vcBOTopRight 23	Ursprungspunkt der Box unten mittig unten links unten rechts mittig mittig mittig links mittig rechts oben mittig oben links oben rechts

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.Origin = VcBoxOrigin.vcBOTopCenter
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Origin = VcBoxOrigin.vcBOTopCenter;
```

Priority

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie die Priorität der Box erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Prioritätsstufe

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.Priority = 3
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Priority = 3;
```

ReferencePoint

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie den Referenzpunkt der Box festlegen oder erfragen, d. h. den Punkt der Box, von dem aus der Abstand zum Ursprung in x- bzw. y-Richtung angegeben wird.

	Datentyp	Beschreibung
Eigenschaftswert	VcBoxReferencePoint	Referenzpunkt der Box
	Mögliche Werte:	
	.vcBRPBottomCenter 28	unten mittig
	.vcBRPBottomLeft 27	unten links
	.vcBRPBottomRight 29	unten rechts
	.vcBRPCenterCenter 25	mittig mittig
	.vcBRPCenterLeft 24	mittig links
	.vcBRPCenterRight 26	mittig rechts
	.vcBRPTopCenter 22	oben mittig
	.vcBRPTopLeft 21	oben links
	.vcBRPTopRight 23	oben rechts

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.ReferencePoint = VcBoxReferencePoint.vcBRPCenterRight
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.ReferencePoint = VcBoxReferencePoint.vcBRPCenterRight;
```

UpdateBehaviorName

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie den Namen des Aktualisierungsverhaltens erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Aktualisierungsverhaltens

Visible

Eigenschaft von VcBox

Mit dieser Eigenschaft können Sie erfragen oder festlegen, ob die Box sichtbar sein soll. Diese Eigenschaft können Sie auch im Dialog **Boxen verwalten** festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Box sichtbar/unsichtbar **Standardwert:** True

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
box.Visible = False
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Visible = false;
```

Methoden

GetActualExtent

Methode von VcBox

Mit dieser Methode können Sie die Breite und Höhe der Box erfragen (Einheit: 1/100 mm).

Werden diese Werte beim XY-Offset berücksichtigt, kann man z.B. den Referenzpunkt der Verankerungslinie ändern, ohne die Position der Box zu verändern.

	Datentyp	Beschreibung
Parameter:		
⇦ width	System.Int32	Breite der Box
⇦ height	System.Int32	Höhe der Box
Rückgabewert	System.Boolean	Ausdehnung der Box wird zurückgegeben/wird nicht zurückgegeben

GetTopLeftPixel

Methode von VcBox

Mit dieser Methode können Sie den gespeicherten XY-Offset für die obere linke Ecke in Pixel umrechnen und ausgeben.

Der x-Wert kann dann z.B. mit der Methode **VcGantt.GetDate** weiter verwendet werden, um ein Datum zu erhalten

	Datentyp	Beschreibung
Parameter:		
⇦ x	System.Int32	x-Wert des Offsets
⇦ y	System.Int32	y-Wert des Offsets
Rückgabewert	System.Boolean	Offset wird zurückgegeben/nicht zurückgegeben

GetXYOffset

Methode von VcBox

Mit dieser Methode können Sie den Abstand zwischen Ursprung und Referenzpunkt in x- und y-Richtung erfragen (Einheit: 1/100 mm).

	Datentyp	Beschreibung
Parameter:		
⇦ xOffset	System.Int32	x-Wert des Offsets
⇦ yOffset	System.Int32	y-Wert des Offsets
Rückgabewert	System.Boolean	Offset wird zurückgegeben/nicht zurückgegeben

IdentifyFormatField

Methode von VcBox

Mit dieser Methode können Sie den Index des an der bezeichneten Position befindlichen Formatfeldes erfragen. Falls sich an der bezeichneten Position ein Feld befindet, wird **True** zurückgegeben, ansonsten **False**.

	Datentyp	Beschreibung
Parameter:		
⇨ x	System.Int32	X-Koordinate der Position
⇨ y	System.Int32	Y-Koordinate der Position

⇦ format	VcBoxFormat	Identifiziertes Format
⇦ formatFieldIndex	System.Int16	Format-Feldindex
Rückgabewert	System.Boolean	Ein Formatfeld befindet sich/befindet sich nicht an der angegebenen Position

SetXYOffset

Methode von VcBox

Mit dieser Methode können Sie den Abstand zwischen Ursprung und Referenzpunkt in x- und y-Richtung festlegen (Einheit: 1/100 mm).

Den Offset können Sie auch im Dialog **Boxen verwalten** festlegen.

	Datentyp	Beschreibung
Parameter:		
⇨ xOffset	System.Int32	x-Wert des Offsets
⇨ yOffset	System.Int32	y-Wert des Offsets
Rückgabewert	System.Boolean	Offset wird gesetzt (True)/nicht gesetzt (False)

Code-Beispiel VB.NET

```
Dim offSet As Boolean
offSet = VcTree1.BoxCollection.FirstBox.SetXYOffset(100, 100)
```

Code-Beispiel C#

```
bool offSet = vcTree1.BoxCollection.FirstBox().SetXYOffset(100, 100);
```

SetXYOffsetByTopLeftPixel

Methode von VcBox

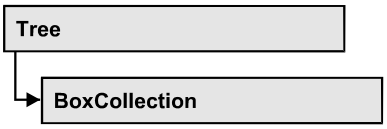
Mit dieser Methode können Sie den angegebenen Pixelwert der oberen linken Ecke intern in einen XY-Offset umrechnen und speichern.

Damit kann man z.B. an einer XY-Koordinate aus einem Ereignis eine Box positionieren.

	Datentyp	Beschreibung
Parameter:		
⇨ x	System.Int32	x-Wert des Offsets
⇨ y	System.Int32	y-Wert des Offsets

Rückgabewert	System.Boolean	Offset wird gesetzt (True)/nicht gesetzt (False)

7.5 VcBoxCollection



Im VcBoxCollection-Objekt sind alle verfügbaren Boxen zusammengefasst. Über **For Each box In BoxCollection** oder die Methoden **First...** und **Next...** können Sie in einer Schleife auf alle Boxen zugreifen. Sie haben Zugriff auf bestimmte Objekte über die Eigenschaften **BoxByName** und **BoxByIndex**. Die Anzahl der im Collection-Objekt vorhandenen Boxen kann über die Eigenschaft **Count** erfragt werden. Die Methoden **Add**, **Copy** und **Remove** ermöglichen das Hinzufügen, Kopieren und Löschen von Boxen.

Eigenschaften

- Count

Methoden

- Add
- AddBySpecification
- BoxByIndex
- BoxByName
- Copy
- FirstBox
- GetEnumerator
- NextBox
- Remove
- Update

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcBoxCollection

Mit dieser Eigenschaft können Sie die Anzahl der Boxobjekte in der Box-Auflistung erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Boxen

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim numberOfBoxes As Integer

boxCltn = VcTree1.BoxCollection
numberOfBoxes = boxCltn.Count
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
int numberOfBoxes = boxCltn.Count;
```

Methoden

Add

Methode von VcBoxCollection

Mit dieser Methode können Sie eine neue Box in der BoxCollection anlegen. Wenn der Name noch nicht verwendet wird, dann wird das neue Boxobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB). Um die neu angelegte Box im Diagramm sichtbar werden zu lassen, muss die Auflistung (Collection) mit **Update** aktualisiert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ boxName	System.String	Name der Box
Rückgabewert	VcBox	Neues Boxobjekt

Code-Beispiel VB.NET

```
newBox = VcTree1.BoxCollection.Add("box1")
```

Code-Beispiel C#

```
newBox = vcTree1.BoxCollection.Add("box1");
```

AddBySpecification

Methode von VcBoxCollection

Mit dieser Methode können Sie eine Box über eine Box-Spezifikation erzeugen. Dies dient der Persistenz von Boxobjekten. Die Spezifikation einer

Box kann erfragt (siehe VcBox-Eigenschaft **Specification**) und gespeichert werden. Bei einer neuen Sitzung kann die gleiche Box mit Hlife der eingelesenen Spezifikation samt des gespeicherten Namens wieder erzeugt werden. Um die neu angelegte Box im Diagramm sichtbar werden zu lassen, muss die Auflistung (Collection) mit **Update** aktualisiert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ specification	System.String	Boxspezifikation
Rückgabewert	VcBox	Neues Boxobjekt

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcTree1.BoxCollection
boxCltn.AddBySpecification(textSpecification)
boxCltn.Update()
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
boxCltn.AddBySpecification(textSpecification);
boxCltn.Update();
```

BoxByIndex

Methode von VcBoxCollection

Mit dieser Methode können Sie auf eine einzelne Box über ihren Index zugreifen. Existiert kein Box-Objekt an dem angegebenen Index, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index der Box
Rückgabewert	VcBox	Ermitteltes Box-Objekt

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcTree1.BoxCollection
box = boxCltn.BoxByIndex(0)
box.LineThickness = 2
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.BoxByIndex(0);
box.LineThickness = 2;
```

BoxByName

Methode von VcBoxCollection

Mit dieser Methode können Sie unter Verwendung des Namens auf eine bestimmte Box zugreifen. Existiert keine Box unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ boxName	System.String	Name der Box
Rückgabewert	VcBox	Box

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcTree1.BoxCollection
box = boxCltn.BoxByName("BoxOne")
box.LineThickness = 3
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.BoxByName("BoxOne");
box.LineThickness = 3;
```

Copy

Methode von VcBoxCollection

Mit dieser Methode können Sie eine Box kopieren. Wenn die Box mit dem angegebenen Namen existiert und der Name der neuen Box noch nicht verwendet wird, wird das neue Boxobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB). Um die kopierte Box im Diagramm sichtbar werden zu lassen, muss die Auflistung (Collection) mit **Update** aktualisiert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ boxName	System.String	Name der zu kopierenden Box
⇨ newBoxName	System.String	Name der neuen Box
Rückgabewert	VcBox	Boxobjekt

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcTree1.BoxCollection
boxCltn.Copy("BoxOne", "NewBox")
boxCltn.Update()
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
boxCltn.Copy("BoxOne", "NewBox");
boxCltn.Update();
```

FirstBox

Methode von VcBoxCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. die erste Box der Box-Auflistung zugreifen, um anschließend in einer Schleife mit der Methode **NextBox** über die nachfolgenden Boxen zu iterieren. Existiert keine Box in der Box-Auflistung, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcBox	Erste Box

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
```

GetEnumerator

Methode von VcBoxCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Box-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim box As VcBox

For Each box In VcTree1.BoxCollection
   ListBox1.Items.Add(box.FormatName)
Next
```

Code-Beispiel C#

```
foreach (VcBox box in vcTree1.BoxCollection)
   listBox1.Items.Add(box.FormatName);
```

NextBox

Methode von VcBoxCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Boxen des BoxCollection-Objekts zugreifen, nachdem Sie mit der Methode **FirstBox** den Initialwert erfasst haben. Sind alle Boxen durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcBox	Nachfolgende Box

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox

While Not box Is Nothing
   ListBox1.Items.Add(box.Name)
   box = boxCltn.NextBox
End While
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();

while (box != null)
   {
   ListBox.Items.Add(box.Name);
   box = boxCltn.NextBox();
   }
```

Remove

Methode von VcBoxCollection

Mit dieser Methode können Sie eine Box löschen. Um das Löschen der Box im Diagramm sichtbar werden zu lassen, muss die Auflistung (Collection) mit **Update** aktualisiert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ boxName	System.String	Name der Box
Rückgabewert	System.Boolean	Box gelöscht (True)/nicht gelöscht (False)

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.BoxByIndex(0)
boxCltn.Remove(box.Name)
boxCltn.Update()
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.BoxByIndex(0);
boxCltn.Remove(box.Name);
boxCltn.Update();
```

Update

Methode von VcBoxCollection

Mit dieser Methode können Sie eine BoxCollection aktualisieren, nachdem Sie sie verändert haben.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Aktualisierung erfolgt (True)/ nicht erfolgt (False)

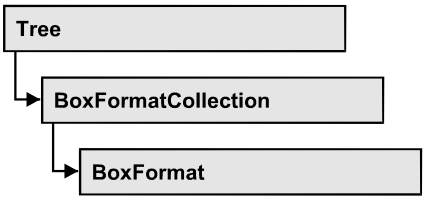
Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.BoxByIndex(0)
boxCltn.Remove(box.Name)
boxCltn.Update()
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.BoxByIndex(0);
boxCltn.Remove(box.Name);
boxCltn.Update();
```

7.6 VcBoxFormat



Ein Objekt vom Typ **VcBoxFormat** beschreibt die Formate von Boxen. Über **For Each formatField In BoxFormat** können Sie in einer Schleife auf alle Boxen zugreifen.

Eigenschaften

- FieldsSeparatedByLines
- FormatField
- FormatFieldCount
- Name
- Specification

Methoden

- CopyFormatField
- GetEnumerator
- RemoveFormatField

Eigenschaften

FieldsSeparatedByLines

Eigenschaft von VcBoxFormat

Mit dieser Eigenschaft können Sie festlegen, ob die Felder durch sichtbare Linien getrennt werden (True) oder nicht (False).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Boxfelder werden durch Linien getrennt (True)/ nicht getrennt (False).

Code-Beispiel VB.NET

```
Dim boxFormat As VcBoxFormat

boxFormat = VcTree1.BoxFormatCollection.FormatByIndex(0)
boxFormat.FieldsSeparatedByLines = True
```

Code-Beispiel C#

```
VcBoxFormat boxFormat = vcTree1.BoxFormatCollection.FormatByIndex(0);
boxFormat.FieldsSeparatedByLines = true;
```

FormatField

Nur-Lese-Eigenschaft von VcBoxFormat

Mit dieser Eigenschaft können Sie ein VcBoxFormatField-Objekt per Index holen. Der Index muss im Bereich von 0 bis FormatFieldCount-1 liegen.

Die Eigenschaft FormatField ist eine indizierte Eigenschaft, die in C# über die Methode get_FormatField (index) angesprochen wird.

	Datentyp	Beschreibung
Parameter:		
index	System.Int16 0 ... .FormatFieldCount-1	Index des Boxformatfeldes
Eigenschaftswert	VcBoxFormatField	Boxformatfeld

Code-Beispiel VB.NET

```
Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcTree1.BoxFormatCollection.FirstFormat
formatField = boxFormat.FormatField(0)
MsgBox(formatField.FormatName)
```

Code-Beispiel C#

```
VcBoxFormat boxFormat = vcTree1.BoxFormatCollection.FirstFormat();
VcBoxFormatField formatField = boxFormat.get_FormatField(0);
MessageBox.Show(formatField.FormatName);
```

FormatFieldCount

Nur-Lese-Eigenschaft von VcBoxFormat

Mit dieser Eigenschaft können Sie die Anzahl der Felder eines Boxformats ermitteln.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Anzahl der Felder im Boxformat

Code-Beispiel VB.NET

```
Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcTree1.BoxFormatCollection.FirstFormat
MsgBox(boxFormat.FormatFieldCount)
```

Code-Beispiel C#

```
VcBoxFormat boxFormat = vcTree1.BoxFormatCollection.FirstFormat();
MessageBox.Show(boxFormat.FormatFieldCount.ToString());
```

Name

Eigenschaft von VcBoxFormat

Mit dieser Eigenschaft können Sie den Namen eines Boxformats erfragen oder setzen. Diese Eigenschaft können Sie auch im Dialog **Boxformate verwalten** festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Boxformats

Code-Beispiel VB.NET

```
Dim boxFormat As VcBoxFormat

For Each boxFormat In VcTree1.BoxFormatCollection
   ListBox1.Items.Add(boxFormat.Name)
Next
```

Code-Beispiel C#

```
foreach (VcBoxFormat boxFormat in vcTree1.BoxFormatCollection)
   listBox1.Items.Add(boxFormat.Name);
```

Specification

Nur-Lese-Eigenschaft von VcBoxFormat

Mit dieser Eigenschaft können Sie die Spezifikation dieses Boxformats auslesen. Die Spezifikation ist ein String, der nur lesbare ASCII-Zeichen im Bereich 32 bis 127 enthält und somit problemlos in Textdateien oder Datenbanken gespeichert werden kann. Dies ermöglicht Persistenz. Eine solche Spezifikation kann später zur Wiederherstellung eines Boxformats mit der Methode **VcBoxFormatCollection.AddBySpecification** benutzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Spezifikation des Boxformats

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormat = boxFormatCltn.FirstBoxFormat
MsgBox(boxFormat.Specification)
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormat boxFormat = boxFormatCltn.FirstBoxFormat();
MessageBox.Show(boxFormat.Specification);
```

Methoden

CopyFormatField

Methode von VcBoxFormat

Mit dieser Methode können Sie ein Boxformatfeld kopieren. Das neue VcBoxFormatField-Objekt wird zurückgegeben. Es erhält den nächsten, noch nicht vergebenen Index.

	Datentyp	Beschreibung
Parameter:		
⇨ position	VcFormatFieldInnerPosition	Position des neuen Boxformatfeldes
	Mögliche Werte: .vcInnerAbove 1 .vcInnerBelow 3 .vcInnerLeftOf 0 .vcInnerRightOf 4	 oberhalb unterhalb links von rechts von
⇨ refIndex	System.Int16	Index des Referenz-Boxformatfeldes
Rückgabewert	VcBoxFormatField	Boxformatfeld-Objekt

Code-Beispiel VB.NET

```
Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcTree1.BoxFormatCollection.FormatByIndex(2)
boxFormat.CopyFormatField(VcFormatFieldInnerPosition.vcInnerRightOf, 0)
```

Code-Beispiel C#

```
VcBoxFormat boxFormat = vcTree1.BoxFormatCollection.FormatByIndex(0);
VcBoxFormatField formatField =
boxFormat.CopyFormatField(VcFormatFieldInnerPosition.vcInnerRightOf, 0);
```

GetEnumerator

Methode von VcBoxFormat

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Boxformatfelder iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcTree1.BoxFormatCollection.FirstFormat
For Each formatField In boxFormat
   ListBox1.Items.Add(formatField.FormatName)
Next
```

Code-Beispiel C#

```
VcBoxFormat boxFormat = vcTree1.BoxFormatCollection.FirstFormat();
foreach(VcBoxFormatField formatField in boxFormat)
   listBox1.Items.Add(formatField.FormatName);
```

RemoveFormatField

Methode von VcBoxFormat

Mit dieser Methode können Sie ein Boxformatfeld über den angegebenen Index löschen. Anschließend wird ggf. der Index aller Boxformatfelder neu festgesetzt, so dass sie wieder fortlaufend numeriert sind.

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des zu löschenden Boxformatfeldes

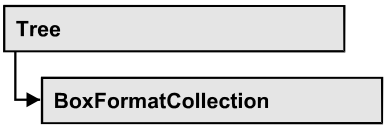
Code-Beispiel VB.NET

```
Dim boxFormat As VcBoxFormat
Dim i As Integer

boxFormat = VcTree1.BoxFormatCollection.FirstFormat

For i = 0 To boxFormat.FormatFieldCount - 1
   boxFormat.RemoveFormatField(i)
Next
```

Code-Beispiel C#

```
VcBoxFormat boxFormat = vcTree1.BoxFormatCollection.FirstFormat();
for (short i=0; i<boxFormat.FormatFieldCount-1; i++)
   boxFormat.RemoveFormatField(i);
```

7.7 VcBoxFormatCollection



Im VcBoxFormatCollection-Objekt sind alle verfügbaren Boxformate zusammengefasst. Über **For Each boxFormat In BoxFormatCollection** oder die Methoden **First...** und **Next...** können Sie in einer Schleife auf alle Formate zugreifen. Sie haben Zugriff auf bestimmte Objekte über die Eigenschaften **BoxFormatByName** und **BoxFormatByIndex**. Die Anzahl der im Collection-Objekt vorhandenen BoxFormate kann über die Eigenschaft **Count** erfragt werden. Die Methoden **Add...**, **Copy...** und **Remove...** ermöglichen das Hinzufügen, Kopieren und Löschen von Boxformaten.

Eigenschaften

- Count

Methoden

- Add
- AddBySpecification
- Copy
- FirstFormat
- FormatByIndex
- FormatByName
- GetEnumerator
- NextFormat
- Remove

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcBoxFormatCollection

Mit dieser Eigenschaft können Sie die Anzahl der Boxformatobjekte in der BoxFormat-Auflistung erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Boxformate

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim numberOfBoxformats As Integer

boxFormatCltn = VcTree1.BoxFormatCollection
numberOfBoxformats = boxFormatCltn.Count
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
int numberOfBoxformats = boxFormatCltn.Count;
```

Methoden

Add

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie ein neues Boxformat in der BoxFormatCollection anlegen. Wenn der Name noch nicht verwendet wird, wird das neue Boxformatobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ formatName	System.String	Name des Boxformats
Rückgabewert	VcBoxFormat	Neues Boxformatobjekt

Code-Beispiel VB.NET

```
Dim newBoxFormat = VcTree1.BoxFormatCollection.Add("boxFormat1")
```

Code-Beispiel C#

```
newBoxFormat = vcTree1.BoxFormatCollection.Add("boxFormat1");
```

AddBySpecification

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie ein Boxformat über eine Boxformat-Spezifikation erzeugen. Dies dient der Persistenz von Boxformat-Objekten. Die Spezifikation eines Boxformats kann erfragt (siehe VcBoxFormat-

Eigenschaft **Specification**) und gespeichert werden. Bei einer neuen Sitzung kann das gleiche Boxformat mit der wieder eingelesenen Spezifikation samt des gespeicherten Namens wieder erzeugt werden.

	Datentyp	Beschreibung
Parameter:		
⇨ formatSpecification	System.String	Boxformatspezifikation
Rückgabewert	VcBoxFormat	Neues Boxformatobjekt

Copy

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie ein Boxformat kopieren. Wenn das Boxformat mit dem angegebenen Namen existiert und der Name des neuen Boxformats noch nicht verwendet wird, wird das neue Boxformatobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ FormatName	System.String	Name des zu kopierenden Boxformats
⇨ newFormatName	System.String	Name des neuen Boxformats
Rückgabewert	VcBoxFormat	Boxformatobjekt

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormat = boxFormatCltn.Copy("CurrentBoxFormat", "NewBoxFormat")
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormat boxFormat = boxFormatCltn.Copy("CurrentBoxFormat", "NewBoxFormat");
```

FirstFormat

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. das erste Boxformat des BoxFormatCollection-Objekts zugreifen, um anschließend in einer Schleife mit der Methode **NextFormat** über die nachfolgenden Boxformate

zu iterieren. Existiert kein Boxformat im BoxFormatCollection-Objekt, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcBoxFormat	Erstes Boxformat

Code-Beispiel VB.NET

```
Dim format As VcBoxFormat

format = VcTree1.BoxFormatCollection.FirstFormat
```

Code-Beispiel C#

```
VcBoxFormat format = vcTree1.BoxFormatCollection.FirstFormat();
```

FormatByIndex

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie auf ein einzelnes Boxformat über ihren Index zugreifen. Existiert kein BoxFormat-Objekt an dem angegebenen Index, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Boxformats
Rückgabewert	VcBoxFormat	Ermitteltes Boxformat-Objekt

Code-Beispiel VB.NET

```
Dim formatBoxCltn As VcBoxFormatCollection
Dim formatBox As VcBoxFormat

formatBoxCltn = VcTree1.BoxFormatCollection
formatBox = formatBoxCltn.FormatByIndex(2)
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormat format = boxFormatCltn.FormatByIndex(2);
```

FormatByName

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie unter Verwendung des Namens auf ein bestimmtes Boxformat zugreifen. Existiert kein Boxformat unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ formatName	System.String	Name des Boxformats
Rückgabewert	VcBoxFormat	Boxformat

Code-Beispiel VB.NET

```
Dim formatBoxCltn As VcBoxFormatCollection
Dim formatBox As VcBoxFormat

formatBoxCltn = VcTree1.BoxFormatCollection
formatBox = formatBoxCltn.FormatByName("Standard")
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormat format = boxFormatCltn.FormatByName("Standard");
```

GetEnumerator

Methode von VcBoxFormatCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Boxformat-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcTree1.BoxFormatCollection
For Each boxFormat In boxFormatCltn
   ListBox1.Items.Add(boxFormat.Name)
Next
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
foreach (VcBoxFormat boxFormat in boxFormatCltn)
   listBox1.Items.Add(boxFormat.Name);
```

NextFormat

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Boxformate des BoxFormatCollection-Objekts zugreifen, nachdem Sie mit

der Methode **FirstFormat** den Initialwert erfasst haben. Sind alle Formate durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcBoxFormat	Folgendes Boxformat

Code-Beispiel VB.NET

```
Dim formatBoxCltn As VcBoxFormatCollection
Dim formatBox As VcBoxFormat

formatBoxCltn = VcTree1.BoxFormatCollection
formatBox = formatBoxCltn.FirstFormat

While Not formatBox Is Nothing
   ListBox1.Items.Add(formatBox.Name)
   formatBox = formatBoxCltn.NextFormat
End While
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormat boxFormat = boxFormatCltn.FirstFormat();

while (boxFormat != null)
   {
   ListBox.Items.Add(boxFormat.Name);
   boxFormat = boxFormatCltn.NextFormat();
   }
```

Remove

Methode von VcBoxFormatCollection

Mit dieser Methode können Sie ein Boxformat löschen. Wenn das Boxformat noch in irgendeinem anderen Objekt benutzt wird, kann es nicht gelöscht werden. In diesem Fall wird False zurückgegeben, sonst True.

	Datentyp	Beschreibung
Parameter:		
⇨ FormatName	System.String	Name des Boxformats
Rückgabewert	System.Boolean	Boxformat gelöscht (True)/nicht gelöscht (False)

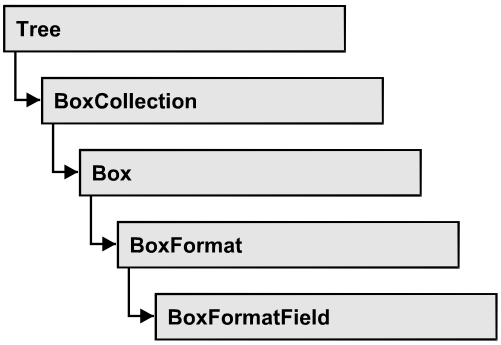
Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormat = boxFormatCltn.FormatByIndex(1)
boxFormatCltn.Remove(boxFormat.Name)
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormat boxFormat = boxFormatCltn.FormatByIndex(1);
boxFormatCltn.Remove(boxFormat.Name);
```

7.8 VcBoxFormatField



Ein Objekt vom Typ **VcBoxFormatField** stellt ein Boxformatfeld, also ein Feld eines VcBoxFormat-Objekts dar. Ein Boxformatfeld besitzt im Gegensatz zu vielen anderen Objekten keinen Namen, sondern nur einen Index, unter dem es im Boxformat untergebracht ist.

Eigenschaften

- Alignment
- FormatName
- GraphicsHeight
- Index
- MaximumTextLineCount
- MinimumTextLineCount
- MinimumWidth
- PatternBackgroundColor
- PatternColorAsARGB
- PatternEx
- TextFont
- TextFontColor
- Type

Eigenschaften

Alignment

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die Ausrichtung des Inhalts im Boxformatfeld festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFormatFieldAlignment **Mögliche Werte:** .vcFFABottom 28 .vcFFABottomLeft 27 .vcFFABottomRight 29 .vcFFACenter 25 .vcFFALeft 24 .vcFFARight 26 .vcFFATop 22 .vcFFATopLeft 21 .vcFFATopRight 23	Ausrichtung des Feldinhalts unten unten links unten rechts unten mittig links rechts oben oben links oben rechts

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Alignment = VcFormatFieldAlignment.vcFFACenter
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Alignment = VcFormatFieldAlignment.vcFFACenter;
```

FormatName

Nur-Lese-Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie den Namen des Boxformats erfragen, zu dem dieses Boxformatfeld gehört.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Boxformats

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
MsgBox(boxFormatField.FormatName)
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
MessageBox.Show(boxFormatField.FormatName);
```

GraphicsHeight

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie beim Typ **vcFFTGraphics** die Höhe der Grafik in dem Boxformatfeld festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 99	Höhe der Grafik in mm 0...200

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics
boxFormatField.GraphicsHeight = 150
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics;
boxFormatField.GraphicsHeight = 150;
```

Index

Nur-Lese-Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie den Index des Boxformatfelds im zugehörigen Boxformat erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Index des Boxformatfeldes

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
MsgBox(boxFormatField.Index)
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
MessageBox.Show(boxFormatField.Index.ToString());
```

MaximumTextLineCount

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die maximale Anzahl der Zeilen in dem Boxformatfeld setzen oder erfragen, falls das Boxformatfeld vom Typ **vcFFTText** ist. Bitte sehen Sie auch die Eigenschaft **MinimumTextLine-Count**.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 9	Maximale Zeilenzahl

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTText
boxFormatField.MaximumTextLineCount = 5
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTText;
boxFormatField.MaximumTextLineCount = 5;
```

MinimumTextLineCount

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die minimale Anzahl der Zeilen in dem Boxformatfeld setzen oder erfragen, falls der Typ des Boxformatfeldes auf **vcFFTText** gesetzt wurde. Ist in einem Knoten mehr Text vorhanden, als in die minimale Anzahl der Zeilen hineinpasst, wird dieses Feld für diesen Knoten dynamisch bis zur maximalen angegebenen Anzahl der Zeilen ausgedehnt. Wenn Sie dieser Eigenschaft einen Wert zuweisen, sollten Sie anschließend auch erneut der Eigenschaft **MaximumTextLineCount** den gewünschten Wert setzen, sonst könnte es vorkommen, dass das Maximum durch das Minimum überschrieben wird.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 9	Minimale Zeilenzahl 0...20

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTText
boxFormatField.MinimumTextLineCount = 3
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTText;
boxFormatField.MinimumTextLineCount = 3;
```

MinimumWidth

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die minimale Breite des Boxformatfeldes in mm festlegen oder erfragen. Die Breite des Feldes kann sich vergrößern, wenn unter oder über dem Feld andere Felder größere minimale Breiten besitzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 9	Minimale Breite des Boxformatfeldes 0...200

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.MinimumWidth = 100
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.MinimumWidth = 100;
```

PatternBackgroundColor

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die Hintergrundfarbe des Boxformatfeldes festlegen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255

(ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt.

Wählen Sie den Wert **-1**, wenn das Feld die Hintergrundfarbe des Boxformats besitzen soll.

	Datentyp	Beschreibung

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.BackgroundColor = Color.Red
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.BackgroundColor = Color.Red;
```

PatternColorAsARGB

Nur-Lese-Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die Musterfarbe des Boxformatfeldes festlegen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt. Bei der Umwandlung eines RGB-Wertes in einen ARGB-Wert muss ein Alpha-Wert von 255 hinzugegeben werden.

Wählen Sie den Wert **-1**, wenn das Feld die Hintergrundfarbe des Boxformats besitzen soll.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	RGB-Farbwerte ({0...255},{0...255},{0...255}) **Standardwert:** -1

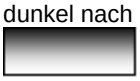
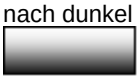
Code-Beispiel VB.NET

```
boxFormatField.PatternColor = RGB(0, 255, 0)
```

PatternEx

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie für den Hintergrund des Boxformatfeldes ein Muster setzen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFieldFillPattern	Mustertyp
	Mögliche Werte:	
	.vcAeroGlassPattern 44	Vertikaler Farbverlauf in der Füllmusterfarbe
	.vcFieldNoPattern 1276	Kein Füllmuster
	.vcFieldVerticalBottomLightedConvexPattern 43	Vertikaler Farbverlauf von dunkel nach hell 
	.vcFieldVerticalConcavePattern 40	Vertikaler Farbverlauf von dunkel über hell nach dunkel
	.vcFieldVerticalConvexPattern 41	Vertikaler Farbverlauf von hell über dunkel nach hell
	.vcFieldVerticalTopLightedConvexPattern 42	Vertikaler Farbverlauf von hell nach dunkel 

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Pattern = VcFillPatternSingleColored.vcSingleColoredNoPatter
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Pattern = VcFillPatternSingleColored.vcSingleColoredNoPattern;
```

TextFont

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die Schriftart des Boxformatfeldes festlegen oder erfragen, falls der Typ des Feldes auf **vcFFTText** gesetzt wurde.

	Datentyp	Beschreibung
Eigenschaftswert	System.DrawingFont	Schriftart des Boxformatfelds

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
MsgBox(boxFormatField.TextFont.FontFamily.ToString())
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
MessageBox.Show(boxFormatField.TextFont.Name.ToString());
```

TextFontColor

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie die Schriftfarbe des Boxformatfeldes festlegen oder erfragen, falls der Typ des Feldes auf **vcFFTText** gesetzt wurde.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	Schriftfarbe des Boxformatfelds **Standardwert:** Color.Black

Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.TextFontColor = Color.Red
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.TextFontColor = Color.Red;
```

Type

Eigenschaft von VcBoxFormatField

Mit dieser Eigenschaft können Sie den Typ des Boxformatfelds erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFormatFieldType **Mögliche Werte:** .vcFFTGraphics 64 .vcFFTText 36	Typ des Boxformatfeldes Grafik Text

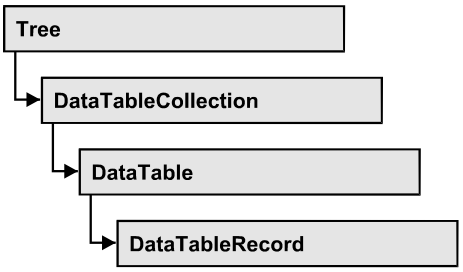
Code-Beispiel VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcTree1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics
boxFormatField.GraphicsHeight = 150
```

Code-Beispiel C#

```
VcBoxFormatCollection boxFormatCltn = vcTree1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics;
boxFormatField.GraphicsHeight = 150;
```

7.9 VcDataRecord



Ein Datensatz ist das logische Grundelement eines Objektes in einem Tree-Diagram, z. B. eines Knotens, eines Gruppenknotens, einer Verbindung, einer Aufgabe, Operation etc. Die Objekte besitzen spezifische Eigenschaften, die in den Feldern des Datensatzes beschrieben werden. Zu den Datenfeldern des Datensatzes existieren entsprechende Beschreibungen, die Datentabellenfelder. Datensätze und Datentabellenfelder werden jeweils zu Collection-Objekten zusammengefasst und bilden eine Datentabelle.

Eigenschaften

- AllData
- DataField
- DataTableName
- ID

Methoden

- Delete
- IdentifyObject
- RelatedDataRecord

Eigenschaften

AllData

Eigenschaft von VcDataRecord

Mit dieser Eigenschaft können alle Daten eines Datansatzes gesetzt oder erfragt werden. Beim Setzen ist ein CSV-String (Semikolon als Trennzeichen) oder der Datentyp "Object" erlaubt, der in einem Array alle Datenfelder des Knotens erhält. Beim Erfragen wird eine Zeichenkette (String) zurückgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.Object	Alle Daten des Datensatzes

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRecValue() As Object
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata1")
dataRecCltn = dataTable.DataRecordCollection
ReDim dataRecValue(dataTable.DataTableFieldCollection.Count)
dataRecValue(0) = 1
dataRecValue(1) = "Node One"

'Object
dataRecord = dataRecCltn.Add(dataRecValue)
'CSV
dataRecord.AllData = "1;Node One;"

dataRecord.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
Object [] dataRecVal = new object[dataTable.DataTableFieldCollection.Count];

dataRecVal[0] = 1;
dataRecVal[1] = "Node One";

//Object
VcDataRecord dataRecord = dataRecordCltn.Add(dataRecVal);
//CSV
dataRecord.AllData = "1;Node One;";

dataRecord.Update();
```

DataField

Eigenschaft von VcDataRecord

Mit dieser Eigenschaft können Sie einem Datenfeld des Datensatzes einen Wert zuweisen oder einen gesetzten Wert erfragen. Wenn ein Datensatz durch diese Methode einen neuen Wert erhalten hat, muss anschließend die grafische Darstellung mit der Methode **UpdateDataRecord** aktualisiert werden.

Die Eigenschaft DataField ist eine indizierte Eigenschaft, die in C# über die beiden Methoden set_DataField (index, pvn) und get_DataField (index) angesprochen wird.

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Datenfeldes
Eigenschaftswert	System.Object	Inhalt des Datenfeldes

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)

dataRecord.DataField(1) = "Node Two"
dataRecord.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);
dataRecord.set_DataField(1, "Node Two");
dataRecord.Update();
```

DataTableName

Nur-Lese-Eigenschaft von VcDataRecord

Mit dieser Eigenschaft können Sie den Namen der Datentabelle erfragen, zu der dieser Datensatz gehört.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der zughörigen Tabelle

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)

MsgBox(dataRecord.DataTableName)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);

MessageBox.Show(dataRecord.DataTableName);
```

ID

Nur-Lese-Eigenschaft von VcDataRecord

Mit dieser Eigenschaft können Sie die ID eines Datensatzes erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Datensatz-ID

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord
dataTable = VcTree1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)
MsgBox(dataRecord.ID)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);
MessageBox.Show(dataRecord.ID);
```

Methoden

Delete

Methode von VcDataRecord

Mit dieser Methode können Sie einen Datensatz löschen.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Datensatz erfolgreich (true) / nicht erfolgreich (false) gelöscht

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)

dataRecord.Delete()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);

dataRecord.Delete();
```

IdentifyObject

Methode von VcDataRecord

Mit dieser Methode kann man erfragen, ob und welches datenbasierte Objekt aus dem Datensatz erzeugt wurde.

Die Methode liefert als Rückgabewert **true**, wenn ein datenbasiertes Objekt ermittelt werden konnte, d.h. wenn aus dem Datensatz für die Grafik ein datenbasiertes Objekt hergestellt wurde.

	Datentyp	Beschreibung
Parameter:		
⇨ establishedObject Param	System.Object	Erkanntes Objekt
establishedObjectTypeParam	VcObjectType	Typ des erkannten Objekts
	Mögliche Werte: .vcObjTypeBox 15 .vcObjTypeNode 2 .vcObjTypeNone 0	 Objekttyp **Box** Objekttyp **Knoten** kein Objekt
Rückgabewert	System.Boolean	datenbasiertes Objekt wurde/ wurde nicht erzeugt

RelatedDataRecord

Methode von VcDataRecord

Mit dieser Eigenschaft können Sie einem Datensatz einen weiteren zuordnen oder einen zugeordneten Datensatz erfragen. Bei der Verwendung von erweiterten Tabellen (extended data tables) können Datensätze einer Tabelle über einen Primärschlüssel den Datensätzen einer anderen Tabelle zugeordnet werden.

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Datenfeldes
Rückgabewert	VcDataRecord	Zugeordneter Datensatz

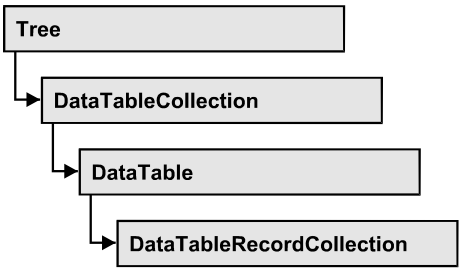
Code-Beispiel VB.NET

```
   Private Sub VcTree1_VcNodeLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftClicking
      Dim dataTable As VcDataTable
      Dim dataRecordCltn As VcDataRecordCollection
      Dim firstDataRecord As VcDataRecord
      Dim secondDataRecord As VcDataRecord

      dataTable = VcTree1.DataTableCollection.DataTableByIndex(0)
      dataRecordCltn = dataTable.DataRecordCollection

      firstDataRecord = dataRecordCltn.DataRecordByID(e.Node.DataField(0))
      secondDataRecord = firstDataRecord.RelatedDataRecord(2)

      MsgBox(secondDataRecord.AllData)
   End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
   {
   VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByIndex(0);
   VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
   VcDataRecord firstDataRecord =
dataRecordCltn.DataRecordByID(e.Node.get_DataField(0));
   VcDataRecord secondDataRecord = firstDataRecord.RelatedDataRecord(2);

   MessageBox.Show(secondDataRecord.AllData.ToString());
   }
```

7.10 VcDataRecordCollection



In einem Objekt vom Typ **VcDataRecordCollection** sind die Datensätze einer Datentabelle zusammengefasst. Mit der Eigenschaft **Count** kann die Anzahl der Datensätze im Collection-Objekt erfragt werden; mit dem Enumerator-Objekt und den Methoden **FirstDataRecord** und **NextDataRecord** können Sie iterativ auf die Datensätze zugreifen sowie mit **DataRecordByID** auf einzelne Datensätze; die Methoden **Add** und **Remove** ermöglichen das Hinzufügen und Entfernen von Datensätzen und mit **Update** können Sie die grafische Darstellung der Datensätze mit neu eingegebenen Daten aktualisieren.

Eigenschaften

- Count

Methoden

- Add
- DataRecordByID
- FirstDataRecord
- GetEnumerator
- NextDataRecord
- Remove
- Update

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcDataRecordCollection

Mit dieser Eigenschaft können Sie die Anzahl der Datensätze in der Data-Record-Auflistung erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Datensätze im Collection-Objekt

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
MsgBox("Number of DataRecords: " & dataRecordCltn.Count)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
MessageBox.Show("Number of DataRecords: " + dataRecordCltn.Count);
```

Methoden

Add

Methode von VcDataRecordCollection

Mit dieser Methode können Sie einen neuen Datensatz in der DataRecordCollection anlegen. Wenn die ID noch nicht verwendet wurde, wird der neue Datensatz zurückgegeben, andernfalls wird eine **VcPrimaryKeyNotUniqueException** erzeugt. Nach dem Anlegen des Datensatzes muss die Methode **VcTree.EndLoading** aufgerufen werden, damit die Änderung wirksam wird.

	Datentyp	Beschreibung
Parameter:		
⇨ dataRecordContent	VcObject	Inhalt des Datensatzes (als Array oder String)
Rückgabewert	VcDataRecord	Neue angelegter Datensatz

Code-Beispiel VB.NET

```
Const Main_ID = 0
Const Main_Name = 1
Const Main_Start = 2
Const Main_Duration = 4
'...

Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim dataRecVal() As Object

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection

Dim dataRec1 As VcDataRecord
ReDim dataRecVal(DataTable.DataTableFieldCollection.Count)

dataRecVal(Main_ID) = 1
dataRecVal(Main_Name) = "Node 1"
dataRecVal(Main_Start) = DateSerial(2014, 1, 8)
dataRecVal(Main_Duration) = 8
dataRec1 = dataRecCltn.Add(dataRecVal)
VcTree1.EndLoading()

' equivalent
'  dataRec1 = dataRecCltn.Add("1;Node 1;01.08.14;;8")
```

Code-Beispiel C#

```
const int Main_ID = 0;
const int Main_Name = 1;
const int Main_Start = 2;
const int Main_Duration = 4;

//...

VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");

VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

Object [] dataRecVal = new object[dataTable.DataTableFieldCollection.Count];

VcDataRecord dataRec1;

dataRecVal[Main_ID] = "1";
dataRecVal[Main_Name] = "Node 1";
dataRecVal[Main_Start] = "08.01.2014";
dataRecVal[Main_Duration] = 8;
dataRec1 = dataRecCltn.Add(dataRecVal);
VcTree1.EndLoading();

// equivalent
// dataRec2 = dataRecCltn.Add("1;Node 1;01.08.14;;8")
```

DataRecordByID

Methode von VcDataRecordCollection

Mit dieser Methode können Sie auf einen einzelnen Datensatz über seine Identifikation zugreifen. Existiert kein Datensatz unter der angegebenen ID, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

Wenn die Identifikation aus mehreren Feldern besteht (zusammengesetzter Primärschlüssel), muss diese mehrteilige ID folgendermaßen angegeben werden:

ID=ID1|ID2|ID3

	Datentyp	Beschreibung
Parameter:		
⇨ dataRecordID	System.String	ID des Datensatzes
Rückgabewert	VcDataRecord	Datensatzobjekt

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(0)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(0);
```

FirstDataRecord

Methode von VcDataRecordCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. den ersten Datensatz der DataRecord-Auflistung zugreifen, um anschließend in einer Schleife mit der Methode **NextDataRecord** über die nachfolgenden Datensätze zu iterieren. Existiert kein Datensatz in der Datensatzauflistung, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcDataRecord	Erster Datensatz

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.FirstDataRecord
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.FirstDataRecord();
```

GetEnumerator

Methode von VcDataRecordCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Datensatz-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Enumerator-Objekt

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection

VcTree1.SuspendUpdate(True)

dataRecord = dataRecordCltn.FirstDataRecord
While Not dataRecord Is Nothing
   dataRecord.DataField(4) = "10"
   dataRecord.Update()
   dataRecord = dataRecordCltn.NextDataRecord
End While

VcTree1.SuspendUpdate(False)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;

vcTree1.SuspendUpdate(true);

foreach (VcDataRecord dataRecord in dataRecordCltn)
{
   dataRecord.set_DataField(4, "10");
   dataRecord.Update();
   dataRecordCltn.NextDataRecord();
}

vcTree1.SuspendUpdate(false);
```

NextDataRecord

Methode von VcDataRecordCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Datensätze des DataRecordCollection-Objekts zugreifen, nachdem Sie mit der Methode **FirstDataRecord** den Initialwert erfasst haben. Sind alle Datensätze durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcDataRecord	Nachfolgende Datensatz

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection

VcTree1.SuspendUpdate(True)

dataRecord = dataRecordCltn.FirstDataRecord
While Not dataRecord Is Nothing
   dataRecord.DataField(4) = "10"
   dataRecord.Update()
   dataRecord = dataRecordCltn.NextDataRecord
End While

VcTree1.SuspendUpdate(False)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;

vcTree1.SuspendUpdate(true);

foreach (VcDataRecord dataRecord in dataRecordCltn)
{
   dataRecord.set_DataField(4, "10");
   dataRecord.Update();
   dataRecordCltn.NextDataRecord();
}

vcTree1.SuspendUpdate(false);
```

Remove

Methode von VcDataRecordCollection

Mit dieser Methode können Sie einen Datensatz löschen. Die Methode liefert **true** wenn gelöscht wurde und **false**, wenn nicht gelöscht wurde. Der Datensatzinhalt im Übergabeparameter wird dazu verwendet, um anhand der Identifizierung das Objekt zu finden.

	Datentyp	Beschreibung
Parameter: ⇨ dataRecordContent	VcObject	Inhalt des Datensatzes (als Array oder String)
Rückgabewert	System.Boolean	true

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecordCltn.Remove("1;1Activity; Y;Z;18.01.14;;5")
VcTree1.EndLoading()

' equivalent
' dataRecord = dataRecordCltn.DataRecordByID(1)
' dataRecord.Delete()
' dataRecord.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

dataRecCltn .Remove("1;1Activity Y;Z;18.01.14;;5");
VcTree1.EndLoading();

// equivalent
// VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);
// dataRecord.Delete();
// dataRecord.Update();
```

Update

Methode von VcDataRecordCollection

Mit dieser Methode können Sie einen Datensatz in der Datensatzliste aktualisieren, nachdem mittels der Methode **Add()** der Datensatz vorher hinzugefügt wurde. Falls der zu aktualisierende Datensatz nicht existiert, d.h. also nicht vorher angelegt wurde, wird er mittels Update() nun neu angelegt. Siehe auch **VcDataRecordCollection.Add()**. Nach dem Aktualisieren des Datensatzes muss die Methode **VcTree.EndLoading** aufgerufen werden, damit die Änderung wirksam wird.

	Datentyp	Beschreibung
Parameter: ⇨ dataRecordContent	VcObject	Inhalt des Datensatzes (als Array oder String)
Rückgabewert	System.Boolean	Aktualisierung erfolgt (true) / nicht erfolgt (false)

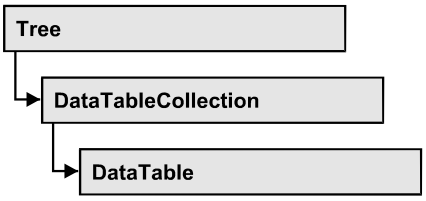
Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecordCltn.Update("1;1.8.2017;;8")
VcTree1.EndLoading()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;
dataRecCltn.Update("1;1.8.2017;;8");
VcTree1.EndLoading();
```

7.11 VcDataTable



Eine Datentabelle umfasst **Datensätze** (data records) mit ihren Datenfeldern und ihren Inhalten sowie die Beschreibungen der Datenfelder, die **Tabellen-datenfelder** (data table fields) genannt werden. Datensätze und Datentabellenfelder können in der Form von eigenen Auflistungen (Collection-Objekten) verwaltet werden.

Datentabellen ihrerseits können ebenfalls in eigenen Auflistungen verwaltet werden.

Eigenschaften

- DataRecordCollection
- DataTableFieldCollection
- Description
- MultiplePrimaryKeysAllowed
- Name

Eigenschaften

DataRecordCollection

Nur-Lese-Eigenschaft von VcDataTable

Diese Eigenschaft gibt die in der Datentabelle enthaltene Datensatz-Auflistung zurück. Die Datensatz-Auflistung enthält alle existierenden Datensätze einer Tabelle. Zu Programmbeginn ist sie leer.

	Datentyp	Beschreibung
Eigenschaftswert	VcDataRecordCollection	DataRecordCollection-Objekt

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcTree1.DataTableCollection.FirstDataTable()
MsgBox(dataTable.DataRecordCollection.Count)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
MessageBox.Show(dataTable.DataRecordCollection.Count.ToString());
```

DataTableFieldCollection

Nur-Lese-Eigenschaft von VcDataTable

Diese Eigenschaft gibt die in der Datentabelle enthaltene Datentabellenfeld-Auflistung zurück. Die Datentabellenfeld-Auflistung enthält die Definition der Datenfelder eines Datensatzes der Tabelle. Zu Programmbeginn enthält sie die bereits zur Designzeit vereinbarten Datenfelder. Weitere Datenfelder können zur Laufzeit über die Methode **Add** des Objekts **DataTableField-Collection** hinzugefügt werden. Die Definition der Datentabellenfelder muss abgeschlossen sein, bevor die Tabelle mit Datensätzen gefüllt wird.

	Datentyp	Beschreibung
Eigenschaftswert	VcDataTableFieldCollection	DataTableFieldCollection-Objekt

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcTree1.DataTableCollection.DataTableByIndex(0)
MsgBox(dataTable.DataTableFieldCollection.Count)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByIndex(0);
MessageBox.Show(dataTable.DataTableFieldCollection.Count.ToString());
```

Description

Eigenschaft von VcDataTable

Mit dieser Eigenschaft können sie eine Beschreibung der Datentabelle setzen oder erfragen. Sprechende Namen, z.B. der Name der Tabelle, sind häufig sehr lang und werden daher bei Previews nicht vollständig angezeigt, so dass ihr Nutzen nicht zum Tragen kommen kann. Damit Sie für die vollständige Anzeige kurze Namen verwenden können und trotzdem nicht auf die gewünschte Information verzichten müssen, können Sie in diesem Feld zusätzliche Informationen zum Tabellennamen speichern. Sein Inhalt wird im Dialog zur Datentabelle angezeigt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Beschreibung der Datentabelle **Standardwert:** Empty string

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcTree1.DataTableCollection.DataTableByName("Maindata")
dataTable.Description = "This table contains data for nodes"
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByName("Maindata");
dataTable.Description = "This table contains data for nodes";
```

MultiplePrimaryKeysAllowed

Eigenschaft von VcDataTable

Mit dieser Eigenschaft können Sie angeben oder erfragen, ob die Verwendung zusammengesetzter Primärschlüssel möglich ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Verwendung von zusammengesetzten Primärschlüsseln erlaubt (true)/nicht erlaubt (false) **Standardwert:** False

Name

Eigenschaft von VcDataTable

Mit dieser Eigenschaft können sie den Namen der Datentabelle setzen oder erfragen. Ein Name für die Datentabelle ist obligat und muss eindeutig sein; zudem ist eine leere Zeichenkette nicht erlaubt. Unterscheidungen in der Groß- und Kleinschreibung führen zu unterschiedlichen Namen. Mit der Methode **DataTableByName** des Objekts **DataTableCollection** können Sie über den Tabellennamen eine Referenz auf das Datentabellenobjekt erhalten.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Datentabelle **Standardwert:** Empty string

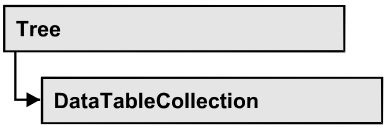
Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcTree1.DataTableCollection.DataTableByIndex(0)
MsgBox(dataTable.Name)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.DataTableByIndex(0);
MessageBox.Show(dataTable.Name);
```

7.12 VcDataTableCollection



In einem Objekt vom Typ VcDataTableCollection sind die vorhandenen Datentabellen zusammengefasst. Mit der Eigenschaft **Count** kann die Anzahl der Tabellen im Collection-Objekt erfragt werden; mit dem Enumerator-Objekt und den Methoden **FirstDataTable** und **NextDataTable** können Sie iterativ auf die Tabellen zugreifen sowie mit **DataTableByName** und **DataTableByIndex** auf einzelne Tabellen; die Methoden **Add** und **Copy** ermöglichen das Hinzufügen und Kopieren von Tabellen und mit **Update** können Sie dem XTree-Objekt die neuen Änderungen der Datenstrukturen bekanntgeben.

Eigenschaften

- Count

Methoden

- Add
- Copy
- DataTableByIndex
- DataTableByName
- FirstDataTable
- GetEnumerator
- NextDataTable
- Update

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcDataTableCollection

Mit dieser Eigenschaft können Sie die Anzahl der Datentabellen in der DataTable-Auflistung erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Datentabellen im Collection-Objekt

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection

dataTableCltn = VcTree1.DataTableCollection
MsgBox(dataTableCltn.Count.ToString())
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
MessageBox.Show(dataTableCltn.Count.ToString());
```

Methoden

Add

Methode von VcDataTableCollection

Mit dieser Methode können Sie eine neue Datentabelle in der DataTable-Auflistung anlegen. Wenn der Tabellenname noch nicht verwendet wurde, wird ein Objekt vom Typ **VcDataTable** zurückgegeben, andernfalls "Nothing" (in Visual Basic) oder "0" (in anderen Sprachen). Nur wenn die Eigenschaft **ExtendedDataTables** auf **True** gesetzt ist, können Tabellen angelegt werden. Maximal 90 Datentabellen können angelegt werden.

	Datentyp	Beschreibung
Parameter:		
⇨ dataTableName	System.String	Name der neuen Tabelle
Rückgabewert	VcDataTable	Neu angelegte Datentabelle

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcTree1.DataTableCollection
dataTable = dataTableCltn.Add("Resources")
dataTableCltn.Update()
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.Add("Resources");
dataTableCltn.Update();
```

Copy

Methode von VcDataTableCollection

Mit dieser Methode können Sie eine Datentabelle kopieren. Es wird nur die Tabellendefinition kopiert, jedoch nicht die eventuell bereits existierende Datensätze. Nur wenn die Eigenschaft **ExtendedDataTables** auf **true** gesetzt ist, können Tabellen angelegt werden. Wenn die Tabelle kopiert werden konnte, wird ein neues Objekt vom Typ **VcDataTable** zurückgegeben, andernfalls **Nothing** (in Visual Basic) oder **0** (in anderen Sprachen). Es wird bei den Tabellennamen generell zwischen Groß- und Kleinschreibung unterschieden.

	Datentyp	Beschreibung
Parameter:		
⇨ dataTableName	System.String	Name der zu kopierenden Datentabelle (Quelltabelle)
⇨ newDataTableName	System.String	Name der neu zu erstellenden Datentabelle (Zieltabelle)
Rückgabewert	VcDataTable	Neu erstelltes Datentabellen-Objekt

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcTree1.DataTableCollection
dataTable = dataTableCltn.Copy("Resources", "NewResources")
dataTableCltn.Update()
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.Copy("Resources", "NewResources");
dataTableCltn.Update();
```

DataTableByIndex

Methode von VcDataTableCollection

Mit dieser Methode können Sie auf eine einzelne Datentabelle über ihren Index zugreifen. Der Index der ersten Tabelle ist 0. Existiert keine Tabelle unter dem angegebenen Index, wird ein Leerobjekt zurückgegeben (**Nothing** in Visual Basic oder **0** in anderen Sprachen).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index der Datentabelle

Rückgabewert	VcDataTable	Ermitteltes Datentabellenobjekt

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcTree1.DataTableCollection
dataTable = dataTableCltn.DataTableByIndex(2)
MsgBox(dataTable.Name)
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.DataTableByIndex(2);
MessageBox.Show(dataTable.Name);
```

DataTableByName

Methode von VcDataTableCollection

Mit dieser Methode können Sie auf eine einzelne Datentabelle über ihren Namen zugreifen. Existiert keine Tabelle unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (**Nothing** in Visual Basic oder **0** in anderen Sprachen).

	Datentyp	**Beschreibung**
Parameter:		
⇨ dataTableName	System.String	Name der Datentabelle
Rückgabewert	VcDataTable	Ermitteltes Datentabellenobjekt

Code-Beispiel VB.NET

```
Dim dataTablecltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTablecltn = VcTree1.DataTableCollection
dataTable = dataTablecltn.DataTableByName("Resources")
MsgBox(dataTable.Description)
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.DataTableByName("Resources");
MessageBox.Show(dataTable.Description);
```

FirstDataTable

Methode von VcDataTableCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. die erste Datentabelle der DataTable-Auflistung zugreifen, um anschließend in einer Schleife mit

der Methode **NextDataTable** über die nachfolgenden Tabellen zu iterieren. Existiert keine Datentabelle in der DataTable-Auflistung, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcDataTable	Erste Datentabelle

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcTree1.DataTableCollection
dataTable = dataTableCltn.FirstDataTable
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
VcDataTable dataTable= dataTableCltn.FirstDataTable();
```

GetEnumerator

Methode von VcDataTableCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Datentabellen iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Enumerator-Objekt

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcTree1.DataTableCollection
For Each dataTable In dataTableCltn
   ListBox1.Items.Add(dataTable.Name)
Next
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
foreach (VcDataTable dataTable in dataTableCltn)
   listBox1.Items.Add(dataTable.Name);
```

NextDataTable

Methode von VcDataTableCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Datentabellen des DataTableCollection-Objekts zugreifen, nachdem Sie mit der Methode **FirstDataTable** den Initialwert erfasst haben. Sind alle Tabellen durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcDataTable	Nachfolgende Datentabelle

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable
Dim i As Integer

dataTableCltn = VcTree1.DataTableCollection
dataTable = dataTableCltn.FirstDataTable
For i = 1 To dataTableCltn.Count
   ListBox1.Items.Add(dataTable.Name)
   dataTable = dataTableCltn.NextDataTable
Next
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.FirstDataTable();
for (int i=0; i<dataTableCltn.Count; i++)
{
   listBox1.Items.Add(dataTable.Name);
   dataTable = dataTableCltn.NextDataTable();
}
```

Update

Methode von VcDataTableCollection

Mit dieser Methode können Sie neue Änderungen an Datenstrukturen aktualisieren. Der Aufruf ist notwendig, damit durchgeführte Änderungen an der Datentabellendefinition und an den Datentabellenfeldern in der VARCHART Komponente wirksam werden. Auf diese Weise werden bei mehreren Änderungen unnötige Aktualisierungen vermeiden.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Aktualisierung erfolgt (true) / nicht erfolgt (false)

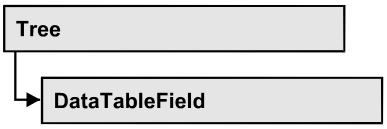
Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcTree1.DataTableCollection
dataTable = dataTableCltn.Add("Resources")
dataTable.DataTableFieldCollection.Add("Id")
dataTableCltn.Update()
```

Code-Beispiel C#

```
VcDataTableCollection dataTableCltn = vcTree1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.Add("Resources");
dataTable.DataTableFieldCollection.Add("Id");
dataTableCltn.Update();
```

7.13 VcDataTableField



Ein Objekt vom Typ **VcDataTableField** legt die Eigenschaften eines Feldes der Datentabelle fest. Zur Definition eines Datentabellenfeldes gehört der Name, der Datentyp und die Festlegung, ob das Datenfeld als Primärschlüssel dient, der zur eindeutigen Identifizierung eines Datensatzes herangezogen werden kann. Unter Verwendung des Primärschlüssels z.B. kann von anderen Datentabellen auf diese Datentabelle Bezug genommen werden. Um eine Beziehung zu einer Datentabelle mit Primärschlüssel aufzubauen, muss der Primärschlüssel im Feld **RelationshipFieldIndex** benannt werden.

Die DataTableField-Objekte einer Datentabelle werden über die Auflistung **DataTableFieldCollection** verwaltet.

