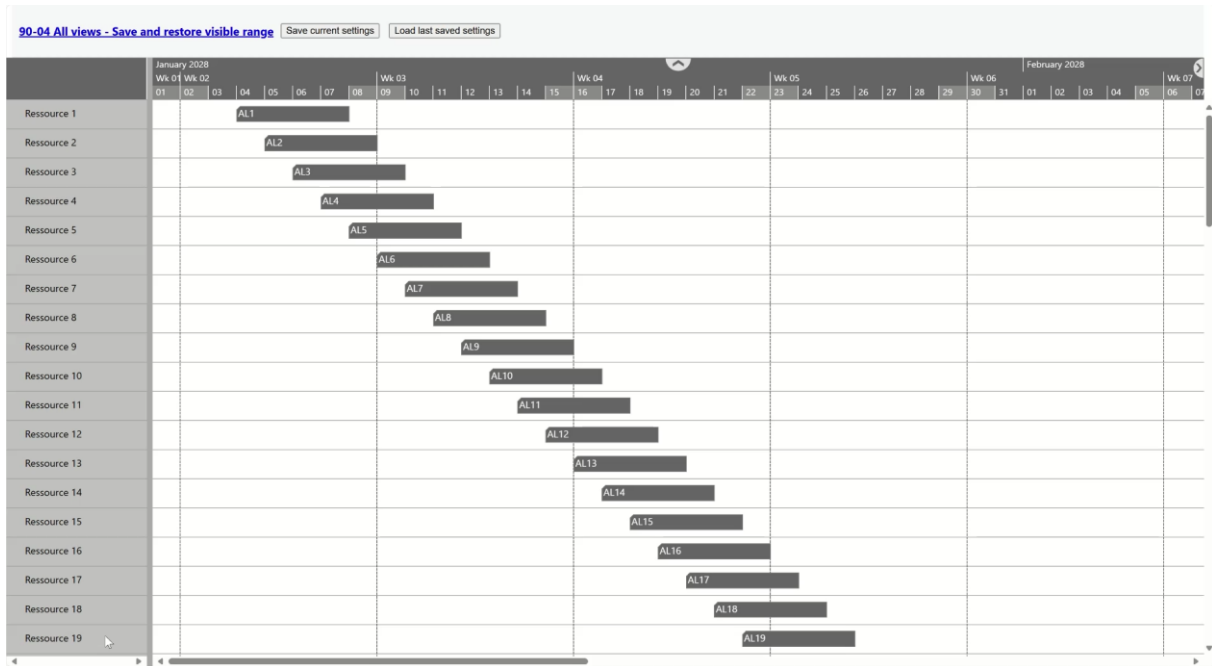


90-04 All views - Save and restore visible range

This example shows how to determine the visible area of a view so that it can be restored later.

Adjust the diagram view using the horizontal and vertical scroll bars. Click **Save Current Settings** to store the current viewport configuration. You can then modify the visible area further, and later restore the previous view by clicking **Load Last Saved Settings**.



There are no properties that can be read directly. Instead, you have to capture the relevant values for the horizontal and vertical changes in the two callbacks and store them in variables.

```
onTimeAreaViewParametersChanged: (args) => {
    horizontalViewStart = args.start;
    timeResolutionUnitCount = args.timeResolutionUnitCount;
    timeResolutionUnit = args.timeResolutionUnit;
},

onVerticalScrollOffsetChanged: (args) => {
    rowObjectAtTop = args.rowObjectAtTop;
    rowObjectTypeAtTop = args.rowObjectTypeAtTop;
    verticalViewStart = args.scrollOffset;
}
```

Um den sichtbaren Bereich wieder herzustellen, sind diese zwei Befehle notwendig. Mit render() werden die Einstellungen im Bild wirksam.

To restore the visible area, these two commands are required.

```
vsWidget.setTimeResolutionForView(savedTimeResolutionUnit,
    savedTimeResolutionUnitCount,
    savedHorizontalViewStart);
```

```
vsWidget.scrollToObject(savedRowObjectTypeAtTop,
                        savedRowObjectAtTop,
                        TargetPositions.Top | TargetPositions.NoHScroll, false);
vsWidget.render();
```

Using **render()** applies the settings to the image.

