

JSComponent

 Apr 06, 2024 16:24

Supported Clients

SmartClient WebClient NGClient MobileClient

Property Summary		
Number	anchors	Enables a component to stick to a specific side of form and/or to grow or shrink when a window is resized.
String	background	The background color of the component.
String	borderType	The type, color and style of border of the component.
CSSPosition	cssPosition	CSS position is a replacement for anchoring system making it more intuitive to place a component.
Boolean	enabled	The enable state of the component, default true.
String	fontType	The font type of the component.
String	foreground	The foreground color of the component.
Number	formIndex	The Z index of this component.
String	groupID	A String representing a group ID for this component.
Number	height	The height in pixels of the component.
String	name	The name of the component.
Number	printSliding	Enables an element to resize based on its content and/or move when printing.
Boolean	printable	Flag that tells if the component should be printed or not when the form is printed.
String	styleClass	The name of the style class that should be applied to this component.
Boolean	transparent	Flag that tells if the component is transparent or not.
Boolean	visible	The visible property of the component, default true.
Number	width	The width in pixels of the component.
Number	x	The x coordinate of the component on the form.
Number	y	The y coordinate of the component on the form.

Methods Summary		
Object	getAttribute(name)	Get the value of an attribute of the element.
Array	getAttributes()	Returns the attribute names of an element.
String	getComment()	Returns the comment of this component.
Object	getDesignTimeProperty(key)	Get a design-time property of an element.
Array	getDesignTimePropertyNames()	Get the design-time properties of an element.
String	getFormName()	Returns the name of the form.
UUID	getUUID()	Returns the UUID of this component.
Object	putDesignTimeProperty(key, value)	Set a design-time property of an element.
String	removeAttribute(name)	Remove the attribute of an element.
Object	removeDesignTimeProperty(key)	Clear a design-time property of an element.
void	setAttribute(name, value)	Set the attribute value of an element.

Property Details

anchors

Enables a component to stick to a specific side of form and/or to grow or shrink when a window is resized.

If opposite anchors are activated then the component with grow or shrink with the window. For example if Top and Bottom are activated, then the component will grow/shrink when the window is vertically resized. If Left and Right are activated then the component will grow/shrink when the window is horizontally resized.

If opposite anchors are not activated, then the component will keep a constant distance from the sides of the window which correspond to the activated anchors.

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var form = solutionModel.newForm('mediaForm', 'db:/example_data/parent_table', null, false, 400, 300);
var stretchAllDirectionsLabel = form.newLabel('Strech all directions', 10, 10, 380, 280);
stretchAllDirectionsLabel.background = 'red';
stretchAllDirectionsLabel.anchors = SM_ANCHOR.ALL;
var stretchVerticallyLabel = form.newLabel('Strech vertically', 10, 10, 190, 280);
stretchVerticallyLabel.background = 'green';
stretchVerticallyLabel.anchors = SM_ANCHOR.WEST | SM_ANCHOR.NORTH | SM_ANCHOR.SOUTH;
var stretchHorizontallyLabel = form.newLabel('Strech horizontally', 10, 10, 380, 140);
stretchHorizontallyLabel.background = 'blue';
stretchHorizontallyLabel.anchors = SM_ANCHOR.NORTH | SM_ANCHOR.WEST | SM_ANCHOR.EAST;
var stickToTopLeftCornerLabel = form.newLabel('Stick to top-left corner', 10, 10, 200, 100);
stickToTopLeftCornerLabel.background = 'orange';
stickToTopLeftCornerLabel.anchors = SM_ANCHOR.NORTH | SM_ANCHOR.WEST; // This is equivalent to SM_ANCHOR.DEFAULT
var stickToBottomRightCornerLabel = form.newLabel('Stick to bottom-right corner', 190, 190, 200, 100);
stickToBottomRightCornerLabel.background = 'pink';
stickToBottomRightCornerLabel.anchors = SM_ANCHOR.SOUTH | SM_ANCHOR.EAST;
```

background

The background color of the component.

Returns

[String](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
// This property can be used on all types of components.
// Here it is illustrated only for labels and fields.
var greenLabel = form.newLabel('Green',10,10,100,50);
greenLabel.background = 'green'; // Use generic names for colors.
var redField = form.newField('parent_table_text',JSField.TEXT_FIELD,10,110,100,30);
redField.background = '#FF0000'; // Use RGB codes for colors.
```

borderType

The type, color and style of border of the component.

