

OpenText™ Formio Help

December 2024

Tree Element

The Tree element is used to design a hierarchy or a category. There are options to configure nodes that allows you to change the design of the form in the desired format.

Figure 1 Tree Element - Data

TREE ELEMENT COMPONENT

Display | **Data** | Validation | API | Conditional | Logic

Layout

Data Source Type: JSON

Data Source Raw JSON

```
1 {
2   "dn": "unique id",
3   "name": "container data",
4   "data": "any meta-data attached with the container",
5   "subContainers": [
6     {
7       "dn": "id1",
8       "name": "container data1",
9       "icon": "glyphicon glyphicon-cloud"
10    },
11    {
12      "dn": "id2",
13      "name": "container data2",
14      "icon": "glyphicon glyphicon-cloud"
15    }
16  ]
17 }
```

Mapper keys for URL response/ Raw JSON

```
1 {
2   "dn": "dn",
3   "name": "name",
4   "data": "data",
5   "subContainers": "subContainers",
6   "icon": "icon"
7 }
```

Default Value: Default Value

Refresh On: [Dropdown]

☐ Clear Value On Refresh

Preview

Tree Element

Save Cancel Remove

NOTE: The **Preview** area displays how the form would render on making the changes or edits to any field.

The following fields are populated in the **Data** tab.

Data Source Type: Select the type of data source, that is JSON or URL.

The following fields appear when the **Data Source Type** is **JSON**:

Data Source Raw JSON: Each node configuration is displayed in a JSON file. You must have the following details mentioned:

dn: It is a unique ID that defines the selected node. The value must be a string. This is the data value for the tree element that you add and is passed to the workflow.

name: Enter the name you wish to be displayed on the User Interface. The value must be a string.

data: (Optional) This saves the metadata to the existing field.

subcontainers: It defines the child node. The value must be an array of JSON. Each element of the array is an object following the same structure as that of a node.

icon: The image class to be used for displaying icon on each node. The value must be a string. If you do not define the value, the node uses the image class selected in **Default Icon Class** under the **Display** tab.

NOTE: All nodes must follow either JSON or customized data structure. The parent and child cannot be of different structure.

If JSON does not follow the default structure, it must be mapped accordingly.

Mapper keys for URL response: Allows you to map the JSON parameters with the URL element parameters.

You can use the **Mapper keys for URL response/ Raw JSON** setting to customize the field names provided under **Data Source RAW JSON**.

For example, if you want the field names as ["id","display","description","child","icon"], then modify the details under **Mapper keys for URL response/ Raw JSON** as follows:

```
{
  dn: "id",
  name: "display",
  data: "description",
  subContainers: "child",
  icon: "icon"
}
```

The following fields appear when the **Data Source Type** is **URL**:

HTTP method: Select the required http method. For example, GET, POST.

Service ID: Select the appropriate service ID. For example, IDM, IG.

Data Source URL: The URL that returns a JSON array to use as the data source.

Mapper keys for URL response: Allows you to map the JSON parameters with the URL element parameters.

Request Headers: Set any headers that should be sent along with the request to the URL. This is useful for authentication. You can have multiple keys for a parameter, and you can have multiple values for a key.

request payload: Enter the request body for the root node in the request payload field. The request body contains the node details that will be used to load the node. It applies in case of POST method.

lazy parameter: Enter the parameter name that will be used to load a sub-node. The lazy parameter such as `nodeid` is appended in the data source URL provided by the user. It applies in case of GET method (Lazy Loading).

Default Value: The entered value is displayed in the field before user interaction. Having a default value overrides the placeholder text.

Refresh On: Refreshes data when another field changes.

Clear Value On Refresh: This text appears below the input field.

Dynamic Entity

This is a multi select drop-down field. This widget allows you to select more than one entity in an entity type. For example, user, group

Figure 2 Dynamic Entity

The screenshot shows the 'Dynamic Entity Component' configuration window. It has a top bar with tabs: 'Display', 'Data', 'Validation', 'API', 'Conditional', and 'Logic'. The 'Display' tab is active. The configuration fields include: 'Entity Type' (a text input with a red asterisk), 'Display Attribute' (a text input with a plus icon), a '+ Add Another' button, 'Service ID' (a dropdown menu showing 'IDM'), 'Limit' (a text input showing '20' with a note 'Maximum number of items to view per page of results.'), 'Default Value' (a text input showing 'Default Value'), 'Refresh On' (a dropdown menu), a checkbox for 'Clear Value On Refresh', and a '+ Custom Default Value' button. On the right, there is a 'Preview' section showing a 'Dynamic Entity' label and a multi-select dropdown. At the bottom of the preview are 'Save', 'Cancel', and 'Remove' buttons.

