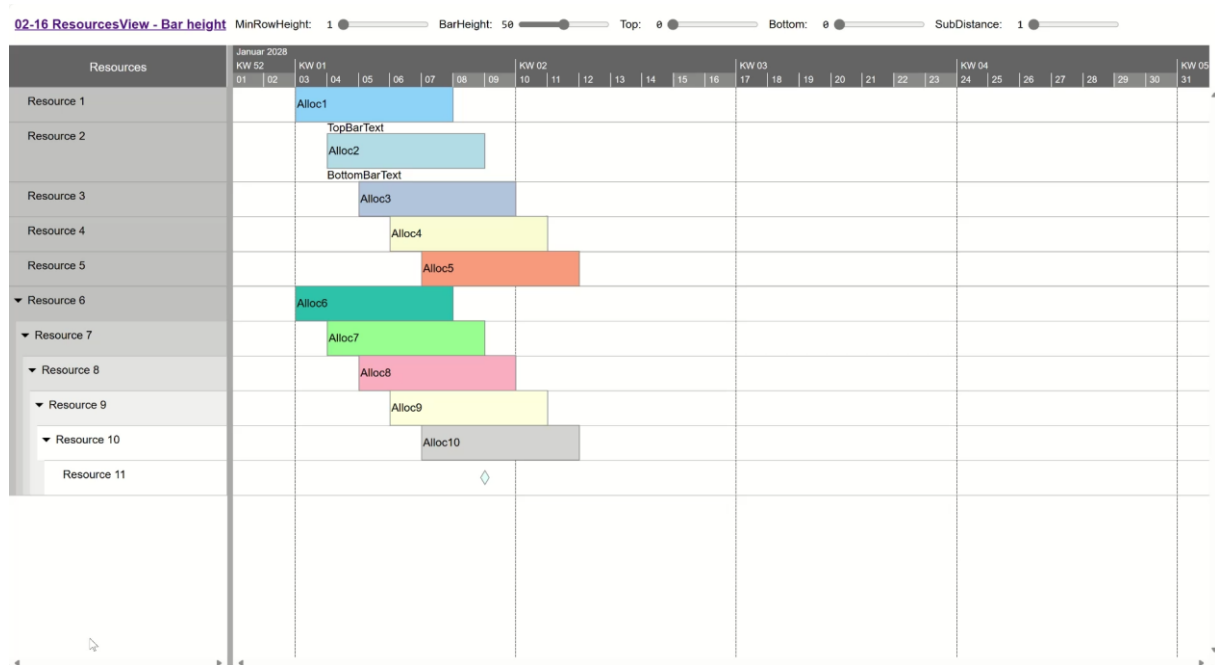


## 21-04 Resources view – Visualization – Row and bar height

This example shows the interaction of various factors that influence the row and bar height. The mode of operation can be tested using the sliders.



There are five options that determine the size of the rows and the bars:

1. **defaultResourceMinimumRowHeight**
2. **defaultAllocationBarHeight**
3. **topRowMarginInTimeArea**
4. **bottomRowMarginInTimeArea**
5. **subRowDistanceInTimeArea**

The **defaultResourceMinimumRowheight** option is the minimum row height that will be maintained for all rows and will never fall below it. As soon as a bar with additional spacing or outside text exceeds the specified minimum row height, the row height is increased individually.

The font size of the bar labels is fixed and consistent across all bars. Since the text remains visible even when the bar height falls below the font size, it is recommended to set the **defaultAllocationBarHeight** to a value of at least 20 px.

The **subRowDistanceInTimeArea** option only becomes effective for collapsed hierarchical structures. It determines the distance between the bars on the subordinate rows.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <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>
    .slider-output {
        display: inline-block;
        width: 3ch;
        font-family: monospace;
        text-align: right;
        white-space: pre;
        font-size: 1.2em;
    }
</style>
</head>
<body>
    <div id="toolbar">
        <label style="margin-left: 8px" for="slider1">
            MinRowHeight:
            <span class="slider-output" id="output1">1</span>
        </label>
        <input type="range" id="slider1" min="1" max="100" value="1" data-output="output1"
            data-option="defaultResourceMinimumRowHeight"/>
        <label style="margin-left: 8px" for="slider2">
            BarHeight:
            <span class="slider-output" id="output2">50</span>
        </label>
        <input type="range" id="slider2" min="0" max="100" value="50" data-output="output2"
            data-option="defaultAllocationBarHeight"/>
        <label style="margin-left: 8px" for="slider3">
            Top:
            <span class="slider-output" id="output3">0</span>
        </label>
        <input type="range" id="slider3" min="0" max="100" value="0" data-output="output3"
            data-option="topRowMarginInTimeArea"/>
        <label style="margin-left: 8px" for="slider4">
            Bottom:
            <span class="slider-output" id="output4">0</span>
        </label>
        <input type="range" id="slider4" min="0" max="100" value="0" data-output="output4"
            data-option="bottomRowMarginInTimeArea"/>
        <label style="margin-left: 8px" for="slider5">
            SubDistance:
            <span class="slider-output" id="output5">1</span>
        </label>
        <input type="range" id="slider5" min="1" max="100" value="1" data-output="output5"
            data-option="subRowDistanceInTimeArea"/>
    </div>
    <div id="VSWidget"></div>
    <script>
        Miscellaneous.ready(() => {
            const { BarDragModes, BarShape, BarSortMode,
                CollapseState, RowDesigns, ViewType } = netronic.nVSW;