Eigenschaften

- DataTableName
- DateFormat
- Editable
- Hidden
- Index
- Name
- PrimaryKey
- RelationshipFieldIndex
- Type

Eigenschaften

DataTableName

Nur-Lese-Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie den Namen der zugehörigen Datentabelle erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Datentabelle

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcTree1.DataTableCollection.FirstDataTable
MsgBox(dataTable.DataTableFieldCollection.FirstDataTableField.DataTableName)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
MessageBox.Show(dataTable.DataTableFieldCollection.FirstDataTableField().DataTab
leName);
```

DateFormat

Nur-Lese-Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie das Datumsformat des Datentabellenfeldes festlegen oder erfragen. Das Datumsformat wird beim Lesen und Schreiben von CSV-Dateien verwendet und wenn der Formattyp **String** beim Hinzufügen eines Datensatzes über die Methode **Add** zur DataRecord-Auflistung verwendet wird. Diese Eigenschaft ist nur wirksam, wenn der Datentyp des Feldes auf **vcDataTableFieldDateTime** eingestellt ist.

Hinweis: Es sollte zuerst die Eigenschaft **Type** gesetzt werden, bevor die Eigenschaft **DateFormat** vereinbart wird.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Datumsformat {DMYhms:;./}

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType
'DateFormat = "01.12.2014"
dataTableField.DateFormat = "DD.MM.YYYY"
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType;
//DateFormat = "01.12.2014"
dataTableField.DateFormat = "DD.MM.YYYY";
vcTree1.DataTableCollection.Update();
```

Editable

Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob das spezifizierte Datenfeld zur Laufzeit in der Tabelle (des Diagramms) und des Dialogs **Knoten bearbeiten** editierbar sein soll.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Feld editierbar (True) / nicht editierbar (False) **Standardwert:** True

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Editable = False
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Editable = false;
VcTree1.DataTableCollection.Update();
```

Hidden

Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob das Datenfeld zur Laufzeit in den Dialogen **EditNode** oder **EditLink** angezeigt wird.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Feld wird nicht angezeigt (True) / angezeigt (False) **Standardwert:** False

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Hidden = True
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Hidden = true;
vcTree1.DataTableCollection.Update();
```

Index

Nur-Lese-Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie den Index des Datentabellenfelds in der zugehörigen Datentabelle erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Index des Datentabellenfeldes.

Name

Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie den Namen des Datenfelds setzen oder erfragen. Der Name des Datenfelds erscheint innerhalb von Laufzeitdialogen wie beispielsweise dem **EditNode**-Dialog. In der API erfolgt der Zugriff auf die Datenfeldinhalte eines Datensatzes jedoch immer über den Index, den dieses Feld im **DataTableFieldCollection**-Objekt besitzt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Feldes **Standardwert:** Empty string

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.DataTableByName("Operation")
dataTableField = dataTable.DataTableFieldCollection.Add("Start")
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.Add("Start");
vcTree1.DataTableCollection.Update();
```

PrimaryKey

Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob dieses Datenfeld den Primärschlüssel enthält, der zur eindeutigen Identifikation eines Datensatz herangezogen werden kann. In einer Datentabelle kann immer nur ein Datenfeld die Kennung Primärschlüssel tragen. Die aktuelle Setzung hebt eine eventuell vorhandene Setzung bei einem anderen Datenfeld der gleichen Tabelle auf. Die Festlegung eines Primärschlüssels ist unerlässlich, wenn eine Beziehung von einer anderen Tabelle zu dieser Tabelle hergestellt werden soll.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Das Feld dient (True) / dient nicht (False) als Primärschlüssel **Standardwert:** False

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField
Dim isPrimaryKey As Boolean

dataTable = VcTree1.DataTableCollection.DataTableByName("Operation")
dataTableField = dataTable.DataTableFieldCollection.DataTableFieldByName("Id")
dataTableField.PrimaryKey = True
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Id");
dataTableField.PrimaryKey = true;
vcTree1.DataTableCollection.Update();
```

RelationshipFieldIndex

Eigenschaft von VcDataTableField

Mit dieser Eigenschaft verbinden Sie ein Datenfeld und seine Beschreibung. Dazu setzen Sie hier den Index des Datensatzfeldes, auf welches sich die gesetzten Eigenschaften des Datentabellenfeldes beziehen sollen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Index des Datensatzfeldes, auf das sich die Datendefinition des Datentabellenfeldes bezieht. **Standardwert:** -1

Code-Beispiel VB.NET

```
Dim dataTableTask As VcDataTable
Dim dataTaskFieldId As VcDataTableField
Dim dataTaskFieldName As VcDataTableField
Dim dataTableOperation As VcDataTable
Dim dataOperationFieldId As VcDataTableField
Dim dataOperationFieldName As VcDataTableField
Dim dataOperationFieldTaskId As VcDataTableField

'Create table Task
dataTableTask = VcTree1.DataTableCollection.Add("Task")
dataTaskFieldId = dataTableTask.DataTableFieldCollection.Add("Id")
dataTaskFieldId.PrimaryKey = True
dataTaskFieldName = dataTableTask.DataTableFieldCollection.Add("Name")
dataTaskFieldName.Type = VcDataTableFieldType.vcDataTableFieldStringType

'Create table Operation
dataTableOperation = VcTree1.DataTableCollection.Add("Operation")
dataOperationFieldId = dataTableOperation.DataTableFieldCollection.Add("Id")
dataOperationFieldId.PrimaryKey = True
dataOperationFieldName = dataTableOperation.DataTableFieldCollection.Add("Name")
dataOperationFieldName.Type = VcDataTableFieldType.vcDataTableFieldStringType
dataOperationFieldTaskId =
dataTableOperation.DataTableFieldCollection.Add("TaskId")
dataOperationFieldTaskId.Type = VcDataTableFieldType.vcDataTableFieldIntegerType

'Node tables Task and Operations
dataOperationFieldTaskId.RelationshipFieldIndex =
VcTree1.DetectFieldIndex("Task", "Id")
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
//Create table Task
VcDataTable dataTableTask = vcTree1.DataTableCollection.Add("Task");
VcDataTableField dataTaskFieldId =
dataTableTask.DataTableFieldCollection.Add("Id");
dataTaskFieldId.PrimaryKey = true;
VcDataTableField dataTaskFieldName =
dataTableTask.DataTableFieldCollection.Add("Name");
dataTaskFieldName.Type = VcDataDefinitionFieldType.vcDefFieldStringType;

//Create table Operation
VcDataTable dataTableOperation = vcTree1.DataTableCollection.Add("Operation");
VcDataTableField dataOperationFieldId =
dataTableOperation.DataTableFieldCollection.Add("Id");
dataOperationFieldId.PrimaryKey = true;
VcDataTableField dataOperationFieldName =
dataTableOperation.DataTableFieldCollection.Add("Name");
 dataOperationFieldName.Type = VcDataDefinitionFieldType.vcDefFieldStringType;
VcDataTableField dataOperationFieldTaskId =
dataTableOperation.DataTableFieldCollection.Add("TaskId");
dataOperationFieldTaskId.Type = VcDataDefinitionFieldType.vcDefFieldIntegerType;

//Node tables Task and Operation
dataOperationFieldTaskId.RelationshipFieldIndex =
vcTree1.DetectFieldIndex("Task","Id");
vcTree1.DataTableCollection.Update();
```

Type

Eigenschaft von VcDataTableField

Mit dieser Eigenschaft können Sie den Datentyp des Feldes setzen oder erfragen.

Hinweis: Durch Setzen der Eigenschaft **Type** kann sich die Eigenschaft **DateFormat** ändern. Durch Setzen des Eigenschaftswertes auf **vcDataTableAlphanumeric** oder **vcDataTableFieldInteger** wird ein eventuell eingestelltes Datumsformat auf "" gesetzt.

	Datentyp	Beschreibung
Eigenschaftswert	VcDataTableFieldType	Datentyp des Feldes, kann maximal 512 Zeichen enthalten **Standardwert:** vcDataTableFieldIntegerType

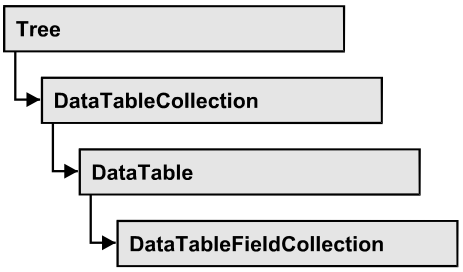
Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

VcTree1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable =
vcTree1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType;
vcTree1.DataTableCollection.Update();
```

7.14 VcDataTableFieldCollection



In einem Objekt vom Typ VcDataTableFieldCollection sind die Datentabellenfelder einer Datentabelle zusammengefasst. Mit der Eigenschaft **Count** kann die Anzahl der Felder im Collection-Objekt erfragt werden; mit dem Enumerator-Objekt und den Methoden **FirstDataField** und **NextDataField** können Sie iterativ auf die Felder zugreifen sowie mit **DataFieldByName** und **DataFieldByIndex**auf einzelne Felder; die Methoden **Add** und **Copy** ermöglichen das Hinzufügen und Kopieren von Feldern.

Eigenschaften

- Count

Methoden

- Add
- Copy
- DataTableFieldByIndex
- DataTableFieldByName
- FirstDataTableField
- GetEnumerator
- NextDataTableField

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcDataTableFieldCollection

Mit dieser Eigenschaft können Sie die Anzahl der Datentabellenfelder in der DataTableField-Auflistung erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Datentabellenfelder im Collection-Objekt

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcTree1.DataTableCollection.FirstDataTable()
MsgBox(dataTable.DataTableFieldCollection.Count.ToString())
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
MessageBox.Show(dataTable.DataTableFieldCollection.Count.ToString());
```

Methoden

Add

Methode von VcDataTableFieldCollection

Mit dieser Methode können Sie ein neues Datentabellenfeld in der DataTableFieldCollection anlegen. Wenn der Name noch nicht verwendet wurde, wird das neue Datenfeld zurückgegeben, andernfalls "Nothing" (Visual Basic) oder "0" (andere Sprachen). Maximal 9.999 Felder können angelegt werden.

	Datentyp	Beschreibung
Parameter:		
⇨ dataTableFieldName	System.String	Name des neuen Datentabellenfeldes
Rückgabewert	VcDataTableField	Neu angelegtes Datentabellenfeld

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.Add("Priority")
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.Add("Priority");
vcTree1.DataTableCollection.Update();
```

Copy

Methode von VcDataTableFieldCollection

Mit dieser Methode können Sie ein Datentabellenfeld kopieren. Die Identifizierung des Feldes erfolgt über den Namen.

	Datentyp	Beschreibung
Parameter:		
⇨ dataTableFieldName	System.String	Name des zu kopierenden Datentabellenfeldes (Quellfeld)
⇨ newDataTableFieldName	System.String	Name des neu zu erstellenden Datentabellenfeldes (Zielfeld)
Rückgabewert	VcDataTableField	Neu angelegtes Datentabellenfeld

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.Copy("Name", "NewName")
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.Copy("Name", "NewName");
vcTree1.DataTableCollection.Update();
```

DataTableFieldByIndex

Methode von VcDataTableFieldCollection

Mit dieser Methode können Sie auf einen einzelnen Datensatz über seinen Index zugreifen. Existiert kein Datensatz an dem angegebenen Index, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Datentabellenfeldes
Rückgabewert	VcDataTableField	Ermitteltes Datentabellenfeld

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.DataTableFieldByIndex(1)
MsgBox(dataTableField.Name)
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByIndex(1);
MessageBox.Show(dataTableField.Name);
```

DataTableFieldByName

Methode von VcDataTableFieldCollection

Mit dieser Methode können Sie auf ein einzelnes Datentabellenfeld über seinen Namen zugreifen. Existiert kein Feld unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ dataTableFieldName	System.String	Name des Datentabellenfeldes
Rückgabewert	VcDataTableField	Ermitteltes Datentabellenfeld

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.DataTableFieldByName("Name")
dataTableField.Editable = False
VcTree1.DataTableCollection.Update()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Name");
dataTableField.Editable = false;
vcTree1.DataTableCollection.Update();
```

FirstDataTableField

Methode von VcDataTableFieldCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. das erste Datenfeld der DataTableField-Auflistung zugreifen, um anschließend in einer Schleife mit der Methode **NextDataTableField** über die nachfolgenden Datenfelder

zu iterieren. Existiert kein Datenfeld in der DataTableField-Auflistung, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcDataTableField	Erstes Datentabellenfeld

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.FirstDataTableField()
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.FirstDataTableField();
```

GetEnumerator

Methode von VcDataTableFieldCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Tabellendatenfeld-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Enumerator-Objekt

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcTree1.DataTableCollection.FirstDataTable()
For Each dataTableField In dataTable.DataTableFieldCollection
   ListBox1.Items.Add(dataTableField.Name)
Next
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
foreach (VcDataTableField dataTableField in dataTable.DataTableFieldCollection)
   listBox1.Items.Add(dataTableField.Name);
```

NextDataTableField

Methode von VcDataTableFieldCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Datentabellenfelder des DataTableFieldCollection-Objekts zugreifen, nachdem Sie mit der Methode **FirstDataTableField** den Initialwert erfasst haben. Sind alle Felder durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcDataTable	Nachfolgendes Datentabellenfeld

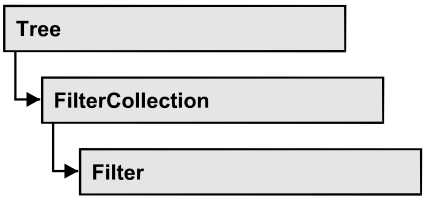
Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableFieldCltn As VcDataTableFieldCollection
Dim dataTableField As VcDataTableField
Dim i As Integer

dataTable = VcTree1.DataTableCollection.FirstDataTable()
dataTableFieldCltn = dataTable.DataTableFieldCollection
dataTableField = dataTableFieldCltn.FirstDataTableField
For i = 1 To dataTableFieldCltn.Count
   ListBox1.Items.Add(dataTableField.Name)
   dataTableField = dataTableFieldCltn.NextDataTableField()
Next
```

Code-Beispiel C#

```
VcDataTable dataTable = vcTree1.DataTableCollection.FirstDataTable();
VcDataTableFieldCollection dataTableFieldCltn =
dataTable.DataTableFieldCollection;
VcDataTableField dataTableField = dataTableFieldCltn.FirstDataTableField();
for (int i=0; i<dataTableFieldCltn.Count; i++)
   {
   listBox1.Items.Add(dataTableField.Name);
   dataTableField = dataTableFieldCltn.NextDataTableField();
   }
```

7.15 VcFilter



Ein Objekt vom Typ VcFilter enthält Filterbedingungen (VcFilterSubCondition), z. B. zulässige Werte für die Datenfelder eines Knotens. Abhängig von den Daten trifft die Filterbedingung für einen Vorgang zu oder nicht. Filter werden verwendet, um z. B. einem Knoten ein bestimmtes Format zuzuweisen.

Nur wenn der Filter nach Änderungen der Filterbedingungen gültig ist, werden diese wirksam. Andernfalls bleiben die bisherigen Filterbedingungen wirksam (prüfbar über die Methoden VcFilter.IsValid und VcFilterSubCondition.IsValid).

Eigenschaften

- DatesWithHourAndMinute
- Name
- Specification
- StringsCaseSensitive
- SubCondition
- SubConditionCount

Methoden

- AddSubCondition
- CopySubCondition
- Evaluate
- GetEnumerator
- IsValid
- RemoveSubCondition

Eigenschaften

DatesWithHourAndMinute

Eigenschaft von VcFilter

Diese Eigenschaft entscheidet, ob beim Vergleich von Datums-Filterbedingungen Stunden und Minuten berücksichtigt werden. Die Einstellung kann nur geändert werden, wenn mindestens eine Unterbedingung mit einem Datumsvergleich vorhanden ist. Ansonsten ist der Eigenschaftswert immer False.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Stunden und Minuten werden berücksichtigt (True)/ nicht berücksichtigt (False)

Name

Eigenschaft von VcFilter

Mit dieser Eigenschaft können Sie den Namen des Filters erfragen oder setzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Filters

Code-Beispiel VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcTree1.FilterCollection

For Each filter In filterCltn
   ListBox1.Items.Add(filter.Name)
Next
```

Code-Beispiel C#

```
VcFilterCollection filterCltn = vcTree1.FilterCollection;

foreach (VcFilter filter in filterCltn)
   {
   ListBox.Items.Add(filter.Name);
   }
```

Specification

Nur-Lese-Eigenschaft von VcFilter

Mit dieser Eigenschaft können Sie die Spezifikation dieses Filters auslesen. Die Spezifikation ist ein String, der nur lesbare ASCII-Zeichen im Bereich 32 bis 127 enthält und somit problemlos in Textdateien oder Datenbanken gespeichert werden kann. Dies ermöglicht Persistenz. Eine solche Spezifikation kann später zur Erzeugung eines Filters mit der Methode **VcFilterCollection.AddBySpecification** benutzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Filterspezifikation

Code-Beispiel VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcTree1.FilterCollection
filter = filterCltn.FirstFilter
MsgBox(filter.Specification)
```

Code-Beispiel C#

```
VcFilterCollection boxCltn = vcTree1.FilterCollection;
VcFilter filter = filterCltn.FirstFilter();
MessageBox.Show(filter.Specification);
```

StringsCaseSensitive

Eigenschaft von VcFilter

Diese Eigenschaft entscheidet, ob bei String-Filterbedingungen der Vergleich mit Unterscheidung von Groß- und Kleinschreibung stattfindet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Vergleich mit Unterscheidung von Groß- und Kleinschreibung findet statt (True)/findet nicht statt (False)

SubCondition

Nur-Lese-Eigenschaft von VcFilter

Mit dieser Eigenschaft können Sie auf ein VcFilterSubCondition-Objekt per Index zugreifen.

Die Eigenschaft SubCondition ist eine indizierte Eigenschaft, die in C# über die Methode get_SubCondition (index) angesprochen wird.

	Datentyp	Beschreibung
Parameter: ⇨ index	System.Int16	Index der Filterbedingung {0 ... VcFilter.SubConditionCount-1}
Eigenschaftswert	VcFilterSubCondition	Filterbedingungsobjekt

SubConditionCount

Nur-Lese-Eigenschaft von VcFilter

Mit dieser Eigenschaft können Sie die Anzahl der Filterbedingungen erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Anzahl der Filterbedingungen

Methoden

AddSubCondition

Methode von VcFilter

Mit dieser Methode können Sie eine neue Filterbedingung an der angegebenen Stelle in der Collection der bestehenden Filterbedingungen erzeugen.

Das entsprechende VcFilterSubCondition-Objekt wird zurückgegeben. Die Eigenschaften dieses Objekt sind standardmäßig folgendermaßen gesetzt:

- DataFieldIndex: -1
- Operator: vcInvalidOp
- ComparisonValueAsString: "<INVALID>"
- ConnectionOperator: vcInvalidConnOp.

	Datentyp	Beschreibung
Parameter:		
⇨ atIndex	System.Int16	Index der neuen Filterbedingung {0 to VcFilter.SubConditionCount and –1 for "at the end of the Collection" (identical with the value VcFilter.SubConditionCount)}
Rückgabewert	VcFilterSubCondition	Filterbedingungsobjekt

CopySubCondition

Methode von VcFilter

Mit dieser Methode können Sie eine Filterbedingung mit Hilfe der Indexangabe kopieren. Die neue Filterbedingung wird an der angegebenen Stelle in der Collection der bestehenden Filterbedingungen eingefügt und als VcFilterSubCondition-Objekt zurückgegeben.

	Datentyp	Beschreibung
Parameter:		
⇨ fromIndex	System.Int16	Index der zu kopierenden Filterbedingung
⇨ atIndex	System.Int16	Index der neuen Filterbedingung {0 to VcFilter.SubConditionCount and –1 for "at the end of the Collection" (identical with the value VcFilter.SubConditionCount)}
Rückgabewert	VcFilterSubCondition	Filterbedingungsobjekt

Evaluate

Methode von VcFilter

Mit dieser Methode kann für einen bestimmten Datensatz geprüft werden, ob der gesetzte Filter zutrifft oder nicht. Es sollten sinnvollerweise nur Objekte übergeben werden, die intern mit Datensätzen der Datentabellen verbunden sind. Dies sind: **VcNode, VcLink, VcGroup, VcDataRecord**. Wird ein hier nicht aufgeführter Objekttyp übergeben, wird eine Exception ausgelöst.

	Datentyp	Beschreibung
Parameter:		
⇨ dataObjectParam	Variant	Datensatzobjekt
Rückgabewert	Boolean	Filter trifft für Datensatz zu (True)/nicht zu (False)

GetEnumerator

Methode von VcFilter

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Bedingungs-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim filter As VcFilter
Dim filterCond As VcFilterSubCondition

filter = VcTree1.FilterCollection.FirstFilter

For Each filterCond In filter
   Debug.Write(filterCond.Index)
Next
```

Code-Beispiel C#

```
VcFilter filter = vcTree1.FilterCollection.FirstFilter();
foreach(VcFilterSubCondition filterCond in filter)
   {
   Console.Write(filterCond.Index);
    }
```

IsValid

Methode von VcFilter

Diese Methode prüft, ob alle Filterbedingungen korrekt formuliert sind. Nur wenn das der Fall ist, werden geänderte Filterbedingungen überhaupt wirksam. Andernfalls bleiben die bisherigen Filterbedingungen wirksam.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Filterbedingungen korrekt (True)/ nicht korrekt (False)

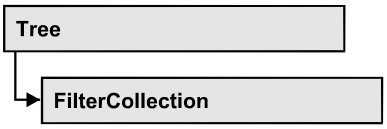
RemoveSubCondition

Methode von VcFilter

Mit dieser Methode können Sie eine Filterbedingung mit Hilfe der Indexangabe löschen.

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index der zu löschenden Filterbedingung

7.16 VcFilterCollection



In einem Objekt vom Typ VcFilterCollection sind automatisch alle verfügbaren Filter zusammengefasst. Über **For Each filter In Filter-Collection** oder die Methoden **First...** und **Next...** können Sie in einer Schleife auf alle Filter zugreifen. Sie haben Zugriff auf bestimmte Objekte über die Eigenschaften **FilterByName** und **FilterByIndex**. Die Anzahl der im Auflistungsobjekt vorhandenen Filter kann über die Eigenschaft **Count** erfragt werden. Die Methoden **Add**, **Copy** und **Remove** ermöglichen das Hinzufügen, Kopieren und Löschen von Filtern.

Eigenschaften

- Count
- MarkedNodesFilter

Methoden

- Add
- AddBySpecification
- Copy
- FilterByIndex
- FilterByName
- FirstFilter
- GetEnumerator
- NextFilter
- Remove

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcFilterCollection

Mit dieser Eigenschaft können Sie die Anzahl der Filterobjekte in der Filter-Auflistung abfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Filter

Code-Beispiel VB.NET

```
Dim filterCltn As VcFilterCollection
Dim numberOfFilters As Integer

filterCltn = VcTree1.FilterCollection
numberOfFilters = filterCltn.Count
```

Code-Beispiel C#

```
VcFilterCollection filterCltn = vcTree1.FilterCollection;
int numberOfFilters = filterCltn.Count;
```

MarkedNodesFilter

Nur-Lese-Eigenschaft von VcFilterCollection

Mit dieser Eigenschaft können Sie einen konstanten Pseudo-Filter holen, der nur bei **ActiveNodeFilter** eingesetzt werden kann und dort das Filtern auf die gerade markierten Knoten auslöst (Teildiagramm).

	Datentyp	Beschreibung
Eigenschaftswert	VcFilter	Pseudo-Filter

Code-Beispiel VB.NET

```
VcTree1.ActiveNodeFilter = VcTree1.FilterCollection.MarkedNodesFilter
```

Code-Beispiel C#

```
vcTree1.ActiveNodeFilter = vcTree1.FilterCollection.MarkedNodesFilter;
```

Methoden

Add

Methode von VcFilterCollection

Mit dieser Methode können Sie einen neuen Filter in der FilterCollection anlegen. Wenn der Name noch nicht verwendet wird, dann wird das neue Filterobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ newName	System.String	Name des Filters
Rückgabewert	VcFilter	Neues Filterobjekt

Code-Beispiel VB.NET

```
newFilter = VcTree1.FilterCollection.Add("foo")
```

Code-Beispiel C#

```
newFilter = vcTree1.FilterCollection.Add("foo");
```

AddBySpecification

Methode von VcFilterCollection

Mit dieser Methode können Sie einen Filter über eine Filter-Spezifikation erzeugen. Dies dient der Persistenz von Filterobjekten. Die Spezifikation eines Filters kann erfragt (siehe VcFilter-Eigenschaft **Specification**) und gespeichert werden. Bei einer neuen Sitzung kann der gleiche Filter mit der wieder eingelesenen Spezifikation samt des gespeicherten Namens wieder erzeugt werden.

	Datentyp	Beschreibung
Parameter:		
⇨ filterSpecification	System.String	Filterspezifikation
Rückgabewert	VcFilter	Neues Filterobjekt

Copy

Methode von VcFilterCollection

Mit dieser Methode können Sie einen Filter kopieren. Wenn der Filter mit dem angegebenen Namen existiert und der Name des neuen Filters noch nicht verwendet wird, wird das neue Filterobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ fromName	System.String	Name des zu kopierenden Filters
⇨ newName	System.String	Name des neuen Filters
Rückgabewert	VcFilter	Filterobjekt

FilterByIndex

Methode von VcFilterCollection

Mit dieser Methode können Sie auf einen einzelnen Filter über ihren Index zugreifen. Existiert kein Filter an dem angegebenen Index, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Filters
Rückgabewert	VcFilter	Ermitteltes Filterobjekt

FilterByName

Methode von VcFilterCollection

Mit dieser Methode können Sie unter Verwendung des Namens auf einen bestimmten Filter zugreifen. Existiert kein Filter unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ filterName	System.String	Name des Filters
Rückgabewert	VcFilter	Filter

Code-Beispiel VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcTree1.FilterCollection
filter = filterCltn.FilterByName("Department A")
```

Code-Beispiel C#

```
VcFilterCollection filterCltn = vcTree1.FilterCollection;
VcFilter filter = filterCltn.FilterByName("Department A");
```

FirstFilter

Methode von VcFilterCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. den ersten Filter des FilterCollection-Objekts zugreifen, um anschließend in einer Schleife mit der Methode **NextFilter** über die nachfolgenden Filter zu iterieren. Existiert kein Filter im FilterCollection-Objekt, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcFilter	Erster Filter

Code-Beispiel VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcTree1.FilterCollection
filter = filtercltn.FirstFilter
```

Code-Beispiel C#

```
VcFilterCollection filterCltn = vcTree1.FilterCollection;
VcFilter filter = filterCltn.FirstFilter();
```

GetEnumerator

Methode von VcFilterCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt verwendet. Mit diesem Objekt können Sie über alle enthaltenen Filter-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim filter As VcFilter
Dim filterCond As VcFilterSubCondition

filter = VcTree1.FilterCollection.FirstFilter

For Each filterCond In filter
   Debug.Write(filterCond.FilterName)
Next
```

Code-Beispiel C#

```
VcFilter filter = vcTree1.FilterCollection.FirstFilter();
foreach(VcFilterSubCondition filterCond in filter)
   {
   Console.Write(filterCond.FilterName);
    }
```

NextFilter

Methode von VcFilterCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Filter des FilterCollection-Objekts zugreifen, nachdem Sie mit der Methode **FirstFilter** den Initialwert erfasst haben. Sind alle Filter durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcFilter	Nächster Filter

Code-Beispiel VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcTree1.FilterCollection
filter = filtercltn.FirstFilter

While Not filter Is Nothing
   ListBox1.Items.Add(filter.Name)
   filter = filterCltn.NextFilter
End While
```

Code-Beispiel C#

```
VcFilterCollection filterCltn = vcTree1.FilterCollection;
VcFilter filter = filterCltn.FirstFilter();

while (filter != null)
   {
   ListBox.Items.Add(filter.Name);
   filter = filterCltn.NextFilter();
   }
```

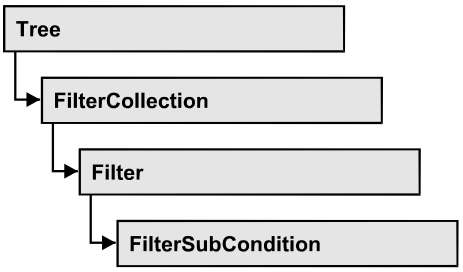
Remove

Methode von VcFilterCollection

Mit dieser Methode können Sie einen Filter löschen. Wenn der Filter noch in irgendeinem anderen Objekt benutzt wird, kann er nicht gelöscht werden. In diesem Fall wird False zurückgegeben, sonst True.

	Datentyp	Beschreibung
Parameter:		
⇨ name	System.String	Name des Filters
Rückgabewert	System.Boolean	Filter gelöscht (True)/nicht gelöscht (False)

7.17 VcFilterSubCondition



Ein Objekt vom Typ VcFilterSubCondition enthält eine einzelne Filterbedingung. Eine Filterbedingung hat im Gegensatz zu vielen anderen Objekten keinen Namen, sondern nur einen Index, der ihre Position im Filter bestimmt.

Im Dialog **Filter bearbeiten** gibt es für jede Filterbedingung eine eigene Zeile. Die dort zur Designzeit dargestellten Eigenschaften sind mit der API hier zur Laufzeit nachträglich veränderbar.

Eigenschaften

- ComparisonValueAsString
- ConnectionOperator
- DataFieldIndex
- FilterName
- Index
- Operator

Methoden

- IsValid

Eigenschaften

ComparisonValueAsString

Eigenschaft von VcFilterSubCondition

Mit dieser Eigenschaft können Sie den Vergleichswert erfragen oder setzen. Dieser String muss einem bestimmten Format entsprechen:

- String: wird in doppelte Anführungszeichen eingeschlossen. Beispiel in VB: """Aachen"""; Beispiel in C/C++: "\"Aachen\""

- Datum: wird in #-Zeichen eingeschlossen. Beispiel: "#18.06.2015; 12:34:56;#" (das Datumsformat ist immer "DD.MM.YYYY;hh:mm:ss;", da es sich hierbei um das interne Standardformat, unabhängig vom Betriebssystem und dessen lokalen Einstellungen, handelt). Ein spezieller Datums-Vergleichswert ist "<TODAY>".

- Datenfeld: wird in eckige Klammern eingeschlossen. Beispiel: "[ID]"

- Zahl: wird direkt angegeben. Beispiel: "52076"

- Liste: bei einem der vc...In-Operatoren: wird in geschweifte Klammern eingeschlossen. Die enthaltenen Werte müssen dann alle vom gleichen Typ (String, Datum oder Zahl) sein und können alle obigen Formate besitzen. Beispiel: "{"NETRONIC", [Name]}"

- Ungültig (z. B. nach Neuerzeugen einer Unterbedingung): "<INVALID>"

Der Typ des Vergleichswerts muss dem Datenfeldtyp und dem Typ des Operators entsprechen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Vergleichswert

ConnectionOperator

Eigenschaft von VcFilterSubCondition

Mit dieser Eigenschaft können Sie den Operator für die Verknüpfung mit der folgenden Unterbedingung erfragen oder setzen. Dabei bindet **vcAnd** stärker als **vcOr**.

	Datentyp	Beschreibung
Eigenschaftswert	VcConnectionOperator	Operator für die Verknüpfung mit der folgenden Filterbedingung
	Mögliche Werte: .vcAnd 1 .vcInvalidConnOp 0 .vcOr 2	 Und-Operator ungültiger Operator Oder-Operator

DataFieldIndex

Eigenschaft von VcFilterSubCondition

Mit dieser Eigenschaft können Sie den Index des Datenfeldes, dessen Inhalt verglichen werden soll, erfragen oder setzen. Der Datenfeldtyp muss dem Typ des Vergleichswerts und des Operators entsprechen.

Sonderwert: -1: kein Datenfeld (ungültig)

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Index des Datenfeldes, dessen Inhalt verglichen werden soll

FilterName

Nur-Lese-Eigenschaft von VcFilterSubCondition

Mit dieser Eigenschaft können Sie den Namen des Filters erfragen, zu dem diese Filterbedingung gehört.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Filters

Index

Nur-Lese-Eigenschaft von VcFilterSubCondition

Mit dieser Eigenschaft können Sie den Index dieser Filterbedingung im zugehörigen Filter erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Index der Filterbedingung im zugehörigen Filter

Operator

Eigenschaft von VcFilterSubCondition

Mit dieser Eigenschaft können Sie den Operator für den Vergleich erfragen oder setzen. Die über die API verfügbaren Operatoren entsprechen den Operatoren im Dialog **Filter bearbeiten**. Der Typ des Operators muss dem Datenfeldtyp des Vergleichswerts entsprechen.

	Datentyp	Beschreibung
Eigenschaftswert	VcOperator **Mögliche Werte:** .vcDateEarlier 27 .vcDateEarlierOrEqual 28 .vcDateEqual 25 .vcDateIn 31 .vcDateLater 29 .vcDateLaterOrEqual 30 .vcDateNotEqual 26 .vcDateNotIn 32 .vcIntEqual 9 .vcIntGreater 13 .vcIntGreaterOrEqual 14 .vcIntIn 15 .vcIntLess 11 .vcIntLessOrEqual 12 .vcIntNotEqual 10 .vcIntNotIn 16 .vcInvalidOp 0 .vcStringBeginsWith 3 .vcStringContains 5 .vcStringEqual 1 .vcStringIn 7 .vcStringNotBeginsWith 4 .vcStringNotContains 6 .vcStringNotEqual 2 .vcStringNotIn 8	Vergleichsoperator Datum früher als Datum früher als oder gleich Datum gleich Datum in Datum später als Datum später als oder gleich Datum ungleich Datum nicht in Integer gleich Integer größer Integer größer oder gleich Integer in Integer kleiner als Integer kleiner als oder größer Integer ungleich Integer nicht in ungültiger Operator String beginnt mit String enthält String gleich String enthält String beginnt nicht mit String enthält nicht String nicht gleich String nicht enthalten in

Methoden

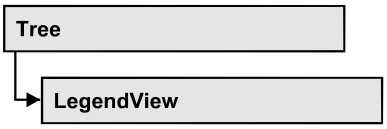
IsValid

Methode von VcFilterSubCondition

Diese Methode prüft, ob die Filterbedingung korrekt formuliert ist.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Filterbedingung korrekt (True)/ nicht korrekt (False)

7.18 VcLegendView



Ein Objekt vom Typ **VcLegendView** bezeichnet das Legendenansicht-Fenster.

Eigenschaften

- Border
- Height
- HeightActualValue
- Left
- LeftActualValue
- ScrollBarMode
- Top
- TopActualValue
- Visible
- Width
- WidthActualValue
- WindowMode

Methoden

- Update

Eigenschaften

Border

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann gesetzt oder erfragt werden, ob die Legendenansicht einen Rahmen besitzt (nicht im Modus **vcPopupWindow**). Die Rahmenfarbe ist **Color.Black**. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Rahmen um die Legendenansicht (True)/kein Rahmen um die Legendenansicht (False) **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.LegendView.Mode = VcLegendViewMode.vcNotFixed
VcTree1.LegendView.Border = True
```

Code-Beispiel C#

```
vcTree1.LegendView.Mode = VcLegendViewMode.vcNotFixed;
vcTree1.LegendView.Border = true;
```

Height

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die vertikale Ausdehnung der Legendenansicht erfragt werden. In den Modi **vcFixedAtTop**, **vcFixedAtBottom**, **vcNotFixed** und **vcPopupWindow** kann sie außerdem gesetzt werden.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Höhe der Legendenansicht **Standardwert:** 100

Code-Beispiel VB.NET

```
VcTree1.LegendView.Height = 100
```

Code-Beispiel C#

```
vcTree1.LegendView.Height = 100;
```

HeightActualValue

Nur-Lese-Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die tatsächlich dargestellte vertikale Ausdehnung der Legendenansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche Höhe der Legendenansicht {0, ...}

Code-Beispiel VB.NET

```
VcTree1.LegendView.Height = 300
```

Code-Beispiel C#

```
vcTree1.LegendView.Height = 100;
```

Left

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die linke Position der Legendenansicht erfragt werden. In den Modi **vcNotFixed** und **vcPopupWindow** kann sie außerdem gesetzt werden.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Linke Position der Legendenansicht **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.LegendView.Left = 200
```

Code-Beispiel C#

```
vcTree1.LegendView.Left = 200;
```

LeftActualValue

Nur-Lese-Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die tatsächlich dargestellte linke Position der Legendenansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche linke Position der Legendenansicht {0, ...}

Code-Beispiel VB.NET

```
VcTree1.LegendView.LeftActualValue = 150
```

Code-Beispiel C#

```
vcTree1.LegendView.LeftActualValue = 150;
```

ScrollBarMode

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann der Scrollbarmodus der Legendenansicht erfragt oder gesetzt werden. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	VcLegendViewScrollBarMode	Scrollbarmodus **Standardwert:** NoScrollBar
	Mögliche Werte:	
	.vcAutomaticScrollBar 3	Anzeige einer horizontalen oder vertikalen Bildlaufleiste, wenn nötig.
	.vcHorizontalScrollBar 1	Anzeige einer horizontalen Bildlaufleiste, wenn nötig.
	.vcNoScrollBar 0	Es wird immer das vollständige Diagramm ohne Bildlaufleisten angezeigt.
	.vcVerticalScrollBar 2	Anzeige einer vertikalen Bildlaufleiste, wenn nötig.

Code-Beispiel VB.NET

```
VcTree1.LegendView.ScrollBarMode = vcAutomaticScrollbar
```

Code-Beispiel C#

```
vcTree1.LegendView.ScrollBarMode = vcAutomaticScrollBar;
```

Top

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die obere Position der Legendenansicht erfragt werden. In den Modi **vcNotFixed** und **vcPopupWindow** kann sie außerdem gesetzt werden.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Obere Position der Legendenansicht **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.LegendView.Top = 20
```

Code-Beispiel C#

```
vcTree1.LegendView.Top = 20;
```

TopActualValue

Nur-Lese-Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die tatsächlich dargestellte obere Position der Legendenansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche obere Position der Legendenansicht {0, ...}

Code-Beispiel VB.NET

```
VcTree1.LegendView.TopActualValue = 40
```

Code-Beispiel C#

```
vcTree1.LegendView.TopActualValue = 40;
```

Visible

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann festgelegt oder erfragt werden, ob die Legendenansicht sichtbar ist. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Legendenansicht sichtbar (True)/unsichtbar (False) **Standardwert:** False

Code-Beispiel VB.NET

```
VcTree1.LegendView.Visible = True
```

Code-Beispiel C#

```
vcTree1.LegendView.Visible = true;
```

Width

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die horizontale Ausdehnung der Legendenansicht erfragt werden. In den Modi **vcFixedAtLeft**, **vcFixedAt-Right**, **vcNotFixed** und **vcPopupWindow** kann diese Eigenschaft außerdem gesetzt werden.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Horizontale Ausdehnung der Legendenansicht **Standardwert:** 100

Code-Beispiel VB.NET

```
VcTree1.LegendView.Width = 200
```

Code-Beispiel C#

```
vcTree1.LegendView.Width = 200;
```

WidthActualValue

Nur-Lese-Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann die tatsächlich dargestellte horizontale Ausdehnung der Legendenansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche horizontale Ausdehnung der Legendenansicht {0, ...}

Code-Beispiel VB.NET

```
VcTree1.LegendView.WidthActualValue = 600
```

Code-Beispiel C#

```
vcTree1.LegendView.WidthActualValue = 600;
```

WindowMode

Eigenschaft von VcLegendView

Mit dieser Eigenschaft kann der Modus der Legendenansicht erfragt oder gesetzt werden. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	VcLegendViewWindowMode	Modus der Gesamtansicht **Standardwert:** vcPopupWindow
	Mögliche Werte:	
	.vcFixedAtBottom 4	Die Legendenansicht wird unten im Fenster des VARCHART .NET Steuerelements angezeigt. Dann kann nur die Höhe festgelegt werden, während Position und Breite vorgegeben sind.
	.vcFixedAtLeft 1	Die Legendenansicht wird links im Fenster des VARCHART .NET Steuerelements angezeigt. Dann kann nur die Breite festgelegt werden, während Position und Höhe vorgegeben sind.
	.vcFixedAtRight 2	Die Legendenansicht wird rechts im Fenster des VARCHART .NET Steuerelements angezeigt. Dann kann nur die Breite festgelegt werden, während Position und Höhe vorgegeben sind.
	.vcFixedAtTop 3	Die Legendenansicht wird oben im Fenster des VARCHART .NET Steuerelements angezeigt. Dann kann nur die Höhe festgelegt werden, während Position und Breite vorgegeben sind.
	.vcNotFixed 5	Die Legendenansicht ist ein untergeordnetes Kindfenster des aktuellen Vaterfensters des VARCHART .NET Steuerelements und kann an beliebiger Position mit beliebiger Ausdehnung angeordnet werden. Das Vaterfenster kann bei Bedarf über die Eigenschaft **VcLegendView.ParentHWnd** geändert werden.
	.vcPopupWindow 6	Die Legendenansicht ist ein Popup-Fenster, das einen eigenen Rahmen besitzt und vom Benutzer in Position und Größe verändert werden kann. Es kann über das Standard-Kontextmenü ein- bzw. ausgeschaltet oder über die **Schließen**-Schaltfläche in der Titelleiste ausgeschaltet werden.

Code-Beispiel VB.NET

```
VcTree1.LegendView.Mode = VcLegendViewMode.vcNotFixed
```

Code-Beispiel C#

```
vcTree1.LegendView.Mode = VcLegendViewMode.vcNotFixed;
```

Methoden

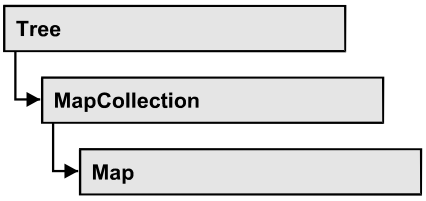
Update

Methode von VcLegendView

Mit dieser Methode wird die Legende aktualisiert.

	Datentyp	Beschreibung

7.19 VcMap



Eine Zuordnungstabelle (Map) legt über Datenfeldeinträge bestimmte Eigenschaften von Knoten, beispielsweise die Hintergrundfarbe, datenfeldabhängig fest.

Sie können in einer Zuordnungstabelle maximal 150 Zuordnungen festlegen. Über **For Each mapEntry In Map** können Sie in einer Schleife auf alle Verbindungen zugreifen.

Eigenschaften

- ConsiderFilterEntries
- Count
- Name
- Specification
- Type

Methoden

- CreateEntry
- DeleteEntry
- FirstMapEntry
- GetMapEntry
- NextMapEntry

Eigenschaften

ConsiderFilterEntries

Nur-Lese-Eigenschaft von VcMap

Mit dieser Eigenschaft können Sie setzen oder erfragen, ob bei der Zuordnung von Datenfeldeinträgen zu einer Zuordnungstabelle Filter berücksichtigt werden, um so auch Wertebereiche als Schlüsselwerte angeben zu können.

	Datentyp	Beschreibung

Count

Nur-Lese-Eigenschaft von VcMap

Mit dieser Eigenschaft kann die Anzahl der Einträge in der Zuordnungstabelle (Map) abgefragt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Mapeinträge

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim numberOfEntries As Integer

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
numberOfEntries = map.Count
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
int numberOfEntries = map.Count;
```

Name

Nur-Lese-Eigenschaft von VcMap

Mit dieser Eigenschaft können Sie den Namen der Zuordnungstabelle (Map) abfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Zuordnungstabelle

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapName As String

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
mapName = map.Name
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
string mapName = map.Name;
```

Specification

Nur-Lese-Eigenschaft von VcMap

Mit dieser Eigenschaft können Sie die Spezifikation dieser Zuordnungstabelle auslesen. Die Spezifikation ist ein String, der nur lesbare ASCII-Zeichen im Bereich 32 bis 127 enthält und somit problemlos in Textdateien oder Datenbanken gespeichert werden kann. Dies ermöglicht Persistenz. Eine solche Spezifikation kann später zur Erzeugung einer Zuordnungstabelle mit der Methode **VcMapCollection.AddBySpecification** benutzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Spezifikation der Zuordnungstabelle

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcTree1.BoxCollection
box = boxCltn.FirstBox
MsgBox(box.Specification)
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
VcBox box = boxCltn.FirstBox();
MessageBox.Show(box.Specification);
```

Type

Eigenschaft von VcMap

Mit dieser Eigenschaft können Sie den Typ der Zuordnungstabelle (Map) abfragen oder setzen.

	Datentyp	Beschreibung
Eigenschaftswert	VcMapType	Typ der Zuordnungstabelle
	Mögliche Werte:	
	.vcAnyMap 0	**Beliebig** (nur zum Selektieren verwendet)
	.vcColorMap 1	**Farben**
	.vcFontMap 8	**Schriften**
	.vcGraphicsFileMap 7	**Grafikdatei**
	.vcMillimeterMap 9	**Millimeter**
	.vcNumberMap 10	**Zahlen**
	.vcPatternMap 3	**Schraffuren**

	.vcTextMap 6	**Text**

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
map.Type = VcMapType.vcPatternMap
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
map.Type = VcMapType.vcPatternMap;
```

Methoden

CreateEntry

Methode von VcMap

Mit dieser Methode kann ein neuer Eintrag (= eine neue Zeile) für die Zuordnungstabelle (Map) erzeugt werden. Damit der neue Eintrag in der Zuordnungstabelle wirksam wird, sollte anschließend die Methode **MapCollection.Update()** aufgerufen werden.

	Datentyp	Beschreibung
Rückgabewert	VcMapEntry	Map-Eintrag

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.CreateEntry
mapCltn.Update
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.CreateEntry();
mapCltn.Update;
```

DeleteEntry

Methode von VcMap

Mit dieser Methode kann ein Eintrag (= eine Zeile) der Zuordnungstabelle (Map) gelöscht werden. Damit die Löschung in der Zuordnungstabelle wirksam wird, sollte anschließend die Methode **MapCollection.Update()** aufgerufen werden.

	Datentyp	Beschreibung
Parameter:		
⇨ mapEntry	VcMapEntry	Eintrag der Map
Rückgabewert	System.Boolean	Map-Eintrag erfolgreich/nicht erfolgreich gelöscht

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
map.DeleteEntry(mapEntry)
mapCltn.Update
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
map.DeleteEntry(mapEntry);
mapCltn.Update;
```

FirstMapEntry

Methode von VcMap

Mit dieser Methode können Sie auf den Initialwert, d.h. den ersten Eintrag der Zuordnungstabelle (Map) zugreifen, um anschließend in einer Schleife mit der Methode **NextMapEntry** über die nachfolgenden Einträge zu iterieren. Existiert kein Eintrag im Map-Objekt, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcMapEntry	Erster Map-Eintrag

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)

map = mapCltn.FirstMap
mapEntry = map.FirstMapEntry
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
VcMapEntry mapEntry = map.FirstMapEntry();
```

GetMapEntry

Methode von VcMap

Diese Methode liefert den entsprechenden Eintrag der Zuordnungstabelle (Map) zu dem im Datenfeld angegebenen Wert.

	Datentyp	Beschreibung
Rückgabewert	System.String	Map-Eintrag entsprechend Feldinhalt

NextMapEntry

Methode von VcMap

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Einträge (Zeilen) des Map-Objekts zugreifen, nachdem Sie mit der Methode **FirstMapEntry** den Initialwert erfasst haben. Sind alle Einträge durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcMapEntry	Nachfolgender Map-Eintrag

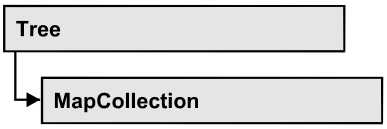
Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
mapEntry = map.FirstMapEntry
While Not mapEntry Is Nothing
   ListBox1.Items.Add(mapEntry.LegendText)
   mapEntry = map.NextMapEntry
End While
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
VcMapEntry mapEntry = map.FirstMapEntry()
while (mapEntry != null)
   {
   listBox1.Items.Add(mapEntry.LegendText);
   mapEntry= map.NextMapEntry();
   }
```

7.20 VcMapCollection



In einem Objekt des Typs VcMapCollection sind die Zuordnungstabellen (Maps) zusammengefasst, die dem Auflistungsobjekt über die Methode **SelectMaps** zugewiesen wurden. Über **For Each map InMapCollection** oder die Methoden **First...** und **Next...** können Sie in einer Schleife auf alle Zuordnungstabellen zugreifen. Sie haben Zugriff auf bestimmte Objekte über die Eigenschaften **MapByName** und **MapByIndex**. Die Anzahl der im Auflistungsobjekt vorhandenen Zuordnungstabellen kann über die Eigenschaft **Count** erfragt werden. Die Methoden **Add**, **Copy** und **Remove** ermöglichen das Hinzufügen, Kopieren und Löschen von Zuordnungstabellen.

Eigenschaften

- Count

Methoden

- Add
- AddBySpecification
- Copy
- FirstMap
- GetEnumerator
- MapByIndex
- MapByName
- NextMap
- Remove
- SelectMaps
- Update

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcMapCollection

Mit dieser Eigenschaft kann die Anzahl der Zuordnungstabellen (Maps) in der Map-Auflistung abgefragt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Maps

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim numberOfMaps As Integer

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
numberOfMaps = mapCltn.Count
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
int numberOfMaps = mapCltn.Count;
```

Methoden

Add

Methode von VcMapCollection

Mit dieser Methode können Sie eine neue Zuordnungstabelle in der MapCollection anlegen. Wenn der Name noch nicht verwendet wird, dann wird das neue Zuordnungstabellenobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ mapName	System.String	Name der Zuordnungstabelle
Rückgabewert	VcMap	Neues Zuordnungstabellenobjekt

Code-Beispiel VB.NET

```
newMap = VcTree1.MapCollection.Add("Map1")
```

Code-Beispiel C#

```
VcMap newMap = vcTree1.MapCollection.Add("Map1");
```

AddBySpecification

Methode von VcMapCollection

Mit dieser Methode können Sie eine Zuordnungstabelle über eine Zuordnungstabellen-Spezifikation erzeugen. Dies dient der Persistenz von Zuordnungstabellen-Objekten. Die Spezifikation einer Zuordnungstabelle kann erfragt werden (siehe VcMap-Eigenschaft **Specification**) und gespeichert werden. Bei einer neuen Sitzung kann die gleiche Zuordnungstabelle mit der wieder eingelesenen Spezifikation samt des gespeicherten Namens wieder erzeugt werden.

	Datentyp	Beschreibung
Parameter:		
⇨ specification	System.String	Zuordnungstabellenspezifikation
Rückgabewert	VcMap	Neues Zuordnungstabellenobjekt

Copy

Methode von VcMapCollection

Mit dieser Methode können Sie eine Zuordnungstabelle kopieren. Wenn die Zuordnungstabelle mit dem angegebenen Namen existiert und der Name der neuen Zuordnungstabelle noch nicht verwendet wird, wird das neue Zuordnungstabellenobjekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ mapName	System.String	Name der zu kopierenden Zuordnungstabelle
⇨ newMapName	System.String	Name der neuen Zuordnungstabelle
Rückgabewert	VcMap	Zuordnungstabellenobjekt

FirstMap

Methode von VcMapCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. die erste Zuordnungstabelle (Map) des MapCollection-Objekts zugreifen, um anschließend in einer Schleife mit der Methode **NextMap** über die nachfolgenden Maps zu iterieren. Existiert keine Zuordnungstabelle (Map) in der MapCollection, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**). Zuvor muss mit der Methode **SelectMaps** die gewünschte Map-Auswahl getroffen worden sein.

	Datentyp	Beschreibung
Rückgabewert	VcMap	Erste Map

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
```

GetEnumerator

Methode von VcMapCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Mit diesem Objekt können Sie über alle enthaltenen Zuordnungstabellen iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

MapByIndex

Methode von VcMapCollection

Mit dieser Methode können Sie auf eine einzelne Zuordnungstabelle über ihren Index zugreifen. Existiert keine Zuordnungstabelle an dem

angegebenen Index, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index der Zuordnungstabelle
Rückgabewert	VcMap	Ermitteltes Zuordnungstabellenobjekt

MapByName

Methode von VcMapCollection

Mit dieser Methode können Sie unter Verwendung des Namens der Zuordnungstabelle (Map) auf eine bestimmte Zuordnungstabelle zugreifen. Zuvor muss mit der Methode **SelectMaps** die gewünschte Auswahl der Zuordnungstabelle getroffen worden sein. Existiert kein Map-Objekt unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ mapName	System.String	Name der Map
Rückgabewert	VcMap	Map

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
```

NextMap

Methode von VcMapCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Zuordnungstabellen (Maps) des MapCollection-Objekts zugreifen, nachdem

Sie mit der Methode **FirstMap** den Initialwert erfasst haben. Sind alle Maps durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcMap	Nachfolgende Map

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
While Not map Is Nothing
   ListBox1.Items.Add(map.Name)
    map = mapCltn.NextMap
End While
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
while (map != null)
   {
   listBox1.Items.Add(map.Name);
   map = mapCltn.NextMap();
   }
```

Remove

Methode von VcMapCollection

Mit dieser Methode können Sie eine Zuordnungstabelle löschen. Wenn die Zuordnungstabelle noch in irgendeinem anderen Objekt benutzt wird, kann sie nicht gelöscht werden. In diesem Fall wird False zurückgegeben, sonst True.

	Datentyp	Beschreibung
Parameter:		
⇨ mapName	System.String	Name der Zuordnungstabelle
Rückgabewert	System.Boolean	Zuordnungstabelle gelöscht (True)/nicht gelöscht (False)

SelectMaps

Methode von VcMapCollection

Mit dieser Methode können Sie steuern, welche Typen von Zuordnungstabellen (Maps) in Ihr MapCollection-Objekt aufgenommen werden.

	Datentyp	Beschreibung
Parameter:		
⇨ selectionType	VcMapType	Auszuwählender Map-Typ
	Mögliche Werte:	
	.vcAnyMap 0	**Beliebig** (nur zum Selektieren verwendet)
	.vcColorMap 1	**Farben**
	.vcFontMap 8	**Schriften**
	.vcGraphicsFileMap 7	**Grafikdatei**
	.vcMillimeterMap 9	**Millimeter**
	.vcNumberMap 10	**Zahlen**
	.vcPatternMap 3	**Schraffuren**
	.vcTextMap 6	**Text**
Rückgabewert	System.Int32	Anzahl der Ausgewählten Maps

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
```

Update

Methode von VcMapCollection

Mit dieser Methode können Sie die Darstellung aller Objekte, die durch die verwendeten Zuordnungstabellen (Maps) bestimmt werden, aktualisieren. Wenn Sie diese Methode nicht aufrufen, werden die Änderungen der Zuordnungstabellen (Maps) zur Laufzeit nicht ausgeführt. Sie sollten diese Methode erst am Ende des Codes zur Festlegung der Zuordnungstabellen und des MapCollection-Objekts verwenden, damit die Aktualisierung nicht schon ausgeführt wird, bevor alle Festlegungen der Zuordnungstabellen ausgeführt worden sind.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Aktualisierung erfolgt (True)/ nicht erfolgt (False)

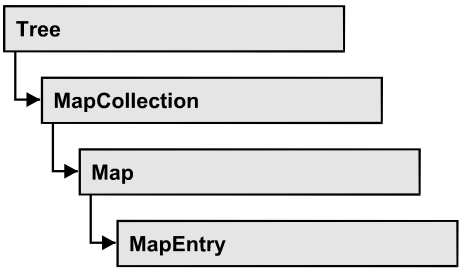
Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
While Not mapEntry.DataFieldValue = "A"
   mapEntry = map.NextMapEntry
End While

mapEntry.Color = Color.Blue
mapCltn.Update()
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
while (mapEntry.DataFieldValue != "A")
   mapEntry = map.NextMapEntry();

mapEntry.Color = Color.LightSteelBlue;
mapCltn.Update();
```

7.21 VcMapEntry



Ein Objekt vom Typ VcMapEntry ist ein Eintrag einer Zuordnungstabelle (Map) und damit das Element einer Zuordnungstabelle. Ein Zuordnungstabelleneintrag enthält die Kombination aus einem Datenfeldinhalt des Vorgangsdatensatzes sowie bestimmten Attributen (z. B. Farbe, Grafikdatei, Schriftattribute).

Sie können in einer Zuordnungstabelle maximal 150 Zuordnungen festlegen. Falls Sie noch mehr Zuordnungen benötigen, erstellen Sie einfach eine neue Zuordnungstabelle, z. B. als Kopie der bereits vorhandenen.

Eigenschaften

- Color
- DataFieldValue
- FontBody
- FontName
- FontSize
- GraphicsFileName
- Number
- Pattern

Eigenschaften

Color

Eigenschaft von VcMapEntry

Für Farben-Zuordnungstabellen: Mit dieser Eigenschaft können Sie die Farbe für den Zuordnungstabelleneintrag festlegen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von

0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	RGB-Farbwerte ({0...255},{0...255},{0...255})

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim colorOfMapEntry As Color

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcColorMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
colorOfMapEntry = mapEntry.Color
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcColorMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
Color colorOfMapEntry = mapEntry.Color;
```

DataFieldValue

Eigenschaft von VcMapEntry

Mit dieser Eigenschaft können Sie den Inhalt des Datenfeldes des Zuordnungstabelleneintrags erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Inhalt des Datenfelds

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim dataFieldValue As String

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry

dataFieldValue = mapEntry.DataFieldValue
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
string dataFieldValue = mapEntry.DataFieldValue;
```

FontBody

Eigenschaft von VcMapEntry

für Schriften-Zuordnungstabellen: Mit dieser Eigenschaft können Sie den Schriftgrad für den Zuordnungstabelleneintrag erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFontBody	Schriftgrad

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim fontBodyOfMapEntry As VcFontBody

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcFontMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
fontBodyOfMapEntry = VcFontBody.vcBold
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcFontMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
VcFontBody fontBodyOfMapEntry = VcFontBody.vcBold;
```

FontName

Eigenschaft von VcMapEntry

für Schriften-Zuordnungstabellen: Mit dieser Eigenschaft können Sie die Schriftart für den Zuordnungstabelleneintrag erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Schriftart

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim fontNameOfMapEntry As String

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcFontMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
fontNameOfMapEntry = "Arial"
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcFontMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
string fontNameOfMapEntry = "Arial";
```

FontSize

Eigenschaft von VcMapEntry

für Schriften-Zuordnungstabellen: Mit dieser Eigenschaft können Sie die Schriftgröße für den Zuordnungstabelleneintrag erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Schriftgröße

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim fontSizeOfMapEntry As Integer

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcFontMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
fontSizeOfMapEntry = 14
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcFontMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
int fontSizeOfMapEntry = 14;
```

GraphicsFileName

Eigenschaft von VcMapEntry

Für Grafikdateien-Zuordnungstabellen: Mit dieser Eigenschaft können Sie den Namen der Grafikdatei des Zuordnungstabelleneintrags erfragen oder festlegen. Mögliche Formate:

*.EMF (Enhanced Metafile oder Enhanced Metafile Plus)

*.GIF (Graphics Interchange Format)

*.JPG (Joint Photographic Experts Group)

*.PNG (Portable Network Graphics)

*.TIF (Tagged Image File Format)

*.VMF (Viewer Metafile)

*.WMF (Microsoft Windows Metafile, ggf. mit eingebautem EMF

Nur EMF, EMF+, VMF und WMF sind Vektorformate, in denen das Diagramm auflösungsunabhängig gespeichert werden kann. Die übrigen Formate sind pixelorientiert und bieten damit nicht beliebige Auflösungen.

