

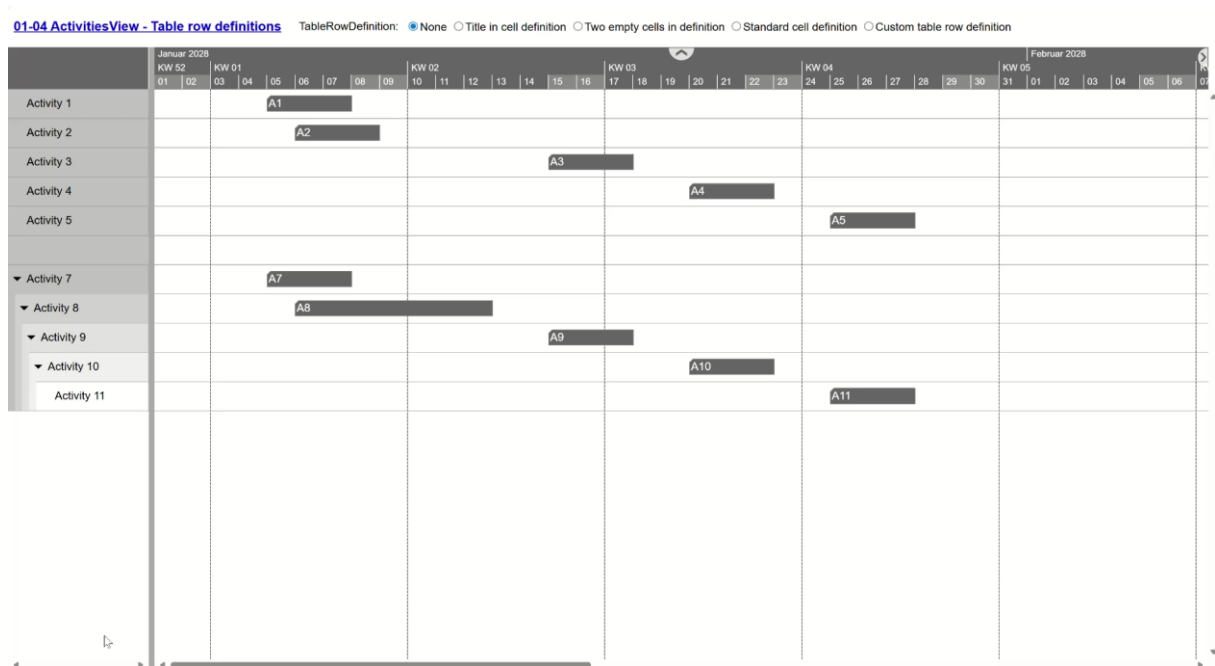
## 10-02 Activities view - Basics - Table row definitions

This example shows how `TableRowDefinitions` are created and used.

The **TableRowDefinition** determines how a table row is divided into data fields and where the displayed data comes from.

Setting the **defaultActivityTableRowDefinitionID** option determines which table row definition is to be used.

Although the option is not set at the beginning, an empty table row definition is used internally, which consists of a data field and displays the content of the **TableText** data field.



However, this definition cannot be addressed or changed as there is no assigned ID. In terms of content, this is the equivalent:

```
const rowDef = {
    ID: "CustomRowDef",
    CellDefinitions: [{}]}];

vswidget.addTableRowDefinitions([rowDef]);
vswidget.option("defaultActivityTableRowDefinitionID", " CustomRowDef");
vswidget.render();
```

It is now possible to add a header text to the table:

```
const rowDef = {
  ID: "CustomRowDef",
  CellDefinitions: [{TitleText: "Activity"}]};
```

Activity
Activity 1
Activity 2
Activity 3
Activity 4
Activity 5

The complete definition behind **CellDefinitions: [{ }]** looks like this:

```
const rowDef = {
  ID: "CustomRowDef",
  CellDefinitions: [
    {
      TitleText: "",
      HorizontalTitleAlignment: HorizontalAlignment.Center,
      TextSource: "TableText",
      Width: 200,
      MinimumWidth: 3,
      VerticalAlignment: VerticalAlignment.FirstLineOnBaseline,
      HorizontalAlignment: HorizontalAlignment.Left,
      WrapMode: TextWrapMode.NoWrap,
    }
  ]
};
```

It becomes interesting when you arrange two empty cells one behind the other:

**CellDefinitions: [{ }, { }]**

Activity 1	Activity 1
Activity 2	Activity 2
Activity 3	Activity 3
Activity 4	Activity 4
Activity 5	Activity 5
▲ Activity 7	Activity 7
▲ Activity 8	Activity 8
▲ Activity 9	Activity 9
▲ Activity 10	Activity 10
Activity 11	Activity 11

In this case, you can see that the first cell has a special role because the drop-down symbols for the hierarchy are added there.

