Control API

There are two UI Controls available in Configurator Services: RuleEditor and VariantsTree. The following sections describe the APIs for each.

RuleEditor

The RuleEditor control enables users to input text using an “input group template”. You can either enter text manually or by using the intellisense menu. You can also customize the view for individual groups or within a group for specific group types.

Constructor

The Constructor supports one composite parameter. The following table describes its properties.

Name	Type	Description
connectId	String	Mandatory parameter. It is the ID of the DOM element container for VariantsTree control. This element should exist in the DOM document. The VariantsTree DOM is placed as a child into the container. T The default is ‘variantsTreeContainer’.
contextMenuEnabled	Boolean	Optional parameter. Turns context menu support on/off (widget creation, events tracking). The default is False.
isEditable	Boolean	Optional parameter. Turns text editing for the control on/off. The default is False.
template	Object	Optional composite parameter. It must include a ‘template’ array property containing the group names sequence and the parameters for each group from ‘template’ keyed with group names. **Example:**

```
{
  template: ['RuleConditionGroup'],
  templateGroups: {
    'RuleConditionGroup': {
      type: 'grammar',
      grammarFile: 'RuleEditor/RuleGrammar.txt',
      title: 'Rule condition expression',
    },
  },
}
```

Public Fields

Name	Type	Description
containerNode	DOM Node	A reference to the DOM Node used as a container for the control DOM.
domNode	DOM Node	A reference to the DOM Node that corresponds to the control.
contextMenu	Object, MenuWidget	A reference to the menu widget used for the context menu.
intelliSenseMenu	Object, MenuWidget	A reference to the menu widget used for the intellisense menu.

Public Methods

This section describes the public methods.

setInputTemplate

Use this method to create a new editor input template. The startNewInput method needs to be called before you can start working with the editor using this template method.

Input parameters

Name	Type	Description
inputTemplate	Object	A template that contains a list of groups used as the 'template' array property as well as settings for each group.

getInputTemplate

Use this method to get the current editor input template.

Return value

Object. Template descriptor or null.

setEditState

Switch edit state for the control.

Input parameters

Name	Type	Description
isEditable	Boolean	The applied value defines how the editor content can be modified.

isEditable

Get edit state of the control. It determines whether content is editable.

Return value

Boolean

toggleCssClasses

This method enables toggling between DOM Node CSS classes.

Input parameters

Name	Type	Description
cssClassNames	String | Array of Strings	Class names that can be added or removed from the Node. You can pass several class names as a string parameter by separating them using the ‘space’ character.
turnStyleOn	Boolean	If true, classes are added to the Node. If false, classes are removed.
targetNode	DOM Node, Nullable	Node that can be modified.

isInputValid

Determines if the values from all the input groups passed validation.

Return value

Boolean

isInputStarted

Determines if the template was set and all input groups were created.

Return value

Boolean

focus

Places focus into the active or first input group.

focusgroup

Places focus into a specific input group.

Input parameters

Name	Type	Description
groupName	String	Group name from template.

focusFirstEditableGroup

Places focus on the first input group, which supports user input.

undoChange

Undoes the previous editor value change.

redoChange

Re-does the change that was undone.

getCurrentState

Gets information about the current editor state.

Return value

Object with editor state information.

resetChanges

Resets all changes. The Editor returns to its initial values.

setStartState

Sets up the editor according to state settings. The current changes queue is dropped.

Input parameters

Name	Type	Description
startState	Object	State information for editor settings restoration.

getValue

Gets the current editor value.

Return value

Object with values from all input groups. The key is the group name from the template.

getStringValue

Gets the editor value as string.

Return value

String. Concatenation of all input group values.

setValue

Sets the editor value.

Input parameters

Name	Type	Description
inputData	Object	Value collection for editor input groups. Key = group name from template.

getErrorString

Gets value validation error as a string.

Return value

String. Concatenation of validation errors from all input groups.

isEditorFocused

Determines which editor control has focus.