The following fields are populated in the **Data** tab:

Entity Key: Specify the entity key for the entity type. For example, for the entity type User, the entity key is user.

Display Expression Attribute: Specify the attributes of the entity type you want to be displayed. You can add multiple display attributes. For example, for the entity type User, the **Display Expression Attribute** can be FirstName, LastName.

Entity Type	Display Expression Attribute
User	FirstName, LastName

Service ID: Select the service ID. For example, OpenText Identity Lifecycle Manager or OpenText Identity Governance.

Limit: Specify the number of entities that you want to be displayed. The default value is 20.

Default Value: Specify the value to be displayed in the field before user interaction. Having a default value overrides the placeholder text.

Refresh On: Refreshes data when another field changes.

DN Query

Allows you to search and retrieve DNs from the Identity Vault. However, with the DNQuery, the object selector content can be driven by the result of a directory abstraction layer Queries object rather than from properties.

NOTE: You must ensure that the query, parameter, and key you specify is present in Designer.

Figure 3 DN Query

The screenshot shows the 'DN Query Component' window. It has a top navigation bar with tabs: Display, Data, Validation, API, Conditional, and Logic. The 'Data' tab is selected. Below the tabs, there's a 'Service ID' dropdown menu with 'IDM' selected. Under 'Parameters', there's a table with 'Key' and 'Value' columns. The 'Value' column has a text input field and a '+ Add Another' button. Below the table is another '+ Add Another' button. There's a 'Query Key' text input field, a 'key of query' text input field, a 'Return Attributes' text input field, and a 'key of return parameter' text input field. Each of these input fields has a '+ Add Another' button. On the right side, there's a 'Preview' panel with a 'DN Query' text input field and three buttons: 'Save' (green), 'Cancel' (grey), and 'Remove' (red).

The following fields are populated in the **Data** tab:

Service ID: Select the required service ID. For example, OpenText Identity Lifecycle Manager or OpenText Identity Governance.

Parameters: Specify the parameter key and its value. You can have multiple keys for a parameter, and you can have multiple values for a key.

NOTE: You must provide the parameter value (static value) while configuring this component. Dynamic parameter value such as `data.<other dynamic value>` is not supported. To use the dynamic parameter value, you can use the `IDVault.globalquery` in the **Select** component.

Query Key: Specify the key of the DAL Queries object.

Return Attributes: Specify the attributes of the entity type you want to be displayed. You can add multiple display attributes.

Default Value: Specify the value to be displayed in the field before user interaction. Having a default value overrides the placeholder text.

Refresh On: Refreshes data when another field changes.

Title Element

This widget is used to design a title for the form.

Figure 4 Title Element

The screenshot shows the 'Title Element Component' configuration window. It has a top bar with tabs: 'Display' (selected), 'Data', 'Validation', 'API', 'Conditional', and 'Logic'. On the right is a 'Preview' section showing a blue 'Title' text. Below the preview are 'Save' (green), 'Cancel' (grey), and 'Remove' (red) buttons. The main configuration area on the left includes: a 'Label' field with 'Title Element'; a 'Hide Label' checkbox; a 'Label Position' dropdown set to 'Top'; a 'Title Content' field with 'Title'; a 'CSS Class' field with 'CSS Class'; and an 'Attributes' table with columns 'Attribute' and 'Value'. At the bottom left is a '+ Add Another' button. A 'Help' icon is in the top right corner.

DN Display

This is used to display a read-only DN. It can display the full DN or a set of attributes associated with the DN depending on the properties you set.

Figure 5 DN Display

The screenshot shows the 'DN Display Component' configuration window. It has a tabbed interface with 'Display', 'Data', 'Validation', 'API', 'Conditional', and 'Logic' tabs. The 'Data' tab is active. On the left, there are several input fields: 'Entity key for DN Expression' (placeholder: 'Entity key for DN Expression'), 'Display Expression' (placeholder: 'Display Expression'), 'DN' (placeholder: 'Enter Fully Qualified DN'), 'Default Value' (placeholder: ':'), and 'Refresh On' (a dropdown menu). Below these are checkboxes for 'Clear Value On Refresh', 'Allow Manual Override of Calculated Value', 'Encrypt', and 'Database Index'. There are also buttons for '+ Custom Default Value' and '+ Calculated Value'. On the right, there is a 'Preview' section showing 'DN Display' with a dropdown arrow, and three buttons: 'Save' (green), 'Cancel' (grey), and 'Remove' (red).

