

JSLayoutContainer



Apr 04, 2024 05:51

Supported Clients

NGClient

Property Summary

String	cssClasses	The css classes to be output for html tag.
String	elementId	The id to be output for html tag.
Number	height	Get/set container height.
String	name	The name of the component.
String	packageName	returns the layouts package name
String	specName	returns the layouts spec name
String	style	The style definition to be output in html tag.
String	tagType	The tag type for html output.
Number	x	Get/set x location.
Number	y	Get/set Y location.

Methods Summary

JSLayoutContainer	findLayoutContainer(name)	Returns a JSLayoutContainer that has the given name throughout the whole form hierarchy.
JSComponent	findWebComponent(name)	Returns a JSWebComponent that has the given name through the whole hierarchy of JSLayoutContainers
String	getAttribute(name)	Returns the comment of this container.
String	getComment()	
JSComponent	getComponent(name)	
Array	getComponents()	Returns a JSComponent that has the given name; if found it will be a JSField, JSLabel, JSButton, JSPortal, JSBean, JSWebComponent or JSTabPanel.
Array	getComponents(returnInheritedElements)	Returns an array of all the JSComponents that a form has; they are of type JSField, JSLabel, JSButton, JSPortal, JSBean, JSWebComponents or JSTabPanel.
JSLayoutContainer	getLayoutContainer(name)	Returns a array of all the JSComponents that a form has; they are of type JSField, JSLabel, JSButton, JSPortal, JSBean, JSWebComponent or JSTabPanel.
Array	getLayoutContainers()	Returns a JSLayoutContainer that has the given name of this container.
Array	getLayoutContainers(returnInheritedElements)	Returns all JSLayoutContainers objects of this container.
JSComponent	getWebComponent(name)	Returns all JSLayoutContainers objects of this container
Array	getWebComponents()	Returns a JSWebComponent that has the given name that is a child of this layout container.
Array	getWebComponents(returnInheritedElements)	Returns all JSWebComponents of this form/container.
JSLayoutContainer	newLayoutContainer()	Returns all JSWebComponents of this form/container.
JSLayoutContainer	newLayoutContainer(position)	Create a new layout container as the last child of its parent container.
JSLayoutContainer	newLayoutContainer(position, spec)	Create a new layout container.
JSLayoutContainer	newLayoutContainer(spec)	Create a new layout container.
JSComponent	newWebComponent(type)	Create a new layout container as the last child in its parent container.
JSComponent	newWebComponent(type, position)	Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form.
JSComponent	newWebComponent(name, type)	Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form.
JSComponent	newWebComponent(name, type, position)	Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form, as the last component in its parent container.
JSComponent	newWebComponent(name, type, x, y, width, height)	Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form.
void	putAttribute(key, value)	Creates a new JSWebComponent (spec based component) object on the form.
void	remove()	Remove a layout container (with all its children) from hierarchy.
Boolean	removeComponent(name)	Removes a component (JSLabel, JSButton, JSField, JSPortal, JSBean, JSTabpanel, JSWebComponent) that has the given name.
Boolean	removeWebComponent(name)	Removes a JSWebComponent that has the specified name.

Property Details

cssClasses

The css classes to be output for html tag.

Returns[String](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.cssClasses = 'myContainer';
```

elementId

The id to be output for html tag.

Returns[String](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.elementId = 'rowCol';
```

height

Get/set container height. This is only used for CSS Position Container.

Returns[Number](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.height = 300;
```

name

The name of the component. Through this name it can also accessed in methods.
Must be a valid javascript name. (no - in the name or start with number)

Returns[String](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.name = 'col1';
```

packageName

returns the layouts package name

Returns[String](#) String**Supported Clients**

NGClient

Sample**specName**

returns the layouts spec name

Returns[String](#) String**Supported Clients**

NGClient

Sample**style**

The style definition to be output in html tag.

Returns[String](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.style = "background-color:red";
```

tagType

The tag type for html output. Default value is 'div'.

Returns[String](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.tagType = 'span';
```

x

Get/set x location. Location is used for ordering in html output.

