RuntimeTextField

Extends

RuntimeComponent

Property Summary

String	bgcolor Gets or sets the background color of a field.
String	border Gets or sets the border attribute(s) of a specified element.
Number	caretPosition Gets or sets the number value (position) of the text caret (text "I" bar) in a field.
Boolean	editable Gets or sets the editable/read-only state of a field; true - editable; false - read-only.
Boolean	enabled Gets or sets the enabled state of a specified field, also known as "grayed".
String	fgcolor Gets or sets the foreground color of a field.
String	font Gets or sets the font name, style, and size of an element.
String	format Gets or sets the display formatting of an element for number and text values; does not affect the actual value stored in the database column.
String	placeholderText Gets or sets the placeholder text in a field.
Boolean	readOnly Gets or sets the editable/read-only state of a field; true - read-only; false - editable; ! - the editable/read-only state is inverted (the opposite).
String	titleText Gets or sets the title text.
String	toolTipText Gets or sets the tool tip text of an element; text displays when the mouse cursor hovers over an element.
Boolean	transparent Gets or sets the transparency of an element; true - transparent; false - not transparent.
Boolean	visible Gets or sets the visibility of an element; true - visible; false - not visible; ! - the visibility state is inverted (the opposite).

Method Summary

Number	getAbsoluteFormLocationY() Returns the absolute form (designed) Y location.
Object	getClientProperty(key) Gets the specified client property for the element based on a key.
String	getDataProviderID() Get the data provider this UI element (display) is showing.
Object	getDesignTimeProperty() Get a design-time property of an element.
String	getElementType() Returns the type of a specified element.
Number	getHeight() Returns the height of the current element.
String[]	getLabelForElementNames() Returns an Array of label element names that has this field filled in as the labelFor.
Number	getLocationX() Returns the x location of the current element.
Number	getLocationY() Returns the y location of the current element.
String	getName() Returns the name of an element.
String	getSelectedText() Returns the currently selected text in the specified text field or text area.
String	getValueListName() Returns the current valuelist name for the specified field; returns NULL if no valuelist.

Number	getWidth() Returns the width of the current element.
void	putClientProperty(key, value) Sets the value for the specified element client property key.
void	replaceSelectedText(s) Replaces the selected text; if no text has been selected, the replaced value will be inserted at the last cursor position.
void	requestFocus() Request the focus in this element.
void	requestFocus(mustExecuteOnFocusGainedMethod) Request the focus in this element.
void	selectAll() Selects all the contents of a field.
void	setLocation(x, y) Sets the location of an element.
void	setSize(width, height) Sets the size of an element.
void	setValueListItems(value) Sets the display/real values to the custom valuelist of the element (if element has custom valuelist).

Property Details

bgcolor

Gets or sets the background color of a field. The color has to be set using the hexadecimal RGB value as used in HTML.
It only returns it's correct value if it was explicitly set.

Returns

String

Sample

```
//sets the background color of the field
forms.customer.elements.customer_id.bgcolor = "#FFFFFF";
//gets the background color of the field
var c = forms.customer.elements.customer_id.bgcolor;
```

border

Gets or sets the border attribute(s) of a specified element.

The border attributes:

borderType - EmptyBorder, EtchedBorder, BevelBorder, LineBorder, TitleBorder, MatteBorder, SpecialMatteBorder.
size - (numeric value) for: bottom, left, right, top.
color - (hexadecimal value) for: bottom, left, right, top.
dash pattern - (numeric value) for selected side(s).
rounding radius - (numeric value) for selected side(s).

It only returns it's correct value if it was explicitly set.

NOTE: Use the same value(s) and order of attribute(s) from the element design time property "borderType".

Returns

String

Sample

```
//sets the border type to "LineBorder"
//sets a 1 px line width for the bottom and left side of the border
//sets the hexadecimal color of the border to "#ccffcc"
forms.customer.elements.customer_id.border = 'LineBorder,1,#ccffcc';
```

caretPosition

Gets or sets the number value (position) of the text caret (text "I" bar) in a field.

Returns

Number

Sample

```
//get the current caretposition
var caretPos = forms.customer.elements.customer_id.caretPosition;
//add one and set it
forms.customer.elements.customer_id.caretPosition = caretPos+1;
```

editable

Gets or sets the editable/read-only state of a field; true - editable; false - read-only.

NOTE the "!" operator can be used to invert the editable state.

Returns

Boolean

Sample

```
var currentState = forms.customer.elements.customer_id.editable;
forms.customer.elements.customer_id.editable = !currentState;
```

enabled

Gets or sets the enabled state of a specified field, also known as "grayed".
true - enabled; false - not enabled; ! - the enabled state is inverted (the opposite).