The following fields are populated in the **Data** tab:

Entity key for DN Expression: Represents the type of entity used for DN display. Leave this value blank if you want to display the full DN or CN value retrieved from the Identity Vault, else enter an entity.

The entity you choose must:

- ♦ Have the directory abstraction layer View property set to **True**.
- ♦ Be the entity of the DN you are working with.

Display Expression: Represents the attribute used for DN display. It also displays multiple attributes, provided the attribute is separated by comma. Leave this value blank if you want to display the full DN or CN value. If you want to mask the DN by displaying attributes, you must first specify an **Entity key for DN expression**.

For example, to show the user entity's first and last name attributes, construct an expression like this:
FirstName LastName.

Ensure the attribute's View, Read, Search, and Required properties are set to True in the directory abstraction layer. For more information, see [Attribute Properties](#) in the *OpenText™ Identity Manager - Administrator's Guide to Designing the Identity Applications*.

DN: Enter the respective DN.

Default Value: The entered value is displayed in the field before user interaction. Having a default value overrides the placeholder text.

Refresh On: Refreshes data when another field changes.

Clear Value On Refresh: When Refresh On field is changed, clear this components value.

Permission Request DN

This is used to provide permission for a default role and resource approval template.

Figure 6 Permission Request DN

The screenshot shows the 'Permission Request DN Component' configuration window. It has a top bar with tabs: 'Display' (selected), 'Data', 'Validation', 'API', 'Conditional', and 'Logic'. A 'Help' icon is in the top right corner. The main configuration area on the left includes: a 'Permission Request Type' dropdown menu; a 'Hide Label' checkbox; a 'Label Position' dropdown menu set to 'Top'; a 'placeholder' text input field; a 'Description' text input field; a 'Tooltip' text input field; an 'Error Label' text input field; and a 'Custom CSS Class' text input field. On the right, there is a 'Preview' section with a visual representation of the label and three buttons: 'Save' (green), 'Cancel' (grey), and 'Remove' (red).

In the **Display** tab, based on the selected **Permission Request type**, the label is automatically populated.

Label Element

This widget allows you to create and design the labels used in the forms.

Figure 7 Label Element

The screenshot shows the 'Label Element Component' configuration window. It has a title bar with a 'Help' icon and a close button. Below the title bar are tabs: 'Display' (selected), 'Data', 'Validation', 'API', 'Conditional', and 'Logic'. The main area contains several input fields: 'Label' (with a question mark icon and a red asterisk), 'Hide Label' (checkbox), 'Label Position' (dropdown menu showing 'Top'), 'Label Width' (text input '30' with a percentage icon), 'Label Content' (text input), 'Widget' (dropdown menu showing 'Select a widget'), 'placeholder' (text input), and 'Description' (text input). On the right, there is a 'Preview' section showing a visual representation of the label. At the bottom right of the preview are 'Save', 'Cancel', and 'Remove' buttons.

Data Item Mapping Element

This widget allows you to map data from the data flow into fields in a form (pre-activity mapping) and to map data from the form back to the data flow (post-activity mapping). If you have specified the Data Item Mapping Value in OpenText Designer, the **Custom Default Value** will overwrite the **Data Mapping Value**. In other words, the **Custom Default Value** takes precedence.

Figure 8 Data Item Mapping Element

The screenshot shows the 'DATA ITEM MAPPING COMPONENT' configuration window. It has a title bar with a 'Help' icon and a close button. Below the title bar are tabs: 'Display', 'Data' (selected), 'Validation', 'API', 'Conditional', 'Logic', and 'Layout'. The main area contains: 'Refresh On' (dropdown menu), 'Clear Value On Refresh' (checkbox), a section with '+ Custom Default Value' and '+ Calculated Value' (both with plus icons), and a checkbox 'Allow the calculated value to be overridden manually'. On the right, there is a 'Preview' section showing a visual representation of the mapping. At the bottom right of the preview are 'Save', 'Cancel', and 'Remove' buttons.