            document.getElementById("slider1").addEventListener("input", changeHeightOption);
            document.getElementById("slider2").addEventListener("input", changeHeightOption);
            document.getElementById("slider3").addEventListener("input", changeHeightOption);
            document.getElementById("slider4").addEventListener("input", changeHeightOption);
            document.getElementById("slider5").addEventListener("input", changeHeightOption);

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

            const options = {
                licenseKey: window.VSW_LICENSE_KEY,

                viewType: ViewType.Resources,

                start: timeAreaStart,
                end: timeAreaEnd,

                defaultResourceExpandedRowDesign: RowDesigns.Bars,
                defaultResourceCollapsedRowDesign: RowDesigns.Bars |
                    RowDesigns.BarsInHiddenDescendantRows |

```

```

        RowDesigns.BarsStacked,

        allocationBarSortModeForStackedRowDesign: BarSortMode.StartAndEnd,

        defaultAllocationAllowedBarDragModes: BarDragModes.None,
        defaultAllocationBarSelectable: false,
        defaultResourceRowSelectable: false,
        multipleSelectionEnabled: false,
        defaultResourceTableRowDefinitionID: "CustomRowDef",
        ignoreCalendarOnAllocationBarInteractions: true,
        defaultAllocationBarShape: BarShape.Rectangle,
        defaultResourceMinimumRowHeight: 1,
        defaultAllocationBarHeight: 50,
        topRowMarginInTimeArea: 0,
        bottomRowMarginInTimeArea: 0,
        subRowDistanceInTimeArea: 1,
        fixedTableColumnWidth: 0,
        animationDuration: 0,
    };

    const vsWidget = Miscellaneous.instantiateVSWidget(document.querySelector("#VSWidget"), options);

    const tableRowDefinitions = createTableRowDefinitions();
    const resources = createResources();
    const allocations = createAllocations();
    vsWidget.addTableRowDefinitions(tableRowDefinitions);
    vsWidget.addResources(resources);
    vsWidget.addAllocations(allocations);
    vsWidget.render();

    function createTableRowDefinitions() {
        const tableRowDefinitions = [
            {
                ID: "CustomRowDef",
                CellDefinitions: [
                    {
                        TitleText: "Resources",
                        Width: 310,
                    },
                ],
            },
        ],
    };

    return tableRowDefinitions;
}

function createResources() {
    const resources = [
        {
            ID: "Res1",
            TableText: "Resource 1",
            CalendarId: "Cal2",
            AllocationRowsCollapseState: CollapseState.Expanded,
        },
        {
            ID: "Res2",
            TableText: "Resource 2",
            CalendarId: "Cal2",
            AllocationRowsCollapseState: CollapseState.Expanded,
        },
        {
            ID: "Res3",
            TableText: "Resource 3",
            CalendarId: "Cal2",
            AllocationRowsCollapseState: CollapseState.Expanded,
        },
        {
            ID: "Res4",
            TableText: "Resource 4",
            CalendarId: "Cal2",
            AllocationRowsCollapseState: CollapseState.Expanded,
        },
        {
            ID: "Res5",

```

```

        TableText: "Resource 5",
        CalendarId: "Cal2",
        AllocationRowsCollapseState: CollapseState.Expanded,
    },
    {
        ID: "Res6",
        TableText: "Resource 6",
        CalendarId: "Cal2",
        AllocationRowsCollapseState: CollapseState.Expanded,
    },
    {
        ID: "Res7",
        ParentID: "Res6",
        TableText: "Resource 7",
        CalendarId: "Cal2",
        AllocationRowsCollapseState: CollapseState.Expanded,
    },
    {
        ID: "Res8",
        ParentID: "Res7",
        TableText: "Resource 8",
        CalendarId: "Cal2",
        AllocationRowsCollapseState: CollapseState.Expanded,
    },
    {
        ID: "Res9",
        ParentID: "Res8",
        TableText: "Resource 9",
        CalendarId: "Cal2",
        AllocationRowsCollapseState: CollapseState.Expanded,
    },
    {
        ID: "Res10",
        ParentID: "Res9",
        TableText: "Resource 10",
        CalendarId: "Cal2",
        AllocationRowsCollapseState: CollapseState.Expanded,
    },
    {
        ID: "Res11",
        ParentID: "Res10",
        TableText: "Resource 11",
        CalendarId: "Cal2",
        AllocationRowsCollapseState: CollapseState.Expanded,
    },
    },
];