If you use your own cell definition, you can also specify a specific output format for the date. To do this, the option for date format must first be set:

```
vsWidget.option("intlDateTimeFormatOptionsMap",
  {"dateFmt": {year: "numeric", month: "2-digit", day: "2-digit" }});
```

Finally, the **TextFormat** property must be used instead of **TextSource**:

```
{
  TitleText: "Start",
  HorizontalTitleAlignment: HorizontalAlignment.Center,
  TextFormat: "{{Start:date:dateFmt}}",
  HorizontalAlignment: HorizontalAlignment.Center,
  Width: 100,
  MinimumWidth: 100,
  MaximumWidth: 100,
},
```

If **MinimumWidth** and **MaximumWidth** are set to the value of **Width**, you get a column whose size cannot be changed interactively.

Activity	Start	End
Activity 1	05.01.2028	08.01.2028
Activity 2	06.01.2028	13.01.2028
Activity 3	15.01.2028	18.01.2028
Activity 4	20.01.2028	23.01.2028
Activity 5	25.01.2028	28.01.2028
▲ Activity 7	05.01.2028	08.01.2028
▲ Activity 8	06.01.2028	13.01.2028
▲ Activity 9	15.01.2028	18.01.2028
▲ Activity 10	20.01.2028	23.01.2028
Activity 11	25.01.2028	28.01.2028

#### Good to know:

- Column widths of the cells can be changed interactively.
- It is not possible to move columns interactively.
- Data fields are purely display fields that do not allow any input.
- Each row can have its own **TableRowDefinition**.

```
<!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>
</head>
<body>
  <div id="toolbar">
    <label style="margin-left: 20px">TableRowDefinition:&nbsp;&nbsp;&nbsp;</label>
    <label>
      <input id="rowDef1" type="radio" name="rowDef" checked/>
      None
    </label>
    <label>
      <input id="rowDef2" type="radio" name="rowDef" value="CustomRowDef1"/>
      Title in cell definition
    </label>
    <label>
      <input id="rowDef3" type="radio" name="rowDef" value="CustomRowDef2"/>
      Two empty cells in definition
    </label>
    <label>
      <input id="rowDef4" type="radio" name="rowDef" value="CustomRowDef3"/>
      Standard cell definition
    </label>
    <label>
      <input id="rowDef5" type="radio" name="rowDef" value="CustomRowDef4"/>
      Custom table row definition
    </label>
  </div>
</body>
</html>
```

```

</div>
<div id="VSWidget"></div>
<script>
    Miscellaneous.ready(() => {
        const { BarDragModes, HorizontalAlignment, RowDesigns,
            TextWrapMode, VerticalAlignment } = netronic.nVSW;

        document.getElementById("rowDef1").addEventListener("click", changeTableRowDefinition);
        document.getElementById("rowDef2").addEventListener("click", changeTableRowDefinition);
        document.getElementById("rowDef3").addEventListener("click", changeTableRowDefinition);
        document.getElementById("rowDef4").addEventListener("click", changeTableRowDefinition);
        document.getElementById("rowDef5").addEventListener("click", changeTableRowDefinition);

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

        let options = {
            licenseKey: window.VSW_LICENSE_KEY,

            start: timeAreaStart,
            end: timeAreaEnd,

            defaultActivityExpandedRowDesign: RowDesigns.Bars,
            defaultActivityCollapsedRowDesign: RowDesigns.Bars |
                RowDesigns.BarsInHiddenDescendantRows |
                RowDesigns.BarsStacked,
            defaultActivityAllowedBarDragModes: BarDragModes.None,
            defaultActivityBarSelectable: false,
            defaultActivityRowSelectable: false,
            multipleSelectionEnabled: false,
            intlDateTimeFormatOptionsMap: {
                dateFmt: {
                    year: "numeric",
                    month: "2-digit",
                    day: "2-digit"
                },
            },
        };

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

        const tableRowDefinitions = createTableRowDefinitions();
        const activities = createActivities();
        vsWidget.addTableRowDefinitions(tableRowDefinitions);
        vsWidget.addActivities(activities);
        vsWidget.render();

        function createTableRowDefinitions() {
            const tableRowDefinitions = [
                {
                    ID: "CustomRowDef1",
                    CellDefinitions: [
                        {
                            TitleText: "Activity",
                        },
                    ],
                },
                {
                    ID: "CustomRowDef2",
                    CellDefinitions: [{}, {}],
                },
                {
                    ID: "CustomRowDef3",
                    CellDefinitions: [
                        {
                            TitleText: "",
                            HorizontalTitleAlignment: HorizontalAlignment.Center,
                            TextSource: "TableText",
                            Width: 200,
                            MinimumWidth: 3,
                            VerticalAlignment: VerticalAlignment.FirstLineOnBaseline,
                            HorizontalAlignment: HorizontalAlignment.Left,
                            WrapMode: TextWrapMode.NoWrap,
                        },
                    ],
                },
            ];
        }
    });