Returns

[String](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
//HINT: To know exactly the notation of this property set it in the designer and then read it once out through
the solution model.
var field = form.newField('my_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
field.borderType = solutionModel.createLineBorder(1,'#ff0000');
```

cssPosition

CSS position is a replacement for anchoring system making it more intuitive to place a component.
 CSS position should be set on form, an absolute position form can either work with anchoring or with css position.
 This is only working in NGClient.

Returns

[CSSPosition](#)

Supported Clients

NGClient

Sample

```
var label = form.newLabel('Label', -1);
label.cssPosition.r("10").b("10").w("20%).h("30px")
```

enabled

The enable state of the component, default true.

Returns

[Boolean](#)

Supported Clients

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var form = solutionModel.newForm('printForm', 'db:/example_data/parent_table', null, false, 400, 300);
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
field.enabled = false;
```

fontType

The font type of the component.

Returns

[String](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var label = form.newLabel('Text here', 10, 50, 100, 20);
label.fontType = solutionModel.createFont('Times New Roman',1,14);
```

foreground

The foreground color of the component.

Returns

[String](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
// This property can be used on all types of components.
// Here it is illustrated only for labels and fields.
var labelWithBlueText = form.newLabel('Blue text', 10, 10, 100, 30);
labelWithBlueText.foreground = 'blue'; // Use generic names for colors.
var fieldWithYellowText = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 50, 100, 20);
fieldWithYellowText.foreground = '#FFFF00'; // Use RGB codes for colors.
```

formIndex

The Z index of this component. If two components overlap, then the component with higher Z index is displayed above the component with lower Z index.

Returns

[Number](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var labelBelow = form.newLabel('Green', 10, 10, 100, 50);
labelBelow.background = 'green';
labelBelow.formIndex = 10;
var fieldAbove = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 30);
fieldAbove.background = '#FF0000';
fieldAbove.formIndex = 20;
```

groupId

A String representing a group ID for this component. If several components have the same group ID then they belong to the same group of components. Using the group itself, all components can be disabled/enabled or made invisible/visible.

The group id should be a javascript compatible identifier to allow access of the group in scripting.

Returns

[String](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var form = solutionModel.newForm('someForm', 'db:/example_data/parent_table', null, false, 400, 300);
var label = form.newLabel('Green', 10, 10, 100, 20);
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 40, 100, 20);
label.groupID = 'someGroup';
field.groupID = 'someGroup';
forms['someForm'].elements.someGroup.enabled = false;
```

height

The height in pixels of the component.

Returns

[Number](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
application.output('original width: ' + field.width);
application.output('original height: ' + field.height);
field.width = 200;
field.height = 100;
application.output('modified width: ' + field.width);
application.output('modified height: ' + field.height);
```

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

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var form = solutionModel.newForm('someForm', 'db:/example_data/parent_table', null, false, 620, 300);
var label = form.newLabel('Label', 10, 10, 150, 150);
label.name = 'myLabel'; // Give a name to the component.
forms['someForm'].controller.show()
// Now use the name to access the component.
forms['someForm'].elements['myLabel'].text = 'Updated text';
```

printSliding

Enables an element to resize based on its content and/or move when printing. The component can move horizontally or vertically and can grow or shrink in height and width, based on its content and the content of neighboring components.

Returns

[Number](#)

Supported Clients

SmartClient, WebClient

Sample

```
var form = solutionModel.newForm('printForm', 'db:/example_data/parent_table', null, false, 400, 300);
var slidingLabel = form.newLabel('Some long text here', 10, 10, 5, 5);
slidingLabel.printSliding = SM_PRINT_SLIDING.GROW_HEIGHT | SM_PRINT_SLIDING.GROW_WIDTH;
slidingLabel.background = 'gray';
forms['printForm'].controller.showPrintPreview();
```

printable

Flag that tells if the component should be printed or not when the form is printed.

By default components are printable.

Returns

[Boolean](#)

Supported Clients

SmartClient, WebClient

Sample

```
var form = solutionModel.newForm('printForm', 'db:/example_data/parent_table', null, false, 400, 300);
var printedField = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
var notPrintedField = form.newField('parent_table_id', JSField.TEXT_FIELD, 10, 40, 100, 20);
notPrintedField.printable = false; // This field won't show up in print preview and won't be printed.
forms['printForm'].controller.showPrintPreview();
```

styleClass

The name of the style class that should be applied to this component.