Returns[Number](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.x = 100;;
```

y

Get/set Y location. Location is used for ordering in html output.

Returns[Number](#)**Supported Clients**

NGClient

Sample

```
layoutContainer.y = 100;;
```

Methods Details**findLayoutContainer(name)**

Returns a JSLayoutContainer that has the given name throughout the whole form hierarchy.

Parameters

[String](#) name the specified name of the container

Returns

[JSLayoutContainer](#) a JSLayoutContainer object

Supported Clients

NGClient

Sample

```
var container = myForm.findLayoutContainer("row1");
application.output(container.name);
```

findWebComponent(name)

Returns a JSWebComponent that has the given name through the whole hierarchy of JSLayoutContainers

Parameters

[String](#) name the specified name of the web component

Returns

[JSComponent](#) a JSWebComponent object

Supported Clients

NGClient

Sample

```
var btn = myForm.findWebComponent("mycomponent");
application.output(mybean.typeName);
```

getAttribute(name)**Parameters**

[String](#) name the attributes name

Returns

[String](#)

Supported Clients

NGClient

Sample

```
layoutContainer.getAttribute('class');
```

getComment()

Returns the comment of this container.

Returns

[String](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var comment = solutionModel.getForm("my_form").getComment();
application.output(comment);
```

getComponent(name)

Returns a JSComponent that has the given name; if found it will be a JSField, JSLabel, JSButton, JSPortal, JSBean, JSWebComponent or JSTabPanel.

Parameters

[String](#) name the specified name of the component

Returns

[JSComponent](#) a JSComponent object (might be a JSField, JSLabel, JSButton, JSPortal, JSBean, JSWebComponent or JSTabPanel)

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var frm = solutionModel.getForm("myForm");
var cmp = frm.getComponent("componentName");
application.output("Component type and name: " + cmp);
```

getComponents()

Returns a array of all the JSComponents that a form has; they are of type JSField, JSLabel, JSButton, JSPortal, JSBean, JSWebComponents or JSTabPanel.

Returns

[Array](#) an array of all the JSComponents on the form.

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var form = solutionModel.getForm("myForm");
var components = form.getComponents();
for (var i in components)
    application.output("Component type and name: " + components[i]);
```

getComponents(returnInheritedElements)

Returns a array of all the JSComponents that a form has; they are of type JSField, JSLabel, JSButton, JSPortal, JSBean, JSWebComponent or JSTabPanel.

Parameters

[Boolean](#) returnInheritedElements true to also return the elements from the parent form

Returns

[Array](#) an array of all the JSComponents on the form.

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var form = solutionModel.getForm("myForm");
var components = form.getComponents();
for (var i in components)
    application.output("Component type and name: " + components[i]);
```

getLayoutContainer(name)

Returns a JSLayoutContainer that has the given name of this container.
Use findLayoutContainer() method to find a JSLayoutContainer through the hierarchy

Parameters

[String](#) name the specified name of the container

Returns

[JSLayoutContainer](#) a JSLayoutContainer object

Supported Clients

NGClient

Sample

```
var container = myForm.getLayoutContainer("row1");
application.output(container.name);
```

getLayoutContainers()

Returns all JSLayoutContainers objects of this container.
Does not return the inherited containers, use #getLayoutContainers(true) to get the inherited as well.

Returns

[Array](#) all JSLayoutContainers objects of this container

Supported Clients

NGClient

Sample

```
var frm = solutionModel.getForm("myForm");
var containers = frm.getLayoutContainers();
for (var c in containers)
{
    var fname = containers[c].name;
    application.output(fname);
}
```

getLayoutContainers(returnInheritedElements)

Returns all JSLayoutContainers objects of this container

Parameters

[Boolean](#) returnInheritedElements true to also return the elements from parent form

Returns

[Array](#) all JSLayoutContainers objects of this container

Supported Clients

NGClient

Sample

```
var frm = solutionModel.getForm("myForm");
var containers = frm.getLayoutContainers();
for (var c in containers)
{
    var fname = containers[c].name;
    application.output(fname);
}
```

getWebComponent(name)

Returns a JSWebComponent that has the given name that is a child of this layout container.
Use findWebComponent() to find a webcomponent through the hierarchy

Parameters

[String](#) name the specified name of the web component

Returns

[JSComponent](#) a JSWebComponent object

Supported Clients

NGClient

Sample

```
var btn = myForm.getWebComponent("mycomponent");
application.output(mybean.typeName);
```

getWebComponents()

Returns all JSWebComponents of this form/container.
If this method is called on a form, then it will return all web components on that form.
If the form is responsive, it will return the web components from all the containers.
It does not return the inherited components, use #getWebComponents(true) to get the inherited as well.

Returns

[Array](#) the list of all JSWebComponent on this forms

Supported Clients

NGClient

Sample

```
var webComponents = myForm.getWebComponents();
for (var i in webComponents)
{
    if (webComponents[i].name != null)
        application.output(webComponents[i].name);
}
```

getWebComponents(returnInheritedElements)

Returns all JSWebComponents of this form/container.
 If this method is called on a form, then it will return all web components on that form.
 If the form is responsive, it will return the web components from all the containers.

Parameters

[Boolean](#) returnInheritedElements true to also return the elements from parent form

Returns

[Array](#) the list of all JSWebComponents on this forms

Supported Clients

NGClient

Sample