Das VMF-Format wird in der Zukunft nicht mehr weiterentwickelt, aus Kompatibilitätsgründen für bestehende Anwendungen aber zunächst noch weiter unterstützt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Grafikdatei

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim exeName As String
Dim exeDir As String

mapCltn = VcTree1.MapCollection
mapCltn.SelectMaps(VcMapType.vcGraphicsFileMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry

exeName = System.Environment.GetCommandLineArgs(0)
exeDir = System.IO.Path.GetDirectoryName(exeName)
mapEntry.GraphicsFileName = exeDir + "\Bitmaps\picture1.bmp"
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcGraphicsFileMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();

String exeName = Environment.GetCommandLineArgs()[0];
mapEntry.GraphicsFileName = System.IO.Path.GetDirectoryName(exeName)+
@"\..\Bitmaps\picture1.bmp";
```

Number

Eigenschaft von VcMapEntry

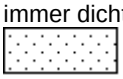
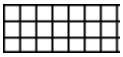
Für numerische Zuordnungstabellen: Mit dieser Eigenschaft können Sie eine Zahl als Zuordnungstabelleneintrag festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Numerischer Wert

Pattern

Eigenschaft von VcMapEntry

Für Schraffuren-Zuordnungstabellen (vcPatternMap): mit dieser Eigenschaft können Sie den Schraffurtyp des Zuordnungstabelleneintrags erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFillPattern	Mustertyp
	Mögliche Werte: .vc05PercentPattern... vc90PercentPattern 01 - 11	Punkte in Vordergrundfarbe auf Hintergrundfarbe; mit steigender Prozentzahl Vordergrundfarbe immer dichter 
	.vcAeroGlassPattern 44	Vertikaler Farbverlauf in der Füllmusterfarbe
	.vcBDiagonalPattern 5	Diagonale Linien von links unten nach rechts oben
	.vcCrossPattern 6	Kreuzschraffur 

.vcDarkDownwardDiagonalPattern 2014	Diagonale Linien von links oben nach rechts unten, 50 % näher zusammen als vcFDiagonalPattern und mit doppelter Liniendicke
.vcDarkHorizontalPattern 2023	Horizontale Linien mit 50% geringerem Abstand als vcHorizontalPattern und doppelter Liniendicke
.vcDarkUpwardDiagonalPattern 2015	Diagonale Linien von links unten nach rechts oben; mit 50%geringerem Abstand als vcBDiagonalPattern und mit doppelter Liniendicke
.vcDarkVerticalPattern 2022	Vertikale Linien mit 50% geringerem Abstand als vcVerticalPattern und doppelter Liniendicke
.vcDashedDownwardDiagonalPattern 2024	Gestrichelte diagonale Linien von links oben nach rechts unten
.vcDashedHorizontalPattern 2026	Horizontale gestrichelte Linien
.vcDashedUpwardDiagonalPattern 2025	Unterbrochene diagonale Linien von links unten nach rechts oben
.vcDashedVerticalPattern 2027	Vertikale gestrichelte Linien
.vcDiagCrossPattern 7	Diagonale Kreuzschraffur, klein
.vcDiagonalBrickPattern 2032	Diagonales Backsteinmuster
.vcDivotPattern 2036	Grassoden-Muster
.vcDottedDiamondPattern 2038	Diagonale Kreuzschraffur aus punktierten Linien
.vcDottedGridPattern 2037	Kreuzschraffur aus punktierten Linien
.vcFDiagonalPattern 4	Diagonale Linien von links oben nach rechts unten
.vcHorizontalBrickPattern 2033	Horizontales Backsteinmuster
.vcHorizontalGradientPattern 52	Horizontaler Farbverlauf
.vcHorizontalPattern 3	Horizontale Linien

.vcLargeCheckerboardPattern 2044	Schachbrettmuster mit doppelt so großen Quadraten wie vcSmall-CheckerBoardPattern
.vcLargeConfettiPattern 2029	Konfetti -Muster, groß
.vcLightDownwardDiagonalPattern 2012	Diagonale Linien von links oben nach rechts unten; mit 50% geringerem Abstand als vcB-DiagonalPattern
.vcLightHorizontalPattern 2019	Horizontale Linien mit 50% geringerem Abstand als bei vc-HorizontalPattern
.vcLightUpwardDiagonalPattern 2013	Diagonale Linien von links unten nach rechts oben; 50 % näher zusammen als vcBDiagonalPattern
.vcLightVerticalPattern 2018	Vertikale Linien mit 50% geringerem Abstand als bei vcVerticalPattern
.vcNarrowHorizontalPattern 2021	Horizontale Linien mit 75% geringerem Abstand als vc-HorizontalPattern
.vcNarrowVerticalPattern 2020	Vertikale Linien mit 75% geringerem Abstand als vcVerticalPattern
.vcNoPattern 1276	Kein Füllmuster
.vcOutlinedDiamondPattern 2045	Diagonale Kreuzschraffur, groß
.vcPlaidPattern 2035	Schottenstoff-Muster
.vcShinglePattern 2039	Diagonales Dachschindel-Muster
.vcSmallCheckerBoardPattern 2043	Schachbrettmuster
.vcSmallConfettiPattern 2028	Konfetti-Muster
.vcSmallGridPattern 2042	Kreuzschraffur mit 50 % geringerem Abstand als vcCrossPattern
.vcSolidDiamondPattern 2046	Schachbrettmuster mit diagonalen Quadraten
.vcSpherePattern 2041	Kugeln schachbrettartig angeordnet
.vcTrellisPattern 2040	Spalier-Muster

.vcVerticalBottomLightedConvexPattern 43	Vertikaler Farbverlauf von dunkel nach hell
.vcVerticalConcavePattern 40	Vertikaler Farbverlauf von dunkel über hell nach dunkel
.vcVerticalConvexPattern 41	Vertikaler Farbverlauf von hell über dunkel nach hell
.vcVerticalGradientPattern 62	Vertikaler Farbverlauf
.vcVerticalPattern 2	Vertikale Linien
.vcVerticalTopLightedConvexPattern 42	Vertikaler Farbverlauf von hell nach dunkel
.vcWavePattern 2031	Horizontales Wellenmuster
.vcWeavePattern 2034	Muster mit verwebten Streifen
.vcWideDownwardDiagonalPattern 2016	Diagonale Linien von links oben nach rechts unten, mit demselben Abstand wie vcFDiagonalPattern, aber mit dreifacher Liniendicke
.vcWideUpwardDiagonalPattern 2017	Diagonale Linien von links oben nach rechts unten, mit demselben Abstand wie vcBDiagonalPattern, aber mit dreifacher Liniendicke
.vcZigZagPattern 2030	Horizontale Zick-Zacklinien

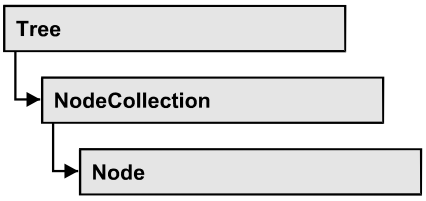
Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim pattern As VcFillPattern

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcPatternMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
pattern = VcFillPattern.vcBDiagonalPattern
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcPatternMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
VcFillPattern pattern = VcFillPattern.vcBDiagonalPattern;
```

7.22 VcNode



Knoten sind die Grundelemente eines Baumdiagramms. Das Aussehen eines Knotens wird über diejenigen NodeAppearance-Objekte bestimmt, deren Filter auf den Knoten zutreffen. Erzeugt werden Knoten über die Methode **VcTree.InsertNodeRecord** oder interaktiv.

Eigenschaften

- AllData
- ChildNodeCollection
- Collapsed
- DataField
- ID
- InCollapsedSubtree
- LeftBrotherNode
- Marked
- ParentNode
- RightBrotherNode
- SubtreeArrangement
- SubtreeNodeCollection

Methoden

- ArrangeSubtree
- CollapseSubtree
- DataRecord
- Delete
- ExpandSubtree
- RelatedDataRecord
- Update

Eigenschaften

AllData

Eigenschaft von VcNode

Mit dieser Eigenschaft können alle Daten auf einmal für den Knoten gesetzt oder erfragt werden. Beim Setzen ist ein CSV-String (Semikolon als Trennzeichen) oder ein Object erlaubt, der in einem Feld (Array) alle Datenfelder des Knotens erhält. Beim Erfragen wird ein String zurückgegeben. (Siehe auch **InsertNodeRecord**.)

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Alle Daten des Datensatzes

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeModifyingEventArgs) Handles VcTree1.VcNodeModifying
   Dim allDataOfNode As String
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse

   allDataOfNode = e.Node.AllData
   MsgBox(allDataOfNode)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeModifying(object sender,
NETRONIC.XGantt.VcNodeModifyingEventArgs e)
   {
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
   string allDataOfNode = e.Node.AllData.ToString();
   MessageBox.Show(allDataOfNode);
   }
```

ChildNodeCollection

Nur-Lese-Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie die direkten Sohnknoten eines Knotens erfragen. Vgl. auch die Eigenschaft **SubtreeNodeCollection**.

	Datentyp	Beschreibung
Eigenschaftswert	VcNodeCollection	NodeCollection-Objekt, enthält Sohnknoten

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   Dim noOfChildren As Integer
   noOfChildren = e.Node.ChildNodeCollection.Count
   MsgBox(noOfChildren)
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
{
   int noOfChildren;
   noOfChildren = e.Node.ChildNodeCollection.Count;
   MessageBox.Show(noOfChildren.ToString());
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

Collapsed

Nur-Lese-Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie erfragen, ob ein Knoten kollabiert ist (True) oder nicht (False).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Knoten kollabiert/expandiert

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   Dim collapseState As Boolean
   collapseState = e.Node.Collapsed
   MsgBox(collapseState)
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
{
   bool collapseState;
   collapseState = e.Node.Collapsed;
   MessageBox.Show(collapseState.ToString());
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

DataField

Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie einem Datenfeld des Knotens einen Wert zuweisen oder einen gesetzten Wert erfragen. Wenn ein Knoten durch diese Methode einen neuen Wert erhalten hat, muss anschließend die Methode **Update** aufgerufen werden.

Die Eigenschaft DataField ist eine indizierte Eigenschaft, die in C# über die beiden Methoden set_DataField (index, pvn) und get_DataField (index) angesprochen wird.

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Datenfeldes
Eigenschaftswert	System.Object	Inhalt des Datenfeldes

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcTree1.VcNodeRightClicking
  If MsgBox("Delete node: " + e.Node.DataField(0), MsgBoxStyle.YesNo, "Delete
node") = MsgBoxResult.Yes Then
     e.Node.Delete()
  End If
     e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
   {
   DialogResult retVal = MessageBox.Show("Delete node: " +
e.Node.get_DataField(0), "Deleting node", MessageBoxButtons.YesNo);
   if (retVal ==  DialogResult.Yes)
      e.Node.Delete();
   else
      e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
   }
```

ID

Nur-Lese-Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie die ID eines Knotens erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Knoten-ID

Code-Beispiel VB.NET

```
VcNode node = VcTree1.NodeCollection.FirstNode()

MsgBox (node.ID)
```

Code-Beispiel C#

```
VcNode node = vcTree1.NodeCollection.FirstNode();

MessageBox.Show(node.ID)
```

InCollapsedSubtree

Nur-Lese-Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie erfragen, ob sich ein Knoten in einem kollabierten Teilbaum befindet (True) und damit unsichtbar ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Knoten liegt/liegt nicht innerhalb eines kollabierten Teilbaumes

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   Dim inCollapsedSubtree As Boolean
   inCollapsedSubtree = e.Node.InCollapsedSubtree
   MsgBox(inCollapsedSubtree)
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
{
   bool inCollapsedSubtree;
   inCollapsedSubtree = e.Node.InCollapsedSubtree;
   MessageBox.Show(inCollapsedSubtree.ToString());
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

LeftBrotherNode

Nur-Lese-Eigenschaft von VcNode

Der linke Bruder des Knotens wird zurückgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	VcNode	Linker Bruderknoten

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   If e.Node.LeftBrotherNode Is Nothing Then
      MsgBox("This node doesn´t have a left brother.")
   Else
      MsgBox(e.Node.LeftBrotherNode.AllData)
   End If

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
{
   if (e.Node.LeftBrotherNode == null)
      MessageBox.Show("This node doesn´t have a left brother.");
   else        MessageBox.Show(e.Node.LeftBrotherNode.AllData.ToString());
}
```

Marked

Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie festlegen oder abfragen, ob ein Knoten markiert ist. Die gesetzte Markierung ist nur dann sichtbar, wenn auf der Eigenschaftenseite **Knoten** unter **Knotenmarkierung** nicht **Ohne** ausgewählt ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Knoten markiert/nicht markiert

Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode
Dim predecessor As VcNode
Dim linkCltn As VcLinkCollection
Dim link As VcLink

nodeCltn = VcTree1.NodeCollection
nodeCltn.SelectNodes(VcSelectionType.vcAll)

For Each node In nodeCltn
   linkCltn = node.IncomingLinks
   For Each link In linkCltn
      predecessor = link.PredecessorNode
      predecessor.Marked = True
   Next
Next
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcAll);
VcNode predecessorNode;
VcLinkCollection linkCltn;
foreach (VcNode node in nodeCltn)
   {
   linkCltn = node.IncomingLinks;
   foreach (VcLink link in linkCltn)
      {
      predecessorNode = link.PredecessorNode;
      predecessorNode.Marked = true;
      }
   }
```

ParentNode

Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie einen Knoten als Sohnknoten unter dem angegebenen Vaterknoten einfügen bzw. den aktuellen Vaterknoten erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcNode	Vaterknoten

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   MsgBox(e.Node.ParentNode.DataField(0))

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
{   MessageBox.Show(e.Node.ParentNode.get_DataField(0).ToString());
}
```

RightBrotherNode

Nur-Lese-Eigenschaft von VcNode

Der rechte Bruder des Knotens wird zurückgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	VcNode	Rechter Bruderknoten

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   If e.Node.RightBrotherNode Is Nothing Then
      MsgBox("This node doesn´t have a left brother.")
   Else
      MsgBox(e.Node.RightBrotherNode.AllData)
   End If

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
      {
         if (e.Node.RightBrotherNode == null)
            MessageBox.Show("This node doesn´t have a left brother.");
         else
            MessageBox.Show(e.Node.RightBrotherNode.AllData.ToString());
      }
```

SubtreeArrangement

Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob der dem Knoten anhängende Teilbaum horizontal oder vertikal angeordnet werden soll.

	Datentyp	Beschreibung
Eigenschaftswert	VcArrangement	Ausrichtung **Standardwert:** vcHorizontal

Code-Beispiel VB.NET

```
VcNode.Arrangement = vcHorizontal
```

Code-Beispiel C#

```
VcNode.Arrangement = vcHorizontal;
```

SubtreeNodeCollection

Nur-Lese-Eigenschaft von VcNode

Mit dieser Eigenschaft können Sie den Teilbaum des Referenzknotens (der Knoten selbst sowie alle direkten und indirekten Sohnknoten des Referenzknotens) erfragen. Siehe auch **ChildNodeCollection**.

	Datentyp	Beschreibung
Eigenschaftswert	VcNodeCollection	Menge der Knoten, die den Teilbaum bilden

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeRightClicking(ByVal sender As System.Object, _
                                ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) _
                                Handles VcTree1.VcNodeRightClicking

         Dim noOfNodes As Integer
         noOfNodes = e.Node.SubtreeNodeCollection.Count
         MsgBox(noOfNodes)
         e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeRightClicking(object sender, VcNodeClickingEventArgs
e)
{
   int noOfNodes = e.Node.SubtreeNodeCollection.Count;
   MessageBox.Show(noOfNodes.ToString());
   e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

Methoden

ArrangeSubtree

Methode von VcNode

Mit dieser Methode können Sie einen Teilbaum horizontal oder vertikal anordnen. Im Unterschied zur Eigenschaft **Arrangement** wird mit dieser Methode zusätzlich die gleichnamige Eigenschaft aller Knoten im Teilbaum gesetzt.

	Datentyp	Beschreibung
Parameter:		
⇨ arrangement	VcArrangement	Ausrichtung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
Dim nodeCltnl As VcNodeCollection
Dim node As VcNode

nodeCltn = VcTree1.NodeCollection
node = VcTree1.GetNodeByID("8")
node.ArrangeSubtree vcVertical
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = VcTree1.NodeCollection;
VcNode node = VcTree1.GetNodeByID("8");
node.ArrangeSubtree vcVertical;
```

CollapseSubtree

Methode von VcNode

Mit dieser Methode können Sie den Knoten einschließlich des darunter hängenden Teilbaums kollabieren. Beim Kollabieren des Teilbaums mit **vcSelf** verschwinden alle Knoten optisch. Beim Kollabieren des Teilbaums mit **vcComplete** verschwinden die Knoten ebenfalls optisch, zusätzlich werden alle die Knoten, die keine Blattknoten sind, in sich kollabiert. Diese müssen dann, falls sie später wieder expandiert werden, einzeln expandiert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ action	VcCollapseExpand	Art der Kollabierung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   If MsgBox("Collapse Node No." & e.Node.DataField(0) & "? ", vbYesNo,
"Collapse node") = vbYes Then
      e.Node.CollapseSubtree(VcCollapseExpand.vcComplete)
   End If
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse
End Sub
```

Code-Beispiel C#

```
      private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
{
   if (MessageBox.Show("Collapse Node No." + e.Node.get_DataField(0) + "? ",
"Collapse node", MessageBoxButtons.OK) == DialogResult.OK)
      e.Node.CollapseSubtree(VcCollapseExpand.vcComplete);
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

DataRecord

Methode von VcNode

Mit dieser Eigenschaft können Sie den Knoten als Datensatzobjekt erfragen. Über die Eigenschaften des Datensatzobjektes haben Sie auch Zugriff auf die entsprechende Datentabelle und Tabellenauflistung.

	Datentyp	Beschreibung
Rückgabewert	VcDataRecord	Zurückgegebener Datensatz

Delete

Methode von VcNode

Mit dieser Methode können Sie einen Knoten löschen.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Knoten erfolgreich/nicht erfolgreich gelöscht

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeRightClicking

   If MsgBox("Delete node: " + e.Node.DataField(0), MsgBoxStyle.YesNo, "Delete
node") = MsgBoxResult.Yes Then
      e.Node.Delete()
      e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
   End If

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeRightClicking(object sender,
NETRONIC.XTree.VcNodeClickingEventArgs e)
   {
   DialogResult retVal = MessageBox.Show("Delete node: " +
e.Node.get_DataField(0), "Deleting node", MessageBoxButtons.YesNo);
   if (retVal ==  DialogResult.Yes)
      {
      e.Node.Delete();
      e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
      }
   }
```

ExpandSubtree

Methode von VcNode

Mit dieser Methode können Sie den Knoten expandieren, ggf. einschließlich seines darunter hängenden Teilbaumes. Bei der Verwendung von **vcSelf** wird nur der Knoten selbst expandiert, aber keine darunter hängenden kollabierten Teilbäume. Bei der Verwendung von **vcComplete** werden alle Knoten des Teilbaums expandiert.

	Datentyp	Beschreibung
Parameter:		
⇨ action	VcCollapseExpand	Art der Expandierung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking

   If MsgBox("Expand Node No." & e.Node.DataField(0) & "? ", vbYesNo, "Expand
node") = vbYes Then
      e.Node.ExpandSubtree(VcCollapseExpand.vcComplete)
   End If
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
VcNodeClickingEventArgs e)
{
   if (MessageBox.Show("Expand Node No." +    e.Node.get_DataField(0) + "? ",
"Expand node", MessageBoxButtons.OK) == DialogResult.OK)
   e.Node.ExpandSubtree(VcCollapseExpand.vcComplete);
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

RelatedDataRecord

Methode von VcNode

Mit dieser Eigenschaft können Sie einen Datensatz aus einer verknüpften Tabelle erfragen, der dem Datensatz der Knotendatentabelle zugeordnet ist. Der im Parameter übergebene Index bezeichnet das Feld im Datensatz, in dem der Schlüssel des zugeordneten Datensatzes steht.

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Datenfeldes, das den Schlüssel enthält

Rückgabewert	VcDataRecord	Zurückgegebener zugeordneter Datensatz

Update

Methode von VcNode

Nachdem Sie ein oder mehrere Datenfelder eines Knotens mit der Eigenschaft **DataField** verändert haben, aktualisieren Sie die Grafik mit **Update**.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Knoten erfolgreich/nicht erfolgreich aktualisiert

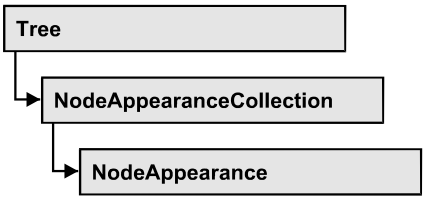
Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcTree1.NodeCollection
node = nodeCltn.FirstNode

node.DataField(12) = "Group A"
node.Update()
```

Code-Beispiel C#

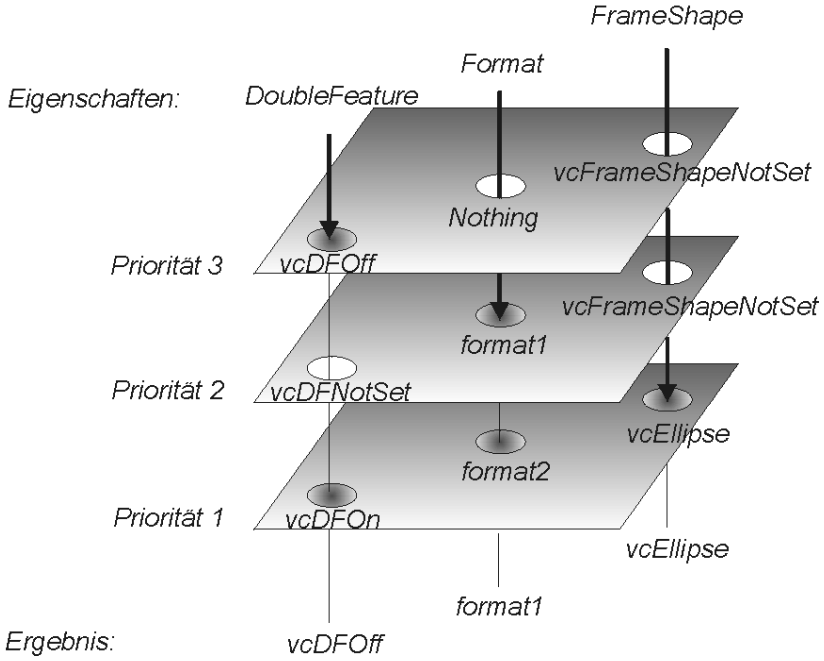
```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
node.set_DataField(12, "Group A");
node.Update();
```

7.23 VcNodeAppearance



Ein NodeAppearance-Objekt bestimmt das Aussehen aller Knoten, deren Daten die dem NodeAppearance-Objekt zugeordneten Filterbedingungen erfüllen. Verschiedene NodeAppearance-Objekte können im Dialogfeld **Knotenaussehen verwalten**, das Sie über die Eigenschaftenseite **Knoten** erreichen, voreingestellt werden.

Die Skizze zeigt, wie sich die Eigenschaften von NodeAppearance-Objekten auf das Aussehen eines Knotens auswirken. Die auf den Knoten zutreffenden NodeAppearances sind nach Priorität absteigend dargestellt. Nicht gesetzte Eigenschaften bei NodeAppearance-Objekten führen dazu, daß die Eigenschaft des nächsttieferen NodeAppearance-Objektes übernommen wird.



Eigenschaften

- BackgroundColor
- LineColorDataFieldIndex
- BackgroundColorMapName
- DoubleFeature
- FilterName
- FormatName

- FrameAroundFieldsVisible
- FrameShape
- LegendText
- LineColor
- LineColorDataFieldIndex
- LineColorMapName
- LineThickness
- LineType
- Name
- Pattern
- PatternColor
- PatternColorDataFieldIndex
- PatternColorMapName
- PatternDataFieldIndex
- PatternMapName
- PileEffect
- Shadow
- ShadowColor
- Specification
- StrikeThrough
- StrikeThroughColor
- ThreeDEffect
- VisibleInLegend

Eigenschaften

BackgroundColor

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie die Hintergrundfarbe eines Knotens einstellen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt.

Bei **-1** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingun-

gen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **-1** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color RGB ({0...255},{0...255},{0...255})	ARGB Farbwerte

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
Set nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance

nodeAppearance.BackColor = RGB(100, 100, 100)
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance;
nodeAppearance.BackColor = RGB(100, 100, 100);
```

BackgroundColorDataFieldIndex

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der Verbindung mit der Eigenschaft **BackColorMapName** benötigt wird. Wenn Sie hier **-1** angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Datenfeldindex

BackgroundColorMapName

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Namen einer Farbzuordnungstabelle für die Hintergrundfarbe setzen oder erfragen. Wird hier "" oder bei der Eigenschaft **BackColorDataFieldIndex -1** angegeben, dann wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Farbzuordnungstabelle

DoubleFeature

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie eine doppelte Umrahmung eines Knotens einstellen oder erfragen. Bei **vcDFNotSet** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **vcDFNotSet** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	VcAppearanceDoubleFeature	Typen von Doppel-Linien
	Mögliche Werte:	
	.vcDFNotSet -1	Schalter für **DoubleFeature nicht gesetzt**
	.vcDFOff 0	Schalter für **DoubleFeature aus**
	.vcDFOn 1	Schalter für **DoubleFeature an**

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance
nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
nodeAppearance.DoubleFrame = VcAppearanceDoubleFrame.vcDFOn
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
nodeAppearance.DoubleFrame = VcAppearanceDoubleFrame.vcDFOn;
```

FilterName

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Filter des NodeAppearance-Objekts setzen oder erfragen. Es gibt Sonderfilter, die unveränderlich sind:

- <ALWAYS>: gilt immer (beim Standard-Aussehen immer gesetzt)
- <NEVER>: gilt niemals

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Filtername

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance
Dim filterOfNodeApp As String

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
filterOfNodeApp = nodeAppearance.filtername
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance =
nodeAppearanceCltn.NodeAppearanceByName("Blue");
string filterOfNodeApp = nodeAppearance.FilterName;
```

FormatName

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie das Format des NodeAppearance-Objekts setzen oder erfragen. Bei **Nothing** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **Nothing** besetzt ist (s. Grafik bei VcNode-Appearance).

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name eines Knotenformat-Objekts oder leere Zeichenkette

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
MsgBox(nodeAppearance.FormatName)
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
MessageBox.Show(nodeAppearance.FormatName);
```

FrameAroundFieldsVisible

Nur-Lese-Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft kann festgelegt werden, ob der Rahmen um die innenliegenden Felder sichtbar ist oder nicht. Die Außenrandlinie der Form

ist davon nicht betroffen, daher wirkt sich diese Eigenschaft bei den möglichen Rahmenformen unterschiedlich aus und hat z.B. beim Typ **vcRectangle** keine Auswirkung.

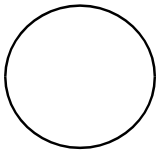
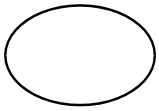
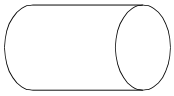
Diese Eigenschaft kann auch im Dialog **Knotenaussehen bearbeiten** gesetzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	VcAppearanceFrameAroundFieldsVisible	Rahmen um Feld **Standardwert:** -1
	Mögliche Werte:	
	.vcFFVNotSet -1	Feldumrandung nicht gesetzt
	.vcFFVOff 0	Schalter für Feldumrandung aus
	.vcFFVOn 1	Schalter für **Feldumrandung** an

FrameShape

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie dem NodeAppearance-Objekt die Rahmenform zuweisen oder erfragen. Bei **vcFrameShapeNotSet** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **vcFrameShapeNotSet** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	AppearanceFrameShapeEnum	Rahmenform
	Mögliche Werte:	
	.vcCircle 11	Rahmenform kreisförmig 
	.vcEllipse 12	Rahmenform elliptisch 
	.vcFile 19	Rahmenform horizontaler Zylinder 
	.vcFrameShapeNotSet -1	Rahmenform nicht gesetzt

.vcLeftArrow 17	Rahmenform pfeilförmig, nach links zeigend
.vcListing 20	Rahmenform Dokument
.vcNoFrameShape 1	keine Rahmenform
.vcOval 4	Rahmenform oval
.vcParallelogram 9	Rahmenform Parallelogramm
.vcPointed 7	Rahmenform an den senkrechten Kanten spitz ausgezogen
.vcRectangle 2	Rahmenform rechteckig
.vcRightArrow 18	Rahmenform pfeilförmig, nach rechts zeigend
.vcRounded 3	Rahmenform rechteckig gerundet
.vcTriangleLeft 10	Rahmenform dreieckig, Spitze links
.vcTriangleUp 13	Rahmenform dreieckig, Spitze oben

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
nodeAppearance.FrameShape = VcAppearanceFrameShape.vcEllipse
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
nodeAppearance.FrameShape = VcAppearanceFrameShape.vcEllipse;
```

LegendText

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie einem Knotenaussehen einen Text zuweisen oder erfragen, der in der Legende für das jeweilige Knotenaussehen angezeigt wird. Steht hier "", dann wird der Inhalt der Eigenschaft **Name** angezeigt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Legendentext des Knotenaussehens **Standardwert:** " " (content of the property **Name**)

LineColor

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie dem NodeAppearance-Objekt eine Linienfarbe zuweisen oder erfragen. Bei **-1** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **-1** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color RGB ({0...255},{0...255},{0...255})	RGB-Farbwerte oder **-1**

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
nodeAppearance.LineColor = Color.LightBlue
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
nodeAppearance.LineColor = Color.LightBlue;
```

LineColorDataFieldIndex

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der bei einer Farbzuordnungstabelle in der Eigenschaft **LineColorMapName** benötigt wird. Wenn Sie hier -1 angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Datenfeldindex

LineColorMapName

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Namen einer Farbzuordnungstabelle für die Linienfarbe setzen oder erfragen. Wird hier "" oder bei der Eigenschaft **LineColorDataFieldIndex -1** angegeben, dann wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Farbzuordnungstabelle

LineThickness

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie die Linienstärke eines NodeAppearance-Objekts erfragen oder festlegen.

Wenn Sie diese Eigenschaft auf Werte zwischen 1 und 4 setzen, wird damit eine absolute Liniendicke in Pixel definiert, d.h. die Linien haben unabhängig vom Zoomfaktor immer die gleiche feste Linienstärke in Pixeln. Dies wird jedoch aufgrund der besseren Lesbarkeit beim Drucken in eine vom Zoomfaktor abhängige Liniendicke umgewandelt:

Wert	Punkte	mm
1	1/2 Punkt	0,09 mm
2	1 Punkt	0,18 mm
3	3/2 Punkt	0,26 mm

Wert	Punkte	mm
4	2 Punkt	0,35 mm

Ein Punkt ist 1/72 Zoll groß und stellt die Maßeinheit für Schriftgrößen dar.

Wenn Sie diese Eigenschaft auf Werte zwischen 5 und 1.000 setzen, wird damit eine Linienstärke in 1/100 mm definiert, d.h. die Linien bekommen eine tatsächliche Dicke in Pixeln, die abhängig vom Zoomfaktor ist.

Wenn Sie diese Eigenschaft auf **-1** setzen, kommt die gleichnamige Eigenschaft des in der Priorität nächst niedrigeren NodeAppearance-Objekts zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **-1** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Linienstärke LineType {1...4}: Werte in Pixeln LineType {5...1000}: Werte in 1/100 mm **Standardwert:** As defined on property page

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.NodeAppearanceByName("Standard")
nodeAppearance.LineThickness = 3
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance =
nodeAppearanceCltn.NodeAppearanceByName("Standard");
nodeAppearance.LineThickness =3;
```

LineType

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie dem NodeAppearance-Objekt einen Linientyp zuweisen oder erfragen. Bei **vcNotSet** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen

und dessen Eigenschaftswert nicht mit **vcNotSet** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	VcLineType	Linientyp
	Mögliche Werte:	
	.vcDashed 4	Linientyp **gestrichelt**
	.vcDashed 4	Linientyp **gestrichelt**
	.vcDashedDotted 5	Linientyp **gestrichelt-gepunktet**
	.vcDashedDotted 5	Linientyp **gestrichelt-gepunktet**
	.vcDotted 3	Linientyp **gepunktet**
	.vcDotted 3	Linientyp **gepunktet**
	.vcLineType0 100	Linientyp 0
	.vcLineType1 101	Linientyp 1
	.vcLineType10 110	Linientyp 10
	.vcLineType11 111	Linientyp 11
	.vcLineType12 112	Linientyp 12
	.vcLineType13 113	Linientyp 13
	.vcLineType14 114	Linientyp 14
	.vcLineType15 115	Linientyp 15
	.vcLineType16 116	Linientyp 16
	.vcLineType17 117	Linientyp 17
	.vcLineType18 118	Linientyp 18
	.vcLineType2 102	Linientyp 2
	.vcLineType3 103	Linientyp 3
	.vcLineType4 104	Linientyp 4
	.vcLineType5 105	Linientyp 5
	.vcLineType6 106	Linientyp 6
	.vcLineType7 107	Linientyp 7
	.vcLineType8 108	Linientyp 8
	.vcLineType9 109	Linientyp 9
	.vcNone 1	Kein Linientyp zugewiesen
	.vcNone 1	Kein Linientyp
	.vcNotSet -1	Kein Linientyp zugewiesen
	.vcSolid 2	Linientyp **durchgezogen**
	.vcSolid 2	Linientyp **durchgezogen**

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance

nodeAppearance.LineType = vcDotted
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance;
nodeAppearance.LineType = vcDotted;
```

Name

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Namen eines NodeAppearance-Objekts erfragen oder setzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Knotenaussehens

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance
Dim nameNodeApp As String

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
NodeApp = nodeAppearance.Name

nameNodeAppName = nodeAppearance.name
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
string nameNodeApp = nodeAppearance.Name;
```

Pattern

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie das Muster des Knotens festlegen oder erfragen. Wenn in der Eigenschaft **PatternMapName** eine Zuordnungstabelle angegeben ist, steuert diese das Muster in Abhängigkeit von den Daten. Bei **-1** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen

Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **-1** besetzt ist (s. Grafik bei VcNodeAppearance).

Werte von vc05PercentPattern bis vc90PercentPattern lauten richtigerweise 2001 bis 2011.

	Datentyp	**Beschreibung**
Eigenschaftswert	VcFillPattern	Mustertyp
	Mögliche Werte: .vc05PercentPattern... vc90PercentPattern 01 - 11	Punkte in Vordergrundfarbe auf Hintergrundfarbe; mit steigender Prozentzahl Vordergrundfarbe immer dichter
	.vcAeroGlassPattern 44	Vertikaler Farbverlauf in der Füllmusterfarbe Engine Cabin Rig & Sail
	.vcBDiagonalPattern 5	Diagonale Linien von links unten nach rechts oben
	.vcCrossPattern 6	Kreuzschraffur
	.vcDarkDownwardDiagonalPattern 2014	Diagonale Linien von links oben nach rechts unten, 50 % näher zusammen als vcFDiagonalPattern und mit doppelter Liniendicke
	.vcDarkHorizontalPattern 2023	Horizontale Linien mit 50% geringerem Abstand als vcHorizontalPattern und doppelter Liniendicke
	.vcDarkUpwardDiagonalPattern 2015	Diagonale Linien von links unten nach rechts oben; mit 50%geringerem Abstand als vcBDiagonalPattern und mit doppelter Liniendicke
	.vcDarkVerticalPattern 2022	Vertikale Linien mit 50% geringerem Abstand als vcVerticalPattern und doppelter Liniendicke
	.vcDashedDownwardDiagonalPattern 2024	Gestrichelte diagonale Linien von links oben nach rechts unten
	.vcDashedHorizontalPattern 2026	Horizontale gestrichelte Linien

.vcDashedUpwardDiagonalPattern 2025	Unterbrochene diagonale Linien von links unten nach rechts oben
.vcDashedVerticalPattern 2027	Vertikale gestrichelte Linien
.vcDiagCrossPattern 7	Diagonale Kreuzschraffur, klein
.vcDiagonalBrickPattern 2032	Diagonales Backsteinmuster
.vcDivotPattern 2036	Grassoden-Muster
.vcDottedDiamondPattern 2038	Diagonale Kreuzschraffur aus punktierten Linien
.vcDottedGridPattern 2037	Kreuzschraffur aus punktierten Linien
.vcFDiagonalPattern 4	Diagonale Linien von links oben nach rechts unten
.vcHorizontalBrickPattern 2033	Horizontales Backsteinmuster
.vcHorizontalGradientPattern 52	Horizontaler Farbverlauf
.vcHorizontalPattern 3	Horizontale Linien
.vcLargeCheckerboardPattern 2044	Schachbrettmuster mit doppelt so großen Quadraten wie vcSmall-CheckerBoardPattern
.vcLargeConfettiPattern 2029	Konfetti -Muster, groß
.vcLightDownwardDiagonalPattern 2012	Diagonale Linien von links oben nach rechts unten; mit 50% geringerem Abstand als vcB-DiagonalPattern
.vcLightHorizontalPattern 2019	Horizontale Linien mit 50% geringerem Abstand als bei vc-HorizontalPattern
.vcLightUpwardDiagonalPattern 2013	Diagonale Linien von links unten nach rechts oben; 50 % näher zusammen als vcBDiagonalPattern
.vcLightVerticalPattern 2018	Vertikale Linien mit 50% geringerem Abstand als bei vcVerticalPattern

.vcNarrowHorizontalPattern 2021	Horizontale Linien mit 75% geringerem Abstand als vc-HorizontalPattern
.vcNarrowVerticalPattern 2020	Vertikale Linien mit 75% geringerem Abstand als vcVerticalPattern
.vcNoPattern 1276	Kein Füllmuster
.vcOutlinedDiamondPattern 2045	Diagonale Kreuzschraffur, groß
.vcPlaidPattern 2035	Schottenstoff-Muster
.vcShinglePattern 2039	Diagonales Dachschindel-Muster
.vcSmallCheckerBoardPattern 2043	Schachbrettmuster
.vcSmallConfettiPattern 2028	Konfetti-Muster
.vcSmallGridPattern 2042	Kreuzschraffur mit 50 % geringerem Abstand als vcCrossPattern
.vcSolidDiamondPattern 2046	Schachbrettmuster mit diagonalen Quadraten
.vcSpherePattern 2041	Kugeln schachbrettartig angeordnet
.vcTrellisPattern 2040	Spalier-Muster
.vcVerticalBottomLightedConvexPattern 43	Vertikaler Farbverlauf von dunkel nach hell
.vcVerticalConcavePattern 40	Vertikaler Farbverlauf von dunkel über hell nach dunkel
.vcVerticalConvexPattern 41	Vertikaler Farbverlauf von hell über dunkel nach hell
.vcVerticalGradientPattern 62	Vertikaler Farbverlauf
.vcVerticalPattern 2	Vertikale Linien
.vcVerticalTopLightedConvexPattern 42	Vertikaler Farbverlauf von hell nach dunkel
.vcWavePattern 2031	Horizontales Wellenmuster
.vcWeavePattern 2034	Muster mit verwebten Streifen

.vcWideDownwardDiagonalPattern 2016	Diagonale Linien von links oben nach rechts unten, mit demselben Abstand wie vcFDiagonalPattern, aber mit dreifacher Liniendicke
.vcWideUpwardDiagonalPattern 2017	Diagonale Linien von links oben nach rechts unten, mit demselben Abstand wie vcBDiagonalPattern, aber mit dreifacher Liniendicke
.vcZigZagPattern 2030	Horizontale Zick-Zacklinien

PatternColor

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie die Musterfarbe des Knotens festlegen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt.

Bei **-1** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **-1** besetzt ist (s. Grafik bei VcNodeAppearance).

Wenn in der Eigenschaft **PatternColorMapName** eine Zuordnungstabelle angegeben ist, steuert diese die Musterfarbe datenabhängig.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	ARGB-Wert ({0...255},{0...255},{0...255},{0...255})

PatternColorDataFieldIndex

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der in Verbindung mit der Eigenschaft **PatternColorMapName** benötigt wird. Wenn Sie hier **-1** angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Datenfeldindex

PatternColorMapName

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Namen einer Farbzuordnungstabelle (Typ vcColorMap) setzen oder erfragen. Wird hier "" angegeben, dann wird keine Zuordnungstabelle verwendet. Nur wenn ein Name einer Farbzuordnungstabelle und ein Datenfeldindex in der Eigenschaft **PatternColorDataFieldIndex** angegeben sind, wird die Musterfarbe des Layers aus der Zuordnungstabelle ausgewählt. Trifft kein Datenfeldeintrag zu, wird die Musterfarbe aus der Eigenschaft **PatternColor** ausgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Farbzuordnungstabelle

PatternDataFieldIndex

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der in Verbindung mit der Eigenschaft **PatternMapName** benötigt wird. Wenn Sie hier **-1** angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	

PatternMapName

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den Namen einer Musterzuordnungstabelle (Typ vcPatternMap) setzen oder erfragen. Wird hier "" angegeben, dann wird keine Zuordnungstabelle verwendet. Nur wenn der Name einer Farbzuordnungstabelle und ein Datenfeldindex in der Eigenschaft **PatternDataFieldIndex** angegeben sind, wird das Muster des Layers aus der Zuordnungstabelle ausgewählt. Trifft kein Datenfeldeintrag zu, wird das Muster aus der Eigenschaft **Pattern** ausgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Zuordnungstabelle

PileEffect

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie die Anzahl von Knotenstapeln im Diagramm erfragen oder festlegen. Bei **-1** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **-1** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl gestapelter Knoten oder **-1**

Code-Beispiel VB.NET

```
Dim nodeAppearanceCollection As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

Set nodeAppearanceCollection = VcTree1.NodeAppearanceCollection
Set nodeAppearance = nodeAppearanceCollection.FirstNodeAppearance

nodeAppearance.Piles = 2
```

Code-Beispiel C#

```
Dim nodeAppearanceCollection As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

Set nodeAppearanceCollection = vcTree1.NodeAppearanceCollection
Set nodeAppearance = nodeAppearanceCollection.FirstNodeAppearance

nodeAppearance.Piles = 2
```

Shadow

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie dem NodeAppearance-Objekt einen Schatten zuweisen oder erfragen. Bei **vcShNotSet** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **vcShNotSet** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	VcAppearanceShadow **Mögliche Werte:** .vcShNotSet -1 .vcShOff 0 .vcShOn 1	Art der Schattensetzung Schalter für **Schatten nicht gesetzt** Schalter für **Schatten aus** Schalter für **Schatten an**

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance
nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
nodeAppearance.Shadow = VcAppearanceShadow.vcShOn
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
nodeAppearance.Shadow = VcAppearanceShadow.vcShOn;
```

ShadowColor

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie die Schattenfarbe des Knotens festlegen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	ARGB-Wert

Specification

Nur-Lese-Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie die Spezifikation dieses Knotenaussehens auslesen. Die Spezifikation ist ein String, der nur lesbare ASCII-Zeichen im Bereich 32 bis 127 enthält und somit problemlos in Textdateien oder Datenbanken gespeichert werden kann. Dies ermöglicht Persistenz. Eine solche Spezifikation kann später zur Wiederherstellung eines Knotenaussehens mit der Methode **VcNodeAppearanceCollection.AddBy-Specification** benutzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Spezifikation des Knotenaussehens

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
MsgBox(nodeAppearance.Specification)
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
MessageBox.Show(nodeAppearance.Specification);
```

StrikeThrough

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie das Durchstreichmuster eines NodeAppearance-Objekts setzen oder erfragen. Bei **vcStrikeThroughNotSet** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **vcStrikeThroughNotSet** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	VcAppearanceStrikeThrough	Durchstreichmuster

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
nodeAppearance.StrikeThrough = VcAppearanceStrikeThrough.vcBackslashed
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
nodeAppearance.StrikeThrough = VcAppearanceStrikeThrough.vcBackslashed;
```

StrikeThroughColor

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie die Farbe des Durchstreichmusters eines NodeAppearance-Objekts setzen oder erfragen. Bei **-1** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **-1** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color RGB ({0...255},{0...255},{0...255})	RGB-Farbwerte oder **-1**

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
nodeAppearance.StrikeThroughColor = Color.LightBlue
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
nodeAppearance.StrikeThroughColor = Color.LightBlue;
```

ThreeDEffect

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie den 3D-Effekt für das NodeAppearance-Objekt erfragen oder festlegen. Bei **vc3DNotSet** kommt die gleichnamige Eigenschaft des in der Priorität nächstniedrigeren NodeAppearance-Objektes zum Tragen, dessen Filterbedingungen für den Knoten ebenfalls zutreffen und dessen Eigenschaftswert nicht mit **vc3DNotSet** besetzt ist (s. Grafik bei VcNodeAppearance).

	Datentyp	Beschreibung
Eigenschaftswert	VcAppearanceThreeDEffect	Art der 3D-Effekt-Setzung

Code-Beispiel VB.NET

```
Dim format As VcTableFormat

format = VcTree1.LeftTable.TableFormatCollection.FormatByName("StandardList")
format.ThreeDEffect = True
```

Code-Beispiel C#

```
VcTableFormat format =
vcTree1.LeftTable.TableFormatCollection.FormatByName("StandardList");
format.ThreeDEffect = true;
```

VisibleInLegend

Eigenschaft von VcNodeAppearance

Mit dieser Eigenschaft können Sie erfragen oder festlegen, ob das NodeAppearance-Objekt in der Legende sichtbar ist. Diese Eigenschaft kann auch im Dialog **Knotenaussehen verwalten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Knotenaussehen in Legende sichtbar (True)/nicht sichtbar (False) **Standardwert:** True

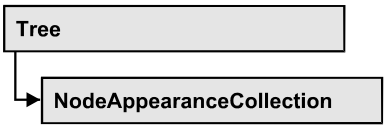
Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.NodeAppearanceByName("Standard")

nodeAppearance.VisibleInLegend = False
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance =
nodeAppearanceCltn.NodeAppearanceByName("Standard");
nodeAppearance.VisibleInLegend = false;
```

7.24 VcNodeAppearanceCollection



In einem Objekt vom Typ VcNodeAppearanceCollection sind automatisch alle definierten NodeAppearance-Objekte zusammengefasst. Sie haben Zugriff auf ein Knotenaussehen über die Methode **NodeAppearanceBy-Name**. Mit der Eigenschaft **Count** können Sie die Anzahl der NodeAppearance-Objekte ermitteln. Über **For Each nodeAppearance In NodeAppearanceCollection** können Sie in einer Schleife auf alle Knotenaussehen zugreifen.

Eigenschaften

- Count

Methoden

- Add
- AddBySpecification
- Copy
- FirstNodeAppearance
- GetEnumerator
- NextNodeAppearance
- NodeAppearanceByIndex
- NodeAppearanceByName
- Remove

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcNodeAppearanceCollection

Mit dieser Eigenschaft können Sie die Anzahl der NodeAppearance-Objekte erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der NodeAppearance-Objekte

Code-Beispiel VB.NET

```
MessageBox.Show(VcTree1.NodeAppearanceCollection.Count)
```

Code-Beispiel C#

```
MessageBox.Show(vcTree1.NodeAppearanceCollection.Count.ToString());
```

Methoden

Add

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie ein neues Knotenaussehen in der NodeAppearanceCollection anlegen. Wenn der Name noch nicht verwendet wird, dann wird das neue VcNodeAppearance-Objekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB). Bei dem neuen Knotenaussehen sind standardmäßig alle Eigenschaften auf transparent gesetzt.

	Datentyp	Beschreibung
Parameter:		
⇨ newName	System.String	Name des NodeAppearance-Objekts
Rückgabewert	VcNodeAppearance	Neues NodeAppearance-Objekt

Code-Beispiel VB.NET

```
newNodeAppearance = VcTree1.NodeAppearanceCollection.Add("nodeapp1")
```

Code-Beispiel C#

```
newNodeAppearance = vcTree1.NodeAppearanceCollection.Add("nodeapp1");
```

AddBySpecification

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie ein Knotenaussehen über eine Knotenaussehen-Spezifikation erzeugen. Dies ermöglicht die Persistenz von Knotenaussehen-Objekten. Die Spezifikation eines Knotenaussehens kann erfragt (siehe VcNodeAppearance-Eigenschaft **Specification**) und gespeichert werden. Bei einer neuen Sitzung kann das gleiche Knotenaussehen mit der wieder eingelesenen Spezifikation samt des gespeicherten Namens wieder erzeugt werden.

	Datentyp	Beschreibung
Parameter:		
⇨ nodeAppearanceSpecification	System.String	Knotenaussehen-Spezifikation
Rückgabewert	VcNodeAppearance	Neues Knotenaussehen-Objekt

Copy

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie ein Knotenaussehen kopieren. Wenn das Knotenaussehen mit dem angegebenen Namen existiert und der Name des neuen Knotenaussehens noch nicht verwendet wird, wird das neue Knotenaussehen-Objekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ fromName	System.String	Name des zu kopierenden Knotenaussehens
⇨ newName	System.String	Name des neuen Knotenaussehens
Rückgabewert	VcNodeAppearance	NodeAppearance-Objekt

FirstNodeAppearance

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. das erste Node-Appearance-Objekt des Collection-Objekts zugreifen, um anschließend mit der Methode **NextNodeAppearance** über die nachfolgenden Objekte zu iterieren. Existiert kein NodeAppearance-Objekt im Collection-Objekt, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcNodeAppearance	Erstes NodeAppearance-Objekt

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance
nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
While Not nodeAppearance Is Nothing
   MessageBox.Show(nodeAppearance.Name)
   nodeAppearance = nodeAppearanceCltn.NextNodeAppearance
End While
```

Code-Beispiel C#

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance
nodeAppearanceCltn = vcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
While Not nodeAppearance Is Nothing
   MessageBox.Show(nodeAppearance.Name)
   nodeAppearance = nodeAppearanceCltn.NextNodeAppearance
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
while (nodeAppearance != null)
{
   MessageBox.Show(nodeAppearance.Name);
   nodeAppearance = nodeAppearanceCltn.NextNodeAppearance();
}
```

GetEnumerator

Methode von VcNodeAppearanceCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Knotenaussehen-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim nodeApp As VcNodeAppearance

For Each nodeApp In VcTree1.NodeAppearanceCollection
   Debug.Print nodeApp.Name
Next
```

Code-Beispiel C#

```
Dim nodeApp As VcNodeAppearance

For Each nodeApp In vcTree1.NodeAppearanceCollection
   Debug.Print nodeApp.Name
Next
```

NextNodeAppearance

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden NodeAppearance-Objekte des Collection-Objekts zugreifen, nachdem Sie mit der Methode **FirstNodeAppearance** den Initialwert erfasst haben. Sind alle Objekte durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcNodeAppearance	Nachfolgendes NodeAppearance-Objekt

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
While Not nodeAppearance Is Nothing
      ListBox1.Items.Add("Name: " + nodeAppearance.Name)
   nodeAppearance = nodeAppearanceCltn.NextNodeAppearance
End While
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
while (nodeAppearance != null)
   {
     listBox1.Items.Add("Name: " + nodeAppearance.Name);
   nodeAppearance = nodeAppearanceCltn.NextNodeAppearance();
   }
```

NodeAppearanceByIndex

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie auf ein einzelnes NodeAppearance-Objekt über seinen Index zugreifen. Existiert kein NodeAppearance-Objekt unter dem angegebenen Index, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Knotenaussehens
Rückgabewert	VcNodeAppearance	Ermitteltes NodeAppearance-Objekt

NodeAppearanceByName

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie ein NodeAppearance-Objekt über den Namen erfragen. Existiert kein NodeAppearance-Objekt unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ nodeAppearanceName	System.String	Name des NodeAppearance-Objekts
Rückgabewert	VcNodeAppearance	NodeAppearance-Objekt

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance

nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.NodeAppearanceByName("NodeAppearanceOne")
nodeAppearance.FrameShape = VcAppearanceFrameShape.vcCircle
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance =
nodeAppearanceCltn.NodeAppearanceByName("NodeAppearanceOne");
nodeAppearance.FrameShape = VcAppearanceFrameShape.vcCircle;
```

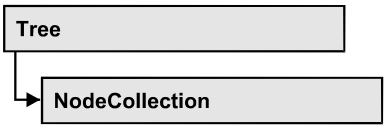
Remove

Methode von VcNodeAppearanceCollection

Mit dieser Methode können Sie ein Knotenaussehen löschen. Wenn das Knotenaussehen noch irgendwo verwendet wird, kann es nicht gelöscht werden. In diesem Fall wird **False** zurückgegeben, sonst **True**.

	Datentyp	Beschreibung
Parameter:		
⇨ name	System.String	Name des Knotenaussehens
Rückgabewert	System.Boolean	Knotenaussehen gelöscht (True)/nicht gelöscht (False)

7.25 VcNodeCollection



Ein Objekt vom Typ VcNodeCollection beinhaltet alle im Diagramm vorhandenen Knoten. Mit der Methode **SelectNodes** können Sie eine Untermenge dieser Knoten selektieren. Über **For Each node InNode-Collection** oder die Methoden **First...** und **Next...** können Sie in einer Schleife auf alle Knoten zugreifen. Die Anzahl der im Auflistungsobjekt vorhandenen Knoten kann über die Eigenschaft **Count** erfragt werden.

Eigenschaften

- Count

Methoden

- FirstNode
- GetEnumerator
- NextNode
- SelectNodes

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcNodeCollection

Mit dieser Eigenschaft können Sie die Anzahl der Knoten in der Knotenauflistung erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl Knoten im NodeCollection-Objekt

Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection

nodeCltn = VcTree1.NodeCollection
MsgBox("Number of nodes: " + nodeCltn.Count)
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
MessageBox.Show("Number of nodes: " + nodeCltn.Count);
```

Methoden

FirstNode

Methode von VcNodeCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. den ersten Knoten des NodeCollection-Objekts zugreifen, um anschließend in einer Schleife mit der Methode **NextNode** über die nachfolgenden Knoten zu iterieren. Existiert kein Knoten im Collection-Objekt, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcNode	Erster Knoten

Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcTree1.NodeCollection
node = nodeCltn.FirstNode
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
```

GetEnumerator

Methode von VcNodeCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Knoten-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

NextNode

Methode von VcNodeCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Knoten des NodeCollection-Objekts zugreifen, nachdem Sie mit der Methode **FirstNode** den Initialwert erfasst haben. Sind alle Knoten durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcNode	Folgeknoten

Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcTree1.NodeCollection
node = nodeCltn.FirstNode
While Not node Is Nothing
   node.Marked = False
   node = nodeCltn.NextNode
End While
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
while (node != null)
   {
   node.Marked = false;
   node = nodeCltn.NextNode;
   }
```

SelectNodes

Methode von VcNodeCollection

Mit dieser Methode können Sie steuern, welche Knoten in das NodeCollection-Objekt aufgenommen werden.

	Datentyp	Beschreibung
Parameter:		
⇨ selType	VcSelectionType	Auszuwählende Knoten
	Mögliche Werte: .vcAll 0 .vcAllVisible 1 .vcMarked 2	 Alle Objekte im Diagramm werden ausgewählt. Alle sichtbaren Objekte werden ausgewählt. Alle markierten Objekte werden ausgewählt.
Rückgabewert	System.Int32	Anzahl ausgewählter Knoten

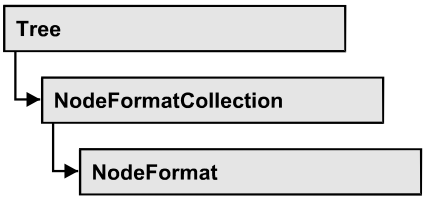
Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcTree1.NodeCollection
nodeCltn.SelectNodes(VcSelectionType.vcSelected)
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcSelected);
```

7.26 VcNodeFormat



Ein Objekt vom Typ VcNodeFormat legt Inhalt und Erscheinungsbild eines Knotens fest. Knotenformate werden zur Designzeit im Dialogfeld **Knotenformate verwalten**, das Sie über die Eigenschaftenseite **Knoten** erreichen, verwaltet und bearbeitet.

Eigenschaften

- FieldsSeparatedByLines
- FormatField
- FormatFieldCount
- Name
- Specification
- WidthOfExteriorSurrounding

Methoden

- CopyFormatField
- GetEnumerator
- RemoveFormatField

Eigenschaften

FieldsSeparatedByLines

Eigenschaft von VcNodeFormat

Mit dieser Eigenschaft können Sie festlegen, ob innenliegende Felder durch sichtbare Linien getrennt werden (True) oder nicht (False).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Innenliegende Felder durch sichtbare Linien getrennt (True)/ nicht getrennt (False)

Code-Beispiel VB.NET

```
Dim nodeFormatCltn As VcNodeFormatCollection
Dim nodeFormat As VcNodeFormat

nodeFormatCltn = VcTree1.NodeFormatCollection
nodeFormat = nodeFormatCltn.FormatByName("FormatOne")
nodeFormat.FieldsSeparatedByLines = True
```

Code-Beispiel C#

```
VcNodeFormatCollection nodeFormatCltn = vcTree1.NodeFormatCollection;
VcNodeFormat nodeFormat = nodeFormatCltn.FormatByName("FormatOne");
nodeFormat.FieldsSeparatedByLines = true;
```

FormatField

Nur-Lese-Eigenschaft von VcNodeFormat

Mit dieser Eigenschaft können Sie ein VcNodeFormatField-Objekt per Index holen. Der Index muss im Bereich von 0 bis FormatFieldCount-1 liegen.

	Datentyp	Beschreibung
Parameter:		
index	System.Int16 0 ... .FormatFieldCount-1	Index des Knotenformatfeldes
Eigenschaftswert	VcNodeFormatField	Knotenformatfeld

FormatFieldCount

Nur-Lese-Eigenschaft von VcNodeFormat

Mit dieser Eigenschaft können Sie die Anzahl der Felder eines Knotenformats ermitteln.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Anzahl der Felder im Knotenformat

Code-Beispiel VB.NET

```
Dim nodeFormat As VcNodeFormat

nodeFormat = VcTree1.NodeFormatCollection.FirstFormat
MsgBox(nodeFormat.FormatFieldCount)
```

Code-Beispiel C#

```
      VcNodeFormat nodeFormat = vcTree1.NodeFormatCollection.FirstFormat();
MessageBox.Show(nodeFormat.FormatFieldCount.ToString());
```

Name

Eigenschaft von VcNodeFormat

Mit dieser Eigenschaft können Sie den Namen des Knotenformats erfragen oder setzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Knotenformatname

Code-Beispiel VB.NET

```
Dim nodeFormat As VcNodeFormat

nodeFormat = VcTree1.NodeFormatCollection.FirstFormat
MsgBox(nodeFormat.Name)
```

Code-Beispiel C#

```
VcNodeFormat nodeFormat = vcTree1.NodeFormatCollection.FirstFormat();
MessageBox.Show(nodeFormat.Name);
```

Specification

Nur-Lese-Eigenschaft von VcNodeFormat

Mit dieser Eigenschaft können Sie die Spezifikation dieses Knotenformats auslesen. Die Spezifikation ist ein String, der nur lesbare ASCII-Zeichen im Bereich 32 bis 127 enthält und somit problemlos in Textdateien oder Datenbanken gespeichert werden kann. Dies ermöglicht Persistenz. Eine solche Spezifikation kann später zur Wiederherstellung eines Knotenformats mit der Methode **VcNodeFormatCollection.AddBySpecification** benutzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Spezifikation des Knotenformats

Code-Beispiel VB.NET

```
Dim nodeFormatCltn As VcNodeFormatCollection
Dim nodeFormat As VcNodeFormat

nodeFormatCltn = VcTree1.NodeFormatCollection
nodeFormat = nodeFormatCltn.FirstNodeFormat
MsgBox(nodeFormat.Specification)
```

Code-Beispiel C#

```
VcNodeFormatCollection nodeFormatCltn = vcTree1.NodeFormatCollection;
VcNodeFormat nodeFormat = nodeFormatCltn.FirstNodeFormat();
MessageBox.Show(nodeFormat.Specification);
```

WidthOfExteriorSurrounding

Eigenschaft von VcNodeFormat

Mit dieser Eigenschaft können Sie die Breite (in mm) des Außenbereichs des Knotenfeldes setzen, d. h. den Abstand in Millimetern, den Knoten mit diesem Knotenformat zu benachbarten Knoten und zum Rand der Darstellung halten sollen. Standardmäßig beträgt die Breite des Außenbereichs 3 mm. Bei kleineren Werten kann es gelegentlich zu Überlagerungen von grafischen Elementen kommen. Daher sollten Sie den Standardwert nur in begründeten Fällen unterschreiten.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 9	Breite des Außenbereichs des Knotenfeldes (in mm)

Methoden

CopyFormatField

Methode von VcNodeFormat

Mit dieser Methode können Sie ein Knotenformatfeld kopieren. Das neue VcNodeFormatField-Objekt wird zurückgegeben. Es erhält den nächsten, noch nicht vergebenen Index.