Return value

Boolean

isMenuFocused

Determines which target menu control has focus.

Input parameters

Name	Type	Description
targetMenu	Object, MenuWidget	Menu widget used in the editor.

Return value

Boolean

isValueModified

Identifies whether the initial editor value was modified.

Return value

Boolean

registerShortcut

Registers the shortcut for the editor.

Input parameters

Name	Type	Description
keyCode	Number	Button key code.
isCtrl	Boolean	Specifies whether the Ctrl key is pressed.
isShift	Boolean	Specifies whether the Shift key is pressed.
callbackHandler	Function	Method that is called when the shortcut is activated.

getGroupIndex

Gets the index for a particular input group.

Input parameters

Name	Type	Description
targetGroup	Object, InputGroup	Registered editor input group.

getGroupByIndex

Gets a particular input group by index.

Input parameters

Name	Type	Description
groupIndex	Number	Group index from template.

Return value

InputGroup

getGroupByDomNode

Gets input group using the related DOM Node.

Input parameters

Name	Type	Description
targetDomNode	DOM Node	DOM Node belonging to the registered editor input group.

Return value

InputGroup

getGroupName

Gets the name of a specific input group.

Input parameters

Name	Type	Description
targetGroup	Object, InputGroup	Registered editor input group.

Return value

String. Group name from template.

getGroupByName

Gets registered editor input group by name from the template.

Input parameters

Name	Type	Description
groupName	String	Group name from active editor template.

Return value

InputGroup

getGroupsCount

Counts the editor input groups.

Return value

Number

getNextGroup

Gets the next input group relative to the target group. If the target group is last, then the first group is returned.

Input parameters

Name	Type	Description
targetGroup	Object, InputGroup	Group relative to which next group is searched.

Return value

InputGroup

getPrevGroup

Gets the previous input group relative to the target group. If the target group is last, then the first group is returned.

Input parameters

Name	Type	Description
targetGroup	Object, InputGroup	Group relatively to which previous group is searched.

Return value

InputGroup

getGroupDomNode

Gets the previous input group relative to the target group. If target group is last, then the first group is returned.

Input parameters

Name	Type	Description
targetGroup	Object,InputGroup | String	Existing editor input group or it id.

Return value

InputGroup

startNewInput

Creates editor input groups based on the current template. All previous input groups are removed.

showIntelliSense

Shows the intellisense menu for a particular input group. ***Input parameters***

Name	Type	Description
targetGroup	Object, InputGroup	Existing editor input group.
allOptions	Boolean	If true, all available group values appear. If false, proposed values are limited based on the current group value.

hideIntelliSenseMenu

Hides active intellisense menu.

Events

Events described in this section are based on the Eventable core module. You can assign Handlers for these events using the **addEventListener** method.

onFocus

Event fires when the editor control has focus.

onBlur

Event fires when the focus moves away from the editor control.

onStateChanged

Event fires after group values are changed or the group loses focus.

onRender

Event fires after the editor creates DOM for input groups. onValueChanged
Event fires after editor creates DOM for input groups.

onGroupValueEntered

Event fires if the group value changed and passed validation. ***Input parameters***

Name	Type	Description
targetGroup	Object, InputGroup	Affected editor input group.
approvedValue	String	Applied value, which passed validation.

Context Menu Events

The events described in this section belong to the dojo context menu widget. Use the dojo.connect and dojo.on modules to attach handlers for them.

onMenuInit

Event raised before context menu creation. Use it to create a list of required menu items.

Input parameters

Name	Type	Description
menuEvent	Event	Corresponding to a native context menu event.
targetGroup	Object, InputGroup	Target editor input group that the context menu is created for.
targetGroupName	String	Input group name from template.

Return value

Array of menu items descriptors.

onMenuItem

Click Event raised when the user clicks on the editor context menu item. ***Input parameters***

Name	Type	Description
commandId	String	ID of selected menu item.
contextData	Any	Context information for menu.