```
var webComponents = myForm.getWebComponents(false);
for (var i in webComponents)
{
    if (webComponents[i].name != null)
        application.output(webComponents[i].name);
}
```

newLayoutContainer()

Create a new layout container as the last child of its parent container.
 This method can only be used in responsive forms.

If you want to use default values and so on from a layout package (like 12grid) or if you use the solution model to create a form that is saved back into the workspace (servoyDeveloper.save(form)) then you have to set the packageName and specName properties. So that it works later on in the designer.

If the packageName and specName are not provided, then:
 the packageName is the same as for the parent container
 the specName is the first allowed child defined in the specification of the parent container

If the specification of the parent container does not defined allowed children, then if it is not empty the packageName and the specName are copied from the first child layout container.

Returns

[JSLayoutContainer](#) the new layout container

Supported Clients

NGClient

Sample

```
var container = form.newLayoutContainer();
container.packageName = "12grid";
container.specName = "row";
```

newLayoutContainer(position)

Create a new layout container. The position is used to determine the generated order in html markup.
 This method can only be used in responsive forms.

If you want to use default values and so on from a layout package (like 12grid) or if you use the solution model to create a form that is saved back into the workspace (servoyDeveloper.save(form)) then you have to set the packageName and specName properties. So that it works later on in the designer.

If the packageName and specName are not provided, then:
 the packageName is the same as for the parent container
 the specName is the first allowed child defined in the specification of the parent container

If the specification of the parent container does not defined allowed children, then if it is not empty the packageName and the specName are copied from the first child layout container.

Parameters

[Number](#) position the position of JSWebComponent object in its parent container

Returns

[JSLayoutContainer](#) the new layout container

Supported Clients

NGClient

Sample

```
var container = form.newLayoutContainer(1);
container.packageName = "12grid";
container.specName = "row";
```

newLayoutContainer(position, spec)

Create a new layout container. The position is used to determine the generated order in html markup. This method can only be used in responsive forms.

Parameters

[Number](#) position the position of JSWebComponent object in its parent container

[String](#) spec a string of the form 'packageName-layoutName', or 'layoutName'

Returns

[JSLayoutContainer](#) the new layout container

Supported Clients

NGClient

Sample

```
var container = form.newLayoutContainer(1, "12grid-row");
container.newLayoutContainer(1, "column");
```

newLayoutContainer(spec)

Create a new layout container as the last child in its parent container. This method can only be used in responsive forms.

Parameters

[String](#) spec a string of the form 'packageName-layoutName', or 'layoutName'

Returns

[JSLayoutContainer](#) the new layout container

Supported Clients

NGClient

Sample

```
var container = form.newLayoutContainer(1, "12grid-row");
container.newLayoutContainer(1, "column");
```

newWebComponent(type)

Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form. Will receive a generated name. Will be added as last position in container.

Parameters

[String](#) type the webcomponent name as it appears in the spec

Returns

[JSComponent](#) a JSWebComponent object

Supported Clients

NGClient

Sample

```
var form = solutionModel.newForm('newForm1', 'db:/server1/table1', null, true, 800, 600);
var container = myForm.getLayoutContainer("row1")
var bean = container.newWebComponent('mypackage-testcomponent');
```

newWebComponent(type, position)

Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form.
Will receive a generated name.

Parameters

[String](#) type the webcomponent name as it appears in the spec
[Number](#) position the position of JSWebComponent object in its parent container

Returns

[JSComponent](#) a JSWebComponent object

Supported Clients

NGClient

Sample

```
var form = solutionModel.newForm('newForm1', 'db:/server1/table1', null, true, 800, 600);
var container = myForm.getLayoutContainer("row1")
var bean = container.newWebComponent('mypackage-testcomponent',1);
```

newWebComponent(name, type)

Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form,
as the last component in its parent container.

Parameters

[String](#) name the specified name of the JSWebComponent object
[String](#) type the webcomponent name as it appears in the spec

Returns

[JSComponent](#) a JSWebComponent object

Supported Clients

NGClient

Sample

```
var form = solutionModel.newForm('newForm1', 'db:/server1/table1', null, true, 800, 600);
var container = myForm.getLayoutContainer("row1")
var bean = container.newWebComponent('bean', 'mypackage-testcomponent');
```

newWebComponent(name, type, position)

Creates a new JSWebComponent (spec based component) object on the RESPONSIVE form.

Parameters

[String](#) name the specified name of the JSWebComponent object
[String](#) type the webcomponent name as it appears in the spec
[Number](#) position the position of JSWebComponent object in its parent container

Returns

[JSComponent](#) a JSWebComponent object

Supported Clients

NGClient

Sample