```

```

    ],
  },
  {
    ID: "CustomRowDef4",
    CellDefinitions: [
      {
        TitleText: "Activity",
        HorizontalTitleAlignment: HorizontalAlignment.Center,
        TextSource: "TableText",
        Width: 170,
      },
      {
        TitleText: "Start",
        HorizontalTitleAlignment: HorizontalAlignment.Center,
        TextFormat: "{{Start:date:dateFmt}}",
        HorizontalAlignment: HorizontalAlignment.Center,
        Width: 100,
        MinimumWidth: 100,
        MaximumWidth: 100,
      },
      {
        TitleText: "End",
        HorizontalTitleAlignment: HorizontalAlignment.Center,
        TextFormat: "{{End:date:dateFmt}}",
        HorizontalAlignment: HorizontalAlignment.Center,
        Width: 100,
        MinimumWidth: 100,
        MaximumWidth: 100,
      },
    ],
  },
];

return tableRowDefinitions;
}

function createActivities() {
  const activities = [
    {
      ID: "Act1",
      TableText: "Activity 1",
      BarText: "A1",
      Start: new Date(2028, 0, 5),
      WorkingDays: 3,
      ParentID: "",
    },
    {
      ID: "Act2",
      TableText: "Activity 2",
      BarText: "A2",
      Start: new Date(2028, 0, 6),
      WorkingDays: 3,
      ParentID: "",
    },
    {
      ID: "Act3",
      TableText: "Activity 3",
      BarText: "A3",
      Start: new Date(2028, 0, 15),
      WorkingDays: 3,
      ParentID: "",
    },
    {
      ID: "Act4",
      TableText: "Activity 4",
      BarText: "A4",
      Start: new Date(2028, 0, 20),
      WorkingDays: 3,
      ParentID: "",
    },
    {
      ID: "Act5",
      TableText: "Activity 5",
      BarText: "A5",
    }
  ]
}

```

```

        Start: new Date(2028, 0, 25),
        WorkingDays: 3,
        ParentID: "",
    },
    {
        ID: "Act6",
        ParentID: "",
    },
    {
        ID: "Act7",
        TableText: "Activity 7",
        BarText: "A7",
        Start: new Date(2028, 0, 5),
        WorkingDays: 3,
        ParentID: "",
    },
    {
        ID: "Act8",
        TableText: "Activity 8",
        BarText: "A8",
        Start: new Date(2028, 0, 6),
        WorkingDays: 7,
        ParentID: "Act7",
    },
    {
        ID: "Act9",
        TableText: "Activity 9",
        BarText: "A9",
        Start: new Date(2028, 0, 15),
        WorkingDays: 3,
        ParentID: "Act8",
    },
    {
        ID: "Act10",
        TableText: "Activity 10",
        BarText: "A10",
        Start: new Date(2028, 0, 20),
        WorkingDays: 3,
        ParentID: "Act9",
    },
    {
        ID: "Act11",
        TableText: "Activity 11",
        BarText: "A11",
        Start: new Date(2028, 0, 25),
        WorkingDays: 3,
        ParentID: "Act10",
    },
    ],
    ];

    const millisecondsPerDay = 24 * 60 * 60 * 1000;
    for (const activity of activities)
        if (activity.Start) {
            activity.End = new Date(activity.Start.getTime() +
                (activity.WorkingDays ?? 1) * millisecondsPerDay);
        }

    return activities;
}

function changeTableRowDefinition(event) {
    const radio = event.target;

    vsWidget.option({
        defaultActivityTableRowDefinitionID: radio.value,
        tableViewWidth: 400,
    });
}

});
</script>
</body>
</html>

```