**Component Name**: Multileveltreetable

**Component Functionality**:

Multileveltreetable component has following features:

1. Display hierarchical data in tree structure
2. Header and footer are configurable.
   1. Position
   2. Visibility
3. Number columns to be displayed can be controlled
4. Tree nodes can be expanded/collapsed
5. Multiple/Single selection of rows using checkboxes or radio buttons
6. Highlighting clicked row
7. Each column can be styled
8. Component can be also used in view mode

**Usage Areas**:

Can be useful to integrate in any web application where a nested data needs to be displayed in a Tree-table structure with node selection and row selection

Ex. To display organization hierarchy or files and folder structure

**Component Technical details**:

* Angular cli version 9.1.7
* Typescript version 3.8.3
* rxjs version: 6.5.4

It has 2 components: app component and multilevel-tree-table component.

App component has selector tag of multilevel-tree-table. The app component also takes external json data from app.service.ts and it is passed to the multileveltreetable component as treeData.

Multileveltreetable component has 2 inputs treeTableConfig and treeData. It takes these inputs and creates multilevel tree table. It uses mat-tree from angular material to draw the tree.

TreetableConfig object has following properties:

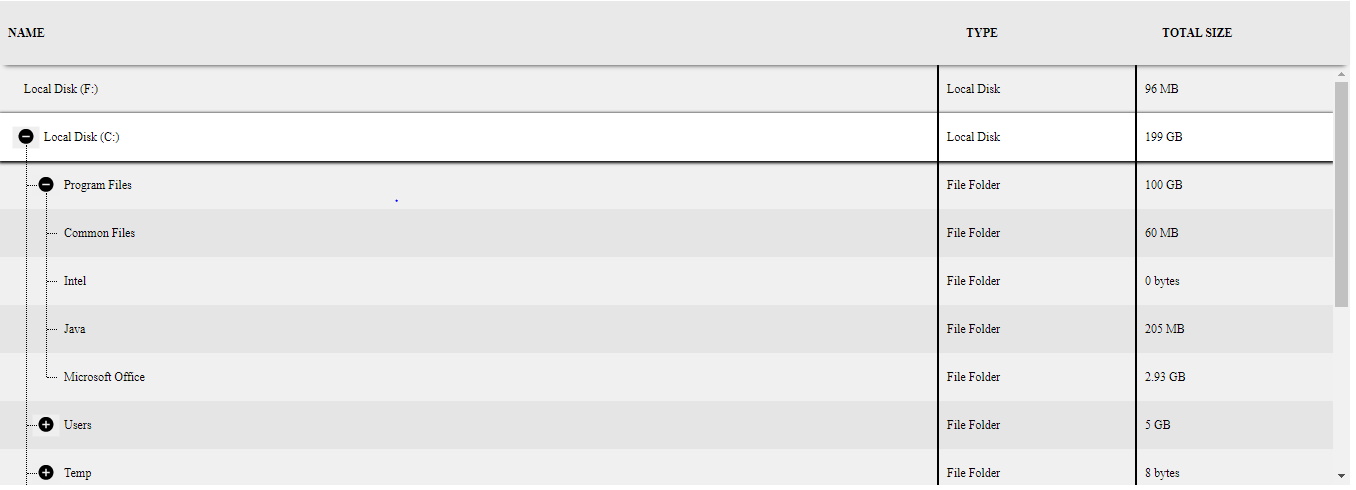
|  |  |  |
| --- | --- | --- |
| **Property** | **Description** | **Required/Optional** |
| 1. childKey: ‘key’ | key of treeData object passed as string which holds the children of tree node. It should be present in all child nodes. | Required |
| 1. bIsTreeCollapsed:   ‘allNodesExpanded’/  ‘allNodesCollapse’/  ‘firstNodeExpanded’/  ‘firstLevelExpanded’ | state of the tree (expanded/collapsed).  allNodesExpanded - all the nodes are expanded by default  allNodesCollapse - all the nodes are collapse by default  firstNodeExpanded - first node is expanded by default  firstLevelExpanded - root nodes are expanded by default | Optional |
| 1. rowSelections: {   selectionType: ‘single/multiple’,  bIsParentNodeSelectable: true/false,  bIsNodeClickable: true/false  } | Allows row selections and row highlight   1. selectionType:   ‘single’ - will allow you to select only one node using radio button  ‘multiple’ – will allow to select multiple nodes using checkboxes | Optional |
| 1. bIsParentNodeSelectable:   true - make parent node selectable.  When some child nodes are selected, parent node becomes indeterminate.  When parent node is selected all the child under it gets selected.  false - only the parent node will be selected, the child under it will not be selected.  Unspecified – if property not present, only the parent node will be selected, the child under it will not be selected |
| 1. bIsNodeClickable:   When this property is used, the selectionType and bIsParentNodeSelectable properties should not be used.  true – allows row to be highlighted when clicked  false – does not allow row highlight |
| 1. columnConfig: [ {   header: {},  content: {},  footer: {},  bIsTreeContent: true/false,  style: {}  }] | Configurations of each column. It has header, content, footer, style and bIsTreeContent properties for each column. It can have single object to display a single column or can have multiple objects with those properties depending upon the number of columns to be displayed | Required |
| Column Config has the following properties: | |  |
| 1. Header |  | Required |
| header: {} | Would not render header at all |  |
| header: {  bIsFixed: true/false,  contents: [  {  type: ‘text’,  label: ‘header1’  }]  } |  |  |
| bIsFixed:  true would render header and footer sticky and only content would be scrollable.  false would make the entire table content scrollable including header and footer. | Optional |
| contents can be multiple in a single table header cell. | Optional |
| type - Type of content in header cell | Required if contents is present |
| label - Content to be displayed in header cell (column name) | Required if contents is present |
| 1. Content |  | Required |
| content: {  type: ‘text/checkbox’,  property: ‘columnName’  } | type - Type of content of tree-table | Required |
| property - Property of treeData object which would be rendered in single cell | Required |
| 1. Footer |  | Required |
| footer: {} | Would not render footer at all |  |
| footer: {  bIsFixed: true/false,  contents: [  {  type: ‘text/button’,  label: ‘footer1’  }  ]  } | bIsFixed: true would render header and footer sticky and only content would be scrollable.  false would make the entire table content scrollable including header and footer. | optional |
| contents can be multiple in a single table footer cell | optional |
| type - Type of content in footer cell | Required if contents is included |
| label - Content to be displayed in footer cell | Required if contents is included |
| 1. bIsTreeContent: true | This indicates which column would have tree structure in the table.  Only one column in the entire config must have this property set as true. | Optional |
| 1. Style   style: {  width: ‘180px’,  property: value  } | style itself is optional, when provided, must have width specified | Optional |
| width if set to auto, resets to 120px by CSS | Required if style is included |
| Other CSS styles are optional |  |

**How this component can be integrated with other Applications**:

We can use the <app-multilevel-tree-table> tag in component, we will also require the multileveltreetable component’s entire code to use the tag. We can modify the component if there any specific requirements.

**Screenshots**:

1. **Component with row highlighted and a node expanded:**



1. **Component with header and footer displayed**:
2. **Component with no header and footer, displaying one column with multiple nodes selection feature enabled**:



1. **Component with single selection feature enabled and all nodes collapsed:**