When defining style classes for specific component types, their names must be prefixed according to the type of the component. For example in order to define a class names 'fancy' for fields, in the style definition the class must be named 'field.fancy'. If it would be intended for labels, then it would be named 'label.fancy'. When specifying the class name for a component, the prefix is dropped however. Thus the field or the label will have its styleClass property set to 'fancy' only.

Returns

[String](#)

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var form = solutionModel.newForm('printForm', 'db:/example_data/parent_table', null, false, 400, 300);
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
var style = solutionModel.newStyle('myStyle', 'field.fancy { background-color: yellow; }');
form.styleName = 'myStyle'; // First set the style on the form.
field.styleClass = 'fancy'; // Then set the style class on the field.
```

transparent

Flag that tells if the component is transparent or not.

The default value is "false", that is the components are not transparent.

Returns

[Boolean](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
// Load an image from disk and create a Media object based on it.
var imageBytes = plugins.file.readFile('d:/ball.jpg');
var media = solutionModel.newMedia('ball.jpg', imageBytes);
// Put on the form a label with the image.
var image = form.newLabel('', 10, 10, 100, 100);
image.imageMedia = media;
// Put two fields over the image. The second one will be transparent and the
// image will shine through.
var nonTransparentField = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 20, 100, 20);
var transparentField = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 50, 100, 20);
transparentField.transparent = true;
```

visible

The visible property of the component, default true.

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

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var form = solutionModel.newForm('printForm', 'db:/example_data/parent_table', null, false, 400, 300);
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
field.visible = false;
```

width

The width in pixels of the component.

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

SmartClient,WebClient,NGClient

Sample

```
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
application.output('original width: ' + field.width);
application.output('original height: ' + field.height);
field.width = 200;
field.height = 100;
application.output('modified width: ' + field.width);
application.output('modified height: ' + field.height);
```

x

The x coordinate of the component on the form.

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

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
application.output('original location: ' + field.x + ', ' + field.y);
field.x = 90;
field.y = 90;
application.output('changed location: ' + field.x + ', ' + field.y);
```

y

The y coordinate of the component on the form.

Returns

[Number](#)

Supported Clients

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
application.output('original location: ' + field.x + ', ' + field.y);
field.x = 90;
field.y = 90;
application.output('changed location: ' + field.x + ', ' + field.y);
```

Methods Details**getAttribute(name)**

Get the value of an attribute of the element.

Parameters

[String](#) name the name of the attribute

Returns

[Object](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
var attributes = fld.getAttributes();
for (var i = 0; i < attributes.length; i++)
{
    application.output(fld.getAttribute(attributes[i]));
}
```

getAttributes()

Returns the attribute names of an element.

Returns

[Array](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
var attributes = fld.getAttributes();
for (var i = 0; i < attributes.length; i++)
{
    application.output(fld.getAttribute(attributes[i]));
}
```

getComment()

Returns the comment of this component.

Returns

[String](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

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

getDesignTimeProperty(key)

Get a design-time property of an element.

Parameters

[String](#) key the name of the property

Returns

[Object](#)

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
var prop = fld.getDesignTimeProperty('myprop')
```

getDesignTimePropertyNames()

Get the design-time properties of an element.

Returns

[Array](#)

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
var propName = fld.getDesignTimePropertyNames()
```

getFormName()

Returns the name of the form. (may be empty string as well)

Returns

[String](#) The name of the form.

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var name = %%elementName%%.getFormName();
```

getUUID()

Returns the UUID of this component.

Returns

[UUID](#)

Supported Clients

SmartClient, WebClient, NGClient

Sample

```
var button_uuid = solutionModel.getForm("my_form").getButton("my_button").getUUID();
application.output(button_uuid.toString());
```

putDesignTimeProperty(key, value)

Set a design-time property of an element.

Parameters

[String](#) key the name of the property
[Object](#) value the value to store

Returns

[Object](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
fld.putDesignTimeProperty('myprop', 'strawberry')
```

removeAttribute(name)

Remove the attribute of an element.

Parameters

[String](#) name the name of the attribute

Returns

[String](#) the deleted attribute value

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
fld.removeAttribute('keylistener')
```

removeDesignTimeProperty(key)

Clear a design-time property of an element.

Parameters

[String](#) key the name of the property

Returns

[Object](#)

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
fld.removeDesignTimeProperty('myprop')
```

setAttribute(name, value)

Set the attribute value of an element.

Parameters

[String](#) name the name of the attribute
[String](#) value the value of the attribute

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
var frm = solutionModel.getForm('orders')
var fld = frm.getField('fld')
fld.setAttribute('keylistener', 'callback')
```