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="utf-8"/>
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <link href="../VSWidget/nwaf-apptools.min.css" rel="stylesheet"/>
  <link href="../VSWidget/nwaf-table.min.css" rel="stylesheet"/>
  <link href="../VSWidget/nwaf-gantt.min.css" rel="stylesheet"/>
  <link href="../VSWidget/nwaf-rab.min.css" rel="stylesheet"/>
  <script src="https://cdn.jsdelivr.net/npm/hammerjs@2.0.8/hammer.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/d3@7.9.0/dist/d3.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/tinycolor2@1.6.0/cjs/tinycolor.min.js"></script>
  <script src="../VSWidget/nwaf-apptools.min.js"></script>
  <script src="../VSWidget/nwaf-table.min.js"></script>
  <script src="../VSWidget/nwaf-gantt.min.js"></script>
  <script src="../VSWidget/nwaf-rab.min.js"></script>
  <script src="../LicenseKey.js"></script>
  <script src="Miscellaneous.js"></script>
  <style>
    body { margin: 0; font: 14px/1.4 system-ui, sans-serif; }
    #toolbar { padding: 10px; background: #f6f7f9; border-bottom: 1px solid #dde1e6; }
    #toolbar button { margin-right: 8px; }
    #VSWidget { height: 420px; } /* feste Höhe, damit man direkt etwas sieht */
  </style>
</head>
<body>
  <div id="toolbar">
    <button id="saveViewSettings">Save current settings</button>
    <button id="loadViewSettings">Load last saved settings</button>
  </div>
  <div id="VSWidget"></div>
</script>
  Miscellaneous.ready(() => {
    const { ViewType, RowDesigns, ObjectType, VisualType, TimeUnit,
           VerticallyScrollableViewArea, TargetPositions } = netronic.nVSW;

    const timeAreaStart = new Date(2028, 0, 1);
    const timeAreaEnd   = new Date(2028, 3, 1);

    let viewStart = timeAreaStart;
    let viewEnd   = timeAreaEnd;
    let centerDistance = 0;

    let horizontalViewStart;
    let timeResolutionUnitCount;
    let timeResolutionUnit;
    let verticalViewStart;
    let rowObjectAtTop;
    let rowObjectTypeAtTop;

    let savedTimeResolutionUnit;
    let savedTimeResolutionUnitCount;
    let savedHorizontalViewStart;
    let savedVerticalViewStart;
    let savedRowObjectAtTop;
    let savedRowObjectTypeAtTop;

    const options = {
      licenseKey: window.VSW_LICENSE_KEY,
      viewType: ViewType.Resources,
      start: timeAreaStart,
      end: timeAreaEnd,
```

```

defaultResourceExpandedRowDesign: RowDesigns.Bars,
defaultResourceCollapsedRowDesign: RowDesigns.Bars,
fixedTableColumnWidth: 0,

scrollOffsetsChangedCallbackTimeDelay: 100,

onClicked: (args) => {
    console.log(`Clicked on ${args.objectType} with ID ${args.objectID}`);
},

onTimeAreaViewParametersChanged: (args) => {
    horizontalViewStart = args.start;
    timeResolutionUnitCount = args.timeResolutionUnitCount;
    timeResolutionUnit = args.timeResolutionUnit;
},

onVerticalScrollOffsetChanged: (args) => {
    rowObjectAtTop = args.rowObjectAtTop;
    rowObjectTypeAtTop = args.rowObjectTypeAtTop;
    verticalViewStart = args.scrollOffset;
}
}
const vsWidget = Miscellaneous.instantiateVSWidget(document.querySelector("#VSWidget"), options);

const dayShift = 86400000; // 1 Tag in Millisekunden
const resources = [];
const allocations = [];

for (let i = 0; i < 90; i++) {
    const start = new Date(timeAreaStart.getTime() + (3 + i) * dayShift);
    const end = new Date(timeAreaStart.getTime() + (7 + i) * dayShift);

    // Create resources
    const resId = `R${i + 1}`;
    resources.push({
        ID: resId,
        TableText: `Ressource ${i + 1}`
    });

    // Create allocations
    const allocId = `AL${i + 1}`;
    allocations.push({
        ID: allocId,
        ResourceID: resId,
        BarText: allocId,
        Entries: [
            {
                Start: start,
                End: end
            }
        ]
    });
}

vsWidget.addResources(resources);
vsWidget.addAllocations(allocations);
vsWidget.render();

setTimeout(() => {
    savedTimeResolutionUnit = timeResolutionUnit;
    savedTimeResolutionUnitCount = timeResolutionUnitCount;
    savedHorizontalViewStart = horizontalViewStart;
    savedVerticalViewStart = verticalViewStart;
    savedRowObjectAtTop = rowObjectAtTop;
    savedRowObjectTypeAtTop = rowObjectTypeAtTop;
}, 200);

document.getElementById("saveViewSettings").addEventListener("click", () => {
    savedHorizontalViewStart = horizontalViewStart;
    savedTimeResolutionUnitCount = timeResolutionUnitCount;
    savedTimeResolutionUnit = timeResolutionUnit;
    savedVerticalViewStart = verticalViewStart;
    savedRowObjectTypeAtTop = rowObjectTypeAtTop;
    savedRowObjectAtTop = rowObjectAtTop;
});

```

```
document.getElementById("loadViewSettings").addEventListener("click", () => {
    vsWidget.setTimeResolutionForView(savedTimeResolutionUnit,
                                      savedTimeResolutionUnitCount,
                                      savedHorizontalViewStart);
    vsWidget.scrollToObject(savedRowObjectTypeAtTop,
                           savedRowObjectAtTop,
                           TargetPositions.Top | TargetPositions.NoHScroll,
                           false);
    vsWidget.render();
});
});
</script>
</body>
</html>
```