NOTE: A disabled element cannot be selected by clicking the element (or by pressing the TAB key even if this option is supported by the operating system).

NOTE: A label or button element will not disable if the "displayType" design time property for a field is set to HTML_AREA.

NOTE: The disabled "grayed" color is dependent on the LAF set in the Servoy Client Application Preferences. For more information see Preferences: Look And Feel in the Servoy Developer User's Guide.

Returns

Boolean

Sample

```
//gets the enabled state of the field
var currState = forms.customer.elements.customer_id.enabled;

//sets the enabled state of the field
forms.customer.elements.customer_id.enabled = !currentState;
```

fgcolor

Gets or sets the foreground color of a field. The color has to be set using the hexadecimal RGB value as used in HTML.
It only returns it's correct value if it was explicitly set.

Returns

String

Sample

```
//sets the foreground color of the field
forms.customer.elements.customer_id.fgcolor = "#000000";

//gets the foreground color of the field
var c = forms.customer.elements.customer_id.fgcolor;
```

font

Gets or sets the font name, style, and size of an element.

font name - the name of the font family.
style - the type of the font. (plain = 0; bold = 1; italic = 2; bold-italic = 3).
size - the size of the font (in points).

It only returns it's correct value if it was explicitly set.

Returns

String

Sample

```
forms.customer.elements.customer_id.font = 'Tahoma,1,11';
```

format

Gets or sets the display formatting of an element for number and text values; does not affect the actual value stored in the database column.

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.
It only returns it's correct value if it was explicitly set, otherwise null.

Returns

String

Sample

```
//sets the display formatting of the field
forms.customer.elements.customer_id.format = '###';

//gets the display formatting of the field
var format = forms.customer.elements.customer_id.format;
```

placeholderText

Gets or sets the placeholder text in a field.

Returns

String

Sample

```
//get the placeholder text
var placeholderText = forms.customer.elements.customer_id.placeholderText;
//set another placeholder
forms.customer.elements.customer_id.placeholderText = 'Search..';
```

readOnly

Gets or sets the editable/read-only state of a field; true - read-only; false - editable; ! - the editable/read-only state is inverted (the opposite).

NOTE: A field set as read-only can be selected by clicking (or pressing the TAB key if this option is supported by the operating system) and the field data can be copied.

Returns

Boolean

Sample

```
//gets the editable/read-only state of the field
var currentState = forms.customer.elements.customer_id.readOnly;

//sets the editable/read-only state of the field
forms.customer.elements.customer_id.readOnly = !currentState;
```

titleText

Gets or sets the title text.

Returns

String

Sample

```
var titleText = forms.customer.elements.customer_id.titleText;
```

toolTipText

Gets or sets the tool tip text of an element; text displays when the mouse cursor hovers over an element.

NOTE: HTML should be used for multi-line tooltips; you can also use any valid HTML tags to format tooltip text.

Returns

String

Sample

```
//gets the tooltip text of the element
var toolTip = forms.customer.elements.customer_id.toolTipText;

//sets the tooltip text of the element
forms.customer.elements.customer_id.toolTipText = "New tip";
forms.customer.elements.customer_id.toolTipText = "<html>This includes <b>bolded text</b> and <font
color='blue'>BLUE</font> text as well.";
```

transparent

Gets or sets the transparency of an element; true - transparent; false - not transparent.

NOTE: transparency can be inverted using ! operator: elements.elementName.transparent = !elements.elementName.transparent;

NOTE: transparency will be mostly used for background color, a transparent element will receive the background of the element "beneath" it, a non transparent one will use its own background color

Returns

Boolean

Sample

```
//gets the transparency of the element
var currentState = forms.customer.elements.customer_id.transparent;

//sets the transparency of the element
forms.customer.elements.customer_id.transparent = !currentState;
```

visible

Gets or sets the visibility of an element; true - visible; false - not visible; ! - the visibility state is inverted (the opposite).

NOTE: The visibility of an element is not persistent; the state of visibility only applies to the current user in his/her current session.

Returns

Boolean

Sample

```
//sets the element as visible
forms.company.elements.faxBtn.visible = true;

//gets the visibility of the element
var currentState = forms.company.elements.faxBtn.visible;

//sets the element as not visible when the current state is visible
forms.company.elements.faxBtn.visible = !currentState;
```

Method Details

getAbsoluteFormLocationY

Number **getAbsoluteFormLocationY** ()

Returns the absolute form (designed) Y location.