```
var form = solutionModel.newForm('newForm1', 'db:/server1/table1', null, true, 800, 600);
var container = myForm.getLayoutContainer("row1")
var bean = container.newWebComponent('bean', 'mypackage-testcomponent',1);
```

newWebComponent(name, type, x, y, width, height)

Creates a new JSWebComponent (spec based component) object on the form.

Parameters

[String](#) name the specified name of the JSWebComponent object
[String](#) type the webcomponent name as it appears in the spec
[Number](#) x the horizontal "x" position of the JSWebComponent object in pixels
[Number](#) y the vertical "y" position of the JSWebComponent object in pixels
[Number](#) width the width of the JSWebComponent object in pixels
[Number](#) height the height of the JSWebComponent object in pixels

Returns

[JSComponent](#) a JSWebComponent object

Supported Clients

NGClient

Sample

```
var form = solutionModel.newForm('newForm1', 'db:/server1/table1', null, true, 800, 600);
var bean = form.newWebComponent('bean', 'mypackage-testcomponent', 200, 200, 300, 300);
forms['newForm1'].controller.show();
```

putAttribute(key, value)**Parameters**

[Object](#) key ;
[String](#) value ;

Supported Clients

NGClient

Sample

```
layoutContainer.putAttribute('class', 'container fluid');
```

remove()

Remove a layout container (with all its children) from hierarchy.

Supported Clients

NGClient

Sample

```
layoutContainer.remove();
```

removeComponent(name)

Removes a component (JSLabel, JSButton, JSField, JSPortal, JSBean, JSTabpanel, JSWebComponent) that has the given name. It is the same as calling "if(!removeLabel(name) && !removeButton(name))". Returns true if removal was successful, false otherwise.

Parameters

[String](#) name the specified name of the component to be deleted

Returns

[Boolean](#) true if component has been successfully deleted; false otherwise

Supported Clients

SmartClient, WebClient, NGClient

Sample

```

var form = solutionModel.newForm('newFormX', 'db:/server1/parent_table', null, true, 1000, 750);
var jsbutton = form.newButton('JSButton to delete', 100, 100, 200, 50, null);
jsbutton.name = 'jsb';
var jslabel = form.newLabel('JSLabel to delete', 100, 200, 200, 50, null);
jslabel.name = 'jsl';
jslabel.transparent = false;
jslabel.background = 'green';
var jsfield = form.newField('scopes.globals.myGlobalVariable', JSField.TEXT_FIELD, 100, 300, 200, 50);
jsfield.name = 'jsf';
var relation = solutionModel.newRelation('parentToChild', 'db:/server1/parent_table', 'db:/server1/child_table',
JSRelation.INNER_JOIN);
relation.newRelationItem('parent_table_id', '=', 'child_table_id');
var jsportal = form.newPortal('jsp', relation, 100, 400, 300, 300);
jsportal.newField('child_table_id', JSField.TEXT_FIELD, 200, 200, 120);
var childOne = solutionModel.newForm('childOne', 'db:/server1/child_table', null, false, 400, 300);
childOne.newField('child_table_id', JSField.TEXT_FIELD, 10, 10, 100, 20);
var childTwo = solutionModel.newForm('childTwo', 'server1', 'other_table', null, false, 400, 300);
childTwo.newField('some_table_id', JSField.TEXT_FIELD, 10, 10, 100, 100);
var jstabpanel = form.newTabPanel('jst', 450, 30, 620, 460);
jstabpanel.newTab('tab1', 'Child One', childOne, relation);
jstabpanel.newTab('tab2', 'Child Two', childTwo);
var jsmethod = form.newMethod("function removeMe(event) { var form = solutionModel.getForm('newFormX');\n if
((form.removeComponent('jsb') == true) && (form.removeComponent('jst') == true) && (form.removeComponent('jsf')
== true) && (form.removeComponent('jsp') == true) & (form.removeComponent('jst') == true)) application.output
('Components removed ok'); else application.output('Some component(s) could not be deleted'); forms['newFormX'].
controller.recreateUI();});");
var removerButton = form.newButton('Click here to remove form components', 450, 500, 250, 50, jsmethod);
removerButton.name = 'remover';
forms['newFormX'].controller.show();

```

removeWebComponent(name)

Removes a JSWebComponent that has the specified name. Returns true if removal was successful, false otherwise.

Parameters

[String](#) name the specified name of the JSWebComponent to be removed

Returns

[Boolean](#) true if the JSWebComponent has been removed; false otherwise

Supported Clients

NGClient

Sample

```

var form = solutionModel.getForm('myform');
form.removeWebComponent('mybean')

```