SmartString Library

July 14, 2015

Edited by Robert A. Rioja

Table of Contents

Description

This library is for adjusting text size automatically or manually.

It is not advisable to use this automatic text size library to adjust every text in your activity. This library adjusts text size by performing many calculations. This can cause high overhead of RAM and CPU usage. Two ways around this problem are:

1: Use SameStyle sub to decrease calculations.
2: Use activity “first time” option.

Credits

Written by Armin KH.

Installation

Required Libraries:

- Java object
- StringUtils
- RichString

Version History

Current Version 1.20

SmartString Object

Methods

IsInitialized

Syntax	**IsInitialized As Boolean**
Description	Returns initialization status of view.
Returns	Flag showing if view is initialized.
Parameters	None

Example:

```
Dim SmartString1 as SmartString, B as boolean
SmartString1.Initialize
B = SmartString1.Initialized
```

Ellipsize

Syntax	**Ellipsize (TargetView As Object, EllipSizeMode As String, MarqueeRepeatLimit As Integer)**	
Description	Cut your text with given mode without changing size if the text is longer than maximum size.	
Parameters	TargetView	The view containing the text.
	EllipSizeMode	"Start", "Middle", "End" or "Marquee".
	MarqueeRepeatLimit	*Only used for "Marquee" mode.* Repeat count for marquee. Set to -1 to repeat indefinitely, or to stop marquee.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.Ellipsize(Label1,"Marquee",-1)
```

Initialize

Syntax	**Initialize**
Description	Initializes the SmartString object.
Parameters	None
Returns	None

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
```

MultiLine1

Syntax	**MultiLine1 (Activity As Activity, TargetView As Object, Font As String, BaseTextSize As Double, MaxLineCount As Int, LineSpaceRate As Float)**	
Description	Increases or decreases text size as long as there is space in width and height and number of lines is smaller than the given maximum lines number.	
Parameters	Activity	Activity containing the view.
	TargetView	View containing the text.
	Font	Font name (in Asset) or blank ("") for default font.
	BaseTextSize	Text size in base layout.
	MaxLineCount	Maximum number of lines.
	LineSpaceRate	Space between lines (0 to set current size).
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.MultiLine1(Activity,Label1,"Roboto.ttf",13,4,0.5)
```

MultiLine2

Syntax	**MultiLine2 (Activity As Activity, TargetView As Object, Font As String, BaseTextSize As Double, MaxLineCount As Int, CanBeLarger As Boolean, LineSpaceRate As Float)**	
Description	Increases or decreases text size as long there is space in width and height, not related to number of lines. NOTE: By using the CanBeLarger option, you have more control over the size of text if it's not larger than view spaces. In other words, this option is useful when you want to keep the base text size in larger devices.	
Parameters	Activity	Activity containing the view.
	TargetView	View containing the text.
	Font	Font name (in Asset) or blank ("") for default font.
	BaseTextSize	Text size in base layout.
	MaxLineCount	Maximum number of lines.
	CanBeLarger	Select False if you don't want to fit your text if your text is smaller than maximum.

	LineSpaceRate	Space betwenn lines (0 to set current size).
Returns	None	

Example:
```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.MultiLine2(Activity,Label1,"Roboto.ttf",13,True,0)
```

RichString1

Syntax	**RichString1 (TargetView As Object, Mode As Int, FirstIndex As Int, LastIndex As Int)**	
Description	Set Underscore, Strikethrough, Subscript, Superscript to selected part of the text.	
Parameters	TargetView	View containing the text.
	Mode	1 = Underscore, 2 = Strikethrough, 3 = Subscript, 4 = Superscript.
	FirstIndex	Index of first character to be acted on.
	LastIndex	Index of last character to be acted on.
Returns	None	

Example:
```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.RichString1(Label1,1,0,4)
```

RichString2

Syntax	**RichString2 (TargetView As Object, Mode As Int, RateOfScale As Float, FirstIndex As Int, LastIndex As Int)**	
Description	Set RelativeSize or ScaleX to selected part of the text.	
Parameters	TargetView	View containing the text.
	Mode	1 = RelativeSize, 2 = ScaleX.
	RateOfScale	Scale selected part of the text with this scale rate.
	FirstIndex	Index of first character to be acted on.
	LastIndex	Index of last character to be acted on.
Returns	None	

Example:
```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.RichString2(Label1,1,1.5,0,4)
```

RichString3

Syntax	**RichString3 (TargetView As Object, Mode As Int, Color As Int, FirstIndex As Int, LastIndex As Int)**	
Description	Change text color or background color in selected part of the text.	
Parameters	TargetView	View containing the text.
	Mode	1 = change text color, 2 = change background color.
	Color	A,B,C are RGB color number
	FirstIndex	Index of first character to be acted on.
	LastIndex	Index of last character to be acted on.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.RichString3(Label1,1,Colors.Red,0,4)
```