	Datentyp	Beschreibung
Parameter:		
⇨ position	VcFormatFieldPosition	Position des neuen Knotenformatfeldes
	Mögliche Werte:	
	.vcAbove 1	oberhalb
	.vcBelow 3	unterhalb
	.vcLeftOf 0	links von
	.vcOutsideAbove 9	außerhalb, oberhalb
	.vcOutsideBelow 11	außerhalb, unterhalb
	.vcOutsideLeftOf 8	außerhalb, links von
	.vcOutsideRightOf 12	außerhalb, rechts von
	.vcRightOf 4	rechts von
⇨ refIndex	System.Int16	Index des Referenz-Knotenformatfeldes
Rückgabewert	VcNodeFormatField	Neu angelegtes Knotenformatfeld-Objekt

GetEnumerator

Methode von VcNodeFormat

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Knotenformatfelder iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim format As VcNodeFormat
For Each format In VcTree1.NodeFormatCollection
    Debug.Write(format.Name)
Next
```

Code-Beispiel C#

```
foreach (VcNodeFormat format in vcTree1.NodeFormatCollection)
     Console.Write(format.Name);
```

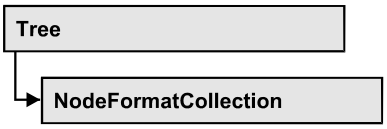
RemoveFormatField

Methode von VcNodeFormat

Mit dieser Methode können Sie ein Knotenformatfeld über den angegebenen Index löschen. Anschließend wird ggf. der Index aller Knotenformatfelder neu festgesetzt, so dass sie wieder fortlaufend numeriert sind.

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des zu löschenden Knotenformatfeldes

7.27 VcNodeFormatCollection



In einem Objekt vom Typ VcNodeFormatCollection sind automatisch alle verfügbaren Knotenformate zusammengefasst. Über **For Each format In-NodeFormatCollection** oder die Methoden **First...** und **Next...** können Sie in einer Schleife auf alle Knotenformate zugreifen. Sie haben Zugriff auf bestimmte Formate über die Eigenschaften **FormatByName**. Die Anzahl der im Auflistungsobjekt vorhandenen Knotenformate kann über die Eigenschaft **Count** erfragt werden.

Eigenschaften

- Count

Methoden

- Add
- AddBySpecification
- Copy
- FirstFormat
- FormatByIndex
- FormatByName
- GetEnumerator
- NextFormat
- Remove

Eigenschaften

Count

Nur-Lese-Eigenschaft von VcNodeFormatCollection

Mit dieser Eigenschaft können Sie die Anzahl der Knotenformatobjekte in der NodeFormat-Auflistung abfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Anzahl der Knotenformate

Code-Beispiel VB.NET

```
Dim formatCltn As VcNodeFormatCollection
Dim numberOfFormats As Integer

formatCltn = VcTree1.NodeFormatCollection
numberOfFormats = formatCltn.Count
```

Code-Beispiel C#

```
VcNodeFormatCollection formatCltn = vcTree1.NodeFormatCollection;
int numberOfFormats = formatCltn.Count;
```

Methoden

Add

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie ein neues Knotenformat in der NodeFormatCollection anlegen. Wenn der Name noch nicht verwendet wird, dann wird das neue VcNodeFormat-Objekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

Das Knotenformat besitzt standardmäßig folgende Eigenschaften:

- ein einziges Feld
- WidthOfExteriorSurrounding: 3 mm

Das Feld hat folgende Eigenschaften:

- Type: vcFFTText
- TextDataFieldIndex: in der Eigenschaftenseite **Allgemeines** festgelegte IDMinimumWidth: 3000
- Alignment: vcFFACenter
- BackColor: -1 (transparent)
- TextFontColor: RGB(0,0,0) (schwarz)
- TextFont: Arial, 10, normal
- LeftMargin, RightMargin, TopMargin, BottomMargin: 0,3 mm

- MinimumTextLineCount, MaximumTextLineCount: 1

	Datentyp	Beschreibung
Parameter: ⇨ newName	System.String	Name des Knotenformats
Rückgabewert	VcNodeFormat	Knotenformat-Objekt

Code-Beispiel VB.NET

```
newNodeFormat = VcTree1.NodeFormatCollection.Add("nodeformat1")
```

Code-Beispiel C#

```
newNodeFormat = vcTree1.NodeFormatCollection.Add("nodeformat1");
```

AddBySpecification

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie ein Knotenformat über eine Knotenformat-Spezifikation erzeugen. Dies ermöglicht die Persistenz von Knotenformat-Objekten. Die Spezifikation eines Knotenformats kann erfragt (siehe VcNodeFormat-Eigenschaft **Specification**) und gespeichert werden. Bei einer neuen Sitzung kann das gleiche Knotenformat mit der wieder eingelesenen Spezifikation samt des gespeicherten Namens wieder erzeugt werden.

	Datentyp	Beschreibung
Parameter: ⇨ formatSpecification	System.String	Knotenformat-Spezifikation
Rückgabewert	VcNodeFormat	Neues Knotenformat-Objekt

Copy

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie ein Knotenformat kopieren. Wenn das Knotenformat mit dem angegebenen Namen existiert und der Name des neuen Knotenformats noch nicht verwendet wird, wird das neue Knotenformat-Objekt zurückgegeben, sonst "Nothing" (Visual Basic) oder "0" (andere Sprachen als VB).

	Datentyp	Beschreibung
Parameter:		
⇨ fromName	System.String	Name des zu kopierenden Knotenformats
⇨ newName	System.String	Name des neuen Knotenformats
Rückgabewert	VcNodeFormat	NodeFormat-Objekt

FirstFormat

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie auf den Initialwert, d.h. das erste Knotenformat des NodeFormatCollection-Objekts zugreifen, um anschließend in einer Schleife mit der Methode **NextFormat** über die nachfolgenden Knotenformate zu iterieren. Existiert kein Knotenformat im NodeFormatCollection-Objekt, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcNodeFormat	Erstes Knotenformat

Code-Beispiel VB.NET

```
Dim format As VcNodeFormat

format = VcTree1.NodeFormatCollection.FirstFormat
```

Code-Beispiel C#

```
VcNodeFormat format = vcTree1.NodeFormatCollection.FirstFormat;
```

FormatByIndex

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie auf ein einzelnes Format über seinen Index zugreifen. Existiert kein Format an dem angegebenen Index, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ index	System.Int16	Index des Knotenformats
Rückgabewert	VcNodeFormat	Ermitteltes Knotenformat

Code-Beispiel VB.NET

```
Dim formatCltn As VcNodeFormatCollection

formatCltn = VcTree1.NodeFormatCollection
format = formatCltn.FormatByIndex(0)
format.WidthOfExteriorSurrounding = 2
```

Code-Beispiel C#

```
VcNodeFormatCollection formatCltn = vcTree1.NodeFormatCollection;
VcNodeFormat format = formatCltn.FormatByIndex(0);
format.WidthOfExteriorSurrounding = 2;
```

FormatByName

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie unter Verwendung des Namens auf ein bestimmtes Knotenformat zugreifen. Existiert kein Knotenformat unter dem angegebenen Namen, wird ein Leerobjekt zurückgegeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Parameter:		
⇨ formatName	System.String	Name des Knotenformats
Rückgabewert	VcNodeFormat	Knotenformat

Code-Beispiel VB.NET

```
Dim formatCltn As VcNodeFormatCollection
Dim format As VcNodeFormat

formatCltn = VcTree1.NodeFormatCollection
format = formatCltn.FormatByName("Standard")
```

Code-Beispiel C#

```
VcNodeFormatCollection formatCltn = vcTree1.NodeFormatCollection;
VcNodeFormat format = formatCltn.FormatByName("Standard");
```

GetEnumerator

Methode von VcNodeFormatCollection

Diese Methode liefert ein Enumerator-Objekt zur Unterstützung von Iterationen durch sprachspezifische Konstrukte. Bei Visual Basic und C# wird es implizit im For...Each-Konstrukt benutzt. Mit diesem Objekt können Sie über alle enthaltenen Knotenformat-Objekte iterieren.

	Datentyp	Beschreibung
Rückgabewert	VcObject	Referenzobjekt

Code-Beispiel VB.NET

```
Dim format As VcNodeFormat

For Each format In VcTree1.NodeFormatCollection
   Debug.Write( format.Name)
Next
```

Code-Beispiel C#

```
foreach (VcNodeFormat format In vcTree1.NodeFormatCollection)
   Console.Write(format.Name);
```

NextFormat

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie in einer Schleife auf die nachfolgenden Knotenformate des NodeFormatCollection-Objekts zugreifen, nachdem Sie mit der Methode **FirstFormat** den Initialwert erfasst haben. Sind alle Formate durchlaufen, wird ein Leerobjekt übergeben (in Visual Basic: **Nothing**).

	Datentyp	Beschreibung
Rückgabewert	VcNodeFormat	Folgendes Knotenformat

Code-Beispiel VB.NET

```
Dim formatCltn As VcNodeFormatCollection
Dim format As VcNodeFormat

formatCltn = VcTree1.NodeFormatCollection
format = formatCltn.Firstformat

While Not format Is Nothing
     ListBox1.Items.Add(format.Name)
      format = formatCltn.NextFormat
End While
```

Code-Beispiel C#

```
VcNodeFormatCollection formatCltn = vcTree1.NodeFormatCollection;
VcNodeFormat format = formatCltn.FirstFormat;

while (format != null)
   {
   listBox1.Items.Add(format.Name);
   format = formatCltn.NextFormat();
   }
```

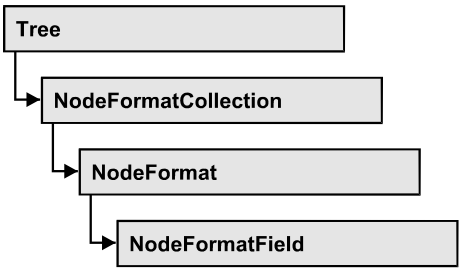
Remove

Methode von VcNodeFormatCollection

Mit dieser Methode können Sie ein Knotenformat löschen. Wenn das Knotenformat noch irgendwo verwendet wird, kann es nicht gelöscht werden. In diesem Fall wird False zurückgegeben, sonst True.

	Datentyp	Beschreibung
Parameter:		
⇨ name	System.String	Name des Knotenformats
Rückgabewert	System.Boolean	Knotenformat gelöscht (True)/nicht gelöscht (False)

7.28 VcNodeFormatField



Ein Objekt vom Typ VcNodeFormatField stellt ein Knotenformatfeld, also ein Feld eines VcNodeFormat-Objekts dar. Ein Knotenformatfeld besitzt im Gegensatz zu vielen anderen Objekten keinen Namen, sondern nur einen Index, unter dem es im Knotenformat untergebracht ist.

Eigenschaften

- Alignment
- BackgroundColor
- BackgroundColorDataFieldIndex
- BackgroundColorMapName
- BottomMargin
- ConstantText
- FormatName
- GraphicsFileName
- GraphicsFileNameDataFieldIndex
- GraphicsFileNameMapName
- GraphicsHeight
- Index
- LeftMargin
- MaximumTextLineCount
- MinimumTextLineCount
- MinimumWidth
- PatternBackgroundColorAsARGB
- PatternBackgroundColorDataFieldIndex
- PatternBackgroundColorMapName
- PatternColorAsARGB
- PatternColorDataFieldIndex
- PatternColorMapName
- PatternEx
- PatternExDataFieldIndex
- PatternExMapName

- RightMargin
- TextAndGraphicsCombined
- TextDataFieldIndex
- TextFont
- TextFontColor
- TextFontDataFieldIndex
- TextFontMapName
- TopMargin
- Type

Eigenschaften

Alignment

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Ausrichtung des Inhalts im Knotenformatfeld festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFormatFieldAlignment	Ausrichtung des Feldinhalts
	Mögliche Werte:	
	.vcFFABottom 28	unten
	.vcFFABottomLeft 27	unten links
	.vcFFABottomRight 29	unten rechts
	.vcFFACenter 25	unten mittig
	.vcFFALeft 24	links
	.vcFFARight 26	rechts
	.vcFFATop 22	oben
	.vcFFATopLeft 21	oben links
	.vcFFATopRight 23	oben rechts

BackgroundColor

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Hintergrundfarbe des Knotenformatfeldes festlegen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt.

Wählen Sie den Wert **-1**, wenn das Feld die Hintergrundfarbe des Knotenformats besitzen soll. Wenn in der Eigenschaft **BackColorMapName** eine Zuordnungstabelle angegeben ist, steuert diese die Hintergrundfarbe in Abhängigkeit von den Daten.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	ARGB-Farbwerte ({0...255},{0...255},{0...255},{0...255}) **Standardwert:** -1

BackgroundColorDataFieldIndex

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der in Verbindung mit der Eigenschaft **BackColorMapName** benötigt wird. Wenn Sie hier **-1** angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Datenfeldindex

BackgroundColorMapName

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Namen einer Farbzuordnungstabelle (Typ vcColorMap) für die Hintergrundfarbe setzen oder erfragen. Wird hier "" angegeben, dann wird keine Zuordnungstabelle verwendet. Wenn der Name einer Farbzuordnungstabelle und zusätzlich ein Datenfeldindex in der Eigenschaft **BackColorDataFieldIndex** angegeben ist, wird die Hintergrundfarbe aus der Zuordnungstabelle ausgewählt. Trifft kein Datenfeldeintrag zu, wird die Hintergrundfarbe aus der Eigenschaft **BackColor** benutzt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Farbzuordnungstabelle

BottomMargin

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Breite des unteren Randes des Knotenformatfeldes in mm festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0...9	Breite des unteren Randes des Knotenformatfeldes in mm

ConstantText

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie einen konstanten Text in dem Knotenformatfeld ausgeben, falls der Typ des Knotenformatfeldes auf **vcFFTText** und falls die Eigenschaft **TextDataFieldIndex** auf **-1** gesetzt wurde.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Konstanter Text

FormatName

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Namen des Knotenformats erfragen, zu dem dieses Knotenformatfeld gehört.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Knotenformats

GraphicsFileName

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie beim Typ **vcFFTGraphics** den Namen einer Grafikdatei setzen oder erfragen, deren Inhalt in dem Knotenformatfeld ausgegeben wird. Der Name muss eine gültige Grafikdatei bezeichnen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Grafikdatei

GraphicsFileNameDataFieldIndex

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie beim Typ **vcFFTGraphics** den Datenfeldindex festlegen oder erfragen, der in der Eigenschaft **GraphicsFileNameMapName** benötigt wird. Beim Wert **-1** wird in dem Knotenformatfeld die Grafikdatei ausgeben, die im entsprechenden Knotenformat angegeben ist. Ist ein gültiger Datenfeldindex angegeben und keine Zuordnungstabelle, dann wird der Grafikdateiname direkt aus dem Inhalt des angegebenen Datenfelds entnommen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Index des Datenfeldes

GraphicsFileNameMapName

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie beim Typ **vcFFTGraphics** den Namen einer Zuordnungstabelle vom Typ **vcGraphicsFileMap** oder "" setzen oder erfragen. Wenn ein Name und zusätzlich ein Datenfeldindex in der Eigenschaft **GraphicsFileNameDataFieldIndex** angegeben ist, wird eine Grafik aus der Zuordnungstabelle angezeigt. Trifft kein Datenfeldeintrag zu, wird die Grafik aus der Eigenschaft **GraphicsFileName** ausgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Grafik-Zuordnungstabelle

GraphicsHeight

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie beim Typ **vcFFTGraphics** die Höhe der Grafik in dem Knotenformatfeld festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 99	Höhe der Grafik in mm

Index

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Index des Knotenformatfelds im zugehörigen Knotenformat erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Index des Knotenformatfeldes

LeftMargin

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Breite des linken Randes des Knotenformatfeldes in mm festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0...9	Breite des linken Randes des Knotenformatfeldes in mm

MaximumTextLineCount

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die maximale Anzahl der Zeilen in dem Knotenformatfeld setzen oder erfragen, falls das Knotenformatfeld vom Typ **vcFFTText** ist. Bitte sehen Sie auch die Eigenschaft **MinimumTextLine-Count**.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 9	Maximale Zeilenzahl

MinimumTextLineCount

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die minimale Anzahl der Zeilen in dem Knotenformatfeld setzen oder erfragen, falls der Typ des Knotenformatfeldes auf **vcFFTText** gesetzt wurde. Ist in einem Knoten mehr Text vorhanden, als in die minimale Anzahl der Zeilen hineinpasst, wird dieses Feld für diesen Knoten dynamisch bis zur maximalen angegebenen Anzahl der Zeilen ausgedehnt. Wenn Sie dieser Eigenschaft einen Wert zuweisen, sollten Sie anschließend auch erneut der Eigenschaft **MaximumTextLineCount** den gewünschten Wert setzen, sonst könnte es vorkommen, dass das Maximum durch das Minimum überschrieben wird.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 9	Minimale Zeilenzahl

MinimumWidth

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die minimale Breite des Knotenformatfeldes in mm festlegen oder erfragen. Die Breite des Feldes kann sich vergrößern, wenn unter oder über dem Feld andere Felder größere minimale Breiten besitzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0 ... 99	Minimale Breite des Knotenformatfeldes in mm

PatternBackgroundColorAsARGB

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Hintergrundfarbe des Knotenformatfeldes festlegen oder erfragen. Farbwerte haben einen Transparenz- oder Alphawert, einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt. Bei der Umwandlung eines RGB-Wertes in einen ARGB-Wert muss ein Alpha-Wert von 255 hinzugegeben werden.

Wählen Sie den Wert **-1**, wenn das Feld die Hintergrundfarbe des Knotenformats besitzen soll.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	ARGB-Farbwerte ({0...255},{0...255},{0...255},{0...255})

PatternBackgroundColorDataFieldIndex

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der in Verbindung mit der Eigenschaft **PatternBackgroundColor-MapName** benötigt wird. Wenn Sie hier -1 angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Datenfeldindex

PatternBackgroundColorMapName

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Namen einer Farbzuordnungstabelle (Typ vcColorMap) setzen oder erfragen. Wird hier "" angegeben, dann wird keine Zuordnungstabelle verwendet. Wenn ein Name einer Farbzuordnungstabelle und zusätzlich ein Datenfeldindex in der Eigenschaft **PatternBackgroundColorDataFieldIndex** angegeben ist, wird die Hintergrundfarbe aus der Zuordnungstabelle ausgewählt. Trifft kein Datenfeldeintrag zu, wird die Hintergrundfarbe aus der Eigenschaft **BackColor** ausgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Farbzuordnungstabelle

PatternColorAsARGB

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Musterfarbe des Knotenformatfeldes erfragen oder festlegen. Farbwerte haben einen Transparenz- oder Alphawert,

einen Rot-, einen Blau- und einen Grünanteil im Zahlenbereich von 0..255 (ARGB-Wert). Ein Alpha-Wert von 0 bedeutet vollständige Transparenz, während der Wert 255 eine voll deckende Farbe erzeugt. Bei der Umwandlung eines RGB-Wertes in einen ARGB-Wert muss ein Alpha-Wert von 255 hinzugegeben werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	ARGB-Farbwerte ({0...255},{0...255},{0...255},{0...255})

PatternColorDataFieldIndex

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der in Verbindung mit der Eigenschaft **PatternColorMapName** benötigt wird. Wenn Sie hier **-1** angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Datenfeldindex

PatternColorMapName

Nur-Lese-Eigenschaft von VcNodeFormatField

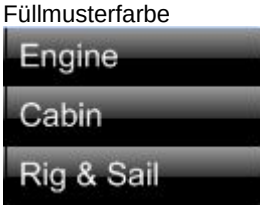
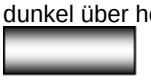
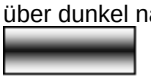
Mit dieser Eigenschaft können Sie den Namen einer Farbzuordnungstabelle (Typ vcColorMap) setzen oder erfragen. Wird hier "" angegeben, dann wird keine Zuordnungstabelle verwendet. Nur wenn ein Name einer Farbzuordnungstabelle und ein Datenfeldindex in der Eigenschaft **PatternColorDataFieldIndex** angegeben sind, wird die Musterfarbe des Kalendergitters aus der Zuordnungstabelle ausgewählt. Trifft kein Datenfeldeintrag zu, wird die Musterfarbe aus der Eigenschaft **PatternColor** ausgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Farbzuordnungstabelle

PatternEx

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie für den Hintergrund des Knotenformatfeldes ein Muster setzen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFieldFillPattern	Mustertyp
	Mögliche Werte:	
	.vcAeroGlassPattern 44	Vertikaler Farbverlauf in der Füllmusterfarbe 
	.vcFieldNoPattern 1276	Kein Füllmuster
	.vcFieldVerticalBottomLightedConvexPattern 43	Vertikaler Farbverlauf von dunkel nach hell
	.vcFieldVerticalConcavePattern 40	Vertikaler Farbverlauf von dunkel über hell nach dunkel 
	.vcFieldVerticalConvexPattern 41	Vertikaler Farbverlauf von hell über dunkel nach hell 
	.vcFieldVerticalTopLightedConvexPattern 42	Vertikaler Farbverlauf von hell nach dunkel

PatternExDataFieldIndex

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der in Verbindung mit der Eigenschaft **PatternExMapName** benötigt wird. Wenn Sie hier **-1** angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Datenfeldindex

PatternExMapName

Nur-Lese-Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Namen einer Muster-Zuordnungstabelle (Typ vcPatternMap) setzen oder erfragen. Wird hier "" angegeben, dann wird keine Zuordnungstabelle verwendet. Wenn ein Name einer Muster-Zuordnungstabelle und zusätzlich ein Datenfeldindex in der Eigenschaft **PatternExDataFieldIndex** angegeben ist, wird das Muster aus der Zuordnungstabelle ausgewählt. Trifft kein Datenfeldeintrag zu, wird das Muster aus der Eigenschaft **PatternEx** ausgegeben.

	Datentyp	Beschreibung
Parameter:		
⇨ Rückgabewert	System.String	Name der Musterzuordnungstabelle
Eigenschaftswert	System.String	Name der Musterzuordnungstabelle

RightMargin

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Breite des rechten Randes des Knotenformatfeldes in mm festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0...9	Breite des rechten Randes des Knotenformatfeldes in mm

TextAndGraphicsCombined

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob das Knotenfeld ein Kombifeld ist. (Vgl. Dialog **Knotenformat bearbeiten**).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Kombifeld (True)/ kein Kombifeld (False)

TextDataFieldIndex

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Index des Datenfelds, dessen Inhalt in dem Knotenformatfeld dargestellt werden soll, erfragen oder setzen, sofern es sich um ein Feld des Datentyps **vcFFTText** handelt. Falls der Index **-1** ist, wird stattdessen der Inhalt der Eigenschaft **ConstantText** ausgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Index des Datenfeldes

TextFont

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Schriftart des Knotenformatfeldes festlegen oder erfragen, falls der Typ des Feldes auf **vcFFTText** gesetzt wurde. Wenn in der Eigenschaft **TextFontMapName** eine Zuordnungstabelle angegeben ist, steuert diese die Schriftart in Abhängigkeit von den Daten.

	Datentyp	Beschreibung
Eigenschaftswert	System.DrawingFont	Schriftart des Knotenformatfelds

TextFontColor

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Schriftfarbe des Knotenformatfeldes festlegen oder erfragen, falls der Typ des Feldes auf **vcFFTText** gesetzt wurde. Wenn über die Eigenschaft **TextFontColorMapName** eine Zuordnungstabelle angegeben ist, steuert diese die Schriftfarbe in Abhängigkeit von den Daten.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	Schriftfarbe des Knotenformatfelds **Standardwert:** -1

TextFontDataFieldIndex

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Datenfeldindex festlegen oder erfragen, der bei einer Schrift-Zuordnungstabelle in der Eigenschaft **TextFontMapName** benötigt wird. Wenn Sie hier -1 angegeben, wird keine Zuordnungstabelle verwendet.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Datenfeldindex

TextFontMapName

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Namen einer Schrift-Zuordnungstabelle (Typ vcFontMap) setzen oder erfragen. Wird hier "" angegeben, dann wird keine Zuordnungstabelle verwendet. Wenn ein Name einer Schrift-Zuordnungstabelle und zusätzlich ein Datenfeldindex in der Eigenschaft **TextFontDataFieldIndex** angegeben ist, wird die Schriftart aus der Zuordnungstabelle ausgewählt. Trifft kein Datenfeldeintrag zu, wird die Schriftart aus der Eigenschaft **TextFont** ausgegeben.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Schrift-Zuordnungstabelle

TopMargin

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie die Breite des oberen Randes des Knotenformatfeldes in mm festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 0...9	Breite des oberen Randes des Knotenformatfeldes in mm

Type

Eigenschaft von VcNodeFormatField

Mit dieser Eigenschaft können Sie den Typ des Knotenformatfelds erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcFormatFieldType **Mögliche Werte:** .vcFFTGraphics 64 .vcFFTText 36	Typ des Knotenformatfeldes Grafik Text

7.29 VcPrinter

Das VcPrinter-Objekt stellt Ihnen Eigenschaften zur Verfügung, die die Seitengestaltung und den Druckvorgang betreffen. Sie können die Randbreiten der Seiten einstellen sowie Seitenrahmen, Seitenzahlen, Seitenbeschriftung, Schnittmarkierungen und Druckdatum setzen. Weiterhin können Sie die Anzahl der Seiten, auf die das Diagram verteilt werden soll, sowie Vergrößerungsfaktor, Druckausrichtung, Hoch- bzw. Querformat, Papierformat und Farbmodus festlegen.

Eigenschaften

- AbsoluteBottomMarginInInches
- AbsoluteLeftMarginInCM
- AbsoluteLeftMarginInInches
- AbsoluteRightMarginInCM
- AbsoluteRightMarginInInches
- AbsoluteTopMarginInCM
- AbsoluteTopMarginInInches
- Alignment
- CurrentHorizontalPagesCount
- CurrentVerticalPagesCount
- CurrentZoomFactor
- CuttingMarks
- DefaultPrinterName
- DocumentName
- FitToPage
- FoldingMarksType
- MarginsShownInInches
- MaxHorizontalPagesCount
- MaxVerticalPagesCount
- Orientation
- PageDescription
- PageDescriptionString
- PageFrame
- PageNumberMode
- PageNumbers
- PagePaddingEnabled
- PaperSize
- PrintDate
- PrinterName

- PrintPreviewWithFirstPage
- TitleAndLegendOnAllPages
- VcCalendarGrid
- ZoomFactorAsDouble

Eigenschaften

AbsoluteBottomMarginInInches

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Höhe des unteren Seitenrandes in der Maßeinheit Zoll setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

Hinweis: Damit im Dialog **Seite einrichten** glattere Werte erzielt werden, beträgt der interne Umrechnungsfaktor 2,5 cm/Zoll statt der eigentlich korrekten 2,54 cm/Zoll (1,5 cm entsprechen dann 0,6 Zoll, 1 cm 0,4 Zoll).

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Absolute Höhe des unteren Seitenrandes in Zoll **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteBottomMarginInInches = 0.5   ' 0.5 inches
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteBottomMarginInInches = 0.5;   // 0.5 inches
```

AbsoluteLeftMarginInCM

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Breite des linken Seitenrandes in der Maßeinheit cm setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Breite des linken Seitenrandes in cm **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteTopMarginInCM = 1.5   ' 2 cm
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteTopMarginInCM = 1.5;   // 1.5 cm
```

AbsoluteLeftMarginInInches

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Breite des linken Seitenrandes in der Maßeinheit Zoll setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

Hinweis: Damit im Dialog **Seite einrichten** glattere Werte erzielt werden, beträgt der interne Umrechnungsfaktor 2,5 cm/Zoll statt der eigentlich korrekten 2,54 cm/Zoll (1,5 cm entsprechen dann 0,6 Zoll, 1 cm 0,4 Zoll).

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Absolute Breite des linken Seitenrandes in Zoll **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteBottomMarginInInches = 0.5   ' 0.5 inches
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteBottomMarginInInches = 0.5;   // 0.5 inches
```

AbsoluteRightMarginInCM

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Breite des rechten Seitenrandes in der Maßeinheit cm setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Breite des rechten Seitenrandes in cm **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteTopMarginInCM = 1.5   ' 2 cm
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteTopMarginInCM = 1.5;   // 1.5 cm
```

AbsoluteRightMarginInInches

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Breite des rechten Seitenrandes in der Maßeinheit Zoll setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

Hinweis: Damit im Dialog **Seite einrichten** glattere Werte erzielt werden, beträgt der interne Umrechnungsfaktor 2,5 cm/Zoll statt der eigentlich korrekten 2,54 cm/Zoll (1,5 cm entsprechen dann 0,6 Zoll, 1 cm 0,4 Zoll).

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Absolute Breite des rechten Seitenrandes in Zoll **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteBottomMarginInInches = 0.5   ' 0.5 inches
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteBottomMarginInInches = 0.5;   // 0.5 inches
```

AbsoluteTopMarginInCM

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Höhe des oberen Seitenrandes in der Maßeinheit cm setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Höhe des oberen Seitenrandes in cm **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteTopMarginInCM = 1.5   ' 2 cm
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteTopMarginInCM = 1.5;   // 1.5 cm
```

AbsoluteTopMarginInInches

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Höhe des oberen Seitenrandes in der Maßeinheit Zoll setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

Hinweis: Damit im Dialog **Seite einrichten** glattere Werte erzielt werden, beträgt der interne Umrechnungsfaktor 2,5 cm/Zoll statt der eigentlich korrekten 2,54 cm/Zoll (1,5 cm entsprechen dann 0,6 Zoll, 1 cm 0,4 Zoll).

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Absolute Höhe des oberen Seitenrandes in Zoll **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteBottomMarginInInches = 0.5   ' 0.5 inches
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteBottomMarginInInches = 0.5;   // 0.5 inches
```

Alignment

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die Ausrichtung des Ausdrucks auf einer Seite setzen oder erfragen. Sie hat nur dann eine Auswirkung, wenn die Gesamtgrafik auf einer einzigen Seite dargestellt wird, oder wenn die Eigenschaft **RepeatTitleAndLegend** eingeschaltet ist. In allen anderen Fällen wird die Gesamtgrafik zentriert ausgerichtet.

	Datentyp	Beschreibung
Eigenschaftswert	VcPrinterAlignment	Ausrichtung des Ausdrucks auf der Seite **Standardwert:** vcPCenterCenter
	Mögliche Werte:	
	.vcPBottomCenter 28	vertikale Ausrichtung: unten, horizontale Ausrichtung: mittig
	.vcPBottomLeft 27	vertikale Ausrichtung: unten, horizontale Ausrichtung: links
	.vcPBottomRight 29	vertikale Ausrichtung: unten, horizontale Ausrichtung: rechts
	.vcPCenterCenter 25	vertikale Ausrichtung: mittig, horizontale Ausrichtung: mittig
	.vcPCenterLeft 24	vertikale Ausrichtung: mittig, horizontale Ausrichtung: links
	.vcPCenterRight 26	vertikale Ausrichtung: mittig, horizontale Ausrichtung: rechts

.vcPTopCenter 22	vertikale Ausrichtung: oben, horizontale Ausrichtung: mittig
.vcPTopLeft 21	vertikale Ausrichtung: oben, horizontale Ausrichtung: links
.vcPTopRight 23	vertikale Ausrichtung: oben, horizontale Ausrichtung: rechts

Code-Beispiel VB.NET

```
VcTree1.Printer.Alignment = VcPrinterAlignment.vcPTopLeft
```

Code-Beispiel C#

```
vcTree1.Printer.Alignment = VcPrinterAlignment.vcPTopLeft;
```

CurrentHorizontalPagesCount

Nur-Lese-Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die tatsächliche Anzahl der Seiten des Ausdrucks in der Breite ermitteln. Siehe auch die Eigenschaften **CurrentVerticalPagesCount** und **MaxHorizontalPagesCount**.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche Anzahl Seiten in horizontaler Richtung

CurrentVerticalPagesCount

Nur-Lese-Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die tatsächliche Anzahl der Seiten des Ausdrucks in der Höhe ermitteln. Siehe auch die Eigenschaften **CurrentHorizontalPagesCount** und **MaxVerticalPagesCount**.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche Anzahl Seiten in vertikaler Richtung

CurrentZoomFactor

Nur-Lese-Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie den tatsächlichen Zoomfaktor in Prozent für die Einstellung **FitToPage = False** erfragen (Zoomfaktor = 100: Originalgröße, Zoomfaktor > 100: Vergrößerung, Zoomfaktor < 100: Verkleinerung).

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Tatsächlicher Zoomfaktor

CuttingMarks

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob Schnittmarkierungen auf eine Seite gedruckt werden sollen (True) oder nicht (False).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Schnittmarken werden (True) / werden nicht (False) gedruckt **Standardwert:** False

Code-Beispiel VB.NET

```
VcTree1.Printer.CuttingMarks = True
```

Code-Beispiel C#

```
vcTree1.Printer.CuttingMarks = true;
```

DefaultPrinterName

Nur-Lese-Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie den Namen des aktuellen Standard-Systemdruckers erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des aktuellen Standard-Systemdruckers

DocumentName

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie den Namen des Dokumentes bestimmen oder auslesen. Der Dokumentenname wird beim Drucker in der Liste der zu druckenden Dokumente angezeigt und hat bei speziellen Druckertreibern wie z. B. einigen, die PDF-Dateien erzeugen, besondere Funktionen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Dokumentenname **Standardwert:** " "

FitToPage

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie setzen oder erfragen, ob die über die Eigenschaften **MaxHorizontalPagesCount** und **MaxVerticalPagesCount** definierte Anzahl von Seiten gedruckt werden soll (True) oder ob das Diagramm in der mit der Eigenschaft **ZoomFactor** eingestellten Größe ausgegeben werden soll (False).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Diagramm wird auf eine definierte Anzahl von Seiten verteilt/wird in der voreingestellten Größe ausgegeben.

Code-Beispiel VB.NET

```
VcTree1.Printer.FitToPage = True
```

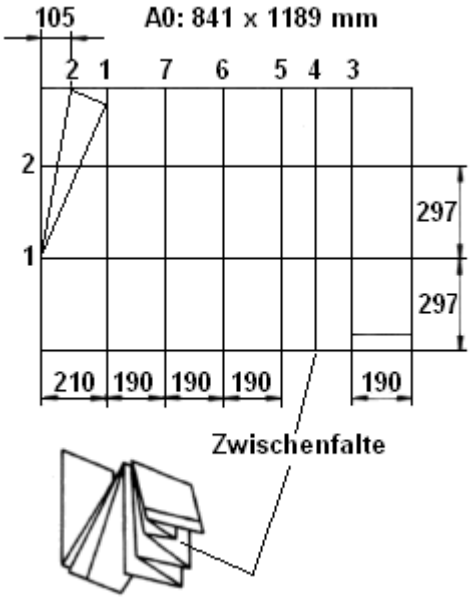
Code-Beispiel C#

```
vcTree1.Printer.FitToPage = true;
```

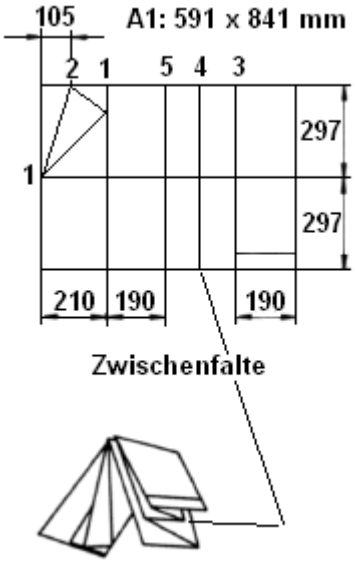
FoldingMarksType

Eigenschaft von VcPrinter

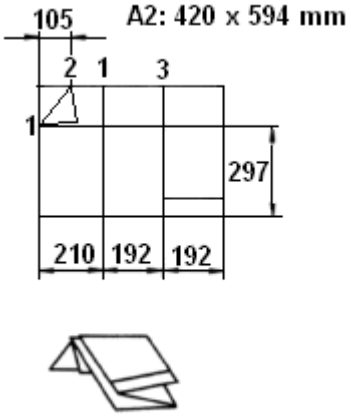
Mit dieser Eigenschaft können Sie folgende Faltmarkierungen nach DIN 824 für den Ausdruck festlegen oder erfragen. Diese ermöglichen das standardisierte Falten für DIN-A-Blattgrößen:



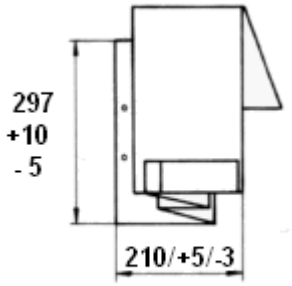
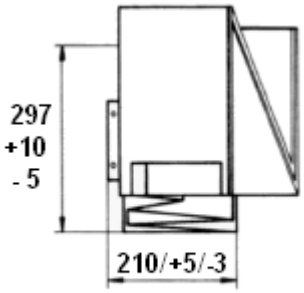
Faltung des DIN-A-0 Formats

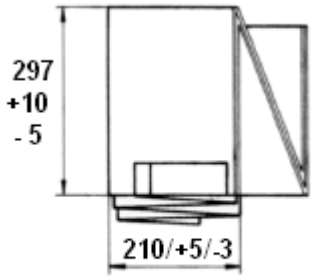


Faltung des DIN-A-1 Formats



Faltung des DIN-A-2 Formats

	Datentyp	Beschreibung
Eigenschaftswert	VcFoldingMarksType	Faltmarkierungen **Standardwert:** vcFMTNone
	Mögliche Werte: .vcFMTDIN824FormA 65	Ausgabe von Faltmarkierungen nach DIN824-A: Die gefaltete Zeichnung kann gelocht und ohne Heftstreifen abgeheftet werden.  **Faltung nach DIN 824-A**
	.vcFMTDIN824FormB 66	Ausgabe von Faltmarkierungen nach DIN824-B: Die gefaltete Zeichnung kann gelocht und mittels Heftstreifens abgeheftet werden.  **Faltung nach DIN 824-B**

.vcFMTDIN824FormC 67	Ausgabe von Faltmarkierungen nach DIN824-C: Die gefaltete Zeichnung wird nicht gelocht, sondern in eine Sichthülle gelegt.  Faltung nach DIN 824-C
.vcFMTNone 0	Keine Ausgabe von Faltmarkierungen

MarginsShownInInches

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob im Dialog **Seite einrichten** die Maßeinheit für Seitenränder in Zoll ein- und ausgegeben wird. (Gegenwärtig nur zur Laufzeit möglich).

Hinweis: Damit im Dialog **Seite einrichten** glattere Werte erzielt werden, beträgt der interne Umrechnungsfaktor 2,5 cm/Zoll statt der eigentlich korrekten 2,54 cm/Zoll (1,5 cm entsprechen dann 0,6 Zoll, 1 cm 0,4 Zoll).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Maßeinheit der Seitenränder im Dialog **Seite einrichten** in Zoll (True)/ in cm (False) **Standardwert:** False

MaxHorizontalPagesCount

Eigenschaft von VcPrinter

Diese Eigenschaft dient der Festlegung oder Erfragung der horizontalen Seitenzahl beim Drucken und für die Druckvorschau. Die Festlegung ist nur wirksam, wenn Sie in der Eigenschaft **ScalingMode** den Wert **vcFitToPage-Count** oder **vcZoomWithHorizontalFit** festgelegt haben. S. auch Eigenschaft **MaxVerticalPagesCount**.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Maximale Anzahl Seiten in horizontaler Richtung **Standardwert:** 1

Code-Beispiel VB.NET

```
VcTree1.Printer.MaxHorizontalPagesCount = 4
```

Code-Beispiel C#

```
vcTree1.Printer.MaxHorizontalPagesCount = 4;
```

MaxVerticalPagesCount

Eigenschaft von VcPrinter

Diese Eigenschaft dient der Festlegung oder Erfragung der vertikalen Seitenzahl beim Drucken und für die Druckvorschau. Diese Festlegung ist nur wirksam, wenn Sie in der Eigenschaft **ScalingMode** den Wert **vcFitToPageCount** festgelegt haben. S. auch Eigenschaft **MaxHorizontalPagesCount**.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Maximale Anzahl Seiten in vertikaler Richtung **Standardwert:** 1

Code-Beispiel VB.NET

```
VcTree1.Printer.MaxVerticalPagesCount = 4
```

Code-Beispiel C#

```
vcTree1.Printer.MaxVerticalPagesCount = 4;
```

Orientation

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen, ob die einzelnen Seiten des Ausdrucks im Hoch- oder Querformat verwendet werden.

	Datentyp	Beschreibung
Eigenschaftswert	VcOrientation	Ausrichtung **Standardwert:** VcPortrait
	Mögliche Werte: .vcLandscape 42 .vcPortrait 41	 Querformat Hochformat

Code-Beispiel VB.NET

```
VcTree1.Printer.Orientation = VcOrientation.vcLandscape
```

Code-Beispiel C#

```
vcTree1.Printer.Orientation = VcOrientation.vcLandscape;
```

PageDescription

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob die Seitenbeschriftung für die linke untere Ecke jeder Seite erscheinen soll (True) oder nicht (False). Den Inhalt der Seitenbeschriftung legen Sie über die Eigenschaft **PageDescriptionString** fest.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Seitenbeschriftung wird (True) / wird nicht gedruckt (False) **Standardwert:** False

Code-Beispiel VB.NET

```
VcTree1.Printer.PageDescription = True
```

Code-Beispiel C#

```
vcTree1.Printer.PageDescription = true;
```

PageDescriptionString

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie eine Seitenbeschriftung für die linke untere Ecke jeder Seite festlegen oder erfragen. Die Ausgabe der Seitenbeschriftung können Sie mit der Eigenschaft **PageDescription** steuern. Für die Seitennummerierung können Sie folgende Platzhalter angeben, die dann beim Ausdruck durch die entsprechenden Inhalte ersetzt werden:

{PAGE} = fortlaufende Seitennummer

{NUMPAGES} = Gesamtanzahl der Seiten

{ROW} = Zeilenposition des Ausschnitts im Gesamtdiagramm

{COLUMN} = Spaltenposition des Ausschnitts im Gesamtdiagramm

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Seitenbeschriftung **Standardwert:** Empty string ""

Code-Beispiel VB.NET

```
VcTree1.Printer.PageDescriptionString = "Tree-Graphics"
```

Code-Beispiel C#

```
vcTree1.Printer.PageDescriptionString = "Tree-Graphics";
```

PageFrame

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob um den Ausdruck ein Rahmen gezogen werden soll (True) oder nicht (False). Wenn die Eigenschaft **TableTimeScaleOnAllPages** eingeschaltet ist, so wird der Rahmen um jede Einzelseite gezogen, anderenfalls wird er um die gesamte Grafik gezogen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Seitenrahmen wird dargestellt (True) / wird nicht dargestellt (False). **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.Printer.PageFrame = True
```

Code-Beispiel C#

```
vcTree1.Printer.PageFrame = true;
```

PageNumberMode

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen bzw. erfragen, wie die Seitennummerierung ausgegeben werden soll: "Seite N von M Seiten" oder "x.y" (Zeilennummer/Spaltennummer).

	Datentyp	Beschreibung
Eigenschaftswert	VcPageNumberMode	Art der Seitennumerierung **Standardwert:** vcPRowColumn
	Mögliche Werte: .vcPageNOfM 1597	"Seite N von M Seiten"

.vcPRowColumn 1596	"x.y" (Zeilennummer.Spaltennummer)

Code-Beispiel VB.NET

```
Dim printer As VcPrinter
printer.Orientation = VcOrientation.vcLandscape
printer.PageNumberMode = VcPageNumberMode.vcPageNOfM
printer.PageNumbers = True
printer.FitToPage = False
VcTree1.ShowPrintPreviewDialog()
```

Code-Beispiel C#

```
VcPrinter printer = vcTree1.Printer;
printer.Orientation = VcOrientation.vcLandscape;
printer.PageNumberMode = VcPageNumberMode.vcPageNOfM;
printer.PageNumbers = true;
printer.FitToPage = false;
vcTree1.ShowPrintPreviewDialog();
```

PageNumbers

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob die Seitenzahl in der linken unteren Ecke einer Seite erscheinen soll (True) oder nicht (False). Die Art der Nummerierung können Sie mit Hilfe der Eigenschaft **PageNumberMode** festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Seitenzahlen werden (True) / werden nicht (False) ausgegeben **Standardwert:** False

Code-Beispiel VB.NET

```
VcTree1.Printer.PageNumbers = True
```

Code-Beispiel C#

```
vcTree1.Printer.PageNumbers = true;
```

PagePaddingEnabled

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob zwischen dem Diagramm und den Boxen für Titel und Legende so viel Platz gelassen werden soll, dass die Boxen auf jeder Druckseite immer in voller Breite gedruckt werden können und fest am Blattrand positioniert sind. Ist die Eigenschaft auf **False** gesetzt, werden die Boxen ohne Zwischenraum am

Diagramm gedruckt und können dann je nach Diagramm auf den verschiedenen Druckseiten in der Breite variieren.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Zwischenraum zwischen Diagramm und Boxen für Legende/Titel wird (True) / wird nicht (False) ausgegeben **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.Printer.PagePaddingEnabled = True
```

Code-Beispiel C#

```
vcTree1.Printer.PagePaddingEnabled = true;
```

PaperSize

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die zu verwendende Papiergröße festlegen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcPaperSize	Papiergröße
	Mögliche Werte:	
	.vcDIN_A2 66	DIN A2
	.vcDIN_A3 8	DIN A3
	.vcDIN_A4 9	DIN A4
	.vcISO_C 24	ISO C
	.vcISO_D 25	ISO D
	.vcISO_E 26	ISO E
	.vcUS_LEGAL 5	US LEGAL
	.vcUS_LETTER 1	US LETTER

Code-Beispiel VB.NET

```
VcTree1.Printer.PaperSize = VcPaperSize.vcDIN_A3
```

Code-Beispiel C#

```
vcTree1.Printer.PaperSize = VcPaperSize.vcDIN_A3;
```

PrintDate

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob das Druckdatum in der linken unteren Ecke jeder Seite erscheinen soll (True) oder nicht (False).

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Druckdatum wird/wird nicht ausgegeben

Code-Beispiel VB.NET

```
VcTree1.Printer.PrintDate = True
```

Code-Beispiel C#

```
vcTree1.Printer.PrintDate = true;
```

PrinterName

Nur-Lese-Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie den Namen des aktuell ausgewählten Druckers auslesen bzw. setzen. Dies kann zum Speichern und Wiederherstellen des Zustands des Printer-Objekts verwendet werden.

Wenn man beim Setzen der Eigenschaft einen leeren String übergibt, wird der im System eingestellte Standarddrucker benutzt.

Hinweis: Bitte beachten Sie, dass bei Netzwerkdruckern der Druckername in UNC-Notation angegeben werden muss, bspw. "\\server01\printer5".

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Druckername

PrintPreviewWithFirstPage

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen bzw. erfragen, wie die Seitenansicht beim Aufruf aussehen soll: als Gesamtansicht über alle Blätter des Diagramms oder als Darstellung der ersten Seite.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Beim Aufruf der Seitenansicht: Darstellung der ersten Seite (True) / Gesamtansicht über alle Blätter des Diagramms (False)

Code-Beispiel VB.NET

```
Dim printer As VcPrinter
printer.Orientation = VcOrientation.vcLandscape
printer.PrintPreviewWithFirstPage = True
printer.FitToPage = False

VcTree1.ShowPrintPreviewDialog()
```

Code-Beispiel C#

```
VcPrinter printer = vcTree1.Printer;
printer.Orientation = VcOrientation.vcLandscape;
printer.PrintPreviewWithFirstPage = true;
printer.FitToPage = false;

vcTree1.ShowPrintPreviewDialog();
```

TitleAndLegendOnAllPages

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob Titel und Legende auf jeder Seite erscheinen sollen (True) oder nicht (False). Außerdem wird hiermit festgelegt, ob die Seitenaufteilung automatisch berechnet wird, so dass die Knoten nicht durchgeschnitten werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Titel und Legende werden auf jeder Seite wiederholt (True) / Titel und Legende werden nur einmal ausgegeben und ggf. beim Seitenumbruch durchtrennt (False) **Standardwert:** False

Code-Beispiel VB.NET

```
VcTree1.Printer.RepeatTitleAndLegend = True
```

Code-Beispiel C#

```
vcTree1.Printer.RepeatTitleAndLegend = true;
```

VcCalendarGrid

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie die absolute Höhe des unteren Seitenrandes in der Maßeinheit cm setzen oder erfragen. Die tatsächliche Breite kann größer sein, wenn der verwendete Drucker nicht randlos drucken kann.

	Datentyp	Beschreibung

Code-Beispiel VB.NET

```
VcTree1.Printer.AbsoluteTopMarginInCM = 1.5   ' 2 cm
```

Code-Beispiel C#

```
vcTree1.Printer.AbsoluteTopMarginInCM = 1.5;   // 1.5 cm
```

ZoomFactorAsDouble

Eigenschaft von VcPrinter

Mit dieser Eigenschaft können Sie den Zoomfaktor in Prozent für die Einstellung **FitToPage = False** setzen oder erfragen (Zoomfaktor = 100: Originalgröße, Zoomfaktor > 100: Vergrößerung, Zoomfaktor < 100: Verkleinerung).

	Datentyp	Beschreibung
Eigenschaftswert	System.Double	Zoomfaktor für das Diagramm

Code-Beispiel VB.NET

```
VcTree1.Printer.ZoomFactor = 150
```

Code-Beispiel C#

```
vcTree1.Printer.ZoomFactor = 150
```

7.30 VcRect

Rect

Ein Objekt vom Typ **VcRect** bezeichnet ein Rechteck-Objekt und kommt nur in VcInPlaceEditorShowing vor.

Eigenschaften

- Bottom
- Height
- Left
- Right
- Top
- Width

Eigenschaften

Bottom

Eigenschaft von VcRect

Diese Eigenschaft gibt die untere Position des Rechteckobjekts an.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Position des unteren Rands des Rechtecks

Height

Nur-Lese-Eigenschaft von VcRect

Diese Eigenschaft gibt die Höhe des Rechteckobjekts an.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Höhe des Rechtecks

Left

Eigenschaft von VcRect

Diese Eigenschaft gibt die linke Position des Rechteckobjekts an.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Position des linken Rands des Rechtecks

Code-Beispiel VB.NET

```
    Private Sub VcTree1_VcInPlaceEditorShowing(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcInPlaceEditorShowingEventArgs) Handles
VcTree1.VcInPlaceEditorShowing
        Dim node As VcNode
        node = e.EditObject
        If e.EditObjectType = VcObjectType.vcObjTypeNodeInTable Then
            e.ReturnStatus = VcReturnStatus.vcRetStatFalse
            Select Case e.FieldIndex
                Case 1  'Name
                    TextBox1.Left = e.FldRectVisible.Left + VcTree1.Left
                    TextBox1.Top = e.FldRectVisible.Top + VcTree1.Top
                    TextBox1.Width = e.FldRectVisible.Width
                    TextBox1.Height = e.FldRectVisible.Height
                    TextBox1.Text = node.DataField(0)
                    TextBox1.Visible = True
                    TextBox1.Focus()
                Case 2, 3   'Start or End
                    DateTimePicker1.Left = e.FldRectVisible.Left + VcTree1.Left
                    DateTimePicker1.Top = e.FldRectVisible.Top + VcTree1.Top
                    DateTimePicker1.Value = node.DataField(0)
                    DateTimePicker1.Visible = True
                    DateTimePicker1.Focus()
                Case 13     'Employee
                    ComboBox1.Left = e.FldRectVisible.Left + VcTree1.Left
                    ComboBox1.Top = e.FldRectVisible.Top + VcTree1.Top
                    ComboBox1.Width = e.FldRectVisible.Width
                    ComboBox1.Height = e.FldRectVisible.Height
                    ComboBox1.Text = node.DataField(0)
                    ComboBox1.Visible = True
                    ComboBox1.Focus()
            End Select
        End If
    End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcInPlaceEditorShowing(object sender,
NETRONIC.XGantt.VcInPlaceEditorShowingEventArgs e)
   {
   VcNode node = (VcNode)e.EditObject;
   if (e.EditObjectType == VcObjectType.vcObjTypeNodeInTable)
      {
      e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
      switch (e.FieldIndex)
         {
         case 1: //Name
           textBox1.Left = e.FldRectVisible.Left + vcTree1.Left;
           textBox1.Top = e.FldRectVisible.Top + vcTree1.Top;
            textBox1.Width = e.FldRectVisible.Width;
            textBox1.Height = e.FldRectVisible.Height;
            textBox1.Text = Convert.ToString(node.get_DataField(0));
            textBox1.Visible = true;
            textBox1.Focus();
            break;
         case 2: //Start or end
           dateTimePicker1.Left = e.FldRectVisible.Left + vcTree1.Left;
           dateTimePicker1.Top = e.FldRectVisible.Top + vcTree1.Top;
           dateTimePicker1.Value = Convert.ToDateTime(node.get_DataField(0));
           dateTimePicker1.Visible = true;
           dateTimePicker1.Focus();
            break;
         case 13: //Employee
           comboBox1.Left = e.FldRectVisible.Left + vcTree1.Left;
            comboBox1.Top = e.FldRectVisible.Top + vcTree1.Top;
            comboBox1.Width = e.FldRectVisible.Width;
            comboBox1.Height = e.FldRectVisible.Height;
            comboBox1.Text = Convert.ToString(node.get_DataField(0));
            comboBox1.Visible = true;
            comboBox1.Focus();
            break;
            }
        }
   }
```

Right

Eigenschaft von VcRect

Diese Eigenschaft gibt die rechte Position des Rechteckobjekts an.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Position des rechten Rands des Rechtecks

Top

Eigenschaft von VcRect

Diese Eigenschaft gibt die obere Position des Rechteckobjekts an.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Position des oberen Rands des Rechtecks

Code-Beispiel VB.NET

```
DateTimePicker1.Top = e.FldRectVisible.Top + VcTree1.Top
```

Code-Beispiel C#

```
dateTimePicker1.Top = e.FldRectVisible.Top + vcTree1.Top;
```

Width

Nur-Lese-Eigenschaft von VcRect

Diese Eigenschaft gibt die Breite des Rechteckobjekts an.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Breite des Rechtecks

Code-Beispiel VB.NET

```
Text1.Width = fldRectVisible.Width
```

Code-Beispiel C#

```
textBox1.Width = e.FldRectVisible.Width;
```

7.31 VcTree

Tree

Ein Objekt vom Typ **VcTree** bezeichnet das VARCHART-XTree-Steuerelement selbst. Über Ereignisse können Sie dessen Interaktionen kontrollieren. Es kann durch eine Vielzahl von Eigenschaften und Methoden den Anforderungen entsprechend konfiguriert werden.

Eigenschaften

- ActiveNodeFilter
- ArrangementDataFieldIndex
- BorderArea
- BoxCollection
- BoxFormatCollection
- CollapseDataFieldIndex
- CtrlCXVProcessingEnabled
- DataTableCollection
- DateOutputFormat
- DiagramBackgroundColor
- DialogFont
- DoubleOutputFormat
- Enabled
- ExtendedDataTablesEnabled
- FilePath
- FilterCollection
- FirstVerticalLevelNumber
- FontAntiAliasingEnabled
- HorizontalNodeDistance
- HorizontalNodeIndentWidth
- InbuiltMouseCursorWhileDraggingEnabled
- InPlaceEditingAllowed
- InteractionMode
- LegendView
- LevelDataFieldIndex
- MapCollection
- MaximumChartRowCount
- MouseProcessingEnabled
- NodeAppearanceCollection
- NodeCollection

- NodeCreationAllowed
- NodeCreationWithDialog
- NodeFormatCollection
- NodesDataTableName
- NodeToolTipTextDataFieldIndex
- ParentNodeIDDataFieldIndex
- PhantomDrawingWhileDraggingEnabled
- Printer
- RoundedLinkSlantsEnabled
- StructureCodeDataFieldIndex
- StructureType
- TextEntrySupplyingEventEnabled
- ToolTipChangeDuration
- ToolTipDuration
- ToolTipPointerDuration
- ToolTipShowAfterClick
- ToolTipTextSupplyingEventEnabled
- TreeViewStyle
- VerticalLevelDistance
- VerticalNodeDistance
- ViewXCoordinate
- ViewYCoordinate
- WaitCursorEnabled
- WorldView
- ZoomFactor
- ZoomingPerMouseWheelAllowed

Methoden

- Arrange
- Clear
- CompleteViewMode
- CopyNodesIntoClipboard
- CutNodesIntoClipboard
- DeleteNodeRecord
- DetectDataTableFieldName
- DetectDataTableName
- DetectFieldIndex
- DumpConfiguration
- EndLoading
- ExportGraphicsToFileEx

- GetAValueFromARGB
- GetBValueFromARGB
- GetGValueFromARGB
- GetNodeByID
- GetRValueFromARGB
- IdentifyFormatField
- IdentifyObjectAt
- ImportConfiguration
- InsertNodeRecord
- InsertNodeRecordEx
- Load
- MakeARGB
- PasteNodesFromClipboard
- PrintEx
- PrintToFile
- Reset
- SaveAsEx
- ScrollToNode
- SetImageResource
- ShowAboutDialog
- ShowExportGraphicsDialog
- ShowNodeEditDialog
- ShowPageSetupDialog
- ShowPrintDialog
- ShowPrinterSetupDialog
- ShowPrintPreviewDialog
- SuspendUpdate
- UpdateNodeRecord
- Zoom
- ZoomOnMarkedNodes

Ereignisse

- KeyDown
- KeyPress
- KeyUp
- VcBoxLeftClicking
- VcBoxLeftDoubleClicking
- VcBoxModified
- VcBoxModifying
- VcBoxRightClicking

- VcDataModified
- VcDataRecordCreated
- VcDataRecordCreating
- VcDataRecordDeleted
- VcDataRecordDeleting
- VcDataRecordModified
- VcDataRecordModifying
- VcDataRecordNotFound
- VcDiagramLeftClicking
- VcDiagramLeftDoubleClicking
- VcDiagramRightClicking
- VcDragCompleting
- VcDragStarting
- VcErrorOccuring
- VcFieldSelecting
- VcHelpRequested
- VcInPlaceEditorShowing
- VcLegendViewClosed
- VcMouseDoubleClicking
- VcMouseDown
- VcMouseMove
- VcMouseUp
- VcNodeCollapsing
- VcNodeCreated
- VcNodeCreating
- VcNodeDeleted
- VcNodeDeleting
- VcNodeExpanding
- VcNodeLeftClicking
- VcNodeLeftDoubleClicking
- VcNodeModified
- VcNodeModifying
- VcNodeRightClicking
- VcNodesMarked
- VcNodesMarking
- VcStatusLineTextShowing
- VcTextEntrySupplying
- VcToolTipTextSupplying
- VcWorldViewClosed
- VcZoomFactorModified

Eigenschaften

ActiveNodeFilter

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den Filter festlegen oder erfragen, der die darzustellenden Knoten selektiert. Es werden dabei immer die ausgefilterten Knoten mitsamt ihrer Teilbäume angezeigt*level*

	Datentyp	Beschreibung
Eigenschaftswert	VcFilter	Filterobjekt **Standardwert:** Nothing

Code-Beispiel VB.NET

```
VcTree1.ActiveNodeFilter = VcTree1.FilterCollection.FilterByName("Milestone")
```

Code-Beispiel C#

```
vcTree1.ActiveNodeFilter = vcTree1.FilterCollection.FilterByName("Milestone");
```

ArrangementDataFieldIndex

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die Art der Anordnung eines Teilbaums synchron in einem Datenfeld halten. Der Inhalt des Datenfelds kann **0** (Teilbaum horizontal angeordnet) oder **1** (Teilbaum vertikal angeordnet) sein. Die horizontale Anordnung ist nur sichtbar, wenn der direkte und alle indirekten Vaterknoten horizontal angeordnet sind. Diese Eigenschaft können Sie auch auf der Eigenschaftenseite **Knoten** festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Definitionsfeldindex oder "-1". Bei "-1" wird die Art der Anordnung nicht in einem Feld abgebildet. **Standardwert:** -1

Code-Beispiel VB.NET

```
Dim subTreeArrangement As Integer

subTreeArrangement = VcTree1.ArrangementDataFieldIndex
```

Code-Beispiel C#

```
int subTreeArrangement = vcTree1.ArrangementDataFieldIndex;
```

BorderArea

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf das BorderArea-Objekt, also auf den Titel- und Legendenbereich.

	Datentyp	Beschreibung
Eigenschaftswert	VcBorderArea	Titel- und Legendenbereich

Code-Beispiel VB.NET

```
Dim borderArea As VcBorderArea

borderArea = VcTree1.BorderArea
```

Code-Beispiel C#

```
VcBorderAreat borderArea = vcTree1.BorderArea;
```

BoxCollection

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf die BoxAuflistung und damit auf die verwendeten Boxen.

	Datentyp	Beschreibung
Eigenschaftswert	VcBoxCollection	BoxCollection-Objekt

Code-Beispiel VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcTree1.BoxCollection
```

Code-Beispiel C#

```
VcBoxCollection boxCltn = vcTree1.BoxCollection;
```

BoxFormatCollection

Nur-Lese-Eigenschaft von VcTree

Mit dieser Eigenschaft haben Sie Zugriff auf die BoxFormat-Auflistung, in der alle zur Verfügung stehenden Box-Formate enthalten sind.

	Datentyp	Beschreibung
Eigenschaftswert	VcBoxFormatCollection	BoxFormatCollection-Objekt

CollapseDataFieldIndex

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den Index eines Datenfeldes setzen, das den Kollabierstatus eines Knotens synchron in einem Datenfeld hält. Der Inhalt des Datenfelds kann **0** (Knoten expandiert) oder **1** (Knoten kollabiert) sein. Der Knoten ist nur sichtbar, wenn der direkte Vaterknoten und jeder indirekte Vaterknoten expandiert sind. Diese Eigenschaft können Sie auch auf der Eigenschaftenseite **Knoten** festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Definitionsfeldindex oder" -1". Bei "-1" wird der Kollabierstatus nicht in einem Feld abgebildet. **Standardwert:** -1

Code-Beispiel VB.NET

```
Dim nodeCollapsed As Integer

nodeCollapsed = VcTree1.CollapseField
```

Code-Beispiel C#

```
int nodeCollapsed = vcTree1.CollapseField;
```

CtrlCXVProcessingEnabled

Eigenschaft von VcTree

Mit dieser Eigenschaft werden die Tastenkombinationen Strg+C, Strg+X und Strg+V automatisch in die Zwischenablage-Operationen **CopyNodesTo-Clipboard**, **CutNodesToClipboard** bzw. **PasteNodesFromClipboard** übersetzt. Dieses Verhalten kann abgeschaltet werden, damit in Visual Basic kein Konflikt mit Bearbeitungsmethoden für Menüpunkte entsteht, die die gleichen Tastaturkombinationen benutzen. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Allgemeines** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Tastenkombinationen werden/werden nicht in Zwischenablage-Operationen übersetzt **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.CtrlCXVProcessing = True
```

Code-Beispiel C#

```
vcTree1.CtrlCXVProcessing = true;
```

DataTableCollection

Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf die DataTable-Auflistung und auf die darin verwendeten Datentabellen.