Returns

Number - The y location of the form in pixels.

Sample

```
var absolute_y = forms.customer.elements.customer_id.getAbsoluteFormLocationY();
```

getClientProperty

Object **getClientProperty** (key)

Gets the specified client property for the element based on a key.

NOTE: Depending on the operating system, a user interface property name may be available.

Parameters

{Object} key - user interface key (depends on operating system)

Returns

Object - The value of the property for specified key.

Sample

```
var property = forms.customer.elements.customer_id.getClientProperty('ToolTipText');
```

getDataProviderID

String **getDataProviderID** ()

Get the data provider this UI element (display) is showing.

Returns

String - The data provider as String.

Sample

```
forms.customer.elements.customer_id.getDataProviderID();
```

getDesignTimeProperty

Object **getDesignTimeProperty** ()

Get a design-time property of an element.

Returns

Object

Sample

```
var prop = forms.orders.elements.mylabel.getDesignTimeProperty('myprop')
```

getElementType

String **getElementType** ()

Returns the type of a specified element.

Returns

String - The display type of the element as String.

Sample

```
var et = forms.customer.elements.customer_id.getElementType();
```

getHeight

Number **getHeight** ()

Returns the height of the current element.
NOTE: getHeight() can be used with getWidth() to set the size of an element using the setSize function. For example:

```
//returns the width (w) and height (h)
var w = forms.company.elements.faxBtn.getWidth();
var h = forms.company.elements.faxBtn.getHeight();

//sets the new size
forms.company.elements.faxBtn.setSize(w,h);

//sets the new size and adds 1 px to both the width and height
forms.company.elements.faxBtn.setSize(w+1,h+1);
```

Returns

Number - The height of the element in pixels.

Sample

```
var ht = forms.customer.elements.customer_id.getHeight();
```

getLabelForElementNames

String[] **getLabelForElementNames** ()

Returns an Array of label element names that has this field filled in as the labelFor.

Returns

String[] - An array with element names.

Sample

```
var array = elements.name_first.getLabelForElementNames();
for (var i = 0; i<array.length; i++)
{
        elements[array[i]].fgcolor = "#ff00ff";
}
```

getLocationX

Number **getLocationX** ()

Returns the x location of the current element.

NOTE: getLocationX() can be used with getLocationY() to set the location of an element using the setLocation function. For Example:

```
//returns the X and Y coordinates
var x = forms.company.elements.faxBtn.getLocationX();
var y = forms.company.elements.faxBtn.getLocationY();

//sets the new location 10 px to the right; 10 px down from the current location
forms.company.elements.faxBtn.setLocation(x+10,y+10);
```

Returns

Number - The x location of the element in pixels.

Sample

```
var x = forms.customer.elements.customer_id.getLocationX();
```

getLocationY

Number **getLocationY** ()

Returns the y location of the current element.

NOTE: getLocationY() can be used with getLocationX() to set the location of an element using the setLocation function. For Example:

```
//returns the X and Y coordinates
var x = forms.company.elements.faxBtn.getLocationX();
var y = forms.company.elements.faxBtn.getLocationY();

//sets the new location 10 px to the right; 10 px down from the current location
forms.company.elements.faxBtn.setLocation(x+10,y+10);
```

Returns

Number - The y location of the element in pixels.

Sample

```
var y =  forms.customer.elements.customer_id.getLocationY();
```

getName

String **getName** ()

Returns the name of an element. (may be null as well)

Returns

String - The name of the element.

Sample

```
var name = forms.customer.elements.customer_id.getName();
```

getSelectedText

String **getSelectedText** ()

Returns the currently selected text in the specified text field or text area.

NOTE: This does not work for text fields in the Web Client.

Returns

String - The selected text from the component.

Sample

```
var my_text = forms.customer.elements.customer_id.getSelectedText();
```

getValueListName

String **getValueListName** ()

Returns the current valuelist name for the specified field; returns NULL if no valuelist.

Returns

String - The valuelist name.

Sample

```
var name = forms.customer.elements.customer_id.getValueListName();
```

getWidth

Number **getWidth** ()

Returns the width of the current element.

NOTE: getWidth() can be used with getHeight() to set the size of an element using the setSize function. For Example:

//returns the width (w) and height (h)
var w = forms.company.elements.faxBtn.getWidth();
var h = forms.company.elements.faxBtn.getHeight();

//sets the new size
forms.company.elements.faxBtn.setSize(w,h);

//sets the new size and adds 1 px to both the width and height
forms.company.elements.faxBtn.setSize(w+1,h+1);

Returns

Number - The width of the element in pixels.