RichString4

Syntax	**RichString4 (TargetView As Object, StyleOrTypeFamily As String, Mode As Int, FirstIndex As Int, LastIndex As Int)**	
Description	Change style or typeface Underscore, Strikethrough, Subscript, Superscript to selected part of the text.	
Parameters	TargetView	View containing the text.
	Mode	1 = Style, 2 = Typeface.
	StyleOrTypeFamily	Enter style for style mode or TypeFaceFamily for typeface mode. Style can be STYLE_BOLD, STYLE_BOLD_ITALIC, STYLE_ITALIC, STYLE_NORMAL. Typeface examples are monospace, serif, sans-serif.
	FirstIndex	Index of first character to be acted on.
	LastIndex	Index of last character to be acted on.
Returns	None	

Example 1:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.RichString4(Label1,1,"STYLE_BOLD",0,4)
```

Example 2:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.RichString4(Label1,2,"monospace",0,4)
```

SetExactLineNumber

Syntax	**SetExactLineNumber (TargetView As Object, LineCount As Int, EllipsizeModeIfNeeded As String, MarqueeRepeatLimitIfNeeded As Int)**	
Description	Set your text to given line number without changing text size.	
Parameters	TargetView	View containing the text.
	LineCount	Cut your string if be larger than given number of line.
	EllipsizeModeIfNeeded	Ellipsize your text if be longer than maximum size.
	MarqueeRepeatLimitIfNeeded	If you choose marquee mode for ellipsize, then you can limit marquee repeat count.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.Ellipsize(Label1,2,"End",0)
```

SetLineSpace

Syntax	**SetLineSpace (TargetView As Object, Multiplier As Float)**	
Description	Change the spaces between lines.	
Parameters	TargetView	View containing the text.
	Multiplier	Multiplier factor of spacing.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.SetLineSpace(Label1,1.5)
```

SetManualSize

Syntax	**SetManualSize (TargetView As Object, BaseLayoutWidth As Double, BaseLayoutHeight As Double, BaseLayoutScale As Double, Font As String, BaseTextSize As Double, RateOfScale As Double)**	
Description	Scale text by given rate. You can give a scale rate for base layout and then the text size will be larger in large devices and smaller in small devices with the given scale rate. NOTE: Does not preserve aspect ratio, and not dependent on available space in width or height.	
Parameters	TargetView	View containing the text.

	BaseLayoutWidth	Base layout density (dot per inch) / 160. For example: if base layout is 320dpi, then scale will be 320 / 160 = 2.
	BaseLayoutHeight	.
	BaseLayoutScale	.
	Font	Font name (in Asset) or leave this option blank and default font will be applied.
	BaseTextSize	Text size in base layout.
	RateOfScale	Text will be scaled by this rate. Enter 0 if you don't want to change the text size in any device.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.SetManualSize(Label1,320,480,1,"Roboto.ttf",13,0.3)
```

SetPadding

Syntax	**SetPadding (TargetView As Object, Left As Int, Top As Int, Right As Int, Bottom As Int)**	
Description	Set text space padding (left, top, right, bottom) of the view.	
Parameters	TargetView	View containing the text.
	Left	Left padding.
	Top	Top padding.
	Right	Right padding.
	Bottom	Bottom padding.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.Padding(Label1,0,0,0,0)
```

SetSameStyle

Syntax	**SetSameStyle (BaseView As Object, TargetView As Object)**	
Description	Set text size and font of a view to that of another view with same height and width. This option will decrease CPU, RAM and battery usage by reducing calculations.	
Parameters	BaseView	View containing the original style.
	TargetView	View to receive style.

Returns	None

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.SingleLine2(Label1,"Roboto.ttf)
SmartString1.SetSameStyle(Label1,Label2)
```

SingleLine1

Syntax	**SingleLine1 (Activity As Activity, TargetView As Object, ViewWidth As Int, Font As String, BaseTextSize As Double)**	
Description	Tries to increase or decrease the single line text size and preserve the aspect ratio as long as the height of the text is not greater than the height of the view. In other words, you can increase text size while there is space in width and height, The text will always be on single line and the aspect ratio is always preserved. Accuracy is acceptable but not 100%. The text size can not be changed with every 1dip. Text in base layout should not be larger than 1 line.	
Parameters	Activity	Activity containing the view.
	TargetView	View containing the text.
	ViewWidth	Width of the view in base layout.
	Font	Font name (in Asset) or blank ("") for default font.
	BaseTextSize	Text size in base layout. Text in the base layout should not be longer than one line.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.SingleLine1(Activity,TargetView,Label1,150dip,"Roboto.ttf",13)
```

SingleLine2

Syntax	**SingleLine2 (Activity As Activity, TargetView As Object, Font As String)**	
Description	Tries to fit your text in view as long as the height of the text Is not greater than the height of the view. It will always be only one line. This sub is similar to SingleLine1 with two differences: 1: Aspect ratio is not preserved and text can fit in the view. 2: Text in base layout can be longer than one line, but after calling SingleLine2 it will be 1 line.	
Parameters	Activity	Activity containing the view.
	TargetView	View containing the text.

	Font	Font name (in Asset) or blank ("") for default font.
Returns	None	

Example:

```
Dim SmartString1 as SmartString
SmartString1.Initialize
SmartString1.SingleLine2(Activity,Label1,"Roboto.ttf")
```