	Datentyp	Beschreibung
Eigenschaftswert	VcDataTableCollection	Ermitteltes Auflistungsobjekt der Datentabellen

Code-Beispiel VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcTree1.DataTableCollection
For Each dataTable In dataTableCltn
   ListBox1.Items.Add(dataTable.Name)
Next
```

Code-Beispiel C#

```
VcDataTableCollection dataTablecltn = vcTree1.DataTableCollection;
foreach(VcDataTable dataTable in dataTablecltn)
   listBox1.Items.Add(dataTable.Name);
```

DateOutputFormat

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie das Ausgabeformat von Terminen einstellen oder erfragen. Für das Datum stehen folgende Kürzel zur Verfügung:

D: Wochentagsname erster Buchstabe

TD: Wochentagsname (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

DD: Tagesnummer zweistellig: 01-31 (nicht anpassbar)

DDD: die ersten drei Buchstaben des Wochentagsnamens (nicht anpassbar)

M: erster Buchstabe des Monatsnamens (nicht anpassbar)

TM: Monatsname (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

MM: zweistellige Monatsnummer: 01-12

MMM: erste drei Buchstaben des Monatsnamens (nicht anpassbar)

YY: zweistellige Jahreszahl

YYYY: vierstellige Jahreszahl

WW: zweistellige Nummer der Kalenderwoche: 01-53

TW: Text für "Kalenderwoche" (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

Q: einstellige Quartalsnummer: 1-4

TQ: Quartalsname (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

hh: Stunde zweistellig im 24-Stunden-Format: 00-23

HH: Stunde zweistellig im 12-Stunden-Format: 01-12

Th: Text für "o' clock" (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

TH: "am" oder "pm" (kann über das Ereignis **VcTextEntrySupplying** angepasst werden)

mm: Minute zweistellig: 00-59

ss: Sekunde zweistellig: 00-59

TS: kurzes Datumsformat, wie in der Windows-Systemsteuerung über die Regions- und Sprachoptionen definiert

TL: langes Datumsformat, wie in der Windows-Systemsteuerung über die Regions- und Sprachoptionen definiert

TT: Zeitformat, wie in der Windows-Systemsteuerung über die Regions- und Sprachoptionen definiert

Hinweis: Bitte setzen Sie allen Zeichen, die nicht als Datumsbestandteile interpretiert werden sollen, einen Backslash '\' voran. '\\' z.B. ergibt '\' in der Ausgabe. Die Sonderzeichen: **:, /, -** und **Leerzeichen** brauchen nicht durch '\' gekennzeichnet zu werden.

Diese Eigenschaft können Sie auch auf der Eigenschaftenseite **Allgemeines** setzen.

	Datentyp	**Beschreibung**
Parameter: ⇨ dateFormat	System.String	Datumsformat
Eigenschaftswert	Date/Time	Datum

Code-Beispiel VB.NET

```
VcTree1.DateOutputFormat = "DD.MM.YY"
```

Code-Beispiel C#

```
vcTree1.DateOutputFormat = "DD.MM.YY";
```

DiagramBackgroundColor

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die Hintergrundfarbe Ihres Baumdiagramms setzen oder erfragen. Die Standard-Farbe ist weiß (=RGB (255,255,255)). Diese Eigenschaft können Sie auch auf der Eigenschaftenseite **Allgemeines** festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	RGB-Farbwerte ({0...255},{0...255},{0...255})

Code-Beispiel VB.NET

```
VcTree1.DiagramBackgroundColor = RGB(200, 100, 150)
```

Code-Beispiel C#

```
vcTree1.DiagramBackgroundColor = RGB(200, 100, 150);
```

DialogFont

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die Schriftgröße und den Schrifttyp der zur Laufzeit erscheinenden Dialogfelder im VARCHART XTree erfragen oder festlegen. Als Objekt wird ein Fontobjekt Ihrer Programmierumgebung erwartet, z. B. bei Visual Basic ein Objekt der Klasse **StdFont**.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Schriftname

Code-Beispiel VB.NET

```
Dim newFont As Font
newFont = Newfont ("Verdana", 14)
VcTree1.DialogFont = newFont
```

Code-Beispiel C#

```
Font newFont = newFont ("Verdana", 14);
vcTree1.DialogFont = newFont;
```

DoubleOutputFormat

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie das Ausgabeformat von Zahlen als double-Wert im Baum-Diagramm einstellen oder erfragen. Das Format kann über folgende Zeichen dargestellt werden:

- Text
- I
- D

sowie die Trennzeichen Komma und Punkt. Dabei steht **Text** für einen beliebigen Text, **I** für die Ziffern vor dem Dezimaltrenner und D für je eine Ziffer hinter dem Dezimaltrenner. Die erlaubte, generelle Reihenfolge ist **Text I D Text**, wobei Komma und Punkt an beliebiger Stelle eingefügt werden können. Als Beispiel sei die Zahl -284901,3458 gegeben. Über das Format **I,DDDD ppm** wird sie als **-284901,3458 ppm** dargestellt. Über das Format **$I,III.DD** wird sie als **$-284,901.35** dargestellt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Zeichenfolge, die das Double-Format beschreibt, z.B. "I,DDDD ppm"

Code-Beispiel VB.NET

```
VcTree1.DoubleOutputFormat = "I,DDDD ppm"
```

Code-Beispiel C#

```
vcTree1.DoubleOutputFormat = "$I,III.DD";
```

Enabled

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie das VARCHART-XTree-Steuerelement so abschalten, dass es auf Maus- und Tastenbefehle nicht reagiert.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	VARCHART-Windows-Forms-Steuerelement angeschaltet/abgeschaltet

Code-Beispiel VB.NET

```
VcTree1.Enabled = False
```

Code-Beispiel C#

```
vcTree1.Enabled = false;
```

ExtendedDataTablesEnabled

Eigenschaft von VcTree

Mit dieser Eigenschaft könne Sie zwischen der Beschränkung auf zwei Datentabellen (Maindata und Relations) und der weiterentwickelten Verwendung von bis zu 90 Datentabellen wechseln. Die Verwendung der zweiten Option wird empfohlen. Diese Eigenschaft muss zu Beginn des Programms gesetzt werden, bevor Datatabellen und Datensätze angelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	**false**: nur zwei Datentabellen (Maindata und Relations) **true**: bis zu 99 Datentabellen **Standardwert:** false

Code-Beispiel VB.NET

```
VcTree1.ExtendedDataTablesEnabled = True
```

Code-Beispiel C#

```
vcTree1.ExtendedDataTablesEnabled = true;
```

FilePath

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie einen Verzeichnispfad setzen, damit auch Grafikdateien, bei denen nur ein relativer Dateiname angegeben ist, in dem angegebenen Verzeichnis gefunden werden. Andernfalls wird die Datei in dem gerade aktiven Arbeitsverzeichnis der Applikation und im Installationsverzeichnis des VARCHART-ActiveX-Steuerelements gesucht.

Üblicherweise wird diese Eigenschaft beim Start der Applikation während des Initialisierens des VARCHART-ActiveX-Steuerelements gesetzt, der Einfachheit halber wohl zumeist mit dem Verzeichnispfad der Applikation oder einem Unterverzeichnis desselben. Dies hat den Vorteil, dass die Applikation in einem beliebigen Verzeichnis installiert sein kann.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Verzeichnispfad **Standardwert:** ""

Code-Beispiel VB.NET

```
Dim exeName As String
Dim exeDir As String

exeName = System.Environment.GetCommandLineArgs(0)
exeDir = System.IO.Path.GetDirectoryName(exeName)
VcTree1.FilePath = exeDir + "\Bitmaps"
```

Code-Beispiel C#

```
string exeName = Environment.GetCommandLineArgs()[0];
vcTree1.FilePath = System.IO.Path.GetDirectoryName(exeName)+ @"\..\Bitmaps";
```

FilterCollection

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf die Filter-Auflistung und damit auf die verwendeten Filter.

	Datentyp	Beschreibung
Eigenschaftswert	VcFilterCollection	FilterCollection-Objekt

Code-Beispiel VB.NET

```
Dim filterCltn As VcFilterCollection
filterCltn = VcTree1.FilterCollection
```

Code-Beispiel C#

```
VcFilterCollection filterCltn = vcTree1.FilterCollection;
```

FirstVerticalLevelNumber

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie erfragen oder festlegen, ob ab einer bestimmten Ebene die Knoten vertikal angeordnet sind. Bei **-1** ist diese Eigenschaft abgeschaltet. Die Anordnung wird über die Methode **Arrange** durchgeführt. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Layout** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Nummer der Ebene, ab welcher der Teilbaum vertikal angeordnet ist **Standardwert:** -1

Code-Beispiel VB.NET

```
VcTree1.FirstVerticalLevel = 3
VcTree1.Arrange
```

Code-Beispiel C#

```
vcTree1.FirstVerticalLevel = 3;
vcTree1.Arrange;
```

FontAntiAliasingEnabled

Nur-Lese-Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie festlegen oder erfragen, ob Schriftzeichen optisch per GDI+ geglättet werden sollen Wenn dies bei bestimmten Schriftarten, insbesondere bei nichtlateinischen Zeichen, zu verminderter Lesefähigkeit führt, sollte die Eigenschaft auf **False** gesetzt werden.

Die Glättung per GDI+ bewirkt zusätzlich, dass die Texte bei jeder Zoomstufe die gleiche relative Ausdehnung haben, so dass immer dieselbe Anzahl Zeichen z.B. in ein Knotenfeld passt. Wenn diese Eigenschaft aber auf **False** steht, dann wird stattdessen die Einstellung des Betriebssystems übernommen (einstellbar in der **Systemsteuerung**, Dialogfeld **Anzeige**, Reiter **Darstellung: Effekte**). Wenn dort eine Kantenglättung eingeschaltet ist, dann werden also Texte weiterhin geglättet. Es kann dann aber sein, dass bei manchen Zoomstufen mehr Text sichtbar ist als bei anderen, weil die systemeigene Kantenglättung dies nicht garantiert.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Allgemeines** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Textzeichen werden optisch geglättet / nicht geglättet. **Standardwert:** true

HorizontalNodeDistance

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den horizontalen Abstand zwischen zwei horizontal angeordneten Knoten erfragen oder festlegen. Die Einheit beträgt mm. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Layout** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Entfernung (mm) **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.HorizontalNodeDistance = 10
```

Code-Beispiel C#

```
vcTree1.HorizontalNodeDistance = 10;
```

HorizontalNodeIndentWidth

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die horizontale Einrückung vertikal angeordneter Knoten erfragen oder festlegen. Die Einheit beträgt mm. Diese Eigenschaft kann auch über die Eigenschaftenseite **Layout** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Entfernung (mm) **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.HorizontalNodeIndent = 30
```

Code-Beispiel C#

```
vcTree1.HorizontalNodeIndent = 30;
```

InbuiltMouseCursorWhileDraggingEnabled

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft bestimmt, ob während eines OLE-Drag-Vorgangs der Cursor in der Zielkomponente gesetzt werden soll. Bei OLE Drag & Drop ist es möglich, den Cursor in der Quellkomponente über das Ereignis

OLEGiveFeedback zu setzen. Daher würde ein Setzen durch die Zielkomponente zu einem Flimmern der konkurrierenden Cursor führen, was man über diese Eigenschaft beeinflussen kann.

Außerdem können bei eingeschaltetem Cursor und der auf **vcOLEDropManual** gesetzten Eigenschaft Objekte außerhalb der Anlagerungsstellen eines Knotens nicht fallengelassen werden, während dies bei ausgeschaltetem Cursor möglich ist.

Sie können diese Eigenschaft auch auf der Eigenschaftenseite **Allgemeines** setzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Mauszeiger wird/wird nicht in der Zielkomponente gesetzt **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.OLEDragWithOwnMouseCursor = False
```

Code-Beispiel C#

```
vcTree1.OLEDragWithOwnMouseCursor = false;
```

InPlaceEditingAllowed

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie erfragen oder festlegen, ob in Knotenfeldern und in Boxen das direkte Editieren möglich ist. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Allgemeines** eingestellt werden.

Hinweis: Falls einzelne Datenfelder nicht editierbar sein sollen, darf im Dialog **Datentabellen verwalten** das Attribut **editierbar** nicht ausgewählt sein.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Direktes Editieren möglich (True) / nicht möglich (False) **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.InPlaceEditingAllowed = True
```

Code-Beispiel C#

```
vcTree1.InPlaceEditingAllowed = true;
```

InteractionMode

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie einen der verfügbaren Interaktionsmodi einstellen oder erfragen.

	Datentyp	Beschreibung
Eigenschaftswert	VcInteractionMode	Interaktionsmodus **Standardwert:** vcPointer
	Mögliche Werte: .vcCreateNode 2 .vcPointer 0	 Erzeugemodus für Knoten Selektiermodus

Code-Beispiel VB.NET

```
VcTree1.InteractionMode = vcCreateNode
```

Code-Beispiel C#

```
vcTree1.InteractionMode = vcCreateNode;
```

LegendView

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf das LegendView-Objekt, das die Legendenansicht des Diagramms definiert.

	Datentyp	Beschreibung
Eigenschaftswert	VcLegendView	LegendView-Objekt

Code-Beispiel VB.NET

```
Dim legendview As VcLegendView
legendview = VcTree1.LegendView
legendview.Visible = True
```

Code-Beispiel C#

```
VcLegendView legendview = vcTree1.LegendView;
legendview.Visible = true;
```

LevelDataFieldIndex

Eigenschaft von VcTree

Mit dieser Eigenschaft legen Sie ein Datenfeld fest, in dem die Ebenennummer der Knoten abgelegt wird. Die Ebenennummern zählen von 1 an aufwärts. Diese Eigenschaft kann auch über die Eigenschaftenseite **Knoten** festgelegt werden.

Hinweis: Zur Laufzeit können Sie nicht die Ebene des Knotens verändern, indem Sie den Wert des Ebenennummer-Datenfeldes ändern.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Ebenennummer **Standardwert:** -1

Code-Beispiel VB.NET

```
VcTree1.LevelField = 4
```

Code-Beispiel C#

```
vcTree1.LevelField = 4;
```

MapCollection

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf das MapCollection-Objekt, in dem eine bestimmte Menge von Zuordnungstabellen zusammengefasst ist. Die Menge der Zuordnungstabellen wird durch die Methode **VcMapCollection.SelectMaps** definiert.

	Datentyp	Beschreibung
Eigenschaftswert	VcMapCollection	MapCollection-Objekt

Code-Beispiel VB.NET

```
Dim mapCltn As VcMapCollection
mapCltn = VcTree1..MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
```

Code-Beispiel C#

```
VcMapCollection mapCltn = vcTree1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
```

MaximumChartRowCount

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die Höhe einer Baumstruktur erfragen oder festlegen. Diese Eigenschaft können Sie auch auf der Eigenschaftenseite **Layout** setzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Maximale Anzahl der Zeilen der Baumstruktur: {0...255}. Bei 0 ist keine Beschränkung wirksam.

Code-Beispiel VB.NET

```
Dim rowLimit As Integer
rowLimit = VcTree1.RowLimit
```

Code-Beispiel C#

```
int rowLimit = vcTree1.RowLimit;
```

MouseProcessingEnabled

Eigenschaft von VcTree

Diese Eigenschaft kann dazu genutzt werden, Ihre eigene Verarbeitung von Mausereignissen zu ermöglichen. Wenn Sie eine eigene Verarbeitung vom Ereignis **VcMouseDown** bis zum **VcMouseUp** durchführen möchten, setzen Sie die Eigenschaft **MouseProcessingEnabled** für diese Zeit auf False. Dann ignoriert VARCHART XTree bis zum Zurücksetzen auf True alle Mausbewegungen und Klicks.

Diese Eigenschaft kann auch in den OnMouse*-Ereignissen gesetzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Eigenschaft in Kraft (True)/ nicht in Kraft (False) **Standardwert:** True

NodeAppearanceCollection

Nur-Lese-Eigenschaft von VcTree

Mit dieser Eigenschaft haben Sie Zugriff auf das NodeAppearanceCollection-Objekt, in dem alle zur Verfügung stehenden NodeAppearance-Objekte zusammengefasst sind.

	Datentyp	Beschreibung
Eigenschaftswert	VcNodeAppearanceCollection	NodeAppearanceCollection-Objekt

Code-Beispiel VB.NET

```
Dim nodeAppearanceCltn As VcNodeAppearanceCollection
Dim nodeAppearance As VcNodeAppearance
nodeAppearanceCltn = VcTree1.NodeAppearanceCollection
nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance
nodeAppearance.BackgroundColor = Color.LightBlue
```

Code-Beispiel C#

```
VcNodeAppearanceCollection nodeAppearanceCltn =
vcTree1.NodeAppearanceCollection;
VcNodeAppearance nodeAppearance = nodeAppearanceCltn.FirstNodeAppearance();
nodeAppearance.BackgroundColor = Color.LightBlue;
```

NodeCollection

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf das NodeCollection-Objekt, in dem abhängig von **SelectNodes** alle Knoten (vcAll), nur die markierten (vcMarked) oder nur die sichtbaren (vcAllVisible) Knoten zusammengefasst sind.

	Datentyp	Beschreibung
Eigenschaftswert	VcNodeCollection	NodeCollection-Objekt

Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection
nodeCltn = VcTree1.NodeCollection
nodeCltn.SelectNodes(VcSelectionType.vcAll)
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcAll);
```

NodeCreationAllowed

Eigenschaft von VcTree

Mit dieser Eigenschaft wird dem Anwender das Anlegen neuer Knoten erlaubt (True) oder gesperrt (False). Wird diese Eigenschaft auf False gesetzt, kann der **Modus: Knoten erzeugen** nicht eingeschaltet werden. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Allgemeines** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Neue Knoten zugelassen/nicht zugelassen **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.NodeCreationAllowed = False
```

Code-Beispiel C#

```
vcTree1.NodeCreationAllowed = false
```

NodeCreationWithDialog

Eigenschaft von VcTree

Mit dieser Eigenschaft wird festgelegt, ob bei der Erzeugung eines neuen Knotens das Dialogfeld **Vorgänge bearbeiten** erscheinen soll oder nicht. Die Eigenschaft **NodeCreationAllowed** muss auf **True** gesetzt sein, damit überhaupt ein neuer Knoten erzeugt werden kann. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Allgemeines** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	**Vorgänge bearbeiten**-Dialog erscheint (True) / erscheint nicht (False) beim Anlegen

Code-Beispiel VB.NET

```
VcTree1.NodeCreationWithDialog = False
```

Code-Beispiel C#

```
vcTree1.NodeCreationWithDialog = false;
```

NodeFormatCollection

Nur-Lese-Eigenschaft von VcTree

Diese Eigenschaft ermöglicht den Zugriff auf das FormatCollection-Objekt und damit auf die verwendeten Formate.

	Datentyp	Beschreibung
Eigenschaftswert	VcNodeFormatCollection	NodeFormatCollection-Objekt

Code-Beispiel VB.NET

```
Dim formatCtln As VcNodeFormatCollection
formatCtln = VcTree1.NodeFormatCollection
```

Code-Beispiel C#

```
VcNodeFormatCollection formatCtln = vcTree1.NodeFormatCollection;
```

NodesDataTableName

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den Namen der Tabelle setzen oder erfragen, die die Felder für die Darstellung der Knoten zur Verfügung stellt.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name der Datentabelle aus der die Knoten kommen

Code-Beispiel VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecord As VcDataRecord

 'create Node DataTable
dataTable = VcTree1.DataTableCollection.Add("NodeDataTable")
VcTree1.NodesDataTableName = dataTable.Name
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = True
'Load Data
dataTable = VcTree1.DataTableCollection.DataTableByName("NodeDataTable")
dataRecord = dataTable.DataRecordCollection.Add("1;Node One;")
dataRecord = dataTable.DataRecordCollection.Add("2;Node Two;")
VcTree1.EndLoading()
```

Code-Beispiel C#

```
VcDataTable dataTable;
VcDataRecord dataRecord;

//create Node DataTable
dataTable = vcTree1.DataTableCollection.Add("NodeDataTable");
vcTree1.NodesDataTableName = dataTable.Name;
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = true;
//Load Data
dataTable = vcTree1.DataTableCollection.DataTableByName("NodeDataTable");
dataRecord = dataTable.DataRecordCollection.Add("1;Node One;");
dataRecord = dataTable.DataRecordCollection.Add("2;Node Two;");
vcTree1.EndLoading();
```

NodeToolTipTextDataFieldIndex

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie erfragen bzw. festlegen, welches Datenfeld eines Knotens für Tooltips in VMF-Dateien genutzt werden soll. Drückt man im WebViewer die rechte Maustaste, erscheint der zugeordnete Text. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Knoten** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Index des Knoten-Datenfelds für Tooltiptexte **Standardwert:** 4

Code-Beispiel VB.NET

```
VcTree1.NodeToolTipTextDataFieldIndex = 1
```

Code-Beispiel C#

```
vcTree1.NodeToolTipTextDataFieldIndex = 1;
```

ParentNodeIDDataFieldIndex

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den Index eines Datenfeldes setzen oder erfragen, das den Strukturcode des Baumdiagramms mit den IDs des Vaterknoten enthält. Dazu sollte der Strukturtyp auf **vcParentChild** gesetzt worden sein (s. Eigenschaft **StructureType**) .

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Index des Datenfelds

PhantomDrawingWhileDraggingEnabled

Eigenschaft von VcTree

Diese Eigenschaft bestimmt, ob während eines OLE-Drag-Vorgangs ein Phantom erscheinen soll oder nicht. Das Abschalten des Phantoms ist für Anwendungen gedacht, die beim Hineindraggen eines Objekts kein neues Objekt erzeugen, sondern z. B. den Knoten, auf dem dann ein Objekt fallengelassen wird, nur neu attributieren.

Sie können diese Eigenschaft auch auf der Eigenschaftenseite **Allgemeines** setzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Phantom erscheint/erscheint nicht **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.OLEDragWithPhantom = False
```

Code-Beispiel C#

```
vcTree1.OLEDragWithPhantom = false;
```

Printer

Eigenschaft von VcTree

Mit diesem Objekt können Sie die Eigenschaften des aktuell verwendeten Druckers erfragen oder setzen.

	Datentyp	Beschreibung
Eigenschaftswert	VcPrinter	Druckerobjekt

Code-Beispiel VB.NET

```
Dim printerZoomfactor As Integer
Dim printerCuttingMarks As String

printerZoomfactor = VcTree1.Printer.ZoomFactor
printerCuttingMarks = VcTree1.Printer.CuttingMarks
```

Code-Beispiel C#

```
int printerZoomfactor = vcTree1.Printer.ZoomFactor;
bool printerCuttingMarks = vcTree1.Printer.CuttingMarks;
```

RoundedLinkSlantsEnabled

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie einstellen oder erfragen, ob die Schrägen bei Verbindungen als Viertelkreise statt als gerade Linien dargestellt werden sollen. Diese Eigenschaft können Sie auch auf der Eigenschaftenseite **Allgemeines** setzen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Schrägen bei Verbindungen werden/werden nicht als Viertelkreise dargestellt **Standardwert:** false

Code-Beispiel VB.NET

```
VcTree1.RoundedLinkSlantsEnabled = True
```

Code-Beispiel C#

```
vcTree1.RoundedLinkSlants.Enabled = true;
```

StructureCodeDataFieldIndex

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den Index eines Datenfeldes setzen oder erfragen, das den nummerierten Strukturcode des Baumdiagramms enthält. Dazu sollte der Strukturtyp auf **vcHierarchy** gesetzt worden sein (s. Eigenschaft **StructureType**) .

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Index des Datenfelds

StructureType

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den Strukturtyp des Baumdiagramms setzen oder erfragen. Die Struktur kann entweder einer Hierachie aus eigenen Ziffern folgen oder über die ID-Nummern der Vaterknoten definiert werden.

	Datentyp	Beschreibung
Eigenschaftswert	Long	Strukturtyp des Baumdiagramms **Standardwert:** vcHierarchy
	Mögliche Werte: .vcHierarchy 3	Der Struktur-Code folgt einer hierarchischen Ordnung nach dem Muster 1, 1.1, 1.1.1 etc.
	.vcParentChild 2	Der Struktur-Code besteht aus den IDs der Vaterknoten

TextEntrySupplyingEventEnabled

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie das Ereignis **VcTextEntrySupplying** aktivieren bzw. deaktivieren. Mit Hilfe dieses Ereignisses können Sie die Texte aller Kontextmenüs, Dialogfelder, Infoboxen, Fehlermeldungen und Monats- und Tagesnamen, die zur Laufzeit erscheinen, verändern, beispielsweise um sie in unterschiedliche Sprachen zu übersetzen. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Allgemeines** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Eigenschaft in Kraft/nicht in Kraft

Code-Beispiel VB.NET

```
VcTree1.TextEntrySupplyingEventEnabled = True
```

Code-Beispiel C#

```
vcTree1.TextEntrySupplyingEventEnabled = true
```

ToolTipChangeDuration

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die Zeitdauer setzen, die vergeht, bevor das nächste ToolTip-Fenster auf dem Bildschirm erscheint, wenn der Mauszeiger auf das nächste Objekt gesetzt wird. Einheit: Millisekunden. Mit der Ziffer -1 stellen Sie den Standardwert von 98 Millisekunden (für Windows XP) ein.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Zeitdauer in Millisekunden. Maximalwert = 32767 ms **Standardwert:** -1

ToolTipDuration

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die Zeitdauer setzen, während der das ToolTip-Fenster sichtbar bleiben soll (sofern der Mauszeiger innerhalb des umgebenden Rechtecks eines Objektes unbewegt bleibt). Einheit: Millisekunden. Mit der Ziffer -1 stellen Sie den Standardwert von 5.000 Millisekunden ein.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Zeitdauer in Millisekunden. Maximalwert = 32767 ms **Standardwert:** -1

ToolTipPointerDuration

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie die Zeitdauer setzen, während der der Mauszeiger innerhalb des umgebenden Rechtecks eines Objektes unbewegt bleiben muss, damit das ToolTip-Fenster erscheint. Einheit: Millisekunden.

Mit der Ziffer -1 stellen Sie den Standardwert von 480 Millisekunden (für Windows XP) ein.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Zeitdauer in Millisekunden **Standardwert:** -1

ToolTipShowAfterClick

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie einstellen, ob das angezeigte ToolTip-Fenster beim Anklicken des Objektes verschwinden soll (Standard-Verhalten) oder entsprechend seiner eingestellten Zeiten weiterhin angezeigt werden soll.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	ToolTip-Fenster verschwindet (false) oder bleibt (true) **Standardwert:** False

ToolTipTextSupplyingEventEnabled

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie das Ereignis **VcToolTipTextSupplying** aktivieren oder deaktivieren. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Allgemeines** festgelegt werden. Mit Hilfe dieses Ereignisses können Sie die Texte der Tooltips bestimmen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Eigenschaft in Kraft/nicht in Kraft **Standardwert:** False

Code-Beispiel VB.NET

```
VcTree1.ToolTipTextSupplyingEventEnabled = True
```

Code-Beispiel C#

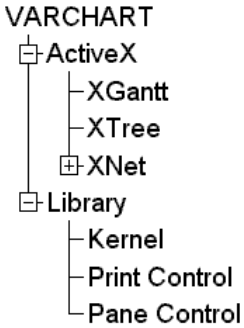
```
vcTree1.ToolTipTextSupplyingEventEnabled = true;
```

TreeViewStyle

Eigenschaft von VcTree

Diese Eigenschaft bestimmt, ob vertikale Ebenen wie in TreeView-Steuerelementen mit **Plus**- oder **Minus**-Zeichen dargestellt werden sollen. Dabei bedeutet ein Plus-Zeichen, dass der darunterliegende Teilbaum kollabiert ist, ein Minus-Zeichen, dass er expandiert ist. Die Zeichen werden nur bei Knoten angezeigt, die Sohnknoten besitzen. Ein Mausklick auf eines der beiden Zeichen überführt den Knoten in den jeweils anderen Kollabierzustand.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Eigenschaft in Kraft/nicht in Kraft



TreeView-Stil: Plussymbol für kollabierten Teilbaum, Minussymbol für expandierten Teilbaum

Code-Beispiel VB.NET

```
VcTree1.TreeViewStyle = True
```

Code-Beispiel C#

```
VcTree1.TreeViewStyle = true;
```

VerticalLevelDistance

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den vertikalen Abstand zwischen zwei horizontal angeordneten Knotenebenen erfragen oder festlegen. Die Einheit beträgt mm. Diese Eigenschaft kann auch über die Eigenschaftenseite **Layout** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Entfernung in Millimeter **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.VerticalLevelDistance = 10
```

Code-Beispiel C#

```
vcTree1.VerticalLevelDistance = 10;
```

VerticalNodeDistance

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie den vertikalen Abstand zwischen zwei vertikal angeordneten Knoten erfragen oder festlegen. Die Einheit beträgt mm. Diese Eigenschaft kann auch über die Eigenschaftenseite **Layout** eingestellt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16	Entfernung in Millimeter **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.VerticalNodeDistance = 10
```

Code-Beispiel C#

```
vcTree1.VerticalNodeDistance = 10;
```

ViewXCoordinate

Eigenschaft von VcTree

Mit dieser Eigenschaft kann der aktuelle Scroll-Offset des angezeigten Ausschnitts in x-Richtung gespeichert werden und bei einem neuen Start der gleichen Anwendung wieder gesetzt werden. Letzteres setzt voraus, dass zuvor auch der Zoomfaktor auf dieselbe Weise gesetzt wird.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Scroll-Offset in x-Richtung

ViewYCoordinate

Eigenschaft von VcTree

Mit dieser Eigenschaft kann der aktuelle Scroll-Offset des angezeigten Ausschnitts in y-Richtung gespeichert werden und bei einem neuen Start der

gleichen Anwendung wieder gesetzt werden. Letzteres setzt voraus, dass zuvor auch der Zoomfaktor auf dieselbe Weise gesetzt wird.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Scroll-Offset in y-Richtung

WaitCursorEnabled

Eigenschaft von VcTree

Mit dieser Eigenschaft kann man steuern, ob bei zeitkritischen Aufrufen (wie ScheduleProject) ein Wartecursor gesetzt werden soll oder nicht.

Die Eigenschaft kann auch auf der Eigenschaftseite **Allgemeines** gesetzt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Wartecursor wird/wird nicht gesetzt **Standardwert:** False

WorldView

Nur-Lese-Eigenschaft von VcTree

Über diese Eigenschaft erhalten Sie Zugriff auf das VcWorldView-Objekt, das die Komplettansicht des Diagramms definiert.

	Datentyp	Beschreibung
Eigenschaftswert	VcWorldView	Komplettansicht-Objekt

Code-Beispiel VB.NET

```
Dim worldview As VcWorldView

worldview = VcTree1.WorldView
worldview.Visible = True
```

Code-Beispiel C#

```
VcWorldView worldview = vcTree1.WorldView;
worldview.Visible = true;
```

ZoomFactor

Eigenschaft von VcTree

Mit dieser Eigenschaft kann der absolute Zoomfaktor der Bildschirmdarstellung in % angegeben oder erfragt werden (Zoomfaktor = 100: Originalgröße, Zoomfaktor > 100: Vergrößerung, Zoomfaktor < 100: Verkleinerung).

	Datentyp	Beschreibung
Eigenschaftswert	System.Int16 {1...10000}	Zoomfaktor

Code-Beispiel VB.NET

```
VcTree1.ZoomFactor = 200
```

Code-Beispiel C#

```
vcTree1.ZoomFactor = 200;
```

ZoomingPerMouseWheelAllowed

Eigenschaft von VcTree

Mit dieser Eigenschaft können Sie setzen oder erfragen, ob das Zoomen per Mausrad zugelassen ist. Um zu zoomen, muss der Anwender dann die Strg-Taste festhalten und das Mausrad drehen.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Zoomen erlaubt (true)/nicht erlaubt (false)

Code-Beispiel VB.NET

```
VcTree1.ZoomingPerMouseWheelAllowed = False
```

Code-Beispiel C#

```
vcTree1.ZoomingPerMouseWheelAllowed = false;
```

Methoden

Arrange

Methode von VcTree

Mit dieser Methode können Sie die Knotenanordnung vornehmen, wie sie über die Eigenschaft **FirstVerticalLevel** festgelegt wurde.

	Datentyp	Beschreibung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
VcTree1.FirstVerticalLevel = 2
VcTree1.Arrange
```

Code-Beispiel C#

```
vcTree1.FirstVerticalLevel = 2;
vcTree1.Arrange;
```

Clear

Methode von VcTree

Diese Methode sollte nur aufgerufen werden, wenn Knoten im Diagramm vorhanden sind. Mit dieser Methode werden alle grafisch darstellbaren Objekte aus dem Diagramm gelöscht und der initiale Zustand der ini-Datei wird wieder hergestellt.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Knoten erfolgreich gelöscht {True}

Code-Beispiel VB.NET

```
VcTree1.Clear
```

Code-Beispiel C#

```
vcTree1.Clear;
```

CompleteViewMode

Methode von VcTree

Mit dieser Methode können Sie die Darstellung so einstellen, dass stets das komplette Diagramm angezeigt wird. Der Zoomfaktor passt sich bei jeder Änderung des Charts automatisch an. Der maximale Zoomfaktor von 100% wird nicht überschritten, die Knoten werden also höchstens in Originalgröße dargestellt. Siehe auch Eigenschaft **ZoomFactor** und Methode **Zoom**.

	Datentyp	Beschreibung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
VcTree1.CompleteViewMode
```

Code-Beispiel C#

```
vcTree1.CompleteViewMode;
```

CopyNodesIntoClipboard

Methode von VcTree

Mit dieser Methode werden die markierten Knoten in den Zwischenspeicher kopiert. Siehe auch die Methoden **CutNodesIntoClipboard** und **PasteNodesFromClipboard**.

	Datentyp	Beschreibung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
VcTree1.CopyNodesIntoClipboard
```

Code-Beispiel C#

```
vcTree1.CopyNodesIntoClipboard;
```

CutNodesIntoClipboard

Methode von VcTree

Mit dieser Methode werden die markierten Knoten ausgeschnitten und in den Zwischenspeicher verschoben. Siehe auch **CopyNodesIntoClipboard** und **PasteNodesFromClipboard**.

	Datentyp	Beschreibung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
VcTree1.CutNodesIntoClipboard
```

Code-Beispiel C#

```
vcTree1.CutNodesIntoClipboard;
```

DeleteNodeRecord

Methode von VcTree

Mit dieser Methode können Sie einen Knoten löschen. Der Knoten wird durch den Primärschlüssel im Datensatz identifiziert. Welches Datenfeld als Identifizierung verwendet wird, wird im Dialog **Datentabellen verwalten** festgelegt.

	Datentyp	Beschreibung
Parameter:		
⇨ nodeRecord	System.Object	Datensatz des Knotens
Rückgabewert	System.Boolean	Datensatz des Knotens erfolgreich/nicht erfolgreich gelöscht

Code-Beispiel VB.NET

```
VcTree1.DeleteNodeRecord "A100;;;;;;"
```

Code-Beispiel C#

```
vcTree1.DeleteNodeRecord "A100;;;;;;";
```

DetectDataTableFieldName

Methode von VcTree

Mit dieser Eigenschaft können Sie über den Index eines Tabellendatenfeldes seinen Namen erfragen.

	Datentyp	Beschreibung
Parameter:		
⇨ fieldIndex	System.Int32	Index des Datentabellenfeldes, dessen Name ermittelt werden soll
Rückgabewert	System.String	Zurückgegebener Name des Datentabellenfeldes

Code-Beispiel VB.NET

```
'Find the name of a DataTableField
Dim fieldName As String

fieldName = VcTree1.DetectDataTableFieldName(0)
```

Code-Beispiel C#

```
//Find the name of a DataTableField
string fieldName = vcTree1.DetectDataTableFieldName(0);
```

DetectDataTableName

Methode von VcTree

Mit dieser Eigenschaft können Sie über den Index einer Datentabelle ihren Namen erfragen.

	Datentyp	Beschreibung
Parameter:		
⇨ fieldIndex	System.Int32	Index der Datentabelle, deren Name ermittelt werden soll
Rückgabewert	System.String	Zurückgegebener Name der Datentabelle

Code-Beispiel VB.NET

```
'Find the name of a DataTable
Dim tableName As String

tableName = VcTree1.DetectDataTableName(0)
```

Code-Beispiel C#

```
//Find the name of a DataTable
string tableName = vcTree1.DetectDataTableName(0);
```

DetectFieldIndex

Methode von VcTree

Mit dieser Eigenschaft können Sie über den Namen einer Datentabelle und den Feldnamen den Index eines Tabellendatenfeldes erfragen.

	Datentyp	Beschreibung
Parameter:		
⇨ dataTableName	System.String	Name der Datentabelle, in der sich das Feld befindet, dessen Index ermittelt werden soll
⇨ dataTableFieldName	System.String	Name des Datentabellenfeldes, dessen Index ermittelt werden soll
Rückgabewert	System.Int32	Zurückgegebener Index des Datentabellenfeldes

Code-Beispiel VB.NET

```
'Find the index of a DataTableField
Dim fieldIndex As Integer

fieldIndex = VcTree1.DetectFieldIndex("Maindata", "Name")
```

Code-Beispiel C#

```
//Find the index of a DataTableField
int fieldIndex = vcTree1.DetectFieldIndex("Maindata", "Name");
```

DumpConfiguration

Methode von VcTree

Mit dieser Methode können Sie die Konfiguration, bestehend aus .INI und .IFD-Datei, speichern.

Die Methode sollte nur zu Diagnosezwecken genutzt werden.

	Datentyp	Beschreibung
Parameter:		
⇨ FileName	System.String	Dateiname, ggf. mit Pfad.
⇨ encoding	VcEncoding	Art der Kodierung
	Mögliche Werte: .vcUnicodeEncoding 2	Wird eine Datei in Unicode-Kodierung gespeichert, ist sie unabhängig von irgendwelchen Einstellungen. Dies Verfahren sollte, wenn möglich, bevorzugt werden. Eine in Unicode-Kodierung gespeicherte Datei erfordert jedoch in Visual Basic 6 eine spezielle Behandlung, wenn sie dort unabhängig vom VARCHART-Steuerelement eingelesen werden soll.
Rückgabewert	System.Boolean	Datei erfolgreich (True)/nicht erfolgreich (False) abgespeichert.

EndLoading

Methode von VcTree

Mit dieser Methode wird das Ende des Ladevorgangs bei **InsertNodeRecord** angezeigt. Dadurch wird eine Aktualisierung der Grafik ausgelöst.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Ladevorgang beendet {True}

Code-Beispiel VB.NET

```
VcTree1.EndLoading()
```

Code-Beispiel C#

```
vcTree1.EndLoading();
```

ExportGraphicsToFileEx

Methode von VcTree

Mit dieser Methode können Sie ein Baum-Diagramm in einer Datei abspeichern, ohne einen **Speichern unter**-Dialog zu erzeugen. Mögliche Formate:

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile oder Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)
- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, ggf. mit eingebautem EMF)

Nur EMF, EMF+, VMF und WMF sind Vektorformate, in denen das Diagramm auflösungsunabhängig gespeichert werden kann. Die übrigen Formate sind pixelorientiert und bieten damit nicht beliebige Auflösungen.

Das VMF-Format wird in der Zukunft nicht mehr weiterentwickelt, aus Kompatibilitätsgründen für bestehende Anwendungen aber zunächst noch weiter unterstützt.

Beim Exportieren in Bitmapgrafikformate kann durch Angabe einer 0 bei der gewünschten Pixelzahl in X- oder Y-Richtung eine verzerrungsfreie Grafik exportiert werden. Sind beide Pixelanzahlen 0, dann wird die Größe der exportierten Grafik in Pixeln von VARCHART XTree wie folgt berechnet:

- PNG: Es wird eine Auflösung von 100 dpi bei einem Zoomfaktor von 100% angenommen. Wird alternativ im Parameter SizeX ein Wert <= -50 angegeben, so wird die absolute Zahl als DPI-Vorgabe genommen. Die DPI-Zahl wird auch in der ausgegebenen PNG-Datei abgelegt, so dass Anzeigeprogramme die richtige Anzeigegröße bei gegebenem Zoomfaktor ermitteln können.

- GIF, TIFF, BMP, JPEG: Es wird eine Auflösung von 100 dpi bei einem Zoomfaktor von 100% angenommen. Wird alternativ im Parameter SizeX ein Wert <= -50 angegeben, so wird die absolute Zahl als DPI-Vorgabe genommen. Es gibt aber zusätzlich eine interne Begrenzung auf 50 MB Größe für die im Speicher für das Exportieren benötigte, nicht komprimierte Ausgangsbitmap, so dass größere Grafiken eine kleinere Auflösung bekommen als gewünscht.

Bei den Vektorgrafikformaten kann keine Pixelanzahl vorgegeben werden, sondern es werden folgende Koordinatenräume benutzt:

- WMF: Es wird eine feste Auflösung angenommen, bei der die größere Ausdehnung die Koordinaten von 0 bis 10.000 benutzt. Die kleinere Ausdehnung benutzt entsprechend einen kleineren Maximalwert für die Koordinaten für eine verzerrungsfreie Darstellung.

- EMF/EMF+: Es wird die volle Auflösung mit Koordinaten in 1/100 mm Abstand in beiden Richtungen X und Y verwendet.

Detaillierte Erläuterungen zu den einzelnen Grafik-Formaten finden Sie im Kapitel: "Wichtige Konzepte: Grafikformate".

	Datentyp	**Beschreibung**
Parameter:		
⇨ fileName	System.String	Dateiname, ggf. mit Pfad.
⇨ printOutputFormat	PrintOutputFormat	Format der abzuspeichernden Datei
	Mögliche Werte:	
	.vcBMP 2	Datei wird im Format BMP geschrieben.
	.vcEMF 9	Datei wird im Format EMF geschrieben.
	.vcEMFPlus 12	Datei wird als *.EMF-Datei geschrieben, beinhaltet aber nur das EMF+-Format.
	.vcEMFWithEMFPlusIncluded 11	Datei wird als *.EMF-Datei geschrieben, beinhaltet aber zusätzlich das EMF+-Format.
	.vcEPS 3	Wird nicht mehr unterstützt.
	.vcGIF 4	Datei wird im Format GIF geschrieben.
	.vcJPG 5	Datei wird im Format JPG geschrieben.
	.vcPCX 6	Wird nicht mehr unterstützt.
	.vcPNG 7	Datei wird im Format PNG geschrieben.
	.vcTIF 8	Datei wird im Format TIF geschrieben.
	.vcVMF 0	Datei wird im Format VMF geschrieben.
	.vcWMF 1	Datei wird im Format WMF geschrieben.
	.vcWMFWithEMFIncluded 10	Datei wird als *.WMF-Datei geschrieben, beinhaltet aber zusätzlich das EMF-Format.
⇨ SizeX	System.Int16	Breite des exportierten Diagramms in Pixeln. Nur bei Pixelformaten möglich. Bei Angabe von 0 wird der Wert unter Beachtung des Seitenverhältnisses berechnet.

⇨ SizeY	System.Int16	Höhe des exportierten Diagramms in Pixeln. Nur bei Pixelformaten möglich. Bei Angabe von 0 wird der Wert unter Beachtung des Seitenverhältnisses berechnet.
Rückgabewert	System.Boolean	Datei erfolgreich (true) / nicht erfolgreich (false) abgespeichert.

Code-Beispiel VB.NET

```
VcTree1.ExportGraphicsToFile "C:\Tmp\test1.vmf", vcVMF,0,0
```

Code-Beispiel C#

```
vcTree1.ExportGraphicsToFile (@"c:\Tmp\test.vmf",
VcPrintOutputFormat.vcVMF,0,0);
```

GetAValueFromARGB

Methode von VcTree

Ein Farbwert setzt sich aus vier Teilen zusammen: A (Alpha), R (Rot), G (Grün) und B (Blau). Der Alpha-Wert 0 bedeutet Volltransparenz und 255 ist ohne Transparenz. Die Farbwerte für R, G und B werden mit höherer Zahl heller, d.h. R,G,B von 0, 0, 0 ist schwarz und 255,255,255 ist weiß. Diese Methode erfragt den Alpha-Wert eines ARGB-Wertes.

	Datentyp	**Beschreibung**
Parameter:		
⇨ argb	System.Int32	ARGB- Wert, aus dem der Alpha-Wert ermittelt werden soll
Rückgabewert	SystemInt.32	Ermittelter Alpha-Wert

Code-Beispiel VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcTree1.MakeARGB(alpha,red,green,blue)
alpha = VcTree1.GetAValueFromARGB(argb)
```

Code-Beispiel C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = vcTree1.MakeARGB(alpha,red,green,blue);
alpha = vcTree1.GetAValueFromARGB(argb);
```

GetBValueFromARGB

Methode von VcTree

Ein Farbwert setzt sich aus vier Teilen zusammen: A (Alpha), R (Rot), G (Grün) und B (Blau). Der Alpha-Wert 0 bedeutet Volltransparenz und 255 ist ohne Transparenz. Die Farbwerte für R, G und B werden mit höherer Zahl heller, d.h. R,G,B von 0, 0, 0 ist schwarz und 255,255,255 ist weiß. Diese Methode erfragt den Blau-Wert eines ARGB-Wertes.

	Datentyp	Beschreibung
Parameter:		
⇨ argb	System.Int32	ARGB- Wert, aus dem der Blau-Wert ermittelt werden soll
Rückgabewert	SystemInt.32	Ermittelter Blau-Wert

Code-Beispiel VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcTree1.MakeARGB(alpha,red,green,blue)
blue = VcTree1.GetBValueFromARGB(argb)
```

Code-Beispiel C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = vcTree1.MakeARGB(alpha,red,green,blue);
blue = vcTree1.GetBValueFromARGB(argb);
```

GetGValueFromARGB

Methode von VcTree

Ein Farbwert setzt sich aus vier Teilen zusammen: A (Alpha), R (Rot), G (Grün) und B (Blau). Der Alpha-Wert 0 bedeutet Volltransparenz und 255 ist ohne Transparenz. Die Farbwerte für R, G und B werden mit höherer Zahl heller, d.h. R,G,B von 0, 0, 0 ist schwarz und 255,255,255 ist weiß. Diese Methode erfragt den Grün-Wert eines ARGB-Wertes.

	Datentyp	Beschreibung
Parameter:		
⇨ argb	System.Int32	ARGB- Wert, aus dem der Grün-Wert ermittelt werden soll
Rückgabewert	SystemInt.32	Ermittelter Grün-Wert

Code-Beispiel VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcTree1.MakeARGB(alpha,red,green,blue)
green = VcTree1.GetRValueFromARGB(argb)
```

Code-Beispiel C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = vcTree1.MakeARGB(alpha,red,green,blue);
green = vcTree1.GetGValueFromARGB(argb);
```

GetNodeByID

Methode von VcTree

Mit dieser Methode können Sie auf einen einzelnen Knoten über seine Identifikation zugreifen, die im Dialog **Datentabellen verwalten** festgelegt wurde. Wenn die Identifikation aus mehreren Feldern besteht (zusammengesetzter Primärschlüssel), muss diese mehrteilige ID folgendermaßen angegeben werden:

ID=ID1|ID2|ID3

	Datentyp	Beschreibung
Parameter: ⇨ nodeID	System.Object	Identifikation des Knotens
Rückgabewert	VcNode	Knoten

Code-Beispiel VB.NET

```
Dim node As VcNode
node = VcTree1.GetNodeByID("10")
```

Code-Beispiel C#

```
VcNode node = vcTree1.GetNodeByID("10");
```

GetRValueFromARGB

Methode von VcTree

Ein Farbwert setzt sich aus vier Teilen zusammen: A (Alpha), R (Rot), G (Grün) und B (Blau). Der Alpha-Wert 0 bedeutet Volltransparenz und 255 ist ohne Transparenz. Die Farbwerte für R, G und B werden mit höherer Zahl heller, d.h. R,G,B von 0, 0, 0 ist schwarz und 255,255,255 ist weiß. Diese Methode erfragt den Rot-Wert eines ARGB-Wertes.

	Datentyp	Beschreibung
Parameter: ⇨ argb	System.Int32	ARGB- Wert, aus dem der Rot-Wert ermittelt werden soll
Rückgabewert	SystemInt.32	Ermittelter Rot-Wert

Code-Beispiel VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcTree1.MakeARGB(alpha,red,green,blue)
red = VcTree1.GetRValueFromARGB(argb)
```

Code-Beispiel C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = vcTree1.MakeARGB(alpha,red,green,blue);
red = vcTree1.GetRValueFromARGB(argb);
```

IdentifyFormatField

Methode von VcTree

Mit dieser Methode können Sie das aktuell verwendete Format des angegebenen Knotens sowie den Index des an der bezeichneten Position befindlichen Formatfeldes erfragen. Falls sich an der bezeichneten Position ein Feld befindet, wird **True** zurückgegeben, andernfalls **False**.

.

	Datentyp	Beschreibung
Parameter:		
⇨ x	System.Int32	X-Koordinate der Position
⇨ y	System.Int32	Y-Koordinate der Position
⇨ node	VcNode	Referenzknoten
⇦ format	VcNodeFormat	Identifiziertes Format
⇦ formatFieldIndex	System.Int16	Format-Feldindex
Rückgabewert	System.Boolean	Ein Formatfeld befindet sich/befindet sich nicht an der angegebenen Position

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftClicking(ByVal sender As System.Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftClicking
   Dim foundFlag As Boolean
   Dim format As VcNodeFormat
   Dim formatFieldIndex As Integer
   foundFlag = VcTree1.IdentifyFormatField(e.X, e.Y, e.Node, format,
formatFieldIndex)
   If foundFlag Then
      MsgBox("You hit the field with the index " + CStr(formatFieldIndex))
   End If
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftClicking(object sender, VcNodeClickingEventArgs
e)
{
   bool foundFlag;
   VcNodeFormat format = null;
   short formatFieldIndex = new short();
   foundFlag = vcTree1.IdentifyFormatField(e.X, e.Y, e.Node, ref format, ref
formatFieldIndex);
   if (foundFlag)
      MessageBox.Show("You hit the field with the index " +
formatFieldIndex.ToString());
}
```

IdentifyObjectAt

Methode von VcTree

Mit dieser Methode können Sie das Objekt, das sich an einer bestimmten Position des Diagramms befindet, identifizieren. Der Typ des Objekts wird zurückgegeben. Derzeit können nur Knoten identifiziert werden.

	Datentyp	**Beschreibung**
Parameter:		
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers
⇦ identifiedObject	System.Object	Erkanntes Objekt
⇦ identifiedObjectType	VcObjectType	Typ des erkannten Objekts
	Mögliche Werte: .vcObjTypeBox 15 .vcObjTypeNode 2 .vcObjTypeNone 0	 Objekttyp **Box** Objekttyp **Knoten** kein Objekt
Rückgabewert	System.Boolean	Objekt identifiziert/Objekt nicht identifiziert

Code-Beispiel VB.NET

```
Private Sub VcTree1_MouseMove(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles VcTree1.MouseMove

      Dim identifiedObject As Object = Nothing
      Dim identifiedObjectType As VcObjectType = VcObjectType.vcObjTypeNone
      Dim node As VcNode = Nothing
      Dim identifiedLayer As VcLayer = Nothing

      VcTree1.IdentifyObjectAt(e.X, e.Y, identifiedObject, identifiedObjectType)

      Select Case identifiedObjectType
         Case VcObjectType.vcObjTypeNodeInDiagram
            node = identifiedObject

            VcTree1.IdentifyLayerAt(e.X, e.Y, node, identifiedLayer)

            If identifiedLayer IsNot Nothing Then
               Label1.Text = "X = " & e.X & "   Y = " & e.Y & vbCrLf & _
                             "Node ID = " & node.DataField(0) & vbCrLf & _
                             "Layer Name = " & identifiedLayer.Name
            End If

         Case Else
            Label1.Text = ""
      End Select

   End Sub
```

Code-Beispiel C#

```
private void VcTree1_MouseMove(object sender, MouseEventArgs e)
      {
         object identifiedObject = null;
         VcObjectType identifiedObjectType = VcObjectType.vcObjTypeNone;
         VcNode node = null;
         VcLayer identifiedLayer = null;

         VcTree1.IdentifyObjectAt(e.X, e.Y, ref identifiedObject, ref
identifiedObjectType);

         switch (identifiedObjectType)
         {
            case VcObjectType.vcObjTypeNodeInDiagram:
               {
                  node = (VcNode)identifiedObject;

                  VcTree1.IdentifyLayerAt(e.X, e.Y,  node, ref identifiedLayer);

                  if (identifiedLayer != null)
                     label1.Text = "X = " + e.X + "   Y = " + e.Y +
                                   "\nNode ID = " + node.get_DataField(0) +
                                   "\nLayer Name = " + identifiedLayer.Name;
                  break;
               }
            default:
               {
                  label1.Text = "";
                  break;
               }
         }
      }
```

ImportConfiguration

Methode von VcTree

Mit dieser Methode können Sie eine Konfigurationsdatei (*.*ini*) laden, aus der alle relevanten Einstellungen übernommen werden. Das schließt eine zugehörige (ggf. andere) Datenschnittstelle (*.*ifd*) ein.

Als Konfigurationsdatei können Sie eine lokale Datei mit Pfad oder eine URL angeben.

Hinweis: Beim Einlesen einer neuen Konfigurationsdatei gehen die bestehenden Daten verloren und müssen ggf. wieder eingelesen werden.

	Datentyp	Beschreibung

Code-Beispiel VB.NET

```
VcTree1.ImportConfiguration ( "c:\VARCHART\XNet\sample.ini")
'or
VcTree1.ImportConfiguration
("http://members.tripod.de/netronic_te/xtree_sample.ini)
```

Code-Beispiel C#

```
vcTree1.ImportConfiguration (@"c:\VARCHART\XTree\sample.ini");
// or
vcTree1.ImportConfiguration
(@"http://members.tripod.de/netronic_te/xtree_sample.ini");
```

InsertNodeRecord

Methode von VcTree

Mit dieser Methode werden die Daten eines Knotens geladen. Die Daten werden als CSV-String (mit Semikolon als Trennzeichen) gemäß der auf der Eigenschaftenseite **Datendefinition** festgelegten Struktur übergeben. **EndLoading** sollte am Ende des kompletten Ladevorgangs einmal aufgerufen werden.

	Datentyp	Beschreibung
Parameter: ⇨ nodeRecord	data field/string	Datensatz des Knotens
Rückgabewert	VcNode	Knoten

Code-Beispiel VB.NET

```
' data format: "Number;Name;Start date;Finish date;Group code;Group name"
VcTree1.InsertNodeRecord("A100;Activity 1;12.09.14;17.09.14;5;Planning")
VcTree1.InsertNodeRecord("A105;Activity 5;13.09.14;18.09.14;7;Testing")
```

Code-Beispiel C#

```
//data format: "Number;Name;Start date;Finish date;Group code;Group name"
vcTree1.InsertNodeRecord("A100;Activity 1;12.09.14;17.09.14;5;Planning");
vcTree1.InsertNodeRecord("A105;Activity 5;13.09.14;18.09.14;7;Testing");
```

InsertNodeRecordEx

Methode von VcTree

Mit dieser Methode können Sie Knoten unter Angabe einer Anlagerungsposition und eines Referenzknotens, an den der neue Knoten angelagert werden soll, einfügen. Die Daten werden dabei als CSV-String (mit Trennzeichen Semikolon) gemäß der auf der Eigenschaftenseite **Datendefinition** festgelegten Struktur übergeben. Die Methode **EndLoading** sollte am Ende des kompletten Ladevorgangs einmal aufgerufen werden.

	Datentyp	Beschreibung
Parameter:		
⇨ nodeRecord	System.Object	Datensatz des Knotens
⇨ position	VcInsertionPosition	Mögliche Anlagerungspositionen
⇨ referenceNode	VcNode	Referenzknoten
Rückgabewert	VcNode	Knoten

Code-Beispiel VB.NET

```
Dim node1 As VcNode
node1 = VcTree1.InsertNodeRecordEx("A2;Node 1;12.09.14;17.09.14;5;Planning",
VcInsertionPosition.vcIPFirstChild, VcTree1.GetNodeByID("A1"))
VcTree1.EndLoading()
```

Code-Beispiel C#

```
VcNode node1 = vcTree1.InsertNodeRecordEx("A2;Node
1;12.09.14;17.09.14;5;Planning", VcInsertionPosition.vcIPFirstChild,
vcTree1.GetNodeByID("A1"));
vcTree1.EndLoading();
```

Load

Methode von VcTree

Mit dieser Methode werden die Daten aus der angegebenen Datei geladen. In der Datei müssen die Daten, wie auf der Eigenschaftenseite **Datendefinition**

festgelegt, im CSV-Format (mit Semikolon als Trennzeichen) gespeichert sein.

	Datentyp	Beschreibung
Parameter:		
⇨ fileName	System.String	Dateiname
Rückgabewert	System.Boolean {True}	Ohne Bedeutung

Code-Beispiel VB.NET

```
VcTree1.Open "C:\Data\project1.wbs"
```

Code-Beispiel C#

```
vcTree1.Open "C:\Data\project1.wbs"
```

MakeARGB

Methode von VcTree

Mit dieser Methode können Sie aus den vier Einzelwerten einer Farbe einen ARGB-Wert bilden.

	Datentyp	Beschreibung
Parameter:		
⇨ alpha	SystemInt.32	Alpha- Wert
⇨ red	SystemInt.32	Rot- Wert
⇨ green	SystemInt.32	Grün- Wert
⇨ blue	SystemInt.32	Blau- Wert
Rückgabewert	System.Int32	Zurückgegebener ARGB- Wert

Code-Beispiel VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = FF
red = A0
green = 34
blue = AB
argb = VcTree1.MakeARGB(alpha,red,green,blue)
```

Code-Beispiel C#

```
long argb;
int alpha = FF;
int red = A0;
int green = 34;
int blue = AB;
argb = vcTree1.MakeARGB(alpha,red,green,blue);
```

PasteNodesFromClipboard

Methode von VcTree

Mit dieser Methode fügen Sie Knoten aus der Zwischenablage an einer bestimmten Position in das Chart ein.

	Datentyp	Beschreibung
Parameter:		
⇨ node	VcNode	Referenzknoten
⇨ pastePosition	VcPastePosition	Position, an der die Knoten aus der Zwischenablage eingefügt werden.
Rückgabewert	Void	

Code-Beispiel VB.NET

```
Dim nodeCltn As VcNodeCollection
nodeCltn = VcTree1.NodeCollection
nodecollection.SelectNodes(VcSelectionType.vcMarked)
If nodecollection.Count = 1 Then
VcTree1.PasteNodesFromClipboard(nodecollection.FirstNode,
VcPastePosition.vcPasteAsLastChild)
End If
```

Code-Beispiel C#

```
VcNodeCollection nodeCltn = vcTree1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcMarked);
if (nodeCltn.Count == 1)
vcTree1.PasteNodesFromClipboard(nodeCltn.FirstNode(),
VcPastePosition.vcPasteAsLastChild);
```

PrintEx

Methode von VcTree

Mit dieser Methode können Sie das Diagramm direkt ausdrucken, ohne dass zuvor ein Dialogfeld erscheint. Der Rückgabewert soll bei nicht erfolgreichem Druck Aufschluss über die Ursache liefern. Dies kann z.B. auch ein Eintrag in einer Logdatei sein.

	Datentyp	Beschreibung
Rückgabewert	VcPrintResultStatus	**Mögliche Werte:**

Name	Parameterposition	Beschreibung
vcPrintingSucceeded	0	Druck verlief erfolgreich
vcNoPrinterInstalled	1	Es wurde weder der über VcPrinter.PrinterName angegebene noch ein im Windows-Betriebssystem als Standarddrucker gekennzeichneter Drucker gefunden.
vcPrintingAbortedByUser	2	Der Druck wurde durch den Anwender abgebrochen.
vcPrintingAbortedByDriver	3	Der Druck wurde durch den Windows-Druckertreiber abgebrochen.
vcUnprintablePageLayout	4	Es konnte nicht gedruckt werden, weil das Seitenlayout zusammen mit den Druckereigenschaften wie Papiergröße und Ränder zu einem nicht druckbaren Layout führten.

Code-Beispiel C#

```
VcPrintResultStatus status = VcTree1.PrintDirectEx();
if (status != VcPrintResultStatus.vcPrintingSucceeded)
   System.Diagnosis.Trace.WriteLine("Printing failed: "+status.ToString);
```

PrintToFile

Methode von VcTree

Mit dieser Methode können Sie das Diagramm direkt in eine Datei drucken. Ob dies gelingt, hängt, da viele PDF-Druckertreiber keine Dateinamen akzeptieren, vom Druckertreiber ab.

	Datentyp	Beschreibung
Parameter:		
⇨ fileName	System.String	Name der Datei
Rückgabewert	Void	

Reset

Methode von VcTree

Mit dieser Methode wird je nach eingestelltem Wert von resetAction der komplette Inhalt sämtlicher Datentabellen gelöscht bzw. der zur Entwurfszeit auf den Eigenschaftenseiten eingestellte Zustand wiederhergestellt

	Datentyp	Beschreibung
Parameter:		
⇨ resetAction	VcResetAction	Objekte, die gelöscht oder neu initialisiert werden
	Mögliche Werte:	
	.vcEmptyAllDataTables 4	Der komplette Inhalt sämtlicher Datentabellen wird gelöscht, die Datentabellen selbst bleiben bestehen.
	.vcReloadConfiguration 2	Komplette Neuinitialisierung mit der INI-Datei. Alle Einstellungen und erzeugten Objekte verfallen.
Rückgabewert	System.Boolean	Objekte im Diagramm erfolgreich gelöscht {True}

Code-Beispiel VB.NET

```
VcTree1.Reset(VcResetAction.vcReloadConfiguration)
```

Code-Beispiel C#

```
vcTree1.Reset(VcResetAction.vcReloadConfiguration);
```

SaveAsEx

Methode von VcTree

Mit dieser Methode werden die aktuellen Daten in die angegebene Datei im CSV-Format gespeichert. Dabei wird die auf der Eigenschaftenseite **Objekte** unter **Datentabellen** festgelegte Struktur verwendet. Datentabellen, die keine Datensätze enthalten, werden nicht gespeichert. Ist kein Name angegeben, wird die zuletzt bei **Open** angegebene Datei überschrieben (entspricht der üblichen **Save**-Funktion).

	Datentyp	Beschreibung
Parameter:		
⇨ fileName	System.String	Dateiname
⇨ encoding	VcEncoding	Art der Kodierung
	Mögliche Werte:	

	.vcUnicodeEncoding 2	Wird eine Datei in Unicode-Kodierung gespeichert, ist sie unabhängig von irgendwelchen Einstellungen. Dies Verfahren sollte, wenn möglich, bevorzugt werden. Eine in Unicode-Kodierung gespeicherte Datei erfordert jedoch in Visual Basic 6 eine spezielle Behandlung, wenn sie dort unabhängig vom VARCHART-Steuerelement eingelesen werden soll.
Rückgabewert	System.Boolean	Speicherung erfolgreich/nicht erfolgreich erfolgt

Code-Beispiel VB.NET

```
VcTree1.SaveAs "C:\Data\project1.wbs"

' or

VcTree1.SaveAs ""
```

Code-Beispiel C#

```
vcTree1.SaveAs "C:\Data\project1.wbs"

' or

vcTree1.SaveAs ""
```

ScrollToNode

Methode von VcTree

Mit dieser Methode können Sie zu der Position eines bestimmten Knotens scrollen.

	Datentyp	**Beschreibung**
Parameter:		
⇨ node	VcNode	Knoten, zu dem gescrollt werden soll
Rückgabewert	System.Boolean	Scrollen erfolgreich/nicht erfolgreich durchgeführt

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftClicking(ByVal sender As System.Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftClicking
   'scroll the diagram so that the node is completely on screen
   VcTree1.ScrollToNode(e.Node)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftClicking(object sender, VcNodeClickingEventArgs
e)
{
   //scroll the diagram so that the node is completely on screen
   vcTree1.ScrollToNode(e.Node);
}
```

SetImageResource

Methode von VcTree

Diese Methode ordnet zur Laufzeit einem angegebenen Namen ein in der Anwendung vorhandenes Image-Objekt zu. Dies ist eine Alternative zur bisherigen Funktionalität, bei der in den XTree-Eigenschaftenseiten angegebene Image-Namen immer dazu führen, dass ein Image-Objekt aus der angesprochenen Datei gelesen wird. Die Methode sollte zu Beginn einer Anwendung, z.B. in der Methode **Form_Load**, ausgeführt werden. Die Image-Namen sind frei wählbar. Zur Unterscheidung von Dateinamen können hier auch Zeichen verwendet werden, die sonst für Dateinamen verboten sind, z.B. das Sternchen (*) . Alle Image-Objekte sind erlaubt: Bitmaps (Formate BMP, JPG, GIF, PNG, TIFF) und Metafiles (Formate WMF, EMF). Wenn der Parameter **image** auf null gesetzt ist, werden vorherige Zuordnungen aufgehoben.