Sample

```
var w = forms.customer.elements.customer_id.getWidth();
```

putClientProperty

void **putClientProperty** (key, value)

Sets the value for the specified element client property key.

NOTE: Depending on the operating system, a user interface property name may be available.

Parameters

{Object} key - user interface key (depends on operating system)
{Object} value - a predefined value for the key

Returns

void

Sample

```
forms.customer.elements.customer_id.putClientProperty('ToolTipText','some text');
```

replaceSelectedText

void **replaceSelectedText** (s)

Replaces the selected text; if no text has been selected, the replaced value will be inserted at the last cursor position.

NOTE: replaceSelectedText applies to text fields and all XXX_AREA displayType text - RTF_AREA, HTML_AREA, or TEXT_AREA.

Parameters

{String} s - The replacement text.

Returns

void

Sample

```
//returns the current selected text
var my_text = forms.customer.elements.customer_id.getSelectedText();

//replaces the current selected text
forms.customer.elements.customer_id.replaceSelectedText('John');
```

requestFocus

void **requestFocus** ()

Request the focus in this element. (Focus is also a text cursor on text components).

Returns

void

Sample

```
//request the focus in this forms.customer.elements.customer_id (focus is also a text cursor on text
components)
forms.customer.elements.customer_id.requestFocus();
```

requestFocus

void **requestFocus** (mustExecuteOnFocusGainedMethod)

Request the focus in this element. (Focus is also a text cursor on text components).

Parameters

{Boolean} mustExecuteOnFocusGainedMethod - If true will execute onFocusGained method, else will not; default value is true.

Returns

void

Sample

```
//request the focus in this forms.customer.elements.customer_id (focus is also a text cursor on text
components), skip on focus gained method call
forms.customer.elements.customer_id.requestFocus(false);
```

selectAll

void **selectAll** ()

Selects all the contents of a field.

Returns

void

Sample

```
forms.customer.elements.customer_id.selectAll();
```

setLocation

void **setLocation** (x, y)

Sets the location of an element. It takes as input the X (horizontal) and Y (vertical) coordinates - starting from the TOP LEFT side of the screen.
Please note that location should not be altered at runtime when an element is anchored. Use the solutionModel in such a situation.

NOTE: getLocationX() can be used with getLocationY() to return the current location of an element; then use the X and Y coordinates with the setLocation function to set a new location. For Example:

```
//returns the X and Y coordinates
var x = forms.company.elements.faxBtn.getLocationX();
var y = forms.company.elements.faxBtn.getLocationY();

//sets the new location 10 px to the right; 10 px down from the current location
forms.company.elements.faxBtn.setLocation(x+10,y+10);
```

Parameters

{Number} x - the X coordinate of the element in pixels.
{Number} y - the Y coordinate of the element in pixels.

Returns

void

Sample

```
forms.customer.elements.customer_id.setLocation(200,200);
```

setSize

void **setSize** (width, height)

Sets the size of an element. It takes as input the width and the height.
Please note that size should not be altered at runtime when an element is anchored. Use the solutionModel in such a situation.

NOTE: getWidth() can be used with getHeight() to set the size of an element using the setSize function. For Example:

```
//returns the width (w) and height (h)
var w = forms.company.elements.faxBtn.getWidth();
var h = forms.company.elements.faxBtn.getHeight();

//sets the new size
forms.company.elements.faxBtn.setSize(w,h);

//sets the new size and adds 1 px to both the width and height
forms.company.elements.faxBtn.setSize(w+1,h+1);
```

Parameters

{Number} width - the width of the element in pixels.
{Number} height - the height of the element in pixels.

Returns

void

Sample

```
forms.customer.elements.customer_id.setSize(20,30);
```

setValueListItems

void **setValueListItems** (value)

Sets the display/real values to the custom valuelist of the element (if element has custom valuelist).
This does not effect the value list with same name list on other elements or value lists at application level.

Should receive a dataset parameter, first column is for display values, second column (optional) is for real values.

NOTE: if you modify values for checkbox/radio field, note that having one value in valuelist is a special case, so switching between one value and 0 /multiple values after form is created may have side effects

Parameters

{Object} value - first column is display value, second column is real value

Returns

void

Sample

```
var dataset = databaseManager.createEmptyDataSet(0,new Array('display_values','optional_real_values'));
dataset.addRow(['aa',1]);
dataset.addRow(['bb',2]);
dataset.addRow(['cc',3]);
// forms.customer.elements.customer_id should have a valuelist attached
forms.customer.elements.customer_id.setValueListItems(dataset);
```