Integration Guide for the NETRONIC Visual Scheduling Widget

To integrate the Visual Scheduling Widget (VSW) into an HTML page the following steps are to be done:

1. Add the following references to CSS and JavaScript files in this order:

```
<link href="jquery-ui.min.css" rel="stylesheet" />
<link href="nwaf-apptools.min.css" rel="stylesheet" />
<link href="nwaf-table.min.css" rel="stylesheet" />
<link href="nwaf-gantt.min.css" rel="stylesheet" />
<link href="nwaf-rab.min.css" rel="stylesheet" />

<script src="jquery-3.1.0.min.js"></script>
<script src="jquery-ui.min.js"></script>
<script src="jquery.mousewheel.min.js"></script>
<script src="hammer.min.js"></script>
<script src="d3.min.js"></script>
<script src="nwaf-apptools.min.js"></script>
<script src="nwaf-table.min.js"></script>
<script src="nwaf-gantt.min.js"></script>
<script src="nwaf-rab.min.js"></script>
```

2. Add a element to your HTML page. This element will be extended by the VSW to generate the diagram:

```
<div id="VSWidget" style="width:1000px; height:300px"></div>
```

3. Inside your code you have to instantiate the widget:

```
var options = {
   "licenseKey": "--- PLEASE INSERT LICENSE KEY HERE ---",
   "start": new Date(2019, 0, 1),
   "end": new Date(2019, 6, 1),
   "viewType": 0 /* 0 = activities, or alternatively 1 = resources */
};
var vsWidget = $("#VSWidget").nVSWidget(options).nVSWidget("instance");
vsWidget.render(); /* now a very first diagram without any data inside
                      will be shown */
```

You will have to fill in the license key that is provided in an additional license file inside the package. If something goes wrong here, then you will see a placeholder image only!

For explanation of the usage of jQuery UI Widgets please see the official documentation: http://api.jqueryui.com/jQuery.widget/.

4. Now, you can add activities as follows:

```
vsWidget.addActivities([{
   "ID": "Act1",
   "Name": "Activity 1",
   "Start": new Date(2019, 0, 1),
   "End": new Date(2019, 0, 6)
}]);
```

If you want to work with the resources view, then you can add resources and allocations:

```
vsWidget.addResources([{
   "ID": "Res1",
   "Name": "Resource 1"
}]);
vsWidget.addAllocations([{
   "ID": "All1",
   "Name": "Allocation 1",
   "ResourceID": "Res1",
   "Entries": [{
      "Start": new Date(2019, 0, 3),
      "End": new Date(2019, 0, 8)
    }],
}]);
```

Please note, to display the resources view you either must change the *viewType* option to 1:

```
vsWidget.option("viewType", 1);
```

or of course, you can set that option when instantiating the VSW.