Beispiel: Man fügt einer Anwendung eine Image- oder Datei-Ressource hinzu (im Visual Studio unter den Projekteigenschaften: Ressourcen/Ressource hinzufügen/Vorhandene Datei hinzufügen) und setzt dann in Form_Load beispielsweise folgenden Code ein:

Für Bitmap-Ressourcen:

*vcTree1.SetImageResource("*PlusImage",*
<namespace>.Properties.Resources.plusImage);

für Metafile-Ressourcen:

*vcTree1.SetImageResource("*MinusImage", new Metafile(new MemoryStream(<namespace>.Properties.Resources.minusImage)));*

	Datentyp	Beschreibung
Parameter:		
⇨ imageName	System.String	Name, der dem Image-Objekt zugeordnet ist
Rückgabewert	System.Drawing.Image	Image-Objekt

ShowAboutDialog

Methode von VcTree

Mit dieser Methode können Sie die **About**-Box aufrufen. Sie enthält eine Übersicht über die jeweils verwendeten Programm- und Bibliotheksdateien mit absolutem Pfad und Versionsnummer. Dies dient der vereinfachten Hotline-Unterstützung. Die Übersicht kann mit der Maus selektiert und mit Strg+C herauskopiert werden, um sie z. B. mit Strg+V in eine Mail einzufügen.

	Datentyp	Beschreibung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
VcTree1.ShowAboutDialog
```

Code-Beispiel C#

```
vcTree1.ShowAboutDialog;
```

ShowExportGraphicsDialog

Methode von VcTree

Mit dieser Methode können Sie das Export-Dialogfeld **Speichern unter** aufrufen, um die Darstellung abzuspeichern. Verfügbare Formate sind:

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile oder Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)
- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, ggf. mit eingebautem EMF)

Nur EMF, EMF+, VMF und WMF sind Vektorformate, in denen das Diagramm auflösungsunabhängig gespeichert werden kann. Die übrigen Formate sind pixelorientiert und bieten damit nicht beliebige Auflösungen.

Das VMF-Format wird in der Zukunft nicht mehr weiterentwickelt, aus Kompatibilitätsgründen für bestehende Anwendungen aber zunächst noch weiter unterstützt.

Detaillierte Erläuterungen zu den einzelnen Grafik-Formaten lesen Sie bitte im Kapitel: **Wichtige Konzepte: Grafikformate**.

Beim Exportieren wird die Größe des exportierten Diagramms in Pixeln wie folgt berechnet:

- PNG: Es wird eine Auflösung von 100 dpi bei einem Zoomfaktor von 100% angenommen. Wird alternativ im Parameter SizeX ein Wert <= -50 angegeben, so wird die absolute Zahl als DPI-Vorgabe genommen.

- GIF, TIFF, BMP, JPEG: Es wird eine Auflösung von 100 dpi bei einem Zoomfaktor von 100% angenommen. Wird alternativ im Parameter SizeX ein Wert <= -50 angegeben, so wird die absolute Zahl als DPI-Vorgabe genommen. Es gibt aber zusätzlich eine interne Begrenzung auf 50 MB Größe für die im Speicher für das Exportieren benötigte, nicht komprimierte Ausgangsbitmap, so dass größere Grafiken eine kleinere Auflösung bekommen als gewünscht.

- WMF: Es wird eine feste Auflösung angenommen, bei der die größere Ausdehnung die Koordinaten von 0 bis 10.000 benutzt. Die kleinere Ausdehnung benutzt entsprechend einen kleineren Maximalwert für die Koordinaten für eine verzerrungsfreie Darstellung.

- EMF/EMF+: Es wird die volle Auflösung mit Koordinaten in 1/100 mm Abstand verwendet.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Grafik erfolgreich (true) /nicht erfolgreich (false) exportiert

Code-Beispiel VB.NET

```
VcTree1.ShowExportGraphicsDialog()
```

Code-Beispiel C#

```
vcTree1.ShowExportGraphicsDialog();
```

ShowNodeEditDialog

Methode von VcTree

Mit dieser Methode wird der Dialog **Vorgänge bearbeiten** für den angegebenen Knoten aufgerufen.

	Datentyp	Beschreibung
Parameter:		
⇨ node	VcNode	Knoten, dessen Daten editiert werden sollen
Rückgabewert	System.Boolean	Knotendaten wurden editiert/Editierung abgebrochen

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftClicking
   VcTree1.ShowNodeEditDialog(node)
End Sub
```

Code-Beispiel C#

```
private void vcvcTree1_VcNodeLeftClicking(object sender,
NETRONIC.XTree.VcNodeClickingEventArgs e)
   {
   vcTree1.ShowNodeEditDialog(e.Node);
   }
```

ShowPageSetupDialog

Methode von VcTree

Mit dieser Methode wird der Dialog **Seite einrichten** aufgerufen.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Ohne Bedeutung {True}

Code-Beispiel VB.NET

```
VcTree1.ShowPageSetupDialog()
```

Code-Beispiel C#

```
vcTree1.ShowPageSetupDialog();
```

ShowPrintDialog

Methode von VcTree

Mit dieser Methode wird der Ausdruck der Grafik ausgelöst. Es werden die unter **ShowPageLayoutDialog** aktuell eingestellten Parameter verwendet.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Ohne Bedeutung {True}

Code-Beispiel VB.NET

```
VcTree1.ShowPrintDialog()
```

Code-Beispiel C#

```
vcTree1.ShowPrintDialog();
```

ShowPrinterSetupDialog

Methode von VcTree

Mit dieser Methode wird das Windows-Dialogfeld **Drucker einrichten** aufgerufen.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Ohne Bedeutung {True}

Code-Beispiel VB.NET

```
VcTree1.ShowPrinterSetupDialog()
```

Code-Beispiel C#

```
vcTree1.ShowPrinterSetupDialog();
```

ShowPrintPreviewDialog

Methode von VcTree

Mit dieser Methode wird die Druckvorschau aufgerufen.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	Ohne Bedeutung

Code-Beispiel VB.NET

```
VcTree1.ShowPrintPreviewDialog()
```

Code-Beispiel C#

```
vcTree1.ShowPrintPreviewDialog();
```

SuspendUpdate

Methode von VcTree

Bei größeren Datenmengen kann es unter Umständen zu lange dauern, wenn man bei einer großen Anzahl von Knoten dieselbe Aktion durchführt. Dies kann man mit Hilfe der Methode **SuspendUpdate** beschleunigen. Klammern Sie den Code für die wiederholte Aktion wie im Code-Beispiel durch **SuspendUpdate (True)** und **SuspendUpdate (False)** ein. Dann wird das Update nicht für jeden Knoten einzeln, sondern für alle gemeinsam durchgeführt, wodurch die Performance erhöht wird.

	Datentyp	Beschreibung
Rückgabewert	System.Boolean	SuspendUpdate(True): Beginn der SuspendUpdate Methode/ SuspendUpdate(False): Ende der SuspendUpdate Methode

Code-Beispiel VB.NET

```
VcTree1.SuspendUpdate (True)

   If updateFlag Then
      For Each node In nodeCltn
         If node.DataField(2) < "07.09.14" Then
            node.DataField(13) = "X"
            node.Update
            counter = counter + 1
         End If
      Next node
   Else
      For Each node In nodeCltn
         If node.DataField(2) < "07.09.14" Then
            node.DataField(13) = ""
            node.Update
            counter = counter + 1
         End If
      Next node
   End If

VcTree1.SuspendUpdate (False)
```

Code-Beispiel C#

```
bool updateFlag = true;
VcNodeCollection nodeCltn = vcvcTree1.NodeCollection;
int counter = 0;

vcvcTree1.SuspendUpdate(true);
if (updateFlag == true)
   {
   foreach (VcNode node in nodeCltn)
      {
      if (DateTime.Compare(Convert.ToDateTime(node.get_DataField(2)),
Convert.ToDateTime("12.09.14")) < 0)
         {
         node.set_DataField(13,"X");
         node.Update();
         counter = counter + 1;
         }
      }
   }
   else
   {
   foreach(VcNode node in nodeCltn)
      {
      if (DateTime.Compare(Convert.ToDateTime(node.get_DataField(2)),
Convert.ToDateTime("12.09.14")) < 0)
         {
         node.set_DataField(13,"");
         node.Update();
         counter = counter + 1;
         }
      }
   }
vcTree1.SuspendUpdate(false);
```

UpdateNodeRecord

Methode von VcTree

Mit dieser Methode können die Daten eines bestehenden Datensatzes eines Knotens verändert werden. Der Datensatz wird durch die auf der Eigenschaftenseite **Datendefinition** festgelegte ID identifiziert. Diese Methode kommt zur Anwendung, wenn externe Änderungen in der Grafik auf dem Bildschirm nachvollzogen werden sollen.

	Datentyp	Beschreibung
Parameter: ⇨ nodeRecord	System.Object	Datensatz des Knotens
Rückgabewert	VcNode	Aktualisierter Knoten

Code-Beispiel VB.NET

```
VcTree1.UpdateNodeRecord ("A100;Activity 1;12.09.14;18.09.14;6;Planning")
```

Code-Beispiel C#

```
vcTree1.UpdateNodeRecord ("A100;Activity 1;12.09.14;18.09.14;6;Planning");
```

Zoom

Methode von VcTree

Mit dieser Methode kann die Bildschirmdarstellung um den angegebenen Prozent-Faktor vergrößert (Zoomfaktor > 100) oder verkleinert (Zoomfaktor < 100) werden.

	Datentyp	Beschreibung
Parameter: ⇨ zoomFactor	System.Int16 {1...1000}	Zoomfaktor {11...999}, anderer Werte bleiben unberücksichtigt
Rückgabewert	System.Boolean	Zoomen erfolgreich durchgeführt {True}

Code-Beispiel VB.NET

```
VcTree1.Zoom(120)
```

Code-Beispiel C#

```
vcTree1.Zoom(120)
```

ZoomOnMarkedNodes

Methode von VcTree

Mit dieser Methode können Sie auf die markierten Knoten zoomen.

	Datentyp	Beschreibung
Rückgabewert	Void	

Code-Beispiel VB.NET

```
VcTree1.ZoomOnMarkedNodes
```

Code-Beispiel C#

```
vcTree1.ZoomOnMarkedNodes,
```

Ereignisse

KeyDown

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender eine Taste drückt, während VARCHART XTree den Fokus hat. Mit Hilfe der Key-Ereignisse können Sie mit Hilfe der Tastatur Funktionen des VARCHART ActiveX auslösen. (Zum Interpretieren von ANSI-Zeichen verwenden Sie das KeyPress-Ereignis.)

	Datentyp	Beschreibung
Eigenschaften:		
⇨ keyCode	System.Int16	Tasten-Code wie vbKeyF1 (F1-Taste) oder vbKeyHome (POS1-Taste)
⇨ shift	System.Int16	Eine Ganzzahl, die dem Zustand der Tasten Umschalt, Strg und Alt zu dem Zeitpunkt entspricht, an dem das Ereignis aufgetreten ist. Das Argument shift ist ein Bitfeld, bei dem die niederstwertigen Bits der Umschalt-Taste (Bit 0), der Strg-Taste (Bit 1) und der Alt-Taste (Bit 2) entsprechen. Diese Bits entsprechen jeweils dem Wert 1, 2 und 4. Einige, alle oder keine dieser Bits können gesetzt werden, um anzuzeigen, dass einige, alle oder keine der Tasten gedrückt sind. Wenn zum Beispiel sowohl Strg als auch Alt gedrückt werden, hat shift den Wert 6.

Code-Beispiel VB.NET

```
Private Sub VcTree1_KeyDown(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyEventArgs) Handles VcTree1.KeyDown
   MsgBox("key pressed")
End Sub
```

Code-Beispiel C#

```
private void vcTree1_KeyDown(object sender, System.Windows.Forms.KeyEventArgs e)
   {
   MessageBox.Show("key pressed");
   }
```

KeyPress

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender eine ANSI-Taste drückt und wieder loslässt, während VARCHART XTree den Fokus hat. Mit Hilfe der Key-Ereignisse können Sie mit Hilfe der Tastatur Funktionen des VARCHART ActiveX auslösen.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ keyAscii	System.Int16	Eine Ganzzahl, die den numerischen Tasten-Code einer Standard-ANSI-Taste zurückgibt. KeyAscii wird als Referenz übergeben. Wird das Argument geändert, wird ein anderes Zeichen an das Objekt gesendet. Das Ändern von KeyAscii auf 0 hebt den Tastenanschlag auf, d.h. das Objekt erhält kein Zeichen.

Code-Beispiel VB.NET

```
Private Sub VcTree1_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles PrivatDummy1.KeyPress
    MsgBox ("Key pressed and released.")
End Sub
```

Code-Beispiel C#

```
private void vcTree1_KeyPress(object sender,
System.Windows.Forms.KeyPressEventArgs e)
   {
   MessageBox.Show("key pressed and released");
   }
```

KeyUp

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender eine Taste loslässt, während VARCHART XTree den Fokus hat. Mit Hilfe der Key-Ereignisse können Sie mit Hilfe der Tastatur Funktionen des VARCHART ActiveX auslösen. (Zum Interpretieren von ANSI-Zeichen verwenden Sie das KeyPress-Ereignis.)

	Datentyp	Beschreibung
Eigenschaften:		
⇨ keyCode	System.Int16	Tasten-Code wie vbKeyF1 (F1-Taste) oder vbKeyHome (POS1-Taste)
⇨ shift	System.Int16	Eine Ganzzahl, die dem Zustand der Tasten Umschalt, Strg und Alt zu dem Zeitpunkt entspricht, an dem das Ereignis aufgetreten ist. Das Argument shift ist ein Bitfeld, bei dem die niederstwertigen Bits der Umschalt-Taste (Bit 0), der Strg-Taste (Bit 1) und der Alt-Taste (Bit 2) entsprechen. Diese Bits entsprechen jeweils dem Wert 1, 2 und 4. Einige, alle oder keine dieser Bits können gesetzt werden, um anzuzeigen, dass einige, alle oder keine der Tasten gedrückt sind. Wenn zum Beispiel sowohl Strg als auch Alt gedrückt werden, hat shift den Wert 6.

Code-Beispiel VB.NET

```
Private Sub VcTree1_KeyUp(KeyCode As Integer, Shift As Integer)
    MsgBox "key released"
End Sub
```

Code-Beispiel C#

```
private void vcTree1_KeyUp(object sender, System.Windows.Forms.KeyEventArgs e)
   {
   MessageBox.Show("key released");
   }
```

VcBoxLeftClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit der linken Maustaste auf eine Box klickt. Das getroffene VcBox-Objekt wird zusammen mit der Position des Mauszeigers (x,y-Koordinaten) als Parameter zurückgegeben.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ box	VcBox	Getroffene Box
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcBoxLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.Xtree.VcBoxClickingEventArgs) Handles VcTree1.VcBoxLeftClicking
   TextBox1.Text = e.Box.FieldText(1)
End Sub
```

Code-Beispiel C#

```
private void vcvcTree1_VcBoxLeftClicking(object sender,
NETRONIC.XTree.VcBoxClickingEventArgs e)
   {
   textBox1.Text = e.Box.get_FieldText(1);
   }
```

VcBoxLeftDoubleClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit der linken Maustaste auf eine Box doppelklickt. Das getroffene VcBox-Objekt wird zusammen mit der Position (x,y-Koordinaten) des Mauszeigers als Parameter zurückgegeben.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ box	VcBox	Getroffene Box
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcBoxLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcBoxClickingEventArgs) Handles VcTree1.VcBoxLeftDoubleClicking
   e.Box.FieldText(0) = TextBox1.Text
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcBoxLeftDoubleClicking(object sender,
NETRONIC.XTree..VcBoxClickingEventArgs e)
   {
   e.Box.set_FieldText(1, textBox1.Text);
   }
```

VcBoxModified

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn die Modifizierung der Box abgeschlossen ist. Das veränderte VcBox-Objekt und der Modifikationstyp werden als Parameter mitgegeben.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcBoxModifiedEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcBoxModifiedEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ box	VcBox	Veränderte Box

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcBoxModified(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcBoxModifiedEventArgs) Handles VcTree1.VcBoxModified
   MsgBox("The box has been modified")
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcBoxModified(object sender,
VcTreeLib.VcBoxModifiedEventArgs e)
   {
   MessageBox.Show("The box has been modified");
   }
```

VcBoxModifying

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn eine Box interaktiv verändert wurde. Das veränderte VcBox-Objekt und der Modifikationstyp werden als Parameter zurückgegeben.

Die mit diesem Ereignis übermittelten Daten dürfen nur gelesen, aber nicht verändert werden. Um sie zu verändern, verwenden Sie bitte das Ereignis **VcBoxModified**.

Durch Setzen des Rückgabestatus auf **vcRetStatFalse** kann die Änderung verhindert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcBoxModifyingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcBoxModifyingEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ box	VcBox	Veränderte Box

⇨ modificationType	VcBoxModificationTypes	Art der Veränderung
	Mögliche Werte:	
	.vcBMTAnything 1	beliebige Veränderung
	.vcBMTNothing 0	keine Veränderung
	.vcBMTTextModified 4	Text der Box verändert
	.vcBMTXYOffsetModified 2	Offset verändert
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte:	
	.vcRetStatDefault 2	Das Default-Verhalten wird nicht verändert.
	.vcRetStatFalse 0	Das Default-Verhalten wird nicht durchgeführt.
	.vcRetStatNoPopup 4	Das Erscheinen des Kontextmenüs wird unterdrückt.
	.vcRetStatOK 1	Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcBoxModifying(ByVal box As VcTreeLib.VcBox, _
                  ByVal modificationType As _
                  NETRONIC.XTree.VcBoxModificationTypes, _
                  returnStatus As Variant)

Select Case modificationType
    Case vcBMTAnything: MsgBox "Box modification"
    Case vcBMTXYOffsetModified: MsgBox "Offset changed"
    Case vcBMTTextModified: MsgBox "Box field text changed"
End Select

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcBoxModifying(obje!ct sender,
NETRONIC.XTree.VcBoxModifyingEventArgs e)
   {
   switch(e.ModificationType)
      {
      case VcBoxModificationTypes.vcBMTAnything:
         MessageBox.Show("Box modification");
         break;
      case VcBoxModificationTypes.vcBMTXYOffsetModified:
         MessageBox.Show("Offset changed");
         break;
      case VcBoxModificationTypes.vcBMTTextModified:
          MessageBox.Show("Box field text changed");
         break;
       }
   }
```

VcBoxRightClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit der rechten Maustaste auf eine Box klickt. Das getroffene Boxobjekt wird zusammen mit der Position (x,y-Koordinaten) des Mauszeigers als Parameter zurückgegeben. Sie können so an der entsprechenden Position Ihr gewünschtes Kontextmenü anzeigen.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ box	VcBox	Getroffene Box
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcBoxRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcBoxClickingEventArgs) Handles VcTree1.VcBoxRightClicking
        PopupMenu.Show(VcTree1, New Point(e.X, e.Y))
    End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcBoxRightClicking(object sender,
NETRONIC.XTree.VcBoxClickingEventArgs e)
   {
   PopupMenu.Show(vcTree1, new Point (e.X, e.Y));
   }
```

VcDataModified

Ereignis von VcTree

Dieses Ereignis tritt immer dann auf, wenn Daten interaktiv im Chart verändert werden, also explizit nach den folgenden Ereignissen:

- VcBoxModified
- VcNodeCreated
- VcNodeDeleting
- VcNodeModified

Mit diesem Ereignis ist es möglich, sich im Anwenderprogramm eine Marke zu setzen, die daran erinnert, dass beim Beenden des Programms die Daten noch gespeichert werden müssen.

	Datentyp	Beschreibung
Eigenschaften:		
⇦ (no parameter)		Kein Parameter

VcDataRecordCreated

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn das interaktive Anlegen eines Objektes beendet ist, das einen Datensatz erzeugt. Das DataRecord-Objekt, die Form des Anlegens (hier nur **vcDataRecordCreated** und **vcDataRecordCreated-ByResourceScheduling**) und die Information, ob der angelegte Datensatz der einzige Datensatz oder der letzte einer Menge ist (derzeit immer **True**), werden als Parameter zurückgegeben, so dass eine Datenvalidierung vorgenommen werden kann.

Wenn eine Verbindung oder ein Knoten angelegt wurde, können Sie zudem auf das analoge Verbindungs- oder Knoten-Ereignis reagieren und hier vor der Weiterverarbeitung zusätzlich grafische Daten prüfen (s. **VcNode-Created** und **VcLinkCreated**).

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDataRecordCreatedEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDataRecordCreatedEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ dataRecord	VcDataRecord	Angelegtes Datensatz-Objekt
⇨ creationType	VcCreationType	Typ des Anlegens von Datensätzen
	Mögliche Werte: .vcDataRecordCreated 6 .vcNodeCreated 1 .vcNodesCloned 4	 Datensatz wurde durch Interaktion angelegt Knoten durch "Stempeln" angelegt Selektierte Knoten wurden durch Interaktion kopiert
⇨ isLast	System.Boolean	**True**: Angelegter Datensatz ist der einzige oder der letzte Datensatz einer Menge **False**: Angelegter Datensatz ist nicht der einzige oder der letzte Datensatz einer Menge

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcDataRecordCreated(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcDataRecordCreatedEventArgs) Handles VcTree1.VcDataRecordCreated
        MsgBox(e.DataRecord.AllData)
    End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcDataRecordCreated(object sender,
NETRONIC.XTree.VcDataRecordCreatedEventArgs e)
   {
   MessageBox.Show(e.DataRecord.AllData.ToString());
   }
```

VcDataRecordCreating

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender interaktiv ein Objekt erzeugt hat, das einen Datensatz generiert. Das neu erzeugte Datensatz-Objekt wird als Parameter zurückgegeben, so dass eine Validierung und ggf. ein Datenbankeintrag vorgenommen werden kann.

Die mit diesem Ereignis übermittelten Daten dürfen nur gelesen, aber nicht verändert werden. Um sie zu verändern, verwenden Sie bitte das Ereignis **VcDataRecordCreated**.

Durch Setzen des Rückgabestatus kann die Anlage verhindert werden.

Wenn eine Verbindung oder ein Knoten angelegt wurde, können Sie zudem auf das analoge Verbindungs- oder Knoten-Ereignis reagieren und hier vor der Weiterverarbeitung zusätzlich grafische Daten prüfen (s. **VcNodeCreating** und **VcLinkCreating**).

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDataRecordCreatingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDataRecordCreatingEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ dataRecord	VcDataRecord	Angelegtes Datensatz-Objekt
⇔ returnStatus	VcReturnStatus	Rückgabestatus

Mögliche Werte:	
.vcRetStatFalse 0	Der Datensatz wird nicht angelegt.
.vcRetStatOK 1	Der Datensatz wird angelegt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcDataRecordCreated(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcDataRecordCreatedEventArgs) Handles VcTree1.VcDataRecordCreated
        MsgBox(e.DataRecord.AllData)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcDataRecordCreated(object sender,
NETRONIC.XTree.VcDataRecordCreatedEventArgs e)
{
MessageBox.Show(e.DataRecord.AllData.ToString());
}
```

VcDataRecordDeleted

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn das Löschen eines Objektes, das auf einem Datensatz-Objekt basiert, beendet ist. Der Datensatz und die Information, ob der betroffene Datensatz der einzige oder der letzte Datensatz einer Menge ist, werden als Parameter zurückgegeben, so dass eine Datenvalidierung vorgenommen werden kann.

Wenn eine Verbindung oder ein Knoten gelöscht wurde, können Sie zudem auf das analoge Verbindungs- oder Knoten-Ereignis reagieren und hier vor der Weiterverarbeitung zusätzlich grafische Daten prüfen (s. **VcNodeDeleted** und **VcLinkDeleted**).

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDataRecordDeletedEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDataRecordDeletedEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ dataRecord	VcDataRecord	Gelöschter Datensatz

⇨ isLast	System.Boolean	**True**: Gelöschter Datensatz ist der einzige oder der letzte Datensatz einer Menge **False**: Gelöschter Datensatz ist nicht der einzige oder der letzte Datensatz einer Menge

VcDataRecordDeleting

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit Hilfe des Kontextmenüs ein Objekt löscht, das auf einem Datensatz-Objekt basiert. Der betroffene Datensatz wird als Parameter zurückgegeben, so dass Sie z. B. noch eine Überprüfung vornehmen und bei negativem Ergebnis dieser Prüfung die Löschung ggf. durch Setzen des Rückgabestatus verhindern können.

	Datentyp	**Beschreibung**
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDataRecordDeletingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDataRecordDeletingEventArgs-Objektes

	Datentyp	**Beschreibung**
Eigenschaften:		
⇨ dataRecord	VcDataRecord	Gelöschtes Datensatz-Objekt
⇦ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatFalse 0 .vcRetStatOK 1	 Der Datensatz wird nicht gelöscht. Der Datensatz wird gelöscht.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcDataRecordDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcDataRecordDeletingEventArgs) Handles
VcTree1.VcDataRecordDeleting
   'deny deletion of data record with a certain value
   If e.DataRecord.DataField(0) = "1" Then
      e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcDataRecordDeleting(object sender,
NETRONIC.XTree.VcDataRecordDeletingEventArgs e)
   {
   // deny deletion of data record with a certain value
   if (e.DataRecord.get_DataField(0).Equals("1"))
      e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
   }
```

VcDataRecordModified

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn die Modifizierung des Datensatzes abgeschlossen ist.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDataRecordModifiedEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDataRecordModifiedEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ dataRecord	VcDataRecord	Veränderter Datensatz

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcDataRecordModified(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcDataRecordModifiedEventArgs) Handles
VcTree1.VcDataRecordModified
   MsgBox("The data record has been modified")
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcDataRecordModified(object sender,
NETRONIC.XTree.VcDataRecordModifiedEventArgs e)
   {
   MessageBox.Show("The data record has been modified");
   }
```

VcDataRecordModifying

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn ein Objekt, dem ein Datensatz zu Grunde liegt, interaktiv verändert wurde. Das veränderte VcDataRecord-Objekt und der Modifikationstyp werden als Parameter zurückgegeben.

Die mit diesem Ereignis übermittelten Daten dürfen nur gelesen, aber nicht verändert werden. Um sie zu verändern, verwenden Sie bitte das Ereignis **VcDataRecordModified**.

Durch Setzen des Rückgabestatus kann die Änderung verhindert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDataRecordModifyingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDataRecordModifyingEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ dataRecord	VcDataRecord	Veränderter Datensatz
⇨ modificationType	VcModificationTypes	Art der Veränderung
	Mögliche Werte: .vcAnything 1 .vcMoved 8 .vcNothing 0	 Änderungstyp nicht näher bestimmt Objekt wurde verschoben. Keine Änderung
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatFalse 0 .vcRetStatOK 1	 Die Veränderung wird rückgängig gemacht. Die Veränderung wird durchgeführt.

VcDataRecordNotFound

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn ein abhängiger Datensatz nicht gefunden wurde. Der Index des Feldes im aktuellen Datensatz, in dem der Schlüsselwert des abhängigen Datensatzes steht, wird zurückgegeben und bietet so Informationen über den nicht gefundenen Datensatz.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	
⇨ e	VcDataRecordNotFoundEventArgs	

	Datentyp	Beschreibung
Eigenschaften:		
⇦ index	System.Int32	Index des Feldes, das den Schlüssel des abhängigen Datensatzes enthält

VcDiagramLeftClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit der linken Maustaste im Diagrammbereich klickt und dabei kein Objekt trifft. Die Position (x,y-Koordinaten des Mauszeigers) wird als Parameter zurückgegeben.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcDiagramLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcDiagramClickingEventArgs) Handles VcTree1.VcDiagramLeftClicking
   MsgBox("x: " + e.X.ToString() + " y: " + e.Y.ToString())
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcDiagramLeftClicking(object sender,
NETRONIC.XTree.VcDiagramClickingEventArgs e)
   {
   MessageBox.Show("x: " + e.X.ToString() + " y: " + e.Y.ToString());
   }
```

VcDiagramLeftDoubleClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit der linken Maustaste im Diagrammbereich doppelklickt und dabei kein Objekt trifft. Die Position (x,y-Koordinaten des Mauszeigers) wird als Parameter zurückgegeben.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcDiagramLeftDoubleClicking(ByVal sender As Object, ByVal e
As NETRONIC.XTree.VcDiagramClickingEventArgs) Handles
VcTree1.VcDiagramLeftDoubleClicking
   VcTree1.Zoom(90)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcDiagramLeftDoubleClicking(object sender,
NETRONIC.XTree.VcDiagramClickingEventArgs e)
   {
   vcTree1.Zoom(90);
   }
```

VcDiagramRightClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit der rechten Maustaste in den Diagrammbereich klickt und dabei weder Stichwertlinie noch Knoten trifft. Die Position (x,y-Koordinaten des Mauszeigers) wird als Parameter übergeben. Sie können so an der entsprechenden Position Ihr gewünschtes Kontextmenü anzeigen. Das integrierte Kontextmenü kann durch Setzen des Rückgabestatus auf **vcRetStatNoPopup** unterdrückt werden.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ x	System.Int32	Y-Koordinate des Mauszeigers

⇨ y	System.Int32	X-Koordinate des Mauszeigers
⇦ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte:	
	.vcRetStatDefault 2	Das Default-Verhalten wird nicht verändert.
	.vcRetStatFalse 0	Das Default-Verhalten wird nicht durchgeführt.
	.vcRetStatNoPopup 4	Das Erscheinen des Kontextmenüs wird unterdrückt.
	.vcRetStatOK 1	Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcDiagramRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcDiagramClickingEventArgs) Handles
VcTree1.VcDiagramRightClicking
        PopupMenu.Show(dummyobject1, New Point(e.X, e.Y))
        e.ReturnStatus = VcTreeLib.VcReturnStatus.vcRetStatNoPopup
    End Sub
    returnStatus = vcRetStatNoPopup

End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcDiagramRightClicking(object sender,
NETRONIC.XTree.VcDiagramClickingEventArgs e)
   {
   PopupMenu.Show(vcTree1, new Point (e.X, e.Y));
   e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
   }
```

VcDragCompleting

Ereignis von VcTree

Dieses Ereignis wird zum Abschluss eines Drag-Vorgangs bei der Quellkomponente ausgelöst und gibt den Drop-Effekt bekannt.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDragCompletingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDragCompletingEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ DropEffect	System.Windows.Forms.DragDropEffects	Auswirkungen einer Drag & Drop-Operation

VcDragStarting

Ereignis von VcTree

Mit diesem Ereignis ist es möglich, die erlaubten DropEffects beim Start eines Drag-Vorgangs zu bestimmen und damit ggf. einzuschränken. Gleichzeitig muss die Eigenschaft **LeavingControlWhileDraggingAllowed** auf **True** gesetzt sein. Die Eigenschaft ist mit dem kombinierten Wert **DragDropEffects.Copy Or DragDropEffects.Move** vorbesetzt. Wenn z.B. Knoten beim Herausziehen aus dem Steuerelement in jedem Fall kopiert und niemals verschoben werden sollen, dann setzt man die Eigenschaft auf **DragDropEffects.Copy**.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcDragStartingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcDragStartingEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ allowedEffects	System.Windows.Forms.AllowedEffects	Erlaubte DropEffects

VcErrorOccuring

Ereignis von VcTree

Dieses Ereignis tritt nur dann auf, wenn ein unvorhergesehener Fehler im Code des VARCHART XTree entdeckt wird. NETRONIC ist bemüht, jeden dieser Fehler zu beseitigen. Um sie auf einfache Art beim Kunden, z. B. in einer Datei, protokollieren zu können, werden sie nun über dieses Ereignis nach außen bekanntgegeben. Das Parameterprofil ist durch den ActiveX-Standard vorgegeben. Dadurch sind die übergebenen Parameter teilweise konstant. Die Nummer sollte im Ereignis immer gegengeprüft werden, um bei zukünftigen Erweiterungen nicht alle Fehlerarten pauschal abzublocken.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ description	System.String	Fehlerbeschreibungstext
⇨ scode	System.Int32	&h800a402f (konstant)

⇨ source	System.String	Name des Controls (konstant)
⇨ helpFile	System.String	Hilfedatei: "" (konstant)
⇨ helpContext	System.Int32	Hilfekontext: 0 (konstant)
⇦ cancelDisplay	System.Boolean	Beim Wert True wird kein normaler Fehler mit der Nummer 71 mehr ausgegeben, der über das 'On Error GoTo'-Konstrukt abfangbar wäre.

Code-Beispiel VB.NET

```
Private Sub VcTree1_Error(Number As Integer, Description As String, _
           Scode As Long, Source As String, HelpFile As String, HelpContext _
           As Long, CancelDisplay As Boolean)

   Debug.Print CStr(Number) + " " + Description

End Sub
```

Code-Beispiel C#

```
Private Sub vcTree1_Error(Number As Integer, Description As String, _
           Scode As Long, Source As String, HelpFile As String, HelpContext _
           As Long, CancelDisplay As Boolean)

   Debug.Print CStr(Number) + " " + Description

End Sub
```

VcFieldSelecting

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn ein Feld einer Box selektiert wird. Die Selektion kann durch Setzen des Rückgabestatus verhindert werden.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcFieldSelectingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcFieldSelectingEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ editObject	VcObject	Editiertes Objekt
⇨ editObjectType	VcObjectType	Objekttyp
	Mögliche Werte: .vcObjTypeBox 15 .vcObjTypeNode 2	 Objekttyp **Box** Objekttyp **Knoten**

	.vcObjTypeNone 0	kein Objekt
⇨ fieldIndex	System.Int32	Feldindex
⇨ objRectComplete	VcRect	Komplettes Rechteck des getroffenen Objekts
⇨ objRectVisible	VcRect	sichtbares Rechteck des getroffenen Objekts
⇨ fldRectComplete	VcRect	Komplettes Rechteck des getroffenen Feldes
⇨ fldRectVisible	VcRect	sichtbares Rechteck des getroffenen Feldes
returnStatus	VcReturnStatus	
	Mögliche Werte:	
	.vcRetStatFalse 0	Das Feld wird nicht ausgewählt.
	.vcRetStatOK 1	Das Feld wird ausgewählt.

VcHelpRequested

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender in einem zur Laufzeit angezeigten Dialog die **F1**-Taste drückt. Die Applikation erhält damit die Möglichkeit, ihr eigenes Hilfesystem aufzurufen, um dialog- und anwendungsbezogene Hilfe anbieten zu können.

	Datentyp	**Beschreibung**
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcHelpRequestedEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcHelpRequestedEventArgs-Objektes

	Datentyp	**Beschreibung**
Eigenschaften:		
⇨ DialogType	VcDialogType	Dialog, für den Hilfe angefordert wurde
	Mögliche Werte:	
	.vcEditDataRecordDialog 5400	Hilfe wurde für den **Datensatz bearbeiten** Dialog angefordert
	.vcPageSetupDialog 4097	Hilfe wurde für den **Seite einrichten** Dialog angefordert
	.vcPrintPreviewDialog 4096	Hilfe wurde für den **Druckvorschau** Dialog angefordert

VcInPlaceEditorShowing

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der im Programm implementierte Editor gestartet wird.

Das Ereignis wird erst aktiviert, wenn die Eigenschaft **InPlaceEditing-Allowed** auf True gesetzt ist.

Wenn Sie den Rückgabestatus auf **False** setzen, können Sie den Start des integrierten Editors verhindern und an den übergebenen Koordinaten ein eigener Editor hochbringen.

	Datentyp	**Beschreibung**
Eigenschaften:		
⇨ editObject	VcObject	Editiertes Objekt
⇨ editObjectType	VcObjectType	Objekttyp
	Mögliche Werte: .vcObjTypeBox 15 .vcObjTypeNode 2 .vcObjTypeNone 0	 Objekttyp **Box** Objekttyp **Knoten** kein Objekt
⇨ fieldIndex	System.Int32	Feldindex
⇨ objRectComplete	VcRect	Komplettes Rechteck des getroffenen Objekts
⇨ objRectVisible	VcRect	Sichtbares Rechteck des getroffenen Objekts
⇨ fldRectComplete	VcRect	Komplettes Rechteck des getroffenen Feldes
⇨ fldRectVisible	VcRect	Sichtbares Rechteck des getroffenen Feldes
returnStatus	VcReturnStatus	
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcInPlaceEditorShowing(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcInPlaceEditorShowingEventArgs) Handles
VcTree1.VcInPlaceEditorShowing

        Dim node As VcNode
        node = e.EditObject
        If e.EditObjectType = VcObjectType.vcObjTypeNodeInTable Then
            e.ReturnStatus = VcReturnStatus.vcRetStatFalse
            Select Case e.FieldIndex
                Case 1  'Name
                    TextBox1.Left = e.FldRectVisible.Left + VcTree1.Left
                    TextBox1.Top = e.FldRectVisible.Top + VcTree1.Top
                    TextBox1.Width = e.FldRectVisible.Width
                    TextBox1.Height = e.FldRectVisible.Height
                    TextBox1.Text = node.DataField(0)
                    TextBox1.Visible = True
                    TextBox1.Focus()
                Case 2, 3   'Start or End
                    DateTimePicker1.Left = e.FldRectVisible.Left + VcTree1.Left
                    DateTimePicker1.Top = e.FldRectVisible.Top + VcTree1.Top
                    DateTimePicker1.Value = node.DataField(0)
                    DateTimePicker1.Visible = True
                    DateTimePicker1.Focus()
                Case 13     'Employee
                    ComboBox1.Left = e.FldRectVisible.Left + VcTree1.Left
                    ComboBox1.Top = e.FldRectVisible.Top + VcTree1.Top
                    ComboBox1.Width = e.FldRectVisible.Width
                    ComboBox1.Height = e.FldRectVisible.Height
                    ComboBox1.Text = node.DataField(0)
                    ComboBox1.Visible = True
                    ComboBox1.Focus()
            End Select
        End If
```

Code-Beispiel C#

```
private voidvcTree1_VcInPlaceEditorShowing(object sender,
NETRONIC.XTree.VcInPlaceEditorShowingEventArgs e)
   {
   VcNode node = (VcNode)e.EditObject;
   if (e.EditObjectType == VcObjectType.vcObjTypeNodeInTable)
         e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
      switch (e.FieldIndex)
          {
          case 1: //Name
            textBox1.Left = e.FldRectVisible.Left + vcTree1.Left;
            textBox1.Top = e.FldRectVisible.Top + vcTree1.Top;
            textBox1.Width = e.FldRectVisible.Width;
            textBox1.Height = e.FldRectVisible.Height;
            textBox1.Text = Convert.ToString(node.get_DataField(0));
            textBox1.Visible = true;
            textBox1.Focus();
            break;
         case 2: //Start or end
            dateTimePicker1.Left = e.FldRectVisible.Left + vcTree1.Left;
            dateTimePicker1.Top = e.FldRectVisible.Top + vcTree1.Top;
            dateTimePicker1.Value = Convert.ToDateTime(node.get_DataField(0));
            dateTimePicker1.Visible = true;
            dateTimePicker1.Focus();
            break;
         case 13: //Employee
            comboBox1.Left = e.FldRectVisible.Left + vcTree1.Left;
            comboBox1.Top = e.FldRectVisible.Top + vcTree1.Top;
            comboBox1.Width = e.FldRectVisible.Width;
            comboBox1.Height = e.FldRectVisible.Height;
            comboBox1.Text = Convert.ToString(node.get_DataField(0));
            comboBox1.Visible = true;
            comboBox1.Focus();
            break;
        }
   }
```

VcLegendViewClosed

Ereignis von VcTree

Dieses Ereignis wird aufgerufen, wenn das Popup-Fenster der Legendenansicht geschlossen wird.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	
⇨ e	VcLEgendViewClosedEventArgs	

	Datentyp	Beschreibung
Eigenschaften:		
⇨ (no parameter)		

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcLegendViewClosed(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcLegendViewClosedEventArgs) Handles VcTree1.VcLegendViewClosed
   MsgBox("Do you want to close the legend view window?", MsgBoxStyle.OKCancel)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcLegendViewClosed(object sender,
NETRONIC.XTree.VcLegendViewClosedEventArgs e)
   {
   DialogResult retVal = MessageBox.Show("Do you want to close the legend view
window?", "Closing legend view window", MessageBoxButtons.OKCancel);
   }
```

VcMouseDoubleClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender eine Maustaste zweimal drückt.

Bitte beachten Sie auch die Eigenschaft **MouseProcessingEnabled**. Wenn Sie den Status der Sondertasten **Shift**, **Strg** und **Alt** der Tastatur erfragen möchten, gibt es hierfür die statische Methode **System.Windows.Forms.-Control.ModifierKeys**.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ button	System.Int16	Zahl, die angibt, welche Maustasten gedrückt sind: **1** (links), **2** (rechts) oder **4** (Mitte).
⇨ shift	System.Int16	Eine Ganzzahl, die dem Zustand der Tasten Umschalt, Strg und Alt zu dem Zeitpunkt entspricht, an dem das Ereignis aufgetreten ist. Das Argument Shift ist ein Bitfeld, bei dem die niederstwertigen Bits der Umschalt-Taste (Bit 0), der Strg-Taste (Bit 1) und der Alt-Taste (Bit 2) entsprechen. Diese Bits entsprechen jeweils dem Wert 1, 2 und 4. Einige, alle oder keine dieser Bits können gesetzt werden, um anzuzeigen, dass einige, alle oder keine der Tasten gedrückt sind. Wenn zum Beispiel sowohl Strg als auch Alt gedrückt werden, hat shift den Wert 6.
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers

VcMouseDown

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender eine Maustaste drückt.

Bitte beachten Sie auch die Eigenschaft **MouseProcessingEnabled**. Wenn Sie den Status der Sondertasten **Shift**, **Strg** und **Alt** der Tastatur erfragen möchten, gibt es hierfür die statische Methode **System.Windows.Forms.-Control.ModifierKeys**.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ button	System.Int16	Zahl, die angibt, welche Maustasten gedrückt sind: **1** (links), **2** (rechts) oder **4** (Mitte).
⇨ shift	System.Int16	Eine Ganzzahl, die dem Zustand der Tasten Umschalt, Strg und Alt zu dem Zeitpunkt entspricht, an dem das Ereignis aufgetreten ist. Das Argument Shift ist ein Bitfeld, bei dem die niederstwertigen Bits der Umschalt-Taste (Bit 0), der Strg-Taste (Bit 1) und der Alt-Taste (Bit 2) entsprechen. Diese Bits entsprechen jeweils dem Wert 1, 2 und 4. Einige, alle oder keine dieser Bits können gesetzt werden, um anzuzeigen, dass einige, alle oder keine der Tasten gedrückt sind. Wenn zum Beispiel sowohl Strg als auch Alt gedrückt werden, hat shift den Wert 6.
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers

VcMouseMove

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender die Maus bewegt.

Bitte beachten Sie auch die Eigenschaft **MouseProcessingEnabled**. Wenn Sie den Status der Sondertasten **Shift**, **Strg** und **Alt** der Tastatur erfragen möchten, gibt es hierfür die statische Methode **System.Windows.Forms.-Control.ModifierKeys**.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ button	System.Int16	Zahl, die angibt, welche Maustasten gedrückt sind: **1** (links), **2** (rechts) oder **4** (Mitte).

⇨ shift	System.Int16	Eine Ganzzahl, die dem Zustand der Tasten Umschalt, Strg und Alt zu dem Zeitpunkt entspricht, an dem das Ereignis aufgetreten ist. Das Argument Shift ist ein Bitfeld, bei dem die niederstwertigen Bits der Umschalt-Taste (Bit 0), der Strg-Taste (Bit 1) und der Alt-Taste (Bit 2) entsprechen. Diese Bits entsprechen jeweils dem Wert 1, 2 und 4. Einige, alle oder keine dieser Bits können gesetzt werden, um anzuzeigen, dass einige, alle oder keine der Tasten gedrückt sind. Wenn zum Beispiel sowohl Strg als auch Alt gedrückt werden, hat shift den Wert 6.
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers

VcMouseUp

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender eine gedrückte Maustaste wieder loslässt.

Bitte beachten Sie auch die Eigenschaft **MouseProcessingEnabled**. Wenn Sie den Status der Sondertasten **Shift**, **Strg** und **Alt** der Tastatur erfragen möchten, gibt es hierfür die statische Methode **System.Windows.Forms.-Control.ModifierKeys**.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ button	System.Int16	Zahl, die angibt, welche Maustasten gedrückt sind: **1** (links), **2** (rechts) oder **4** (Mitte).
⇨ shift	System.Int16	Eine Ganzzahl, die dem Zustand der Tasten Umschalt, Strg und Alt zu dem Zeitpunkt entspricht, an dem das Ereignis aufgetreten ist. Das Argument Shift ist ein Bitfeld, bei dem die niederstwertigen Bits der Umschalt-Taste (Bit 0), der Strg-Taste (Bit 1) und der Alt-Taste (Bit 2) entsprechen. Diese Bits entsprechen jeweils dem Wert 1, 2 und 4. Einige, alle oder keine dieser Bits können gesetzt werden, um anzuzeigen, dass einige, alle oder keine der Tasten gedrückt sind. Wenn zum Beispiel sowohl Strg als auch Alt gedrückt werden, hat shift den Wert 6.
⇨ x	System.Int32	X-Koordinate des Mauszeigers
⇨ y	System.Int32	Y-Koordinate des Mauszeigers

VcNodeCollapsing

Ereignis von VcTree

Dieses Ereignis wird aufgerufen, wenn ein Knoten durch eine Benutzeraktion kollabiert wird. Steht im Parameter **action** der Wert **vcSelf**, so ist der angegebene, nun kollabierte Knoten auch der selektierte. Mit ihm wird die Operation per Kontextmenü durchgeführt. Steht im Parameter **action** der Wert **vcComplete**, befindet sich der selektierte Knoten in dem Teilbaum unter dem Knoten.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Knotenobjekt
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.
⇨ action	VcCollapseExpand	Typ des Kollabierens

VcNodeCreated

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn das interaktive Anlegen eines Knotens abgeschlossen ist. Das Knotenobjekt, der Typ des Knotenanlegens und die Information, ob der angelegte Knoten der einzige Knoten bzw. der letzte Knoten einer Menge ist, werden als Parameter zurückgegeben, so dass eine Datenvalidierung vorgenommen werden kann.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Angelegter Knoten
⇨ creationType	VcCreationType	Typ des Knotenanlegens
	Mögliche Werte: .vcDataRecordCreated 6 .vcNodeCreated 1 .vcNodesCloned 4	 Datensatz wurde durch Interaktion angelegt Knoten durch "Stempeln" angelegt Selektierte Knoten wurden durch Interaktion kopiert
⇨ isLast	System.Boolean	Angelegter Knoten ist/ist nicht der einzige Knoten bzw. der letzte Knoten einer Menge

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeCreated(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeCreatedEventArgs) Handles VcTree1.VcNodeCreated
        MsgBox(e.Node.AllData)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeCreated(object sender,
NETRONIC.XTree.VcNodeCreatedEventArgs e)
{
MessageBox.Show(e.Node.AllData.ToString());
}
```

VcNodeCreating

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender interaktiv einen Knoten erzeugt. Das Knotenobjekt wird als Parameter übergeben, so dass eine Datenvalidierung vorgenommen werden kann. (Der Dialog **Daten bearbeiten** wurde ggf. vorher über die Eigenschaft **VcTree.NodeCreationWithDialog** aktiviert.) Durch Setzen des Rückgabestatus auf **vcRetStatFalse** wird der Knoten nicht erzeugt.

Dieses Ereignis sollte nur verwendet werden, um Daten des aktuellen Knotens auszulesen. Um sie zu verändern, verwenden Sie bitte das Ereignis **OnNodeCreateCompleteEx**.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Anzulegender Knoten
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte:	
	.vcRetStatDefault 2	Das Default-Verhalten wird nicht verändert.
	.vcRetStatFalse 0	Das Default-Verhalten wird nicht durchgeführt.
	.vcRetStatNoPopup 4	Das Erscheinen des Kontextmenüs wird unterdrückt.
	.vcRetStatOK 1	Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeCreating(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeCreatingEventArgs) Handles VcTree1.VcNodeCreating
        MsgBox("Show your own dialog")
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeCreating(object sender,
NETRONIC.XTree.VcNodeCreatingEventArgs e)
   {
   MessageBox.Show("Show your own dialog");
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
   }
```

VcNodeDeleted

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn das interaktive Löschen eines Knotens abgeschlossen ist. Das Knotenobjekt und die Information, ob der gelöschte Knoten der zuletzt gelöschte einer Menge ist, werden als Parameter zurückgegeben, so dass eine Datenvalidierung vorgenommen werden kann.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Gelöschter Knoten
⇨ isLast	System.Boolean	Gelöschter Knoten ist/ist nicht der letzte Knoten einer Menge

VcNodeDeleting

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit Hilfe des Kontextmenüs einen Knoten löscht. Der getroffene Knoten wird als Parameter übergeben, so dass noch eine Überprüfung vorgenommen werden kann. Durch Setzen des Rückgabestatus auf **vcRetStatFalse** wird der Knoten nicht gelöscht.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Knotenobjekt
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte:	
	.vcRetStatDefault 2	Das Default-Verhalten wird nicht verändert.
	.vcRetStatFalse 0	Das Default-Verhalten wird nicht durchgeführt.
	.vcRetStatNoPopup 4	Das Erscheinen des Kontextmenüs wird unterdrückt.
	.vcRetStatOK 1	Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeDeletingEventArgs) Handles VcTree1.VcNodeDeleting
        'deny the deletion of the last node in the chart
        If VcTree1.NodeCollection.Count = 1 Then
            e.ReturnStatus = VcReturnStatus.vcRetStatFalse
            MsgBox("The last node in the chart cannot be deleted.")
        End If
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeDeleting(object sender,
NETRONIC.XTree.VcNodeDeletingEventArgs e)
   {
   //deny the deletion of the last node in the chart
   if (vcTree1.NodeCollection.Count == 1)
      e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
      MessageBox.Show("The last node in the chart cannot be deleted.");
   }
```

VcNodeExpanding

Ereignis von VcTree

Dieses Ereignis wird aufgerufen, wenn ein Knoten durch eine Benutzeraktion expandiert wird. Steht im Parameter **action** der Wert **vcSelf**, so ist der angegebene, nun kollabierte Knoten auch der selektierte. Mit ihm wird die Operation per Kontextmenü durchgeführt. Steht hier **vcComplete**, gehört der angegebene Knoten zu der Menge der Sohnknoten des selektierten Knotens. Wird der Rückgabestatus auf **vcRetStatFalse** gesetzt, wird die Aktion abgebrochen und die angegebenen Knoten werden nicht expandiert.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Knotenobjekt
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.
⇨ action	VcCollapseExpand	Typ des Kollabierens

VcNodeLeftClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender mit der linken Maustaste auf einen Knoten klickt. Das getroffene Knotenobjekt wird zusammen mit der Position (x,y-Koordinaten) des Mauszeigers als Parameter übergeben.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Knotenobjekt
⇨ location	VcLocation	Bereich im Chart
	Mögliche Werte: .vcInDiagram 1	 im Knotenbereich
⇨ x	System.Int32	X-Koordinate
⇨ y	System.Int32	Y-Koordinate
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles dummyobject1.VcNodeLeftClicking
        'change data field of the node
        e.Node.DataField(4) = 1 - Convert.ToInt64(e.Node.DataField(4))
    End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftClicking(object sender,
NETRONIC.XTree.VcNodeClickingEventArgs e)
   {
   //change data field of the node
    e.Node.set_DataField(4,Convert.ToInt64(e.Node.get_DataField(4)));
   }
```

VcNodeLeftDoubleClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender auf einen Knoten mit der linken Maustaste doppelt klickt. Das getroffene Knotenobjekt wird zusammen mit der Position des Mauszeigers (x,y-Koordinaten) als Parameter übergeben. Durch Setzen des Rückgabestatus können Sie das Erscheinen des Dialogfelds **Vorgänge bearbeiten** unterdrücken.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Knotenobjekt
⇨ x	System.Int32	X-Koordinate
⇨ y	System.Int32	Y-Koordinate
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeLeftDoubleClicking
        MsgBox("Show your own dialog")
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeLeftDoubleClicking(object sender,
NETRONIC.XTree.VcNodeClickingEventArgs e)
   {
   MessageBox.Show("Show your own dialog");
   e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
   }
```

VcNodeModified

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn die Modifizierung des angegebenen Knotens abgeschlossen ist.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcNodeModifiedEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcNodeModifiedEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Angelegter Knoten

⇨ isLast	System.Boolean	Angelegter Knoten ist/ist nicht der einzige Knoten bzw. der letzte Knoten einer Menge

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeModifying(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeModifyingEventArgs) Handles VcTree1.VcNodeModifying
   'revoke the modification if the node would change the group
   If e.ModificationType And VcModificationTypes.vcChangedGroup Then
      MsgBox("The node cannot be moved to a different group.")
   End If
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeModifying(object sender,
NETRONIC.XTree.VcNodeModifyingEventArgs e)
   {
   //revoke the modification if the node would change the group
   if (VcModificationTypes.vcChangedGroup.Equals(true))
      {
      MessageBox.Show("The node cannot be moved into another group.");
       }
   }
```

VcNodeModifying

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender interaktiv einen Knoten verändert. Dabei kann der Knoten verschoben oder ein Wert im Dialogfeld **Vorgänge bearbeiten** verändert worden sein. Die Daten des Knotens vor und nach der Veränderung werden als Parameter zurückgegeben. Über den Parameter **modificationType** erhalten Sie nähere Informationen über die Art der Veränderung.

Durch Setzen des Rückgabestatus auf **vcRetStatFalse** kann die Änderung verhindert werden.

Die mit diesem Ereignis übermittelten Daten dürfen nur gelesen, aber nicht verändert werden. Um sie zu verändern, verwenden Sie bitte das Ereignis **VcNodeModified**.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objekt
⇨ e	VcNodeModifiedEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcNodeModifiedEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ oldNode	VcNode	Knoten vor der Veränderung
⇨ node	VcNode	zu verändernder Knoten
⇨ modificationType	VcModificationTypes	Art der Veränderung
	Mögliche Werte: .vcAnything 1 .vcMoved 8 .vcNothing 0	 Änderungstyp nicht näher bestimmt Objekt wurde verschoben. Keine Änderung
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeModifying(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeModifyingEventArgs) Handles VcTree1.VcNodeModifying
        ' revoke the modification if the node would change the group
        If e.ModificationType And VcModificationTypes.vcChangedGroup Then
            MsgBox("The node cannot be moved to a different group.")
            e.ReturnStatus = VcReturnStatus.vcRetStatFalse
        End If
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeModifying(object sender,
NETRONIC.Tree.VcNodeModifyingEventArgs e)
   {
   //revoke the modification if the node would change the group
   if (e.ModificationType == VcModificationTypes.vcChangedGroup)
      {
      MessageBox.Show("The node cannod be moved into another group.");
      e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
      }
   }
```

VcNodeRightClicking

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender auf einen Knoten mit der rechten Maustaste klickt. Das getroffene Knotenobjekt wird zusammen mit der Position des Mauszeigers (x,y-Koordinaten) als Parameter übergeben. Sie können so an der entsprechenden Position Ihr gewünschtes Kontextmenü anzeigen. Das integrierte Kontextmenü kann durch Setzen des Rückgabestatus auf **vcRetStatNoPopup** unterdrückt werden.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ node	VcNode	Knotenobjekt
⇨ location	VcLocation	Bereich im Chart
	Mögliche Werte: .vcInDiagram 1	 im Knotenbereich
⇨ x	System.Int32	X-Koordinate
⇨ y	System.Int32	Y-Koordinate
⇦ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodeClickingEventArgs) Handles VcTree1.VcNodeRightClicking
        PopupMenu.Show(VcTree1, New Point(e.X, e.Y))
        e.ReturnStatus = NETRONIC.XTree.VcReturnStatus.vcRetStatNoPopup
    End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodeRightClicking(object sender,
NETRONIC.XTree.VcNodeClickingEventArgs e)
   {
   PopupMenu.Show(vcTree1, new Point (e.X, e.Y));
   e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
   }
```

VcNodesMarked

Ereignis von VcTree

Mit diesem Ereignis wird das Ende einer Markier- oder Demarkieroperation eines Knotens angezeigt.

	Datentyp	Beschreibung
Eigenschaften:		
⇦ (no parameter)		Kein Parameter

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodesMarked(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodesMarkedEventArgs) Handles VcTree1.VcNodesMarked
   MsgBox("Nodes have been marked successfully.")
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodesMarked(object sender,
NETRONIC.XTree.VcNodesMarkedEventArgs e)
   {
   MessageBox.Show("Nodes have been marked successfully.");
   }
```

VcNodesMarking

Ereignis von VcTree

Mit diesem Ereignis wird bekanntgegeben, dass der Benutzer Knoten zum Markieren ausgewählt oder markierte Knoten durch einen Klick in den leeren Diagrammbereich demarkiert hat. In der Knotenauflistung (NodeCollection-Objekt) sind die beim letzten Markiervorgang ausgewählten Knoten verzeichnet. Falls durch einen Klick ins leere Diagramm demarkiert wurde, ist die NodeCollection leer.

Wenn Sie den Rückgabestatus auf **vcRetStatFalse** setzen, muss das Markieren oder Demarkieren selbst übernommen werden.

Die mit diesem Ereignis übermittelten Daten dürfen nur gelesen, aber nicht verändert werden. Um sie zu verändern, verwenden Sie bitte das Ereignis **VcNodesMarked**

	Datentyp	**Beschreibung**
Eigenschaften:		
⇨ nodeCollection	VcNodeCollection	NodeCollection, die die vom Anwender selektierten Knoten enthält. Wenn in das Diagramm geklickt wurde, ist die Collection leer.
⇨ button	System.Int16	Zahl, die angibt, auf welche Weise markiert wurde: **0**: über die Tastatur, **1**: linke Maustaste, **2**: rechte Maustaste, **4**: mittlere Maustaste
⇨ shift	System.Int16	Zahl, die den Zustand der Modifizierungstasten zu dem Zeitpunkt angibt, zu dem Daten über das Drop-Ziel gezogen werden. Die gültigen Modifizierungstasten sind die <Umschalt>-, <Strg>- und <Alt>-Tasten, die den Zahlen **1**, **2** und **4** entsprechen. Der Parameter **shift** zeigt den Status dieser Tasten an; es können einige, alle oder keines der drei Zahlen gesetzt werden, was jeweils anzeigt, dass einige, alle oder keine der Tasten gedrückt wird. Wenn beispielsweise die <Strg>- und die <Alt>-Tasten gedrückt werden, ist der Wert von **shift** "6".
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte:	
	.vcRetStatDefault 2	Das Default-Verhalten wird nicht verändert.
	.vcRetStatFalse 0	Das Default-Verhalten wird nicht durchgeführt.
	.vcRetStatNoPopup 4	Das Erscheinen des Kontextmenüs wird unterdrückt.

.vcRetStatOK 1	Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcNodesMarking(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcNodesMarkingEventArgs) Handles VcTree1.VcNodesMarking
   If MsgBox("Mark this node?", MsgBoxStyle.YesNo, "Marking nodes") =
MsgBoxResult.No Then
      e.ReturnStatus = VcReturnStatus.vcRetStatFalse
   End If
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcNodesMarking(object sender,
NETRONIC.XTree.VcNodesMarkingEventArgs e)
   {
   DialogResult retVal = MessageBox.Show("Mark this node?", "Marking nodes",
MessageBoxButtons.YesNo);
   if (retVal ==  DialogResult.No)
      e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
   }
```

VcStatusLineTextShowing

Ereignis von VcTree

Dieses Ereignis tritt immer dann ein, wenn eine Information von allgemeiner Bedeutung bereitgestellt wird. Das kann ein funktionaler Hinweis sein (z. B. beim Laden), aber auch die ID des Knotens, auf dem der Cursor gerade steht.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ text	System.String	Text
⇨ paneNo	System.Int16	Dummy

Code-Beispiel VB.NET

```
Private Sub VcNet1_VcStatusLineTextShowing(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcStatusLineTextShowingEventArgs) Handles
VcTree1.VcStatusLineTextShowing
   TextBox1.Text = e.Text
End Sub
```

Code-Beispiel C#

```
private void vcvcTree1_VcStatusLineTextShowing(object sender,
NETRONIC.XTree.VcStatusLineTextShowingEventArgs e)
   {
   textBox1.Text = e.Text;
   }
```

VcTextEntrySupplying

Ereignis von VcTree

Das Ereignis tritt auf, wenn ein Text ausgegeben werden soll. Voraussetzung ist, dass Sie die Eigenschaft **EnableSupplyTextEntryEvent** auf **True** gesetzt haben. Sie können hier alle vorgegebenen Texte von Monats- und Tagesnamen durch eigene Texte ersetzen, z. B. um sie in unterschiedliche Sprachen zu übersetzen.