return resources;
}

function createAllocations() {
    const allocations = [
        {
            ID: "Alloc1",
            ResourceID: "Res1",
            TableText: "Allocation 1",
            BarText: "Alloc1",
            BarTextColor: "black",
            Entries: [
                {
                    Start: new Date(2028, 0, 3),
                    End: new Date(2028, 0, 8),
                    Color: "lightskyblue",
                },
            ],
        },
        {
            ID: "Alloc2",
            ResourceID: "Res2",
            TableText: "Allocation 2",
            BarText: "Alloc2",
            BarTextColor: "black",
            BarTopOutsideText: "TopBarText",
            BarBottomOutsideText: "BottomBarText",
        },
    ],
    },
    {

```

```

Entries: [
  {
    Start: new Date(2028, 0, 4),
    End: new Date(2028, 0, 9),
    Color: "lightblue",
  },
],
},
{
  ID: "Alloc3",
  ResourceID: "Res3",
  TableText: "Allocation 3",
  BarText: "Alloc3",
  BarTextColor: "black",
  Entries: [
    {
      Start: new Date(2028, 0, 5),
      End: new Date(2028, 0, 10),
      Color: "lightsteelblue",
    },
  ],
},
{
  ID: "Alloc4",
  ResourceID: "Res4",
  TableText: "Allocation 4",
  BarText: "Alloc4",
  BarTextColor: "black",
  Entries: [
    {
      Start: new Date(2028, 0, 6),
      End: new Date(2028, 0, 11),
      Color: "lightgoldenrodyellow",
    },
  ],
},
{
  ID: "Alloc5",
  ResourceID: "Res5",
  TableText: "Allocation 5",
  BarText: "Alloc5",
  BarTextColor: "black",
  Entries: [
    {
      Start: new Date(2028, 0, 7),
      End: new Date(2028, 0, 12),
      Color: "lightsalmon",
    },
  ],
},
{
  ID: "Alloc6",
  ResourceID: "Res6",
  TableText: "Allocation 6",
  BarText: "Alloc6",
  BarTextColor: "black",
  Entries: [
    {
      Start: new Date(2028, 0, 3),
      End: new Date(2028, 0, 8),
      Color: "lightseagreen",
    },
  ],
},
{
  ID: "Alloc7",
  ResourceID: "Res7",
  TableText: "Allocation 7",
  BarText: "Alloc7",
  BarTextColor: "black",
  Entries: [
    {
      Start: new Date(2028, 0, 4),
      End: new Date(2028, 0, 9),

```

```

        Color: "lightgreen",
    },
],
},
{
    ID: "Alloc8",
    ResourceID: "Res8",
    TableText: "Allocation 8",
    BarText: "Alloc8",
    BarTextColor: "black",
    Entries: [
        {
            Start: new Date(2028, 0, 5),
            End: new Date(2028, 0, 10),
            Color: "lightpink",
        },
    ],
},
{
    ID: "Alloc9",
    ResourceID: "Res9",
    TableText: "Allocation 9",
    BarText: "Alloc9",
    BarTextColor: "black",
    Entries: [
        {
            Start: new Date(2028, 0, 6),
            End: new Date(2028, 0, 11),
            Color: "lightyellow",
        },
    ],
},
{
    ID: "Alloc10",
    ResourceID: "Res10",
    TableText: "Allocation 10",
    BarText: "Alloc10",
    BarTextColor: "black",
    Entries: [
        {
            Start: new Date(2028, 0, 7),
            End: new Date(2028, 0, 12),
            Color: "lightgray",
        },
    ],
},
{
    ID: "Alloc11",
    ResourceID: "Res11",
    TableText: "Allocation 11",
    BarText: "Alloc11",
    Color: "lightcyan",
    BarTextColor: "black",
    Start: new Date(2028, 0, 9),
    End: new Date(2028, 0, 13),
    BarShape: BarShape.Symbol,
},
];

return allocations;
}

function changeHeightOption(event) {
    const slider = event.currentTarget;
    document.getElementById(slider.dataset.output).innerText = padStartNbsp(slider.value, 3);
    vswidget.option(slider.dataset.option, slider.value);
}

function padStartNbsp(value, width) {
    const s = String(value);
    const missing = Math.max(0, width - s.length);
    return "\u00A0".repeat(missing) + s; // NBSP
}
});

```

```
</script>  
</body>  
</html>
```