Return value

Array of menu items descriptors.

VariantsTree

The VariantsTree Control enables users to create and interact with a layered view. The view contains the following default layers:

- TreeVisualization
- GroupVisualization

The view is scalable. Context menus are supported.

Constructor

The Constructor supports one composite parameter that can contain the properties described in the following table:

Name	Type	Description
connectId	String	Mandatory property. It contains the DOM element container ID for the VariantsTree control. The Element should exist in the document. The VariantsTree DOM will be placed as a child into container. The default is ‘variantsTreeContainer’.

layerVisualizationParameters Object	Optional composite parameter.It contains a collection of additional visual settings for view layers, which can be loaded into view. Collection key = layer id. Example: `{ 'groups': { labelWidth: 200 } }`

Public Fields

Name	Type	Description
layeredView	Object, MultiLayeredView	Reference to MultiLayeredView control. Most parts of the VariantsTree functionality are based on this core control. It is initialized in the constructor.

Public Methods

This section describes the public methods.

loadTree

The loadtree method creates TreeLayer and GroupLayer instances, loads them into the control and then centers the view. The TreeLayer instance is initialized with the treeData input parameter. TreeLayer gets the ‘variantsTree’ ID and GroupLayer gets the ‘groups’ ID. The Method can be overridden in order to change the list of loaded layers.

Input parameters

Name	Type	Description
treeData	Object	Hierarchical object data structure with a single root, where the next level children are placed into property children.

isTreeLoaded

This method verifies that the default view layers are already created and loaded into the view control (loadTree method was called).

Return value

Boolean

getParentNode

Returns the SVG Node related to the control.

Return value

DOM Node

setGroupsLayerVisibility

Allows the Groups layer to be either shown or hidden.

Input parameters

Name	Type	Description
isVisible	Boolean	Required layer visibility state.

setTreeLayerData

Sets new data for the layer.

Input parameters

Name	Type	Description
layerId	String | Number	Layer id or index, which should be updated.
newData	Any	New layer data.

getLeafDataNodesCount

Gets count of leaf data items from TreeLayer.

Return value

Number of leafs.

setScale

Gets count of leaf data items from TreeLayer.

Input parameters

Name	Type	Description
newScale	Number	Sets new scale value for layered view control.

getScale

Get count of leaf data items from TreeLayer.

Return value

Number

setSpacing

Sets new spacing value for Tree layer. This parameter effects distance between rendered item nodes, which are on adjacent hierarchy levels (horizontal distance).

Input parameters

Name	Type	Description
spacingValue	Number	Spacing distance in pixels

centerView

Calculates and sets the optimal view scale when layers content fits client viewport.

adjustViewBounds

Recalculates SVG node size based on current view content bounds and scale. This is a complex operation, which should be used after scaling or significant content changes.

Events

The following sections describe the events that can be used as part of the VariantsTree Control.

onScaleChanged

Event fires when view scale is changed.

Input parameters

Name	Type	Description
newScale	Number	Scale value that was applied.

onGroupsSwitched

Event fires when view scale is changed.

Input parameters

Name	Type	Description
newScale	Number	Scale value that was applied.

Context Menu Events

The events described in this section belong to the dojo context menu widget. Use the dojo.connect and dojo.on modules to attach handlers for them.

onTreeMenuInit

Event handler for onMenuInit, which generates array of descriptors for all available context menu actions.

Return value

Array of menu items descriptors.

onTreeMenuSetup

Event handler for setup context menu item's state based on target layer and node.

Input parameters

Name	Type	Description
menuControl	Object, MenuWidget	Menu control, which should be updated.
targetLayer	Object, VisualizationLayer	Layer that contains Node for which menu should be displayed.
targetDataNode	Object	Data object that bound to the target DOM Node.

onMenuItemClick

Handler for onMenuItemClick event.

Input parameters

Name	Type	Description

commandId	String	Menu item id.
targetDataNode	Object	Data object that is bounded to the target DOM Node.