	Datentyp	Beschreibung
Parameter:		
⇨ sender	VcTree	Verweis auf das ereignisauslösende Objek
⇨ e	VcTextEntrySupplyingEventArgs	Ereignisspezifisches Objekt

Eigenschaften des VcTextEntrySupplyingEventArgs-Objektes

	Datentyp	Beschreibung
Eigenschaften:		
⇨ controlIndex	VcTextEntryIndex	Zu ersetzender Text
	Mögliche Werte:	
	.vcTXECtxmenArrowMode 2116	Text im Kontextmenü: **Selektier-Modus**
	.vcTXECtxmenCollapse 2184	Text im Kontextmenü: **Kollabieren**
	.vcTXECtxmenCopyNodes 2152	Text im Kontextmenü: **Knoten kopieren**
	.vcTXECtxmenCreateNodeMode 2117	Text im Kontextmenü: **Knoten erzeugen**
	.vcTXECtxmenCutNodes 2151	Text im Kontextmenü: **Knoten ausschneiden**
	.vcTXECtxmenDeleteNode 2101	Text im Kontextmenü: **Knoten löschen**
	.vcTXECtxmenEditNode 2100	Text im Kontextmenü: **Daten bearbeiten**
	.vcTXECtxmenExpand 2185	Text im Kontextmenü: **Expandieren**
	.vcTXECtxmenExpandComplete 2186	Text im Kontextmenü: **Vollständig expandieren**
	.vcTXECtxmenFilePrint 2122	Text im Kontextmenü: **Drucken**
	.vcTXECtxmenFilePrintPreview 2121	Text im Kontextmenü: **Druckvorschau**
	.vcTXECtxmenFilePrintSetup 2120	Text im Kontextmenü: **Drucker einrichten**
	.vcTXECtxmenFirstChild 2182	Text im Kontextmenü: **Knoten einfügen als ersten Sohnknoten**
	.vcTXECtxmenFullDiagram 2156	Text im Kontextmenü **Gesamtbaum wiederherstellen**
	.vcTXECtxmenGraphicExport 2123	Text im Kontextmenü: **Grafik exportieren**
	.vcTXECtxmenHorizontal 2187	Text im Kontextmenü: **Horizontal**
	.vcTXECtxmenHorizontalComplete 2188	Text im Kontextmenü: **Horizontal vollständig**
	.vcTXECtxmenLastChild 2182	Text im Kontextmenü: **Knoten einfügen als letzten Sohnknoten**
	.vcTXECtxmenPageLayout 2119	Text im Kontextmenü: **Seite einrichten**
	.vcTXECtxmenPasteNodesAfter 2180	Text im Kontextmenü: **Knoten einfügen hinter**

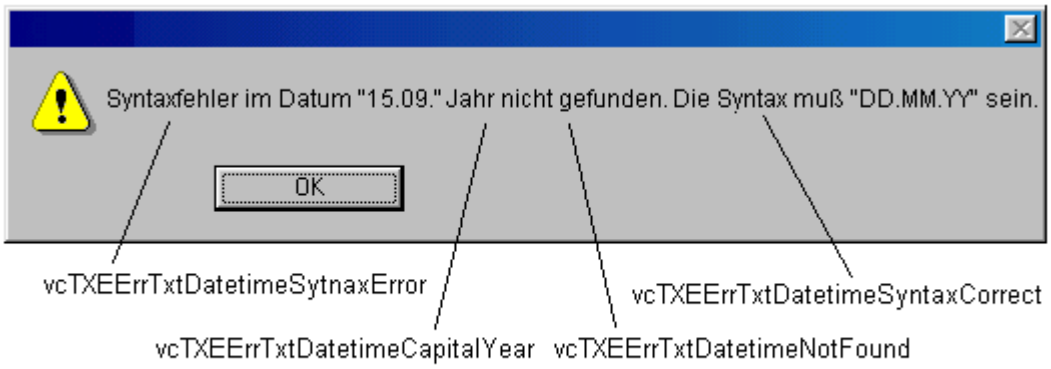
.vcTXECtxmenPasteNodesBefore 2181	Text im Kontextmenü: **Knoten einfügen vor**
.vcTXECtxmenPasteNodesFirstChild 2182	Text im Kontextmenü: **Knoten als ersten Sohnknoten einfügen**
.vcTXECtxmenPasteNodesLastChild 2182	Text im Kontextmenü: **Knoten als letzten Sohnknoten einfügen**
.vcTXECtxmenShowLegendView 2157	Text im Kontextmenü **Legendenansicht anzeigen**
.vcTXECtxmenShowWorldView 2157	Text im Kontextmenü **Komplettansicht anzeigen**
.vcTXECtxmenSubDiagram 2155	Text im Kontextmenü **Teilbaum erstellen**
.vcTXECtxmenVertical 2189	Text im Kontextmenü: **Vertikal**
.vcTXEDateAM 2225	Ausgabetext für **vormittags**
.vcTXEDateCW 2223	Ausgabetext für **Kalenderwoche**
.vcTXEDateDay0 2212	Ausgabetext für **Montag**
.vcTXEDateDay1 2213	Ausgabetext für **Dienstag**
.vcTXEDateDay2 2214	Ausgabetext für **Mittwoch**
.vcTXEDateDay3 2215	Ausgabetext für **Donnerstag**
.vcTXEDateDay4 2216	Ausgabetext für **Freitag**
.vcTXEDateDay5 2217	Ausgabetext für **Samstag**
.vcTXEDateDay6 2218	Ausgabetext für **Sonntag**
.vcTXEDateMonth0 2200	Ausgabetext für **Januar**
.vcTXEDateMonth1 2201	Ausgabetext für **Februar**
.vcTXEDateMonth10 2210	Ausgabetext für **November**
.vcTXEDateMonth11 2211	Ausgabetext für **Dezember**
.vcTXEDateMonth2 2202	Ausgabetext für **März**
.vcTXEDateMonth3 2203	Ausgabetext für **April**
.vcTXEDateMonth4 2204	Ausgabetext für **Mai**
.vcTXEDateMonth5 2205	Ausgabetext für **Juni**
.vcTXEDateMonth6 2206	Ausgabetext für **Juli**
.vcTXEDateMonth7 2207	Ausgabetext für **August**
.vcTXEDateMonth8 2208	Ausgabetext für **September**
.vcTXEDateMonth9 2209	Ausgabetext für **Oktober**
.vcTXEDateOClock 2224	Ausgabetext für **Uhr**
.vcTXEDatePM 2226	Ausgabetext für **nachmittags**
.vcTXEDateQuarter0 2219	Ausgabetext für **1. Quartal**
.vcTXEDateQuarter1 2220	Ausgabetext für **2. Quartal**
.vcTXEDateQuarter2 2221	Ausgabetext für **3. Quartal**
.vcTXEDateQuarter3 2222	Ausgabetext für **4. Quartal**
.vcTXEDlgLegArrangement 2046	Text im Dialog **Legendenattribute**: **Anordnung**
.vcTXEDlgLegBottomMargin 2052	Text im Dialog **Legendenattribute**: **Unterer Rand:**
.vcTXEDlgLegFixedToColumns 2048	Text im Dialog **Legendenattribute**: **nach Spaltenanzahl**
.vcTXEDlgLegFixedToRows 2047	Text im Dialog **Legendenattribute**: **nach Zeilenanzahl**
.vcTXEDlgLegFixedToRowsAndColumns 2049	Text im Dialog **Legendenattribute**: **nach Zeilen- und Spaltenanzahl**
.vcTXEDlgLegIdcancel 2042	Schaltfläche im Dialog **Legendenattribute**: **Abbrechen**
.vcTXEDlgLegIdd 2040	Dialog Legendenattribute Beschriftung der Titelzeile
.vcTXEDlgLegIdok 2041	Schaltflächentext im Dialog **Legendenattribute**: **OK**
.vcTXEDlgLegLegendElements 2045	Text im Dialog **Legendenattribute**: **Legendenelemente**
.vcTXEDlgLegLegendFont 2053	Schaltfläche im Dialog **Legendenattribute**: **Schriftart...** für Legende
.vcTXEDlgLegLegendTitleFont 2044	Schaltfläche im Dialog **Legendenattribute**: **Schriftart...** für Legendentitel
.vcTXEDlgLegLegendTitleVisible 2043	Text im Dialog **Legendenattribute**: **Legendentitel sichtbar**

.vcTXEDlgLegMargins 2050	Text im Dialog **Legendenattribute**: **Ränder**
.vcTXEDlgLegTopMargin 2051	Text im Dialog **Legendenattribute**: **Oberer Rand:**
.vcTXEDlgNedCaptionPrefix 2024	Dialog **Vorgänge bearbeiten**,: Text für Beschriftungszeile: “Knoten”
.vcTXEDlgNedIdapply 2027	Dialog **Vorgänge bearbeiten**, "Übernehmen"-Schaltfläche
.vcTXEDlgNedIdcancel 2016	Text im Dialog **Vorgänge bearbeiten**: **Abbrechen**
.vcTXEDlgNedIdclose 2029	Dialog **Vorgänge bearbeiten**: **Schließen**-Schaltfläche
.vcTXEDlgNedIdd 2014	Überschrift des Dialogs **Vorgänge bearbeiten**
.vcTXEDlgNedIdhelp 2028	Dialog **Vorgänge bearbeiten**: **Hilfe**-Schaltfläche
.vcTXEDlgNedIdok 2015	Text im Dialog **Vorgänge bearbeiten**: **OK**
.vcTXEDlgNedNamesColStr 2018	Text im Dialog **Vorgänge bearbeiten**: **Datenfelder**
.vcTXEDlgNedTTGotoFirst 2032	Dialog **Vorgänge bearbeiten**: Tooltiptext **Ersten ausgewählten Vorgang anzeigen**
.vcTXEDlgNedTTGotoLast 2035	Dialog **Vorgänge bearbeiten**, Tooltip “Letzten ausgewählten Vorgang anzeigen“
.vcTXEDlgNedTTGotoNext 2034	Dialog **Vorgänge bearbeiten**, Tooltiptext **Nächsten ausgewählten Vorgang anzeigen**
.vcTXEDlgNedTTGotoPrev 2033	Dialog **Vorgänge bearbeiten**: Tooltiptext **Vorherigen ausgewählten Vorgang anzeigen**
.vcTXEDlgNedValuesColStr 2019	Text im Dialog **Vorgänge bearbeiten**: **Werte**
.vcTXEErrTxtEntryTooLong 2730	Meldungstext: "Eintrag ist zu lang, %s Zeichen sind möglich."
.vcTXEPrctBtApply 2318	Schaltflächen-Text im **Seite einrichten**-Dialogs: **Anwenden**
.vcTXEPrctBtCancel 2302	Schaltflächen-Text im Druck-Info-Fenster: **Abbrechen**
.vcTXEPrctBtClose 2303	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Schließen**
.vcTXEPrctBtFitToPage 2308	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Einpassen**
.vcTXEPrctBtNext 2305	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Weiter**
.vcTXEPrctBtOk 2301	Schaltflächen-Text des Seitenlayout-Dialogs: **OK**
.vcTXEPrctBtPageLayout 2311	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Seite einrichten**
.vcTXEPrctBtPrevious 2304	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Vorher**
.vcTXEPrctBtPrint 2313	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Drucken**
.vcTXEPrctBtPrinterSetup 2312	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Drucker einrichten**
.vcTXEPrctBtSingle 2307	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Einzelseite**
.vcTXEPrctBtZoomPrint 2319	Schaltflächen-Text des **Druckvorschau**-Dialogs: **Ausschnitt drucken…**
.vcTXEPrctDtAddCuttingMarks 2514	Text des **Seite einrichten**-Dialogs: **Zuschnittmarken**

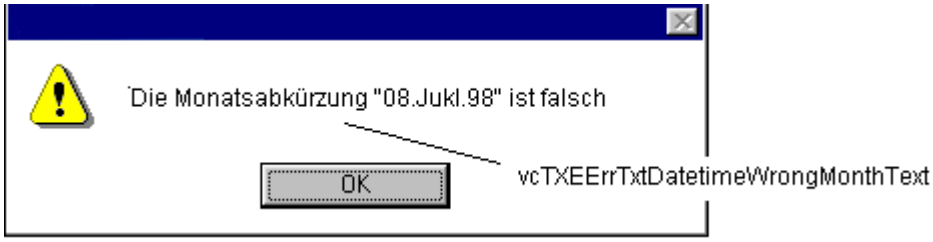
.vcTXEPrctDtAlignment 2526	Text des **Seite einrichten**-Dialogs: **Ausrichtung**
.vcTXEPrctDtAlignmentItems 2583	Text des **Seite einrichten**-Dialogs: **Oben links|Oben|Oben rechts|Links|Mittig|Rechts|Unten links|Unten|Unten rechts**
.vcTXEPrctDtApplicationName 2501	Text im Druck-Info-Fenster: **Name der Anwendung**
.vcTXEPrctDtBottom 2521	Text des **Seite einrichten**-Dialogs: **Unten**
.vcTXEPrctDtCm 2530	Text des **Seite einrichten**-Dialogs: **cm**
.vcTXEPrctDtCurrentValues 2581	Text des **Seite einrichten**-Dialogs: **Aktuell**
.vcTXEPrctDtEnableDiagram 2559	Text des **Seite einrichten**-Dialogs: **Diagramm anzeigen**
.vcTXEPrctDtFitToPage 2508	Text des **Seite einrichten**-Dialogs: **Anpassen an Seitenzahl**
.vcTXEPrctDtFoldingMarksItems 2577	Text des **Seite einrichten**-Dialogs: **Form A|Form B|Form C**
.vcTXEPrctDtFoldingMarksText 2576	Text des **Seite einrichten**-Dialogs:**"&Faltmarkierungen (DIN 824)**
.vcTXEPrctDtFooterGroup 2584	Text des **Seite einrichten**-Dialogs: **Fußzeile**
.vcTXEPrctDtFrameOutside 2515	Text des **Seite einrichten**-Dialogs: **Rahmen außen**
.vcTXEPrctDtInch 2588	Text des **Seite einrichten**-Dialogs: **Zoll**
.vcTXEPrctDtLeft 2520	Text des **Seite einrichten**-Dialogs: **Links**
.vcTXEPrctDtMargins 2529	Text des **Seite einrichten**-Dialogs: **Mindestgrößen für die Seitenränder**
.vcTXEPrctDtMaxPages 2580	Text des **Seite einrichten**-Dialogs: **Seiten**
.vcTXEPrctDtOff 2557	Text **Aus** Dialog
.vcTXEPrctDtOptions 2528	Text des **Seite einrichten**-Dialogs: **Seitenaufteilung**
.vcTXEPrctDtOutput 2531	Text des **Seite einrichten**-Dialogs: **Ausdruck**
.vcTXEPrctDtPageDescription 2562	Text des **Seite einrichten**-Dialogs: **Text**
.vcTXEPrctDtPageLayout 2532	Fenstertitel des **Seite einrichten**-Dialogs
.vcTXEPrctDtPageNumberingItems 2582	Text des **Seite einrichten**-Dialogs: **Zeile.Spalte|Spalte.Zeile|Seite/Anzahl**
.vcTXEPrctDtPageNumbers 2518	Text des **Seite einrichten**-Dialogs: **Seiten&nummerierung**
.vcTXEPrctDtPagePadding 2585	Text des **Seite einrichten**-Dialogs: **Seiten mit Leerraum auff&üllen**
.vcTXEPrctDtPagePreview 2533	Fenstertitel des Dialogs **Druckvorschau**
.vcTXEPrctDtPagesMaxHeight 2511	Text des **Seite einrichten**-Dialogs: **Maximale Höhe**
.vcTXEPrctDtPagesMaxWidth 2510	Text des **Seite einrichten**-Dialogs: **Maximale Breite**
.vcTXEPrctDtPercent 2509	Text des **Seite einrichten**-Dialogs: **%**
.vcTXEPrctDtPrintDate 2564	Text des **Seite einrichten**-Dialogs: **&Druckdatum**
.vcTXEPrctDtPrintingPage 2556	Text im Druck-Info-Fenster: **Seite %1 von %2 wird gedruckt auf**
.vcTXEPrctDtReduceExpand 2507	Text des **Seite einrichten**-Dialogs: **Zoomfaktor**
.vcTXEPrctDtRepeatTable 2565	Text des **Seite einrichten**-Dialogs: **Tabelle wiederholen**
.vcTXEPrctDtRight 2522	Text des **Seite einrichten**-Dialogs: **Rechts**

	.vcTXEPrctDtScaling 2527	Text des **Seite einrichten**-Dialogs: **Skalierung**
	.vcTXEPrctDtScalingMode 2578	Text des **Seite einrichten**-Dialogs **&Modus:**
	.vcTXEPrctDtStatusBarCurrentValues 2586	**Statuszeilentext** des **Druckvorschau**-Dialogs: **%1 Seiten in %2 Zeilen und %3 Spalten**
	.vcTXEPrctDtStatusBarSelectedPage 2587	**Statuszeilentext** des **Druckvorschau**-Dialogs: **Seite %1 selektiert (in Zeile %2, Spalte %3)**
	.vcTXEPrctDtSuppressEmptyPages 2517	Text des **Seite einrichten**-Dialogs: **Leerseiten unterdrücken**
	.vcTXEPrctDtTableColumnRange 2575	Text des **Seite einrichten**-Dialogs: **Tabellenspalten (1-5;7)**
	.vcTXEPrctDtTop 2519	Text des **Seite einrichten**-Dialogs: **Oben**
	.vcTXEPrctDtZoomFactor 2579	Text des **Seite einrichten**-Dialogs:**&Zoomfaktor:**
	.vcTXEPrctMtAdjustBottomAndTopMargin 2437	Meldungstext: **Der untere Rand ist außerhalb des Wertebereichs und wird deshalb auf %1 cm reduziert.\r\nAußerdem wird der obere Rand auf %2 cm reduziert.**
	.vcTXEPrctMtAdjustLeftAndRightMargin 2434	Meldungstext: **Der linke Rand ist außerhalb des Wertebereichs und wird deshalb auf %1 cm reduziert.\r\nAußerdem wird der rechte Rand auf %2 cm reduziert.**
	.vcTXEPrctMtAdjustRightAndLeftMargin 2435	Meldungstext: **Der rechte Rand ist außerhalb des Wertebereichs und wird deshalb auf %1 cm reduziert.\r\nAußerdem wird der linke Rand auf %2 cm reduziert.**
	.vcTXEPrctMtAdjustTopAndBottomMargin 2436	Meldungstext: **Der obere Rand ist außerhalb des Wertebereichs und wird deshalb auf %1 cm reduziert.\r\nAußerdem wird der untere Rand auf %2 cm reduziert.**
	.vcTXEPrctMtBottomMargin 2409	Meldungstext: **Unterer Rand ...**
	.vcTXEPrctMtIncompatibleVcVersion 2414	Meldungstext: **VcVersion inkompatibel**
	.vcTXEPrctMtLeftMargin 2406	**Der linke Rand ist außerhalb des Wertebereichs und wird deshalb auf %s cm reduziert.**
	.vcTXEPrctMtPrinterNotInstalled 2411	Meldungstext: **Kein Drucker installiert**
	.vcTXEPrctMtPrintingNotPossible 2402	Meldungstext: **Drucken z. Zt nicht möglich**
	.vcTXEPrctMtRightMargin 2408	Meldungstext: **Der rechte Rand ist außerhalb des Wertebereichs und wird deshalb auf %s cm reduziert.**
	.vcTXEPrctMtSelectPaperSize 2413	Meldungstext: **Gewählte Blattgröße zu klein**
	.vcTXEPrctMtTopMargin 2407	Meldungstext **Der obere Rand ist außerhalb des Wertebereichs und wird deshalb auf %s cm reduziert.**
	.vcTXEPrctMtValueOutOfRange 2404	Meldungstext: **Außerhalb des Wertebereichs %1 bis %2**
	.vcTXEPrctMtWillBeAdjustedTo 2410	Meldungstext: **Wird korrigiert auf**
⇨ textEntry	System.String	Text, der den Standardtext ersetzen soll
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte:	
	.vcRetStatDefault 2	Das Default-Verhalten wird nicht verändert.

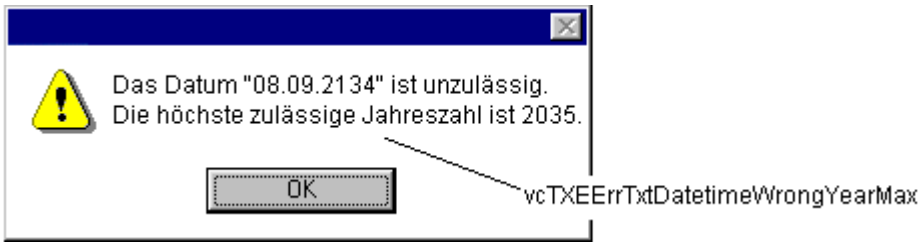
.vcRetStatFalse 0	Das Default-Verhalten wird nicht durchgeführt.
.vcRetStatNoPopup 4	Das Erscheinen des Kontextmenüs wird unterdrückt.
.vcRetStatOK 1	Das Default-Verhalten wird durchgeführt.



Konstanten der Fehlermeldung ***Syntaxfehler im Datum***

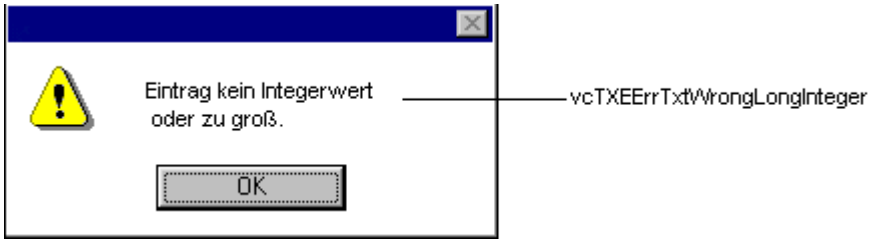


Konstanten der Fehlermeldung ***Fehler im Datum, Monat falsch***

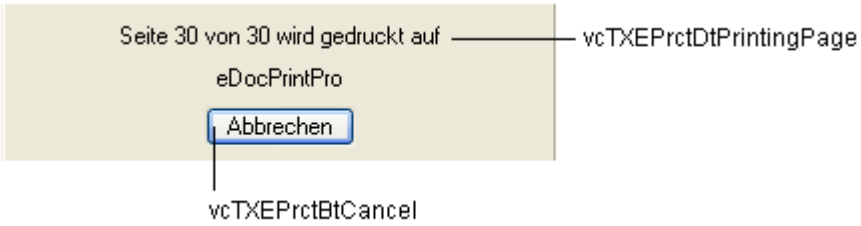


Konstante der Fehlermeldung ***Fehler im Datum, Jahr zu groß***

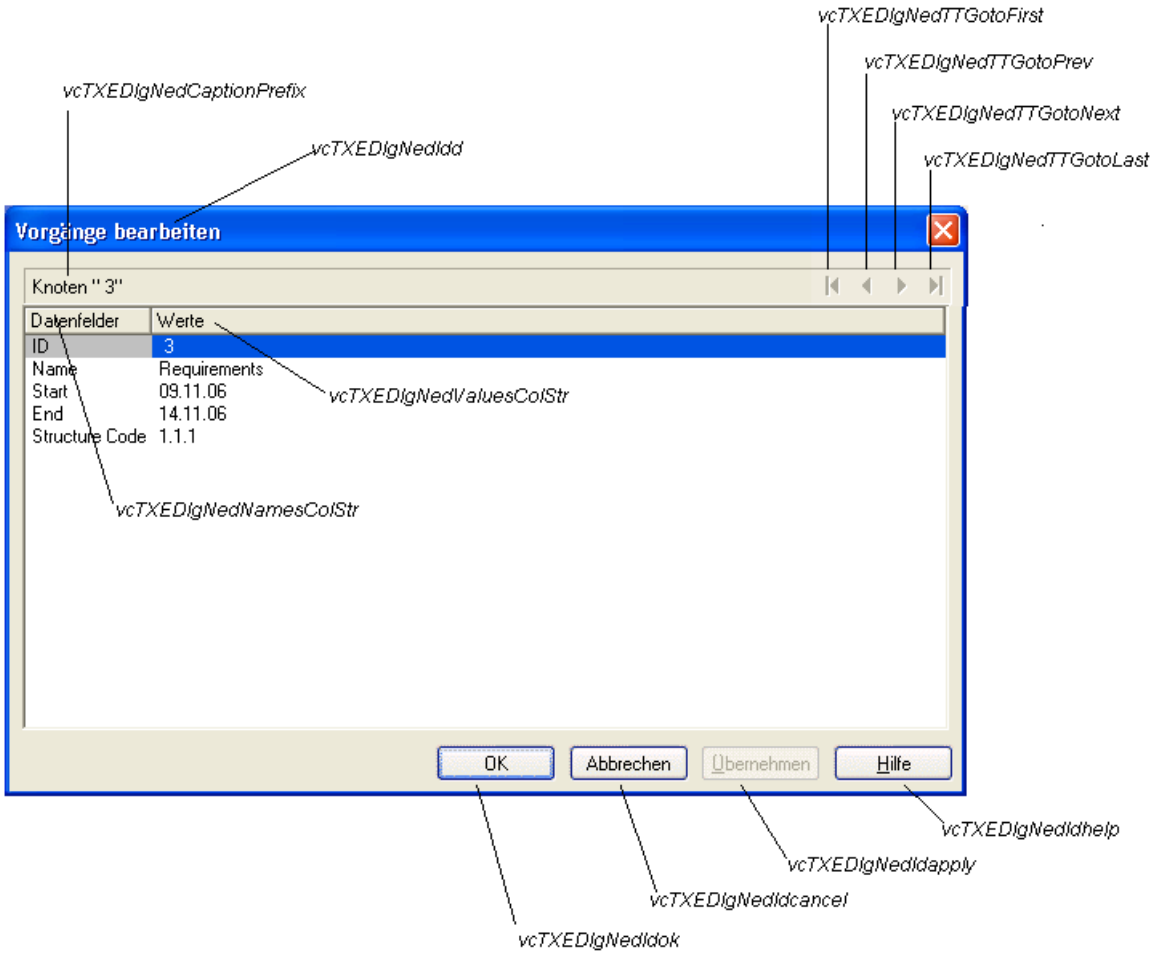
Konstanten der Fehlermeldung ***Wert zu groß***



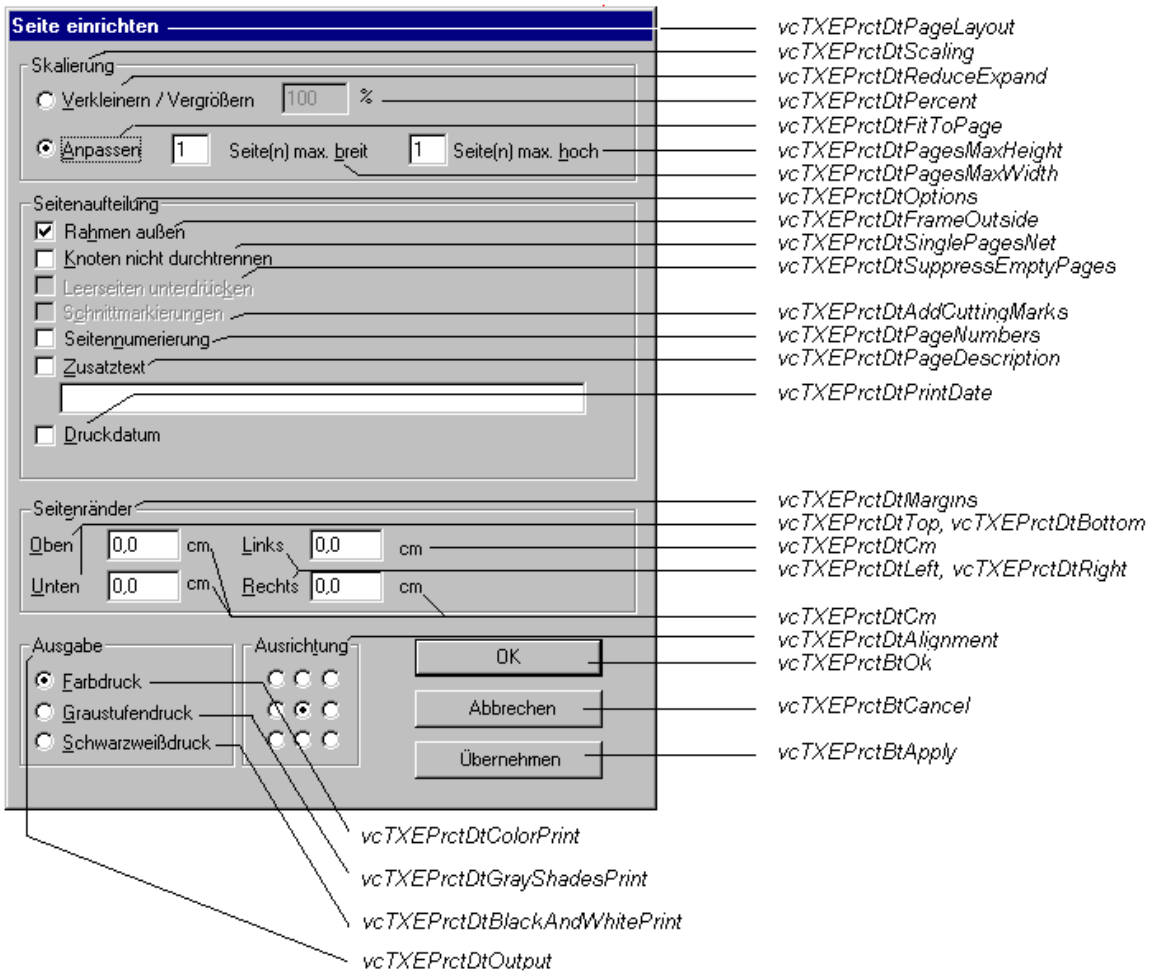
*Konstanten der Fehlermeldung **Wert kein Integer***



*Konstanten der Infobox **Drucken***

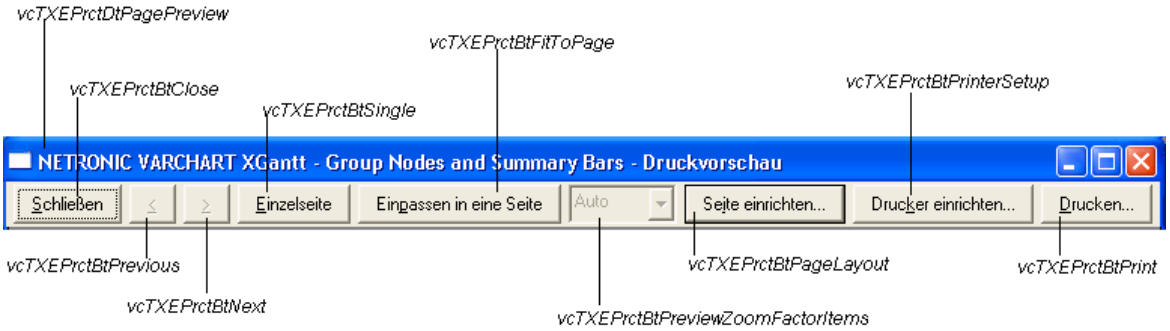


*Konstanten des Dialogs **Vorgänge bearbeiten***

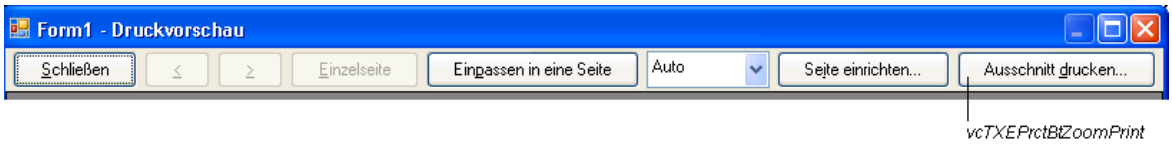


Konstanten des Dialogs ***Seite einrichten***

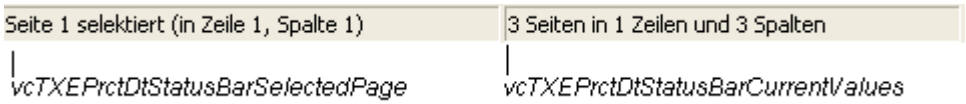
Konstanten der Tastenbeschriftungen in der ***Druckvorschau im Übersichtsmodus***



Konstanten der Tastenbeschriftungen in der ***Druckvorschau im Einzelansichtsmodus***



Konstanten der Tastenbeschriftungen in der ***Druckvorschau im Einzelansichtsmodus bei interaktiv ausgewählten Ausschnitt***



Konstanten der Statuszeile im Dialog ***Druckvorschau***

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcTextEntrySupplying(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcTextEntrySupplyingEventArgs) Handles
VcTree1.VcTextEntrySupplying
  Select Case e.ControlIndex
     Case VcTextEntryIndex.vcTXEPrctBtNext
        e.Text = "Next page"
     Case VcTextEntryIndex.vcTXEPrctBtPrevious
        e.Text = "Previous page"
      End Select
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcTextEntrySupplying(object sender,
NETRONIC.XTree.VcTextEntrySupplyingEventArgs e)
   {
   switch (e.ControlIndex)
      {
      case VcTextEntryIndex.vcTXEPrctBtNext:
         e.Text = "Next page";
         break;
      case VcTextEntryIndex.vcTXEPrctBtPrevious:
         e.Text = "Previous page";
         break;
       }
   }
```

VcToolTipTextSupplying

Ereignis von VcTree

Dieses Ereignis tritt nur auf, wenn Sie die VcTree-Eigenschaft **ToolTipText-SupplyingEventEnabled** auf **True** gesetzt haben. Das Ereignis tritt auf, sobald der Cursor auf ein XTree-Objekt bewegt wird. Es liefert Informationen über das Objekt, den Objekttyp und die Koordinaten des Cursors. Sie können mit Hilfe dieses Ereignisses die vorgegebenen Texte durch eigene Texte ersetzen, z. B. um sie in unterschiedliche Sprachen zu übersetzen. Durch Setzen des Rückgabestatus auf **vcRetStatFalse** oder Leerlassen des Textstrings "" können Sie den Tooltip an dieser Stelle unterdrücken.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ hitObject	VcObject	Objekt
⇨ hitObjectType	VcObjectType	Objekttyp
	Mögliche Werte: .vcObjTypeBox 15 .vcObjTypeNode 2 .vcObjTypeNone 0	 Objekttyp **Box** Objekttyp **Knoten** kein Objekt
⇨ x	System.Int32	X-Koordinate
⇨ y	System.Int32	Y-Koordinate
⇨ toolTipText	System.String	Anzuzeigender Text, ASP-Editionen: unbeschränkte Länge Übrige Editionen: maximal 1024 Zeichen lang
⇔ returnStatus	VcReturnStatus	Rückgabestatus
	Mögliche Werte: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	 Das Default-Verhalten wird nicht verändert. Das Default-Verhalten wird nicht durchgeführt. Das Erscheinen des Kontextmenüs wird unterdrückt. Das Default-Verhalten wird durchgeführt.

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcToolTipTextSupplying(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcToolTipTextSupplyingEventArgs) Handles
VcTree1.VcToolTipTextSupplying

        Dim node As VcNode
        If Convert.ToString(e.HitObject) = "VcTreeLib.VcNode" Then
            node = DirectCast(e.HitObject, VcNode)
            Select Case e.HitObjectType
                Case VcObjectType.vcObjTypeNodeInDiagram
                    e.Text = Convert.ToString(node.DataField(1))
                Case VcObjectType.vcObjTypeNodeInTable
                    e.Text = Convert.ToString(node.DataField(1))
            End Select
        End If
    End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcToolTipTextSupplying(object sender,
NETRONIC.XTree.VcToolTipTextSupplyingEventArgs e)
   {
   VcNode node;
   if (e.HitObject.ToString() == "NETRONIC.XTree.VcNode")
      {
      node = (VcNode)e.HitObject;
      switch(e.HitObjectType)
         {
         case VcObjectType.vcObjTypeNodeInDiagram:
            e.Text = Convert.ToString(node.get_DataField(1));
            break;
         case VcObjectType.vcObjTypeNodeInTable:
            e.Text = Convert.ToString(node.get_DataField(1));
            break;
         }
      }
   }
```

VcWorldViewClosed

Ereignis von VcTree

Dieses Ereignis wird aufgerufen, wenn das Popup-Fenster der Komplettansicht geschlossen wird.

	Datentyp	Beschreibung
Eigenschaften:		
⇨ (no parameter)		

Code-Beispiel VB.NET

```
Private Sub VcTree1_VcWorldViewClosed(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcWorldViewClosedEventArgs) Handles VcTree1.VcWorldViewClosed
   MsgBox("Do you want to close the worldview window?", MsgBoxStyle.OKCancel)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcWorldViewClosed(object sender,
NETRONIC.XTree.VcWorldViewClosedEventArgs e)
   {
   DialogResult retVal = MessageBox.Show("Do you want to close the worldview
window?", "Closing worldview window", MessageBoxButtons.OKCancel);
   }
```

VcZoomFactorModified

Ereignis von VcTree

Dieses Ereignis tritt ein, wenn der Anwender in der Komplettansicht (WorldView) die Größe des Rechtecks verändert hat oder markierte Objekte gezoomt hat. Sie können stufenlos zoomen, indem Sie bei gedrückter Strg-Taste das Mausrad drehen. In bestimmten Schritten können Sie zoomen,

indem Sie bei gedrückter Strg-Taste die Plus- bzw. Minus-Tasten des Ziffernblocks der Tastatur drücken.

	Datentyp	Beschreibung
Eigenschaften: ⇨ (no parameter)		

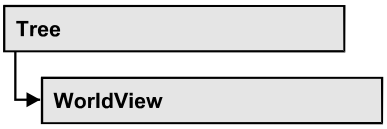
Code-Beispiel VB.NET

```
Private Sub VcTree1_VcZoomFactorModified(ByVal sender As Object, ByVal e As
NETRONIC.XTree.VcZoomFactorModifiedEventArgs) Handles
VcTree1.VcZoomFactorModified
   MsgBox("Zoomfactor: " + dummyobject1.ZoomFactor)
End Sub
```

Code-Beispiel C#

```
private void vcTree1_VcZoomFactorModified(object sender,
NETRONIC.XTree.VcZoomFactorModifiedEventArgs e)
   {
   MessageBox.Show("Zoomfactor: " + vcTree1.ZoomFactor.ToString());
   }
```

7.32 VcWorldView



Ein Objekt vom Typ **VcWorldView** bezeichnet das Komplettansicht-Fenster.

Eigenschaften

- Border
- Height
- HeightActualValue
- Left
- LeftActualValue
- MarkingColor
- Mode
- ParentHWnd
- ScrollBarMode
- Top
- TopActualValue
- UpdateBehaviorName
- Visible
- Width
- WidthActualValue

Eigenschaften

Border

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann gesetzt oder erfragt werden, ob die Komplettansicht einen Rahmen besitzt (nicht im Modus **vcPopupWindow**). Die Rahmenfarbe ist **Color.Black**. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Rahmen um die Komplettansicht (True)/kein Rahmen um die Komplettansicht (False) **Standardwert:** True

Code-Beispiel VB.NET

```
VcTree1.WorldView.Mode = VcWorldViewMode.vcNotFixed
VcTree1.WorldView.Border = True
```

Code-Beispiel C#

```
vcTree1.WorldView.Mode = VcWorldViewMode.vcNotFixed;
vcTree1.WorldView.Border = true;
```

Height

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die vertikale Ausdehnung der Komplettansicht erfragt werden. In den Postionen **vcFixedAtTop**, **vcFixedAtBottom**, **vcNot-Fixed** und **vcPopupWindow** der Eigenschaft **Mode** kann sie außerdem gesetzt werden. Die Koordinaten werden in Pixel, bezogen auf den Bildschirm, angegeben.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Höhe der Komplettansicht **Standardwert:** 100

Code-Beispiel VB.NET

```
VcTree1.WorldView.Height = 100
```

Code-Beispiel C#

```
vcTree1.WorldView.Height = 100;
```

HeightActualValue

Nur-Lese-Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die tatsächlich dargestellte vertikale Ausdehnung der Komplettansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**,

vcLVFixedAtTop von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32 {0, ...}	Tatsächliche Höhe der Komplettansicht {0, ...} **Standardwert:** 100

Code-Beispiel VB.NET

```
VcTree1.LegendView.Height = 300
```

Code-Beispiel C#

```
vcTree1.LegendView.Height = 100;
```

Left

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die linke Position der Komplettansicht erfragt werden. In den Postionen **vcNotFixed** und **vcPopupWindow** der Eigenschaft **Mode** kann sie außerdem gesetzt werden. Die Koordinaten werden in Pixel, bezogen auf den Bildschirm, angegeben.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Linke Position der Komplettansicht **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.WorldView.Left = 200
```

Code-Beispiel C#

```
vcTree1.WorldView.Left = 200;
```

LeftActualValue

Nur-Lese-Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die tatsächlich dargestellte linke Position der Komplettansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**,

vcLVFixedAtTop von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche linke Position der Komplettansicht {0, ...} **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.LegendView.LeftActualValue = 150
```

Code-Beispiel C#

```
vcTree1.LegendView.LeftActualValue = 150;
```

MarkingColor

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die Farbe der Linie des Rechtecks erfragt oder gesetzt werden, das in der Komplettansicht den aktuell gewählten Ausschnitt anzeigt. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Drawing.Color	RGB-Farbwerte ({0...255},{0...255},{0...255}) **Standardwert:** RGB(0, 0, 255)

Code-Beispiel VB.NET

```
VcTree1.WorldView.MarkingColor = Color.Red
```

Code-Beispiel C#

```
vcTree1.WorldView.MarkingColor = Color.Red;
```

Mode

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann der Modus der Komplettansicht erfragt oder gesetzt werden. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	VcWorldViewMode	Modus der Gesamtansicht **Standardwert:** vcPopupWindow
	Mögliche Werte:	
	.vcFixedAtBottom 4	Die Komplettansicht wird unten im Fenster des Steuerelements angezeigt. Das Bezugssystem der Koordinaten ist das Steuerelement. Bei dieser Einstellung kann nur die Höhe festgelegt werden, während Position und Breite vorgegeben sind.
	.vcFixedAtLeft 1	Die Komplettansicht wird links im Fenster des Steuerelements angezeigt. Das Bezugssystem der Koordinaten ist das Steuerelement. Bei dieser Einstellung kann nur die Breite festgelegt werden, während Position und Höhe vorgegeben sind.
	.vcFixedAtRight 2	Die Komplettansicht wird rechts im Fenster des Steuerelements angezeigt. Das Bezugssystem der Koordinaten ist das Steuerelement. Bei dieser Einstellung kann nur die Breite festgelegt werden, während Position und Höhe vorgegeben sind.
	.vcFixedAtTop 3	Die Komplettansicht wird oben im Fenster des Steuerelements angezeigt. Das Bezugssystem der Koordinaten ist das Steuerelement. Bei dieser Einstellung kann nur die Höhe festgelegt werden, während Position und Breite vorgegeben sind.
	.vcNotFixed 5	Die Komplettansicht ist ein untergeordnetes Kindfenster des aktuellen Vaterfensters des Steuerelements und kann an beliebiger Position mit beliebiger Ausdehnung angeordnet werden. Das Bezugssystem der Koordinaten ist das Fenster der Anwendung. Das Kindfenster ist ohne eigenen Fensterrahmen und kann vom Benutzer nicht interaktiv verschoben werden. Das Vaterfenster kann bei Bedarf über die Eigenschaft **VcWorldView.ParentHWnd** geändert werden.
	.vcPopupWindow 6	Die Komplettansicht ist ein Popup-Fenster, das einen eigenen Rahmen besitzt und vom Benutzer in Position und Größe verändert werden kann. Das Bezugssystem der Koordinaten ist der Bildschirm. Es kann über das Standard-Kontextmenü ein- bzw. ausgeschaltet oder über die **Schließen**-Schaltfläche in der Titelleiste ausgeschaltet werden.

Code-Beispiel VB.NET

```
VcTree1.WorldView.Mode = VcWorldViewMode.vcFixedAtBottom
```

Code-Beispiel C#

```
vcTree1.WorldView.Mode = VcWorldViewMode.vcFixedAtBottom;
```

ParentHWnd

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann im Modus **vcNotFixed** das HWnd-Handle des Vaterfensters festgelegt werden, wenn die Komplettansicht beispielsweise in einem selbst implementierten Rahmenfenster erscheinen soll. Standardmäßig

steht dies auf dem HWnd-Handle des Vaterfensters des VARCHART-Windows-Forms-Hauptfensters. Diese Eigenschaft kann nur zur Laufzeit verwendet werden.

	Datentyp	Beschreibung
Eigenschaftswert	OLE_HANDLE	Zugriffsnummer

ScrollBarMode

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann der Scrollbarmodus der Gesamtansicht erfragt oder gesetzt werden. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	VcWorldViewScrollBarMode	Scrollbarmodus **Standardwert:** NoScrollBar
	Mögliche Werte:	
	.vcAutomaticScrollBar 3	Anzeige einer horizontalen oder vertikalen Bildlaufleiste, wenn nötig.
	.vcHorizontalScrollBar 1	Anzeige einer horizontalen Bildlaufleiste, wenn nötig.
	.vcNoScrollBar 0	Es wird immer das vollständige Diagramm ohne Bildlaufleisten angezeigt.
	.vcVerticalScrollBar 2	Anzeige einer vertikalen Bildlaufleiste, wenn nötig.

Code-Beispiel VB.NET

```
VcTree1.WorldView.ScrollBarMode = vcAutomaticScrollbar
```

Code-Beispiel C#

```
vcTree1.WorldView.ScrollBarMode = vcAutomaticScrollBar;
```

Top

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die obere Position der Komplettansicht erfragt werden. In den Positionen **vcNotFixed** und **vcPopupWindow** der Eigenschaft **Mode** kann sie außerdem gesetzt werden. Die Koordinaten werden in Pixel, bezogen auf den Bildschirm, angegeben.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Obere Position der Komplettansicht **Standardwert:** 0

Code-Beispiel VB.NET

```
VcTree1.WorldView.Top = 20
```

Code-Beispiel C#

```
vcTree1.WorldView.Top = 20;
```

TopActualValue

Nur-Lese-Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die tatsächlich dargestellte obere Position der Komplettansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Tatsächliche obere Position der Komplettansicht {0, ...}

Code-Beispiel VB.NET

```
VcTree1.LegendView.TopActualValue = 40
```

Code-Beispiel C#

```
vcTree1.LegendView.TopActualValue = 40;
```

UpdateBehaviorName

Nur-Lese-Eigenschaft von VcWorldView

Mit dieser Eigenschaft können Sie den Namen des Aktualisierungsverhaltens erfragen oder festlegen.

	Datentyp	Beschreibung
Eigenschaftswert	System.String	Name des Aktualisierungsverhaltens

Visible

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann festgelegt oder erfragt werden, ob die Komplettansicht sichtbar ist. Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Boolean	Komplettansicht sichtbar (True)/unsichtbar (False) **Standardwert:** False

Code-Beispiel VB.NET

```
VcTree1.WorldView.Visible = True
```

Code-Beispiel C#

```
vcTree1.WorldView.Visible = true;
```

Width

Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die horizontale Ausdehnung der Komplettansicht erfragt werden. In den Positionen **vcFixedAtLeft**, **vcFixedAtRight**, **vcNot-Fixed** und **vcPopupWindow** der Eigenschaft **Mode** kann diese Eigenschaft außerdem gesetzt werden. Die Koordinaten werden in Pixel, bezogen auf den Bildschirm, angegeben.

Diese Eigenschaft kann auch auf der Eigenschaftenseite **Zusätzliche Ansichten** festgelegt werden.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32	Horizontale Ausdehnung der Komplettansicht **Standardwert:** 100

Code-Beispiel VB.NET

```
VcTree1.WorldView.Width = 200
```

Code-Beispiel C#

```
vcTree1.WorldView.Width = 200;
```

WidthActualValue

Nur-Lese-Eigenschaft von VcWorldView

Mit dieser Eigenschaft kann die tatsächlich dargestellte horizontale Ausdehnung der Komplettansicht erfragt werden. Dieser tatsächliche Wert kann in den Modi **vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** von dem eingestellten Wert abweichen, da in diesen Modi je nach Einstellung die Höhe oder Breite vorgegeben ist.

	Datentyp	Beschreibung
Eigenschaftswert	System.Int32 {0, ...}	Tatsächliche horizontale Ausdehnung der Komplettansicht {0, ...} **Standardwert:** 100

Code-Beispiel VB.NET

```
VcTree1.LegendView.WidthActualValue = 600
```

Code-Beispiel C#

```
vcTree1.LegendView.WidthActualValue = 600;
```

8 Index

A

AbsoluteBottomMarginInInches
Eigenschaft von
VcPrinter 461
AbsoluteLeftMarginInCM
Eigenschaft von
VcPrinter 461
AbsoluteLeftMarginInInches
Eigenschaft von
VcPrinter 462
AbsoluteRightMarginInCM
Eigenschaft von
VcPrinter 462
AbsoluteRightMarginInInches
Eigenschaft von
VcPrinter 463
AbsoluteTopMarginInCM
Eigenschaft von
VcPrinter 463
AbsoluteTopMarginInInches
Eigenschaft von
VcPrinter 464
ActiveNodeFilter
Eigenschaft von
VcTree 487
Add
Methode von
VcBoxCollection 273
VcBoxFormatCollection 286
VcDataRecordCollection 308
VcDataTableCollection 320
VcDataTableFieldCollection 334
VcFilterCollection 347
VcMapCollection 373
VcNodeAppearanceCollection 425
VcNodeFormatCollection 440
AddBySpecification
Methode von
VcBoxCollection 273
VcBoxFormatCollection 286
VcFilterCollection 348
VcMapCollection 374
VcNodeAppearanceCollection 425
VcNodeFormatCollection 441
AddSubCondition
Methode von
VcFilter 342
Alignment
Eigenschaft von
VcBorderBox 250
VcBoxFormatField 292
VcNodeFormatField 447
VcPrinter 464
AllData
Eigenschaft von
VcDataRecord 301
VcNode 390
Anordnung
horizontal/vertikal 85
Arrange
Methode von
VcTree 513
ArrangementDataFieldIndex
Eigenschaft von
VcTree 487
ArrangeSubtree
Methode von

VcNode 397
Auslieferung 17

B

BackgroundColor
Eigenschaft von
VcNodeAppearance 403
VcNodeFormatField 447
BackgroundColorDataFieldIndex
Eigenschaft von
VcNodeAppearance 404
VcNodeFormatField 448
BackgroundColorMapName
Eigenschaft von
VcNodeAppearance 404
VcNodeFormatField 448
Baum-Diagramm
maximale Höhe 108, 143
Baumstrukturen 45
kollabieren/expandieren 51
Vertikale/horizontale Anordnung 47
Benutzerkonten
Problem mit Control 245
Bildschirmausschnitt
verschieben 138
Border
Eigenschaft von
VcLegendView 357
VcWorldView 591
BorderArea
Eigenschaft von
VcTree 488
siehe auch
VcBorderArea 248
BorderBox
Ausrichtung 250
Methode von
VcBorderArea 248
siehe auch
VcBorderBox 250
Bottom
Eigenschaft von
VcRect 479
BottomMargin
Eigenschaft von
VcNodeFormatField 449
Box
markieren 264
Mehrfachmarkierung zulassen 135
siehe auch
VcBox 259
über Index 274
BoxByIndex
Methode von
VcBoxCollection 274
BoxByName
Methode von
VcBoxCollection 275
BoxCollection
Eigenschaft von
VcTree 488
siehe auch
VcBoxCollection 272
Boxen 61
Ausdehnung 268
Offset in Pixel umrechnen 269
Pixel in Offset umrechnen 270
Boxformat
über Index 288
BoxFormat
siehe auch
VcBoxFormat 279
BoxFormatCollection
Eigenschaft von

VcTree 488
siehe auch
VcBoxFormatCollection 285
Boxformatfeld
Ausrichtung 292
Hintergrundfarbe 297
Höhe Grafik 294
Index 294
maximale Zeilenzahl 295
Mindestbreite 296
minimale Zeilenzahl 295
Muster 298
Musterfarbe 297
Name des Formats 293
Schriftart 299
Schriftfarbe 299
Typ 300
BoxFormatField
siehe auch
VcBoxFormatField 292

C

ChildNodeCollection
Eigenschaft von
VcNode 390
Clear
Methode von
VcTree 514
Collapsed
Eigenschaft von
VcNode 391
CollapseDataFieldIndex
Eigenschaft von
VcTree 489
CollapseSubtree
Methode von
VcNode 398
Color
Eigenschaft von
VcMapEntry 380
ComparisonValueAsString
Eigenschaft von
VcFilterSubCondition 353
CompleteViewMode
Methode von
VcTree 514
ConnectionOperator
Eigenschaft von
VcFilterSubCondition 354
ConsiderFilterEntries
Eigenschaft von
VcMap 365
ConstantText
Eigenschaft von
VcNodeFormatField 449
Copy
Methode von
VcBoxCollection 275
VcBoxFormatCollection 287
VcDataTableCollection 321
VcDataTableFieldCollection 335
VcFilterCollection 348
VcMapCollection 374
VcNodeAppearanceCollection 426
VcNodeFormatCollection 441
CopyFormatField
Methode von
VcBoxFormat 282
VcNodeFormat 437
CopyNodesIntoClipboard
Methode von
VcTree 515
CopySubCondition
Methode von

VcFilter 343
Count
Eigenschaft von
VcBoxCollection 272
VcBoxFormatCollection 285
VcDataRecordCollection 307
VcDataTableCollection 319
VcDataTableFieldCollection 333
VcFilterCollection 346
VcMap 366
VcMapCollection 373
VcNodeAppearanceCollection 424
VcNodeCollection 430
VcNodeFormatCollection 439
CreateEntry
Methode von
VcMap 368
CtrlCXVProcessingEnabled
Eigenschaft von
VcTree 489
CurrentHorizontalPagesCount
Eigenschaft von
VcPrinter 465
CurrentVerticalPagesCount
Eigenschaft von
VcPrinter 465
CurrentZoomFactor
Eigenschaft von
VcPrinter 465
CutNodesIntoClipboard
Methode von
VcTree 515
CuttingMarks
Eigenschaft von
VcPrinter 466

D

DataField
Eigenschaft von
VcDataRecord 302
VcNode 392
DataFieldIndex
Eigenschaft von
VcFilterSubCondition 355
DataFieldValue
Eigenschaft von
VcMapEntry 381
DataRecord
Methode von
VcNode 399
siehe auch
VcDataRecord 301
DataRecordByID
Methode von
VcDataRecordCollection 309
DataRecordCollection
Eigenschaft von
VcDataTable 315
siehe auch
VcDataRecordCollection 307
DataTable
siehe auch
VcDataTable 315
DataTableByIndex
Methode von
VcDataTableCollection 321
DataTableByName
Methode von
VcDataTableCollection 322
DataTableCollection
Eigenschaft von
VcTree 490

siehe auch
VcDataTableCollection 319
DataTableField
siehe auch
VcDataTableField 326
DataTableFieldByIndex
Methode von
VcDataTableFieldCollection 335
DataTableFieldByName
Methode von
VcDataTableFieldCollection 336
DataTableFieldCollection
Eigenschaft von
VcDataTable 316
siehe auch
VcDataTableFieldCollection 333
DataTableName
Eigenschaft von
VcDataRecord 303
VcDataTableField 326
DateFormat
Eigenschaft von
VcDataTableField 327
Daten
aus Datei einlesen 32
Datenfeld
für Ebenennummer 145
für Kollabierzustand 144
für Tooltiptext 145
für Unterbaum-Anordnung 145
Datenfelder
Knoten 203, 204
Datenfelder für Baumstruktur 91
Datensatz
abhängiger Datensatz nicht gefunden 555
Aktualisierung 313
alle Daten 301
Anzahl in Collection 307
aus Collection entfernen 312
datenbasiertes Objekt 305
Datenfeld 302
Enumerator-Objekt 311
ID 304
Iteration, Erstwert 310
Iteration, Folgewert 312
löschen 304
Name der zugehörigen Tabelle 303
über ID 310
zu Collection hinzufügen 308
zugeordneter Datensatz 305
Datentabelle
Aktualisierung 324
Anzahl in Collection 319
Beschreibung 316
Datensatz-Auflistung 315
Enumerator-Objekt 323
Erweiterte Datentabellen setzen 494
innerhalb der Collection kopieren 321
Iteration, Erstwert 323
Iteration, Folgewert 324
Name 517
Name 317
Tabellendatenfeld-Auflistung 316
über Index 321
über Name 322
zu Collection hinzufügen 320
Datentabellen 65
erweiterte Nutzung 135
Datentabellenfeld
angezeigt 328
Anzahl in Collection 333
Datentyp 332
Datumsformat 327

editierbar 328
Enumerator-Objekt 337
Index 517
Index 329
Index des Bezugsfeldes 330
Iteration, Erstwert 337
Iteration, Folgewert 338
kopieren 335
Name 516
Name 329
Primärschlüssel 330
über Index 335
über Name 336
zu Collection hinzufügen 334
zugehöriger Tabellenname 326
Datentablle
Auflistung 490
DateOutputFormat
Eigenschaft von
VcTree 490
DatesWithHourAndMinute
Eigenschaft von
VcFilter 340
Datumsausgabeformat 132
DefaultPrinterName
Eigenschaft von
VcPrinter 466
Delete
Methode von
VcDataRecord 304
VcNode 399
DeleteEntry
Methode von
VcMap 369
DeleteNodeRecord
Methode von
VcTree 516
Description
Eigenschaft von
VcDataTable 316
DetectDataTableFieldName
Methode von
VcTree 516
DetectDataTableName
Methode von
VcTree 517
DetectFieldIndex
Methode von
VcTree 517
Deutsche Version 19
DiagramBackgroundColor
Eigenschaft von
VcTree 492
Diagramm
Ausrichtung 222
exportieren 59, 229
Hintergrundfarbe 132
Dialogfeld
Boxen bearbeiten 176
Boxen verwalten 173
Boxformat bearbeiten 180
Boxformate verwalten 178
Druckvorschau 224
Filter bearbeiten 155
Filter verwalten 153
Knotenaussehen bearbeiten 169
Knotenaussehen verwalten 165
Knotenformat bearbeiten 183
Knotenformate verwalten 178
Linie bearbeiten 188
Lizenzierung 196
Lizenzinformationen anfordern 198
Muster 189
Seite einrichten 220

Texte, Grafiken und Legende festlegen 190
Zuordnung einstellen 163
Zuordnungstabelle bearbeiten 161
Zuordnungstabellen verwalten 159
DialogFont
Eigenschaft von
VcTree 492
DocumentName
Eigenschaft von
VcPrinter 466
Doppelklick
auf Knoten 203
Double-Ausgabeformat 134
DoubleFeature
Eigenschaft von
VcNodeAppearance 405
DoubleOutputFormat
Eigenschaft von
VcTree 493
Drag & Drop 76
Druckdatum 223
Drucken 58, 228
Absolute Breite des linken Seitenrandes in cm 461
Absolute Breite des linken Seitenrandes in Zoll 462
Absolute Breite des rechten Seitenrandes in cm 462
Absolute Breite des rechten Seitenrandes in Zoll 463
Absolute Höhe des oberen Seitenrandes in cm 463
Absolute Höhe des oberen Seitenrandes in Zoll 464
Absolute Höhe des unteren Seitenrandes in cm 478
Absolute Höhe des unteren Seitenrandes in Zoll 461
aktueller Drucker 466
an Seitenzahlvorgabe anpassen 221
Drucker einrichten 228
Faltmarkierungen 469
in eine Datei 532
Zoomfaktor 221, 465, 478
Druckvorschau 224, 228
DumpConfiguration
Methode von
VcTree 518

E

Ebenen
vertikal 123
Ebenenabstand
vertikaler 143
Ebenennummer 145
Editable
Eigenschaft von
VcDataTableField 328
Eigenschaften
AbsoluteBottomMarginInInches
VcPrinter 461
AbsoluteLeftMarginInCM
VcPrinter 461
AbsoluteLeftMarginInInches
VcPrinter 462
AbsoluteRightMarginInCM
VcPrinter 462
AbsoluteRightMarginInInches
VcPrinter 463
AbsoluteTopMarginInCM
VcPrinter 463
AbsoluteTopMarginInInches
VcPrinter 464
ActiveNodeFilter
VcTree 487

Alignment
VcBorderBox 250
VcBoxFormatField 292
VcNodeFormatField 447
VcPrinter 464
AllData
VcDataRecord 301
VcNode 390
ArrangementDataFieldIndex
VcTree 487
BackgroundColor
VcNodeAppearance 403
VcNodeFormatField 447
BackgroundColorDataFieldIndex
VcNodeAppearance 404
VcNodeFormatField 448
BackgroundColorMapName
VcNodeAppearance 404
VcNodeFormatField 448
Border
VcLegendView 357
VcWorldView 591
BorderArea
VcTree 488
Bottom
VcRect 479
BottomMargin
VcNodeFormatField 449
BoxCollection
VcTree 488
BoxFormatCollection
VcTree 488
ChildNodeCollection
VcNode 390
Collapsed
VcNode 391
CollapseDataFieldIndex
VcTree 489
Color
VcMapEntry 380
ComparisonValueAsString
VcFilterSubCondition 353
ConnectionOperator
VcFilterSubCondition 354
ConsiderFilterEntries
VcMap 365
ConstantText
VcNodeFormatField 449
Count
VcBoxCollection 272
VcBoxFormatCollection 285
VcDataRecordCollection 307
VcDataTableCollection 319
VcDataTableFieldCollection 333
VcFilterCollection 346
VcMap 366
VcMapCollection 373
VcNodeAppearanceCollection 424
VcNodeCollection 430
VcNodeFormatCollection 439
CtrlCXVProcessingEnabled
VcTree 489
CurrentHorizontalPagesCount
VcPrinter 465
CurrentVerticalPagesCount
VcPrinter 465
CurrentZoomFactor
VcPrinter 465
CuttingMarks
VcPrinter 466
DataField
VcDataRecord 302
VcNode 392
DataFieldIndex

