JSPassword

Apr 26, 2024 12:11

Supported Clients

MobileClient

Property Summary

String	dataProviderID	The dataprovider of the component.
Number	displayType	The type of display used by the field.
Boolean	enabled	The enable state of the component, default true.
String	format	The format that should be applied when displaying the data in the component.
String	name	The name of the component.
JSMethod	onAction	The method that is executed when the component is clicked.
JSMethod	onDataChange	Method that is executed when the data in the component is successfully changed.
Boolean	visible	The visible property of the component, default true.
Number	x	The x coordinate of the component on the form.
Number	y	The y coordinate of the component on the form.

Property Details

dataProviderID

The dataprovider of the component.

Returns

String

Supported Clients

SmartClient,WebClient,NGClient,MobileClient

Sample

```
// Normally the dataprovider is specified when a component is created.
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 40, 100, 20);
// But it can be modified later if needed.
field.dataProviderID = 'parent_table_id';
```

displayType

The type of display used by the field. Can be one of CALENDAR, CHECKS, COMBOBOX, HTML_AREA, IMAGE_MEDIA, PASSWORD, RADIOS, RTF_AREA, TEXT_AREA, TEXT_FIELD, TYPE_AHEAD, LIST_BOX, MULTISELECT_LISTBOX or SPINNER.

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient,MobileClient

Sample

```
// The display type is specified when the field is created.
var cal = form.newField('my_table_date', JSField.CALENDAR, 10, 10, 100, 20);
// But it can be changed if needed.
cal.dataProviderID = 'my_table_text';
cal.displayType = JSField.TEXT_FIELD;
```

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;
```

format

The format that should be applied when displaying the data in the component.
There are different options for the different dataprovider types that are assigned to this field.
For Integer fields, there is a display and an edit format, using http://docs.oracle.com/javase/7/docs/api/java/text/DecimalFormat.html and the max (string) length can be set.
For Text/String fields, there are options to force uppercase,lowercase or only numbers. Or a mask can be set that restrict the input the pattern chars can be found here: http://docs.oracle.com/javase/7/docs/api/javax/swing/text/MaskFormatter.html
A mask can have a placehoder (what is shown when there is no data) and if the data must be stored raw (without literals of the mask). A max text length can also be set to force
the max text length input, this doesn't work on mask because that max length is controlled with the mask.
For Date fields a display and edit format can be set by using a pattern from here: http://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html, you can also say this must behave like a mask (the edit format)
A mask only works with when the edit format is exactly that mask (1 char is 1 number/char), because for example MM then only 2 numbers are allowed MMM that displays the month as a string is not supported as a mask.
Some examples are "dd-MM-yyyy", "MM-dd-yyyy", etc.
The format property is also used to set the UI Converter, this means that you can convert the value object to something else before it gets set into the field, this can also result in a type change of the data.
So a string in scripting/db is converted to a integer in the ui, then you have to set an integer format.
This property is applicable only for types: TEXT_FIELD, COMBOBOX, TYPE_AHEAD, CALENDAR and SPINNER.

Returns

String

Supported Clients

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var field = form.newField('my_table_number', JSField.TEXT_FIELD, 10, 10, 100, 20);
field.format = '$#.00';
```

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';
```

onAction

The method that is executed when the component is clicked.

Returns

JSMethod

Supported Clients

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var doNothingMethod = form.newMethod('function doNothing() { application.output("Doing nothing."); }');
var onClickMethod = form.newMethod('function onClick(event) { application.output("I was clicked at " + event.
getTimestamp()); }');
var onDoubleClickMethod = form.newMethod('function onDoubleClick(event) { application.output("I was double-
clicked at " + event.getTimestamp()); }');
var onRightClickMethod = form.newMethod('function onRightClick(event) { application.output("I was right-clicked
at " + event.getTimestamp()); }');
// At creation the button has the 'doNothing' method as onClick handler, but we'll change that later.
var btn = form.newButton('I am a button', 10, 40, 200, 20, doNothingMethod);
btn.onAction = onClickMethod;
btn.onDoubleClick = onDoubleClickMethod;
btn.onRightClick = onRightClickMethod;
```

onDataChange

Method that is executed when the data in the component is successfully changed.

Returns

JSMethod

Supported Clients

SmartClient,WebClient,NGClient,MobileClient

Sample

```
var form = solutionModel.newForm('someForm', 'db:/example_data/parent_table', null, false, 620, 300);
var onDataChangeMethod = form.newMethod('function onDataChange(oldValue, newValue, event) { application.output
("Data changed from " + oldValue + " to " + newValue + " at " + event.getTimestamp()); }');
var field = form.newField('parent_table_text', JSField.TEXT_FIELD, 10, 10, 100, 20);
field.onDataChange = onDataChangeMethod;
forms['someForm'].controller.show();
```

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;
```

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);
```