VcFilterSubCondition 355
DataFieldValue
VcMapEntry 381
DataRecordCollection
VcDataTable 315
DataTableCollection
VcTree 490
DataTableFieldCollection
VcDataTable 316
DataTableName
VcDataRecord 303
VcDataTableField 326
DateFormat
VcDataTableField 327
DateOutputFormat
VcTree 490
DatesWithHourAndMinute
VcFilter 340
DefaultPrinterName
VcPrinter 466
Description
VcDataTable 316
DiagramBackgroundColor
VcTree 492
DialogFont
VcTree 492
DocumentName
VcPrinter 466
DoubleFeature
VcNodeAppearance 405
DoubleOutputFormat
VcTree 493
Editable
VcDataTableField 328
Enabled
VcTree 493
ExtendedDataTablesEnabled
VcTree 494
FieldsSeparatedByLines
VcBoxFormat 279
VcNodeFormat 434
FieldText
VcBox 260
FilePath
VcTree 494
FilterCollection
VcTree 495
FilterName
VcFilterSubCondition 355
VcNodeAppearance 405
FirstVerticalLevelNumber
VcTree 495
FitToPage
VcPrinter 467
FoldingMarksType
VcPrinter 467
FontAntiAliasingEnabled
VcTree 496
FontBody
VcMapEntry 382
FontName
VcMapEntry 382
FontSize
VcMapEntry 383
FormatField
VcBoxFormat 280
VcNodeFormat 435
FormatFieldCount
VcBoxFormat 280
VcNodeFormat 435
FormatName
VcBox 260
VcBoxFormatField 293
VcNodeAppearance 406

VcNodeFormatField 449
FrameAroundFieldsVisible
VcNodeAppearance 406
FrameShape
VcNodeAppearance 407
GraphicsFileName
VcBorderBox 251
VcMapEntry 384
VcNodeFormatField 449
GraphicsFileNameDataFieldIndex
VcNodeFormatField 450
GraphicsFileNameMapName
VcNodeFormatField 450
GraphicsHeight
VcBoxFormatField 294
VcNodeFormatField 450
Height
VcLegendView 358
VcRect 479
VcWorldView 592
HeightActualValue
VcLegendView 358
VcWorldView 592
Hidden
VcDataTableField 328
HorizontalNodeDistance
VcTree 497
HorizontalNodeIndentWidth
VcTree 497
ID
VcDataRecord 304
VcNode 392
InbuiltMouseCursorWhileDraggingEnabled
VcTree 497
InCollapsedSubtree
VcNode 393
Index
VcBoxFormatField 294
VcDataTableField 329
VcFilterSubCondition 355
VcNodeFormatField 451
InPlaceEditingAllowed
VcTree 498
InteractionMode
VcTree 499
Left
VcLegendView 359
VcRect 480
VcWorldView 593
LeftActualValue
VcLegendView 359
VcWorldView 593
LeftBrotherNode
VcNode 393
LeftMargin
VcNodeFormatField 451
LegendElementsArrangement
VcBorderBox 252
LegendElementsBottomMargin
VcBorderBox 252
LegendElementsMaximumColumnCount
VcBorderBox 253
LegendElementsMaximumRowCount
VcBorderBox 253
LegendElementsTopMargin
VcBorderBox 253
LegendFont
VcBorderBox 253
LegendText
VcNodeAppearance 409
LegendTitle
VcBorderBox 254

LegendTitleFont
VcBorderBox 254
LegendTitleVisible
VcBorderBox 255
LegendView
VcTree 499
LevelDataFieldIndex
VcTree 500
LineColor
VcBox 261
VcNodeAppearance 409
LineColorDataFieldIndex
VcNodeAppearance 410
LineColorMapName
VcNodeAppearance 410
LineThickness
VcBox 261
VcNodeAppearance 410
LineType
VcBox 262
VcNodeAppearance 411
MapCollection
VcTree 500
MarginsShownInInches
VcPrinter 470
Marked
VcBox 264
VcNode 394
MarkedNodesFilter
VcFilterCollection 347
MarkingColor
VcWorldView 594
MaxHorizontalPagesCount
VcPrinter 470
MaximumChartRowCount
VcTree 501
MaximumTextLineCount
VcBoxFormatField 295
VcNodeFormatField 451
MaxVerticalPagesCount
VcPrinter 471
MinimumTextLineCount
VcBoxFormatField 295
VcNodeFormatField 452
MinimumWidth
VcBoxFormatField 296
VcNodeFormatField 452
Mode
VcWorldView 594
MouseProcessingEnabled
VcTree 501
Moveable
VcBox 264
MultiplePrimaryKeysAllowed
VcDataTable 317
Name
VcBox 265
VcBoxFormat 281
VcDataTable 317
VcDataTableField 329
VcFilter 340
VcMap 366
VcNodeAppearance 413
VcNodeFormat 436
NodeAppearanceCollection
VcTree 501
NodeCollection
VcTree 502
NodeCreationAllowed
VcTree 502
NodeCreationWithDialog
VcTree 503
NodeFormatCollection
VcTree 503

NodesDataTableName
VcTree 504
NodeToolTipTextDataFieldIndex
VcTree 504
Number
VcMapEntry 385
Operator
VcFilterSubCondition 355
Orientation
VcPrinter 471
Origin
VcBox 265
PageDescription
VcPrinter 472
PageDescriptionString
VcPrinter 472
PageFrame
VcPrinter 473
PageNumberMode
VcPrinter 473
PageNumbers
VcPrinter 474
PagePaddingEnabled
VcPrinter 474
PaperSize
VcPrinter 475
ParentHWnd
VcWorldView 595
ParentNode
VcNode 395
ParentNodeIDDataFieldIndex
VcTree 505
Pattern
VcMapEntry 385
VcNodeAppearance 413
PatternBackgroundColor
VcBoxFormatField 296
PatternBackgroundColorAsARGB
VcNodeFormatField 452
PatternBackgroundColorDataFieldIndex
VcNodeFormatField 453
PatternBackgroundColorMapName
VcNodeFormatField 453
PatternColor
VcNodeAppearance 417
PatternColorAsARGB
VcBoxFormatField 297
VcNodeFormatField 453
PatternColorDataFieldIndex
VcNodeAppearance 417
VcNodeFormatField 454
PatternColorMapName
VcNodeAppearance 418
VcNodeFormatField 454
PatternDataFieldIndex
VcNodeAppearance 418
PatternEx
VcBoxFormatField 298
VcNodeFormatField 455
PatternExDataFieldIndex
VcNodeFormatField 455
PatternExMapName
VcNodeFormatField 456
PatternMapName
VcNodeAppearance 418
PhantomDrawingWhileDraggingEnabled
VcTree 505
PileEffect
VcNodeAppearance 419
PrimaryKey
VcDataTableField 330
PrintDate

VcPrinter 476
Printer
VcTree 506
PrinterName
VcPrinter 476
PrintPreviewWithFirstPage
VcPrinter 476
Priority
VcBox 266
ReferencePoint
VcBox 267
RelationshipFieldIndex
VcDataTableField 330
Right
VcRect 481
RightBrotherNode
VcNode 395
RightMargin
VcNodeFormatField 456
RoundedLinkSlantsEnabled
VcTree 506
ScrollBarMode
VcLegendView 360
VcWorldView 596
Shadow
VcNodeAppearance 419
ShadowColor
VcNodeAppearance 420
Specification
VcBoxFormat 281
VcFilter 341
VcMap 367
VcNodeAppearance 420
VcNodeFormat 436
StrikeThrough
VcNodeAppearance 421
StrikeThroughColor
VcNodeAppearance 422
StringsCaseSensitive
VcFilter 341
StructureCodeDataFieldIndex
VcTree 507
StructureType
VcTree 507
SubCondition
VcFilter 341
SubConditionCount
VcFilter 342
SubtreeArrangement
VcNode 396
SubtreeNodeCollection
VcNode 396
Text
VcBorderBox 255
TextAndGraphicsCombined
VcNodeFormatField 456
TextDataFieldIndex
VcNodeFormatField 457
TextEntrySupplyingEventEnabled
VcTree 507
TextFont
VcBorderBox 256
VcBoxFormatField 299
VcNodeFormatField 457
TextFontColor
VcBoxFormatField 299
VcNodeFormatField 457
TextFontDataFieldIndex
VcNodeFormatField 458
TextFontMapName
VcNodeFormatField 458
ThreeDEffect
VcNodeAppearance 422
TitleAndLegendOnAllPages

VcPrinter 477
ToolTipChangeDuration
VcTree 508
ToolTipDuration
VcTree 508
ToolTipPointerDuration
VcTree 508
ToolTipShowAfterClick
VcTree 509
ToolTipTextSupplyingEventEnabled
VcTree 509
Top
VcLegendView 360
VcRect 481
VcWorldView 596
TopActualValue
VcLegendView 361
VcWorldView 597
TopMargin
VcNodeFormatField 458
TreeViewStyle
VcTree 510
Type
VcBorderBox 257
VcBoxFormatField 300
VcDataTableField 332
VcMap 367
VcNodeFormatField 459
UpdateBehaviorName
VcBox 267
VcWorldView 597
VcCalendarGrid
VcPrinter 478
VerticalLevelDistance
VcTree 510
VerticalNodeDistance
VcTree 511
ViewXCoordinate
VcTree 511
ViewYCoordinate
VcTree 511
Visible
VcBox 268
VcLegendView 361
VcWorldView 598
VisibleInLegend
VcNodeAppearance 423
WaitCursorEnabled
VcTree 512
Width
VcLegendView 362
VcRect 482
VcWorldView 598
WidthActualValue
VcLegendView 362
VcWorldView 599
WidthOfExteriorSurrounding
VcNodeFormat 437
WindowMode
VcLegendView 363
WorldView
VcTree 512
ZoomFactor
VcTree 513
ZoomFactorAsDouble
VcPrinter 478
ZoomingPerMouseWheelAllowed
VcTree 513
Eigenschaftenseite
Allgemeines 132
Außenbereich 140
Knoten 144
Layout 142
Objekte 151

Zusätzliche Ansichten 147
Enabled
Eigenschaft von
VcTree 493
EndLoading
Methode von
VcTree 518
Ereignisobjekte
VcBoxModifiedEventArgs 547
VcBoxModifyingEventArgs 547
VcDataRecordCreatedEventArgs 550
VcDataRecordCreatingEventArgs 551
VcDataRecordDeletedEventArgs 552
VcDataRecordDeletingEventArgs 553
VcDataRecordModifiedEventArgs 554
VcDataRecordModifyingEventArgs 555
VcDataRecordNotFoundEventArgs 556
VcDragCompletingEventArgs 558
VcDragStartingEventArgs 559
VcFieldSelectingEventArgs 560
VcHelpRequestedEventArgs 561
VcLEgendViewClosedEventArgs 564
VcNodeModifiedEventArgs 573, 574
VcTextEntrySupplyingEventArgs 579
Ereignisse 78
KeyDown
VcTree 543
KeyPress
VcTree 543
KeyUp
VcTree 544
VcBoxLeftClicking
VcTree 545
VcBoxLeftDoubleClicking
VcTree 546
VcBoxModified
VcTree 546
VcBoxModifying
VcTree 547
VcBoxRightClicking
VcTree 548
VcDataModified
VcTree 549
VcDataRecordCreated
VcTree 550
VcDataRecordCreating
VcTree 551
VcDataRecordDeleted
VcTree 552
VcDataRecordDeleting
VcTree 553
VcDataRecordModified
VcTree 554
VcDataRecordModifying
VcTree 555
VcDataRecordNotFound
VcTree 555
VcDiagramLeftClicking
VcTree 556
VcDiagramLeftDoubleClicking
VcTree 557
VcDiagramRightClicking
VcTree 557
VcDragCompleting
VcTree 558
VcDragStarting
VcTree 559
VcErrorOccuring
VcTree 559
VcFieldSelecting

VcTree 560
VcHelpRequested
VcTree 561
VcInPlaceEditorShowing
VcTree 562
VcLegendViewClosed
VcTree 564
VcMouseDoubleClicking
VcTree 565
VcMouseDown
VcTree 566
VcMouseMove
VcTree 566
VcMouseUp
VcTree 567
VcNodeCollapsing
VcTree 568
VcNodeCreated
VcTree 568
VcNodeCreating
VcTree 569
VcNodeDeleted
VcTree 570
VcNodeDeleting
VcTree 570
VcNodeExpanding
VcTree 571
VcNodeLeftClicking
VcTree 572
VcNodeLeftDoubleClicking
VcTree 572
VcNodeModified
VcTree 573
VcNodeModifying
VcTree 574
VcNodeRightClicking
VcTree 575
VcNodesMarked
VcTree 576
VcNodesMarking
VcTree 577
VcStatusLineTextShowing
VcTree 578
VcTextEntrySupplying
VcTree 579
VcToolTipTextSupplying
VcTree 587
VcWorldViewClosed
VcTree 589
VcZoomFactorModified
VcTree 589
Erzeugemodus 227
Evaluate
Methode von
VcFilter 343
Expandieren 99, 231
ExpandSubtree
Methode von
VcNode 400
Export 229
ExportGraphicsToFileEx
Methode von
VcTree 519
ExtendedDataTablesEnabled
Eigenschaft von
VcTree 494

F

Fehlermeldungen 244
FieldsSeparatedByLines
Eigenschaft von
VcBoxFormat 279
VcNodeFormat 434
FieldText

Eigenschaft von
VcBox 260
FilePath
Eigenschaft von
VcTree 494
Filter 79
bearbeiten 155
für Knoten 35
siehe auch
VcFilter 339
über Index 349
Vergleichswert 156
verwalten 153
FilterByIndex
Methode von
VcFilterCollection 349
FilterByName
Methode von
VcFilterCollection 349
FilterCollection
Eigenschaft von
VcTree 495
siehe auch
VcFilterCollection 346
FilterName
Eigenschaft von
VcFilterSubCondition 355
VcNodeAppearance 405
FilterSubCondition
siehe auch
VcFilterSubCondition 353
FirstBox
Methode von
VcBoxCollection 276
FirstDataRecord
Methode von
VcDataRecordCollection 310
FirstDataTable
Methode von
VcDataTableCollection 322
FirstDataTableField
Methode von
VcDataTableFieldCollection 336
FirstFilter
Methode von
VcFilterCollection 350
FirstFormat
Methode von
VcBoxFormatCollection 287
VcNodeFormatCollection 442
FirstMap
Methode von
VcMapCollection 375
FirstMapEntry
Methode von
VcMap 369
FirstNode
Methode von
VcNodeCollection 431
FirstNodeAppearance
Methode von
VcNodeAppearanceCollection 426
FirstVerticalLevelNumber
Eigenschaft von
VcTree 495
FitToPage
Eigenschaft von
VcPrinter 467
FoldingMarksType
Eigenschaft von
VcPrinter 467
FontAntiAliasingEnabled
Eigenschaft von
VcTree 496

FontBody
Eigenschaft von
VcMapEntry 382
FontName
Eigenschaft von
VcMapEntry 382
Fonts 246
FontSize
Eigenschaft von
VcMapEntry 383
FormatByIndex
Methode von
VcBoxFormatCollection 288
VcNodeFormatCollection 442
FormatByName
Methode von
VcBoxFormatCollection 288
VcNodeFormatCollection 443
FormatField
Eigenschaft von
VcBoxFormat 280
VcNodeFormat 435
FormatFieldCount
Eigenschaft von
VcBoxFormat 280
VcNodeFormat 435
FormatName
Eigenschaft von
VcBox 260
VcBoxFormatField 293
VcNodeAppearance 406
VcNodeFormatField 449
Formular
anpassen 25
FrameAroundFieldsVisible
Eigenschaft von
VcNodeAppearance 406
FrameShape
Eigenschaft von
VcNodeAppearance 407

G

Gesamtbaum 228, 232
GetActualExtent
Methode von
VcBox 268
GetAValueFromARGB
Methode von
VcTree 521
GetBValueFromARGB
Methode von
VcTree 522
GetEnumerator
Methode von
VcBoxCollection 276
VcBoxFormat 283
VcBoxFormatCollection 289
VcDataRecordCollection 311
VcDataTableCollection 323
VcDataTableFieldCollection 337
VcFilter 344
VcFilterCollection 350
VcMapCollection 375
VcNodeAppearanceCollection 427
VcNodeCollection 431
VcNodeFormat 438
VcNodeFormatCollection 443
GetGValueFromARGB
Methode von
VcTree 523
GetMapEntry
Methode von
VcMap 370
GetNodeByID

Methode von
VcTree 523
GetRValueFromARGB
Methode von
VcTree 524
GetTopLeftPixel
Methode von
VcBox 269
GetXYOffset
Methode von
VcBox 269
Grafik
exportieren 59
Grafik exportieren 229
Grafikformat 80
GraphicsFileName
Eigenschaft von
VcBorderBox 251
VcMapEntry 384
VcNodeFormatField 449
GraphicsFileNameDataFieldIndex
Eigenschaft von
VcNodeFormatField 450
GraphicsFileNameMapName
Eigenschaft von
VcNodeFormatField 450
GraphicsHeight
Eigenschaft von
VcBoxFormatField 294
VcNodeFormatField 450

H

Height
Eigenschaft von
VcLegendView 358
VcRect 479
VcWorldView 592
HeightActualValue
Eigenschaft von
VcLegendView 358
VcWorldView 592
Hidden
Eigenschaft von
VcDataTableField 328
Hintergrundfarbe 132
Horizontal anordnen 85, 232
Horizontale Knoteneinrückung 143
Horizontaler Knotenabstand 143
HorizontalNodeDistance
Eigenschaft von
VcTree 497
HorizontalNodeIndentWidth
Eigenschaft von
VcTree 497

I

ID
Eigenschaft von
VcDataRecord 304
VcNode 392
IdentifyFormatField
Methode von
VcBox 269
VcTree 525
IdentifyObject
Methode von
VcDataRecord 305
IdentifyObjectAt
Methode von
VcTree 526
ImportConfiguration
Methode von
VcTree 528

InbuiltMouseCursorWhileDraggingEnabled
Eigenschaft von
VcTree 497
InCollapsedSubtree
Eigenschaft von
VcNode 393
Index
Eigenschaft von
VcBoxFormatField 294
VcDataTableField 329
VcFilterSubCondition 355
VcNodeFormatField 451
InPlaceEditingAllowed
Eigenschaft von
VcTree 498
InsertNodeRecord
Methode von
VcTree 528
InsertNodeRecordEx
Methode von
VcTree 529
Installation 14
InteractionMode
Eigenschaft von
VcTree 499
Internet 59, 125, 197, 229
IsValid
Methode von
VcFilter 344
VcFilterSubCondition 356

K

KeyDown
Ereignis von
VcTree 543
KeyPress
Ereignis von
VcTree 543
KeyUp
Ereignis von
VcTree 544
Knoten 90
3D-Effekt 170
ausschneiden 210
bearbeiten 211
Datensatz 399
Doppelrahmen 169, 170
einfügen 210, 230
erzeugen 206
Gestaffelt 172
Hintergrundfarbe 170
horizontaler Knotenabstand 143
ID 392
interaktiv erzeugen 240, 241
Knotenaussehen 37, 94
Knotenaussehen bearbeiten 169
Knotenform 169
Knotenformat 41, 96
kopieren 210
löschen 210
markieren 34, 209, 396
Markierungstyp 145, 209
mit Teilbaum umhängen 212
neue Knoten bearbeiten 136
neue zulassen 136
nicht durchtrennen 221
Schatten 171
vertikaler Knotenabstand 143
zugeordneter Datensatz 400
Knotenaussehen
3D-Effekt 422
Doppelte Umrahmung 405
Durchstreichmuster 421

Durchstreichmuster, Farbe 422
Filter 405
Format 406
Hintergrundfarbe 404
Knotenstapel 419
Legende 423
Legendentext 409
Liniendicke 411
Linienfarbe 409, 410
Linienfarbenzuordnungstabelle 410
Linientyp 412
Name 413
Rahmen um Felder 407
Rahmenform 407
Schatten 419
Spezifikation 420
Knotenaussehen-Auflistung
Anzahl 424
Enumerator 427
erstes Knotenaussehen 426
hinzufügen 425
hinzufügen über Spezifikation 425
kopieren 426
löschen 429
nächstes Knotenaussehen 428
Zugriff über Index 428
Zugriff über Name 429
Knotenformat-Auflistung
Anzahl 439
Enumerator 443
Erstes Format 442
hinzufügen 441
hinzufügen über Spezifikation 441
kopieren 441
löschen 445
nächstes Format 444
Zugriff über Index 442
zugriff über Name 443
Knotenformatfeld
Hintergrundfarbe 453
Musterfarbe 454
Kollabieren 99, 231
Kollabierzustand 144
Komplettansicht 102, 228
Konfiguration 60, 134
speichern 518
Kontextmenü
abschalten 242
für das Diagramm 227
für Knoten 230
Kundendienst 21

L

Leerseiten unterdrücken 221
Left
Eigenschaft von
VcLegendView 359
VcRect 480
VcWorldView 593
LeftActualValue
Eigenschaft von
VcLegendView 359
VcWorldView 593
LeftBrotherNode
Eigenschaft von
VcNode 393
LeftMargin
Eigenschaft von
VcNodeFormatField 451
Legende
Anordnung 194
Anordung 195
erweiterte Attribute 194
Schrift 195

Titel 194
LegendElementsArrangement
Eigenschaft von
VcBorderBox 252
LegendElementsBottomMargin
Eigenschaft von
VcBorderBox 252
LegendElementsMaximumColumnCount
Eigenschaft von
VcBorderBox 253
LegendElementsMaximumRowCount
Eigenschaft von
VcBorderBox 253
LegendElementsTopMargin
Eigenschaft von
VcBorderBox 253
Legendenansicht 106, 228
LegendFont
Eigenschaft von
VcBorderBox 253
LegendText
Eigenschaft von
VcNodeAppearance 409
LegendTitle
Eigenschaft von
VcBorderBox 254
LegendTitleFont
Eigenschaft von
VcBorderBox 254
LegendTitleVisible
Eigenschaft von
VcBorderBox 255
Legendview 106
LegendView
Eigenschaft von
VcTree 499
siehe auch
VcLegendView 357
LevelDataFieldIndex
Eigenschaft von
VcTree 500
LineColor
Eigenschaft von
VcBox 261
VcNodeAppearance 409
LineColorDataFieldIndex
Eigenschaft von
VcNodeAppearance 410
LineColorMapName
Eigenschaft von
VcNodeAppearance 410
LineThickness
Eigenschaft von
VcBox 261
VcNodeAppearance 410
LineType
Eigenschaft von
VcBox 262
VcNodeAppearance 411
Lizenzierung 16, 139, 239
Load
Methode von
VcTree 529

M

MakeARGB
Methode von
VcTree 530
Map
siehe auch
VcMap 365
MapByIndex
Methode von

VcMapCollection 375
MapByName
Methode von
VcMapCollection 376
MapCollection
Eigenschaft von
VcTree 500
siehe auch
VcMapCollection 372
MapEntry
siehe auch
VcMapEntry 380
MarginsShownInInches
Eigenschaft von
VcPrinter 470
Marked
Eigenschaft von
VcBox 264
VcNode 394
MarkedNodesFilter
Eigenschaft von
VcFilterCollection 347
Markiermodus 227
Markierungstyp 145
Knoten 34
MarkingColor
Eigenschaft von
VcWorldView 594
MaxHorizontalPagesCount
Eigenschaft von
VcPrinter 470
maximale Höhe des Baum-Diagramms 108, 143
MaximumChartRowCount
Eigenschaft von
VcTree 501
MaximumTextLineCount
Eigenschaft von
VcBoxFormatField 295
VcNodeFormatField 451
MaxVerticalPagesCount
Eigenschaft von
VcPrinter 471
Methoden
Add
VcBoxCollection 273
VcBoxFormatCollection 286
VcDataRecordCollection 308
VcDataTableCollection 320
VcDataTableFieldCollection 334
VcFilterCollection 347
VcMapCollection 373
VcNodeAppearanceCollection 425
VcNodeFormatCollection 440
AddBySpecification
VcBoxCollection 273
VcBoxFormatCollection 286
VcFilterCollection 348
VcMapCollection 374
VcNodeAppearanceCollection 425
VcNodeFormatCollection 441
AddSubCondition
VcFilter 342
Arrange
VcTree 513
ArrangeSubtree
VcNode 397
BorderBox
VcBorderArea 248
BoxByIndex
VcBoxCollection 274
BoxByName
VcBoxCollection 275
Clear

VcTree 514
CollapseSubtree
VcNode 398
CompleteViewMode
VcTree 514
Copy
VcBoxCollection 275
VcBoxFormatCollection 287
VcDataTableCollection 321
VcDataTableFieldCollection 335
VcFilterCollection 348
VcMapCollection 374
VcNodeAppearanceCollection 426
VcNodeFormatCollection 441
CopyFormatField
VcBoxFormat 282
VcNodeFormat 437
CopyNodesIntoClipboard
VcTree 515
CopySubCondition
VcFilter 343
CreateEntry
VcMap 368
CutNodesIntoClipboard
VcTree 515
DataRecord
VcNode 399
DataRecordByID
VcDataRecordCollection 309
DataTableByIndex
VcDataTableCollection 321
DataTableByName
VcDataTableCollection 322
DataTableFieldByIndex
VcDataTableFieldCollection 335
DataTableFieldByName
VcDataTableFieldCollection 336
Delete
VcDataRecord 304
VcNode 399
DeleteEntry
VcMap 369
DeleteNodeRecord
VcTree 516
DetectDataTableFieldName
VcTree 516
DetectDataTableName
VcTree 517
DetectFieldIndex
VcTree 517
DumpConfiguration
VcTree 518
EndLoading
VcTree 518
Evaluate
VcFilter 343
ExpandSubtree
VcNode 400
ExportGraphicsToFileEx
VcTree 519
FilterByIndex
VcFilterCollection 349
FilterByName
VcFilterCollection 349
FirstBox
VcBoxCollection 276
FirstDataRecord
VcDataRecordCollection 310
FirstDataTable
VcDataTableCollection 322
FirstDataTableField
VcDataTableFieldCollection 336
FirstFilter
VcFilterCollection 350

FirstFormat
VcBoxFormatCollection 287
VcNodeFormatCollection 442
FirstMap
VcMapCollection 375
FirstMapEntry
VcMap 369
FirstNode
VcNodeCollection 431
FirstNodeAppearance
VcNodeAppearanceCollection 426
FormatByIndex
VcBoxFormatCollection 288
VcNodeFormatCollection 442
FormatByName
VcBoxFormatCollection 288
VcNodeFormatCollection 443
GetActualExtent
VcBox 268
GetAValueFromARGB
VcTree 521
GetBValueFromARGB
VcTree 522
GetEnumerator
VcBoxCollection 276
VcBoxFormat 283
VcBoxFormatCollection 289
VcDataRecordCollection 311
VcDataTableCollection 323
VcDataTableFieldCollection 337
VcFilter 344
VcFilterCollection 350
VcMapCollection 375
VcNodeAppearanceCollection 427
VcNodeCollection 431
VcNodeFormat 438
VcNodeFormatCollection 443
GetGValueFromARGB
VcTree 523
GetMapEntry
VcMap 370
GetNodeByID
VcTree 523
GetRValueFromARGB
VcTree 524
GetTopLeftPixel
VcBox 269
GetXYOffset
VcBox 269
IdentifyFormatField
VcBox 269
VcTree 525
IdentifyObject
VcDataRecord 305
IdentifyObjectAt
VcTree 526
ImportConfiguration
VcTree 528
InsertNodeRecord
VcTree 528
InsertNodeRecordEx
VcTree 529
IsValid
VcFilter 344
VcFilterSubCondition 356
Load
VcTree 529
MakeARGB
VcTree 530
MapByIndex
VcMapCollection 375
MapByName
VcMapCollection 376
NextBox

VcBoxCollection 277
NextDataRecord
VcDataRecordCollection 312
NextDataTable
VcDataTableCollection 324
NextDataTableField
VcDataTableFieldCollection 338
NextFilter
VcFilterCollection 351
NextFormat
VcBoxFormatCollection 289
VcNodeFormatCollection 444
NextMap
VcMapCollection 376
NextMapEntry
VcMap 370
NextNode
VcNodeCollection 432
NextNodeAppearance
VcNodeAppearanceCollection 428
NodeAppearanceByIndex
VcNodeAppearanceCollection 428
NodeAppearanceByName
VcNodeAppearanceCollection 429
PasteNodesFromClipboard
VcTree 531
PrintEx
VcTree 531
PrintToFile
VcTree 532
RelatedDataRecord
VcDataRecord 305
VcNode 400
Remove
VcBoxCollection 277
VcBoxFormatCollection 290
VcDataRecordCollection 312
VcFilterCollection 351
VcMapCollection 377
VcNodeAppearanceCollection 429
VcNodeFormatCollection 445
RemoveFormatField
VcBoxFormat 283
VcNodeFormat 438
RemoveSubCondition
VcFilter 345
Reset
VcTree 533
SaveAsEx
VcTree 533
ScrollToNode
VcTree 534
SelectMaps
VcMapCollection 378
SelectNodes
VcNodeCollection 432
SetImageResource
VcTree 535
SetXYOffset
VcBox 270
SetXYOffsetByTopLeftPixel
VcBox 270
ShowAboutDialog
VcTree 536
ShowExportGraphicsDialog
VcTree 536
ShowNodeEditDialog
VcTree 538
ShowPageSetupDialog
VcTree 538
ShowPrintDialog
VcTree 539
ShowPrinterSetupDialog
VcTree 539

ShowPrintPreviewDialog
VcTree 539
SuspendUpdate
VcTree 540
Update
VcBoxCollection 278
VcDataRecordCollection 313
VcDataTableCollection 324
VcLegendView 364
VcMapCollection 378
VcNode 401
UpdateNodeRecord
VcTree 541
Zoom
VcTree 542
ZoomOnMarkedNodes
VcTree 542
MinimumTextLineCount
Eigenschaft von
VcBoxFormatField 295
VcNodeFormatField 452
MinimumWidth
Eigenschaft von
VcBoxFormatField 296
VcNodeFormatField 452
Mode
Eigenschaft von
VcWorldView 594
MouseProcessingEnabled
Eigenschaft von
VcTree 501
Moveable
Eigenschaft von
VcBox 264
MultiplePrimaryKeysAllowed
Eigenschaft von
VcDataTable 317

N

Name
Eigenschaft von
VcBox 265
VcBoxFormat 281
VcDataTable 317
VcDataTableField 329
VcFilter 340
VcMap 366
VcNodeAppearance 413
VcNodeFormat 436
Navigation
Tastatur 200, 205
NextBox
Methode von
VcBoxCollection 277
NextDataRecord
Methode von
VcDataRecordCollection 312
NextDataTable
Methode von
VcDataTableCollection 324
NextDataTableField
Methode von
VcDataTableFieldCollection 338
NextFilter
Methode von
VcFilterCollection 351
NextFormat
Methode von
VcBoxFormatCollection 289
VcNodeFormatCollection 444
NextMap
Methode von
VcMapCollection 376
NextMapEntry

Methode von
VcMap 370
NextNode
Methode von
VcNodeCollection 432
NextNodeAppearance
Methode von
VcNodeAppearanceCollection 428
Node
siehe auch
VcNode 389
NodeAppearance
siehe auch
VcNodeAppearance 402
NodeAppearanceByIndex
Methode von
VcNodeAppearanceCollection 428
NodeAppearanceByName
Methode von
VcNodeAppearanceCollection 429
NodeAppearanceCollection
Eigenschaft von
VcTree 501
siehe auch
VcNodeAppearanceCollection 424
NodeCollection
Eigenschaft von
VcTree 502
siehe auch
VcNodeCollection 430
NodeCreationAllowed
Eigenschaft von
VcTree 502
NodeCreationWithDialog
Eigenschaft von
VcTree 503
NodeFormat
siehe auch
VcNodeFormat 434
NodeFormatCollection
Eigenschaft von
VcTree 503
siehe auch
VcNodeFormatCollection 439
NodeFormatField
siehe auch
VcNodeFormatField 446
NodesDataTableName
Eigenschaft von
VcTree 504
NodeToolTipTextDataFieldIndex
Eigenschaft von
VcTree 504
Number
Eigenschaft von
VcMapEntry 385

O

Objekte
VcBorderArea 248
VcBorderBox 250
VcBox 259
VcBoxCollection 272
VcBoxFormat 279
VcBoxFormatCollection 285
VcBoxFormatField 292
VcDataRecord 301
VcDataRecordCollection 307
VcDataTable 315
VcDataTableCollection 319
VcDataTableField 326
VcDataTableFieldCollection 333
VcFilter 339
VcFilterCollection 346

VcFilterSubCondition 353
VcLegendView 357
VcMap 365
VcMapCollection 372
VcMapEntry 380
VcNode 389
VcNodeAppearance 402
VcNodeAppearanceCollection 424
VcNodeCollection 430
VcNodeFormat 434
VcNodeFormatCollection 439
VcNodeFormatField 446
VcPrinter 460
VcRect 479
VcTree 483
VcWorldView 591

Operator
Eigenschaft von
VcFilterSubCondition 355

Orientation
Eigenschaft von
VcPrinter 471

Origin
Eigenschaft von
VcBox 265

P

PageDescription
Eigenschaft von
VcPrinter 472

PageDescriptionString
Eigenschaft von
VcPrinter 472

PageFrame
Eigenschaft von
VcPrinter 473

PageNumberMode
Eigenschaft von
VcPrinter 473

PageNumbers
Eigenschaft von
VcPrinter 474

PagePaddingEnabled
Eigenschaft von
VcPrinter 474

PaperSize
Eigenschaft von
VcPrinter 475

ParentHWnd
Eigenschaft von
VcWorldView 595

ParentNode
Eigenschaft von
VcNode 395

ParentNodeIDDataFieldIndex
Eigenschaft von
VcTree 505

PasteNodesFromClipboard
Methode von
VcTree 531

Pattern
Eigenschaft von
VcMapEntry 385
VcNodeAppearance 413

PatternBackgroundColor
Eigenschaft von
VcBoxFormatField 296

PatternBackgroundColorAsARGB
Eigenschaft von
VcNodeFormatField 452

PatternBackgroundColorDataFieldIndex
Eigenschaft von
VcNodeFormatField 453

PatternBackgroundColorMapName
Eigenschaft von
VcNodeFormatField 453
PatternColor
Eigenschaft von
VcNodeAppearance 417
PatternColorAsARGB
Eigenschaft von
VcBoxFormatField 297
VcNodeFormatField 453
PatternColorDataFieldIndex
Eigenschaft von
VcNodeAppearance 417
VcNodeFormatField 454
PatternColorMapName
Eigenschaft von
VcNodeAppearance 418
VcNodeFormatField 454
PatternDataFieldIndex
Eigenschaft von
VcNodeAppearance 418
PatternEx
Eigenschaft von
VcBoxFormatField 298
VcNodeFormatField 455
PatternExDataFieldIndex
Eigenschaft von
VcNodeFormatField 455
PatternExMapName
Eigenschaft von
VcNodeFormatField 456
PatternMapName
Eigenschaft von
VcNodeAppearance 418
PDF-Dateien
Export 113
Performance 243
PhantomDrawingWhileDraggingEnabled
Eigenschaft von
VcTree 505
PileEffect
Eigenschaft von
VcNodeAppearance 419
Plattformen x86 und x64 109
Ports 85
Primärschlüssel
zusammengesetzt 317
PrimaryKey
Eigenschaft von
VcDataTableField 330
PrintDate
Eigenschaft von
VcPrinter 476
Printer
Eigenschaft von
VcTree 506
siehe auch
VcPrinter 460
PrinterName
Eigenschaft von
VcPrinter 476
PrintEx
Methode von
VcTree 531
PrintPreviewWithFirstPage
Eigenschaft von
VcPrinter 476
PrintToFile
Methode von
VcTree 532
Priorität 166
Boxen 175
Priority

Eigenschaft von
VcBox 266
Publizieren im Internet 59, 125, 197, 229

R

Rahmen
außen 222
Rect
siehe auch
VcRect 479
ReferencePoint
Eigenschaft von
VcBox 267
RelatedDataRecord
Methode von
VcDataRecord 305
VcNode 400
RelationshipFieldIndex
Eigenschaft von
VcDataTableField 330
Remove
Methode von
VcBoxCollection 277
VcBoxFormatCollection 290
VcDataRecordCollection 312
VcFilterCollection 351
VcMapCollection 377
VcNodeAppearanceCollection 429
VcNodeFormatCollection 445
RemoveFormatField
Methode von
VcBoxFormat 283
VcNodeFormat 438
RemoveSubCondition
Methode von
VcFilter 345
Reset
Methode von
VcTree 533
Return Status 78
Right
Eigenschaft von
VcRect 481
RightBrotherNode
Eigenschaft von
VcNode 395
RightMargin
Eigenschaft von
VcNodeFormatField 456
RoundedLinkSlantsEnabled
Eigenschaft von
VcTree 506
Rückgabewerte 78

S

SaveAsEx
Methode von
VcTree 533
Schnittmarkierungen 222
Schnittstelle
einrichten 26
Schriften
Anti-Aliasing 496
ScrollBarMode
Eigenschaft von
VcLegendView 360
VcWorldView 596
ScrollToNode
Methode von
VcTree 534
Seite einrichten 227
Seitennummerierung 222
Seitenränder 223

SelectMaps
Methode von
VcMapCollection 378
SelectNodes
Methode von
VcNodeCollection 432
SetImageResource
Methode von
VcTree 535
SetXYOffset
Methode von
VcBox 270
SetXYOffsetByTopLeftPixel
Methode von
VcBox 270
Shadow
Eigenschaft von
VcNodeAppearance 419
ShadowColor
Eigenschaft von
VcNodeAppearance 420
ShowAboutDialog
Methode von
VcTree 536
ShowExportGraphicsDialog
Methode von
VcTree 536
ShowNodeEditDialog
Methode von
VcTree 538
ShowPageSetupDialog
Methode von
VcTree 538
ShowPrintDialog
Methode von
VcTree 539
ShowPrinterSetupDialog
Methode von
VcTree 539
ShowPrintPreviewDialog
Methode von
VcTree 539
Sicherheitsrichtlinien
Laufzeit 104
Specification
Eigenschaft von
VcBoxFormat 281
VcFilter 341
VcMap 367
VcNodeAppearance 420
VcNodeFormat 436
Sprachanpassung 116
Statuszeilentext 118
Strg-C, -X und -V verarbeiten 135
StrikeThrough
Eigenschaft von
VcNodeAppearance 421
StrikeThroughColor
Eigenschaft von
VcNodeAppearance 422
StringsCaseSensitive
Eigenschaft von
VcFilter 341
StructureCodeDataFieldIndex
Eigenschaft von
VcTree 507
StructureType
Eigenschaft von
VcTree 507
Struktur 119
SubCondition
Eigenschaft von
VcFilter 341
SubConditionCount

Eigenschaft von
VcFilter 342
SubtreeArrangement
Eigenschaft von
VcNode 396
SubtreeNodeCollection
Eigenschaft von
VcNode 396
SuspendUpdate
Methode von
VcTree 540

T

Teilbaum 215, 218, 228, 232
expandieren 231
kollabieren 231
Teilstrukturen
horizontal und vertikal anordnen 215
kollabieren und expandieren 218
Text
Eigenschaft von
VcBorderBox 255
TextAndGraphicsCombined
Eigenschaft von
VcNodeFormatField 456
TextDataFieldIndex
Eigenschaft von
VcNodeFormatField 457
TextEntrySupplyingEventEnabled
Eigenschaft von
VcTree 507
TextFont
Eigenschaft von
VcBorderBox 256
VcBoxFormatField 299
VcNodeFormatField 457
TextFontColor
Eigenschaft von
VcBoxFormatField 299
VcNodeFormatField 457
TextFontDataFieldIndex
Eigenschaft von
VcNodeFormatField 458
TextFontMapName
Eigenschaft von
VcNodeFormatField 458
ThreeDEffect
Eigenschaft von
VcNodeAppearance 422
Titel
in Darstellung wiederholen 221
TitleAndLegendOnAllPages
Eigenschaft von
VcPrinter 477
Tooltip
Datenfeld für Text 145
ToolTip
Dauer bis zur Anzeige 509
Erscheinungsdauer 508
Verschwinden auf Klick 509
Wechseldauer 508
ToolTipChangeDuration
Eigenschaft von
VcTree 508
ToolTipDuration
Eigenschaft von
VcTree 508
ToolTipPointerDuration
Eigenschaft von
VcTree 508
Tooltips 136
zur Laufzeit 121
ToolTipShowAfterClick
Eigenschaft von

VcTree 509
ToolTipTextSupplyingEventEnabled
Eigenschaft von
VcTree 509
Top
Eigenschaft von
VcLegendView 360
VcRect 481
VcWorldView 596
TopActualValue
Eigenschaft von
VcLegendView 361
VcWorldView 597
TopMargin
Eigenschaft von
VcNodeFormatField 458
Tree
siehe auch
VcTree 483
TreeView-Stil 55, 122, 142
TreeViewStyle
Eigenschaft von
VcTree 510
Type
Eigenschaft von
VcBorderBox 257
VcBoxFormatField 300
VcDataTableField 332
VcMap 367
VcNodeFormatField 459

U

Unterbaum
Anordnung in Datenfeld speichern 88
komplett expandieren 231
komplett horizontal anordnen 232
Unterbaum-Anordnung in Datenfeld 145
Update
Methode von
VcBoxCollection 278
VcDataRecordCollection 313
VcDataTableCollection 324
VcLegendView 364
VcMapCollection 378
VcNode 401
UpdateBehaviorName
Eigenschaft von
VcBox 267
VcWorldView 597
UpdateNodeRecord
Methode von
VcTree 541

V

VARCHART XNet
automatisch skalieren 25
VARCHART XTree
automatisch skalieren 25
im Formular plazieren 24
VcBorderArea 248
BorderBox 248
VcBorderBox 250
Alignment 250
GraphicsFileName 251
LegendElementsArrangement 252
LegendElementsBottomMargin 252
LegendElementsMaximumColumnCount 253
LegendElementsMaximumRowCount 253
LegendElementsTopMargin 253
LegendFont 253

LegendTitle 254
LegendTitleFont 254
LegendTitleVisible 255
Text 255
TextFont 256
Type 257
VcBox 259
FieldText 260
FormatName 260
GetActualExtent 268
GetTopLeftPixel 269
GetXYOffset 269
IdentifyFormatField 269
LineColor 261
LineThickness 261
LineType 262
Marked 264
Moveable 264
Name 265
Origin 265
Priority 266
ReferencePoint 267
SetXYOffset 270
SetXYOffsetByTopLeftPixel 270
UpdateBehaviorName 267
Visible 268
VcBoxCollection 272
Add 273
AddBySpecification 273
BoxByIndex 274
BoxByName 275
Copy 275
Count 272
FirstBox 276
GetEnumerator 276
NextBox 277
Remove 277
Update 278
VcBoxFormat 279
CopyFormatField 282
FieldsSeparatedByLines 279
FormatField 280
FormatFieldCount 280
GetEnumerator 283
Name 281
RemoveFormatField 283
Specification 281
VcBoxFormatCollection 285
Add 286
AddBySpecification 286
Copy 287
Count 285
FirstFormat 287
FormatByIndex 288
FormatByName 288
GetEnumerator 289
NextFormat 289
Remove 290
VcBoxFormatField 292
Alignment 292
FormatName 293
GraphicsHeight 294
Index 294
MaximumTextLineCount 295
MinimumTextLineCount 295
MinimumWidth 296
PatternBackgroundColor 296
PatternColorAsARGB 297
PatternEx 298
TextFont 299
TextFontColor 299
Type 300
VcBoxLeftClicking
Ereignis von

VcTree 545
VcBoxLeftDoubleClicking
Ereignis von
VcTree 546
VcBoxModified
Ereignis von
VcTree 546
VcBoxModifiedEventArgs
Ereignisobjekt von
VcBoxModified 547
VcBoxModifying
Ereignis von
VcTree 547
VcBoxModifyingEventArgs
Ereignisobjekt von
VcBoxModifying 547
VcBoxRightClicking
Ereignis von
VcTree 548
VcCalendarGrid
Eigenschaft von
VcPrinter 478
VcDataModified
Ereignis von
VcTree 549
VcDataRecord 301
AllData 301
DataField 302
DataTableName 303
Delete 304
ID 304
IdentifyObject 305
RelatedDataRecord 305
VcDataRecordCollection 307
Add 308
Count 307
DataRecordByID 309
FirstDataRecord 310
GetEnumerator 311
NextDataRecord 312
Remove 312
Update 313
VcDataRecordCreated
Ereignis von
VcTree 550
VcDataRecordCreatedEventArgs
Ereignisobjekt von
VcDataRecordCreated 550
VcDataRecordCreating
Ereignis von
VcTree 551
VcDataRecordCreatingEventArgs
Ereignisobjekt von
VcDataRecordCreating 551
VcDataRecordDeleted
Ereignis von
VcTree 552
VcDataRecordDeletedEventArgs
Ereignisobjekt von
VcDataRecordDeleted 552
VcDataRecordDeleting
Ereignis von
VcTree 553
VcDataRecordDeletingEventArgs
Ereignisobjekt von
VcDataRecordDeleting 553
VcDataRecordModified
Ereignis von
VcTree 554
VcDataRecordModifiedEventArgs
Ereignisobjekt von
VcDataRecordModified 554
VcDataRecordModifying
Ereignis von

VcTree 555
VcDataRecordModifyingEventArgs
Ereignisobjekt von
VcDataRecordModifying 555
VcDataRecordNotFound
Ereignis von
VcTree 555
VcDataRecordNotFoundEventArgs
Ereignisobjekt von
VcDataRecordNotFound 556
VcDataTable 315
DataRecordCollection 315
DataTableFieldCollection 316
Description 316
MultiplePrimaryKeysAllowed 317
Name 317
VcDataTableCollection 319
Add 320
Copy 321
Count 319
DataTableByIndex 321
DataTableByName 322
FirstDataTable 322
GetEnumerator 323
NextDataTable 324
Update 324
VcDataTableField 326
DataTableName 326
DateFormat 327
Editable 328
Hidden 328
Index 329
Name 329
PrimaryKey 330
RelationshipFieldIndex 330
Type 332
VcDataTableFieldCollection 333
Add 334
Copy 335
Count 333
DataTableFieldByIndex 335
DataTableFieldByName 336
FirstDataTableField 336
GetEnumerator 337
NextDataTableField 338
VcDiagramLeftClicking
Ereignis von
VcTree 556
VcDiagramLeftDoubleClicking
Ereignis von
VcTree 557
VcDiagramRightClicking
Ereignis von
VcTree 557
VcDragCompleting
Ereignis von
VcTree 558
VcDragCompletingEventArgs
Ereignisobjekt von
VcDragCompleting 558
VcDragStarting
Ereignis von
VcTree 559
VcDragStartingEventArgs
Ereignisobjekt von
VcDragStarting 559
VcErrorOccuring
Ereignis von
VcTree 559
VcFieldSelecting
Ereignis von
VcTree 560
VcFieldSelectingEventArgs
Ereignisobjekt von

VcFieldSelecting 560

VcFilter 339

AddSubCondition 342

CopySubCondition 343

DatesWithHourAndMinute 340

Evaluate 343

GetEnumerator 344

IsValid 344

Name 340

RemoveSubCondition 345

Specification 341

StringsCaseSensitive 341

SubCondition 341

SubConditionCount 342

VcFilterCollection 346

Add 347

AddBySpecification 348

Copy 348

Count 346

FilterByIndex 349

FilterByName 349

FirstFilter 350

GetEnumerator 350

MarkedNodesFilter 347

NextFilter 351

Remove 351

VcFilterSubCondition 353

ComparisonValueAsString 353

ConnectionOperator 354

DataFieldIndex 355

FilterName 355

Index 355

IsValid 356

Operator 355

VcHelpRequested

Ereignis von

VcTree 561

VcHelpRequestedEventArgs

Ereignisobjekt von

VcHelpRequested 561

VcInPlaceEditorShowing

Ereignis von

VcTree 562

VcLegendView 357

Border 357

Height 358

HeightActualValue 358

Left 359

LeftActualValue 359

ScrollBarMode 360

Top 360

TopActualValue 361

Update 364

Visible 361

Width 362

WidthActualValue 362

WindowMode 363

VcLegendViewClosed

Ereignis von

VcTree 564

VcLEgendViewClosedEventArgs

Ereignisobjekt von

VcLegendViewClosed 564

VcMap 365

ConsiderFilterEntries 365

Count 366

CreateEntry 368

DeleteEntry 369

FirstMapEntry 369

GetMapEntry 370

Name 366

NextMapEntry 370

Specification 367

Type 367

VcMapCollection 372
Add 373
AddBySpecification 374
Copy 374
Count 373
FirstMap 375
GetEnumerator 375
MapByIndex 375
MapByName 376
NextMap 376
Remove 377
SelectMaps 378
Update 378
VcMapEntry 380
Color 380
DataFieldValue 381
FontBody 382
FontName 382
FontSize 383
GraphicsFileName 384
Number 385
Pattern 385
VcMouseDoubleClicking
Ereignis von
VcTree 565
VcMouseDown
Ereignis von
VcTree 566
VcMouseMove
Ereignis von
VcTree 566
VcMouseUp
Ereignis von
VcTree 567
VcNode 389
AllData 390
ArrangeSubtree 397
ChildNodeCollection 390
Collapsed 391
CollapseSubtree 398
DataField 392
DataRecord 399
Delete 399
ExpandSubtree 400
ID 392
InCollapsedSubtree 393
LeftBrotherNode 393
Marked 394
ParentNode 395
RelatedDataRecord 400
RightBrotherNode 395
SubtreeArrangement 396
SubtreeNodeCollection 396
Update 401
VcNodeAppearance 402
BackgroundColor 403
BackgroundColorDataFieldIndex 404
BackgroundColorMapName 404
DoubleFeature 405
FilterName 405
FormatName 406
FrameAroundFieldsVisible 406
FrameShape 407
LegendText 409
LineColor 409
LineColorDataFieldIndex 410
LineColorMapName 410
LineThickness 410
LineType 411
Name 413
Pattern 413
PatternColor 417
PatternColorDataFieldIndex 417
PatternColorMapName 418

PatternDataFieldIndex 418
PatternMapName 418
PileEffect 419
Shadow 419
ShadowColor 420
Specification 420
StrikeThrough 421
StrikeThroughColor 422
ThreeDEffect 422
VisibleInLegend 423
VcNodeAppearanceCollection 424
Add 425
AddBySpecification 425
Copy 426
Count 424
FirstNodeAppearance 426
GetEnumerator 427
NextNodeAppearance 428
NodeAppearanceByIndex 428
NodeAppearanceByName 429
Remove 429
VcNodeCollapsing
Ereignis von
VcTree 568
VcNodeCollection 430
Count 430
FirstNode 431
GetEnumerator 431
NextNode 432
SelectNodes 432
VcNodeCreated
Ereignis von
VcTree 568
VcNodeCreating
Ereignis von
VcTree 569
VcNodeDeleted
Ereignis von
VcTree 570
VcNodeDeleting
Ereignis von
VcTree 570
VcNodeExpanding
Ereignis von
VcTree 571
VcNodeFormat 434
CopyFormatField 437
FieldsSeparatedByLines 434
FormatField 435
FormatFieldCount 435
GetEnumerator 438
Name 436
RemoveFormatField 438
Specification 436
WidthOfExteriorSurrounding 437
VcNodeFormatCollection 439
Add 440
AddBySpecification 441
Copy 441
Count 439
FirstFormat 442
FormatByIndex 442
FormatByName 443
GetEnumerator 443
NextFormat 444
Remove 445
VcNodeFormatField 446
Alignment 447
BackgroundColor 447
BackgroundColorDataFieldIndex 448
BackgroundColorMapName 448
BottomMargin 449
ConstantText 449
FormatName 449

GraphicsFileName 449
GraphicsFileNameDataFieldIndex 450
GraphicsFileNameMapName 450
GraphicsHeight 450
Index 451
LeftMargin 451
MaximumTextLineCount 451
MinimumTextLineCount 452
MinimumWidth 452
PatternBackgroundColorAsARGB 452
PatternBackgroundColorDataFieldIndex 453
PatternBackgroundColorMapName 453
PatternColorAsARGB 453
PatternColorDataFieldIndex 454
PatternColorMapName 454
PatternEx 455
PatternExDataFieldIndex 455
PatternExMapName 456
RightMargin 456
TextAndGraphicsCombined 456
TextDataFieldIndex 457
TextFont 457
TextFontColor 457
TextFontDataFieldIndex 458
TextFontMapName 458
TopMargin 458
Type 459

VcNodeLeftClicking
Ereignis von
VcTree 572

VcNodeLeftDoubleClicking
Ereignis von
VcTree 572

VcNodeModified
Ereignis von
VcTree 573

VcNodeModifiedEventArgs
Ereignisobjekt von
VcNodeModified 573
VcNodeModifying 574

VcNodeModifying
Ereignis von
VcTree 574

VcNodeRightClicking
Ereignis von
VcTree 575

VcNodesMarked
Ereignis von
VcTree 576

VcNodesMarking
Ereignis von
VcTree 577

VcPrinter 460
AbsoluteBottomMarginInInches 461
AbsoluteLeftMarginInCM 461
AbsoluteLeftMarginInInches 462
AbsoluteRightMarginInCM 462
AbsoluteRightMarginInInches 463
AbsoluteTopMarginInCM 463
AbsoluteTopMarginInInches 464
Alignment 464
CurrentHorizontalPagesCount 465
CurrentVerticalPagesCount 465
CurrentZoomFactor 465
CuttingMarks 466
DefaultPrinterName 466
DocumentName 466
FitToPage 467
FoldingMarksType 467
MarginsShownInInches 470
MaxHorizontalPagesCount 470

MaxVerticalPagesCount 471
Orientation 471
PageDescription 472
PageDescriptionString 472
PageFrame 473
PageNumberMode 473
PageNumbers 474
PagePaddingEnabled 474
PaperSize 475
PrintDate 476
PrinterName 476
PrintPreviewWithFirstPage 476
TitleAndLegendOnAllPages 477
VcCalendarGrid 478
ZoomFactorAsDouble 478

VcRect 479
Bottom 479
Height 479
Left 480
Right 481
Top 481
Width 482

VcStatusLineTextShowing
Ereignis von
VcTree 578

VcTextEntrySupplying
Ereignis von
VcTree 579

VcTextEntrySupplying-Ereignisse 136

VcTextEntrySupplyingEventArgs
Ereignisobjekt von
VcTextEntrySupplying 579

VcToolTipTextSupplying
Ereignis von
VcTree 587

VcToolTipTextSupplying-Ereignisse 136

VcTree 483
ActiveNodeFilter 487
Arrange 513
ArrangementDataFieldIndex 487
BorderArea 488
BoxCollection 488
BoxFormatCollection 488
Clear 514
CollapseDataFieldIndex 489
CompleteViewMode 514
CopyNodesIntoClipboard 515
CtrlCXVProcessingEnabled 489
CutNodesIntoClipboard 515
DataTableCollection 490
DateOutputFormat 490
DeleteNodeRecord 516
DetectDataTableFieldName 516
DetectDataTableName 517
DetectFieldIndex 517
DiagramBackgroundColor 492
DialogFont 492
DoubleOutputFormat 493
DumpConfiguration 518
Enabled 493
EndLoading 518
ExportGraphicsToFileEx 519
ExtendedDataTablesEnabled 494
FilePath 494
FilterCollection 495
FirstVerticalLevelNumber 495
FontAntiAliasingEnabled 496
GetAValueFromARGB 521
GetBValueFromARGB 522
GetGValueFromARGB 523
GetNodeByID 523

GetRValueFromARGB 524
HorizontalNodeDistance 497
HorizontalNodeIndentWidth 497
IdentifyFormatField 525
IdentifyObjectAt 526
ImportConfiguration 528
InbuiltMouseCursorWhileDraggingEnabled 497
InPlaceEditingAllowed 498
InsertNodeRecord 528
InsertNodeRecordEx 529
InteractionMode 499
KeyDown 543
KeyPress 543
KeyUp 544
LegendView 499
LevelDataFieldIndex 500
Load 529
MakeARGB 530
MapCollection 500
MaximumChartRowCount 501
MouseProcessingEnabled 501
NodeAppearanceCollection 501
NodeCollection 502
NodeCreationAllowed 502
NodeCreationWithDialog 503
NodeFormatCollection 503
NodesDataTableName 504
NodeToolTipTextDataFieldIndex 504
ParentNodeIDDataFieldIndex 505
PasteNodesFromClipboard 531
PhantomDrawingWhileDraggingEnabled 505
Printer 506
PrintEx 531
PrintToFile 532
Reset 533
RoundedLinkSlantsEnabled 506
SaveAsEx 533
ScrollToNode 534
SetImageResource 535
ShowAboutDialog 536
ShowExportGraphicsDialog 536
ShowNodeEditDialog 538
ShowPageSetupDialog 538
ShowPrintDialog 539
ShowPrinterSetupDialog 539
ShowPrintPreviewDialog 539
StructureCodeDataFieldIndex 507
StructureType 507
SuspendUpdate 540
TextEntrySupplyingEventEnabled 507
ToolTipChangeDuration 508
ToolTipDuration 508
ToolTipPointerDuration 508
ToolTipShowAfterClick 509
ToolTipTextSupplyingEventEnabled 509
TreeViewStyle 510
UpdateNodeRecord 541
VcBoxLeftClicking 545
VcBoxLeftDoubleClicking 546
VcBoxModified 546
VcBoxModifying 547
VcBoxRightClicking 548
VcDataModified 549
VcDataRecordCreated 550
VcDataRecordCreating 551
VcDataRecordDeleted 552
VcDataRecordDeleting 553
VcDataRecordModified 554
VcDataRecordModifying 555
VcDataRecordNotFound 555

VcDiagramLeftClicking 556
VcDiagramLeftDoubleClicking 557
VcDiagramRightClicking 557
VcDragCompleting 558
VcDragStarting 559
VcErrorOccuring 559
VcFieldSelecting 560
VcHelpRequested 561
VcInPlaceEditorShowing 562
VcLegendViewClosed 564
VcMouseDoubleClicking 565
VcMouseDown 566
VcMouseMove 566
VcMouseUp 567
VcNodeCollapsing 568
VcNodeCreated 568
VcNodeCreating 569
VcNodeDeleted 570
VcNodeDeleting 570
VcNodeExpanding 571
VcNodeLeftClicking 572
VcNodeLeftDoubleClicking 572
VcNodeModified 573
VcNodeModifying 574
VcNodeRightClicking 575
VcNodesMarked 576
VcNodesMarking 577
VcStatusLineTextShowing 578
VcTextEntrySupplying 579
VcToolTipTextSupplying 587
VcWorldViewClosed 589
VcZoomFactorModified 589
VerticalLevelDistance 510
VerticalNodeDistance 511
ViewXCoordinate 511
ViewYCoordinate 511
WaitCursorEnabled 512
WorldView 512
Zoom 542
ZoomFactor 513
ZoomingPerMouseWheelAllowed 513
ZoomOnMarkedNodes 542

VcWorldView 591
Border 591
Height 592
HeightActualValue 592
Left 593
LeftActualValue 593
MarkingColor 594
Mode 594
ParentHWnd 595
ScrollBarMode 596
Top 596
TopActualValue 597
UpdateBehaviorName 597
Visible 598
Width 598
WidthActualValue 599

VcWorldViewClosed
Ereignis von
VcTree 589

VcZoomFactorModified
Ereignis von
VcTree 589

Verbindungen
abgerundete Schrägen 137, 139
Verbindungsaussehen 44

Verbindungsaussehen 142

VerticalLevelDistance
Eigenschaft von
VcTree 510

VerticalNodeDistance
Eigenschaft von

VcTree 511
Vertikal ab Ebene 143
Vertikal anordnen 85, 231
Vertikale Ebenen 123
Vertikaler Ebenenabstand 143
Vertikaler Knotenabstand 143
Viewer Metafile (*.vmf) 125
ViewXCoordinate
Eigenschaft von
VcTree 511
ViewYCoordinate
Eigenschaft von
VcTree 511
Visible
Eigenschaft von
VcBox 268
VcLegendView 361
VcWorldView 598
VisibleInLegend
Eigenschaft von
VcNodeAppearance 423

W

WaitCursorEnabled
Eigenschaft von
VcTree 512
Width
Eigenschaft von
VcLegendView 362
VcRect 482
VcWorldView 598
WidthActualValue
Eigenschaft von
VcLegendView 362
VcWorldView 599
WidthOfExteriorSurrounding
Eigenschaft von
VcNodeFormat 437
WindowMode
Eigenschaft von
VcLegendView 363
Worldview 102
WorldView
Eigenschaft von
VcTree 512
Name UpdateBehavior 597
siehe auch
VcWorldView 591

Z

Zeitkritische Operationen
Wartecursor 138
Zeitskala
anpassen 221
Zeitumstellung 74
Zoom
Methode von
VcTree 542
Zoomen 201
per Mausrad 135
ZoomFactor
Eigenschaft von
VcTree 513
ZoomFactorAsDouble
Eigenschaft von
VcPrinter 478
ZoomingPerMouseWheelAllowed
Eigenschaft von
VcTree 513
ZoomOnMarkedNodes
Methode von
VcTree 542
Zuordnungstabelle
über Index 376

Zuordnungstabellen 126
Angabe von Wertebereichen durch Filter 365

Zusatztext 223