KJAVA API

1.0

Note:

The classes provided in package `com.sun.kjava` are not part of the CLDC reference implementation. These classes have been provided to facilitate porting and testing efforts, and may change or may be removed in future releases of the CLDC/KVM software.

Contents

Package
com.sun.kjava

Description
The test GUI classes for KVM.

Class Summary	
Interfaces	
DialogOwner	A simple interface to be used by anything wishing to display a modal dialog.
ScrollOwner	Interface between something that scrolls and something that cares about that something that scrolls.
Classes	
Bitmap	An object of this class represents a black and white bitmap.
Button	Button: a simple button user interface object.
Caret	Class Caret implements a caret ("|") for use as a marker for the current insertion point in a TextField.
CheckBox	A checkbox user interface object.
Database	This class serves as an interface to the PalmOS database manager.
Dialog	A pop-up modal dialog that displays a title string, text box full of text, and a dismiss button.
Graphics	This class contains various methods for drawing on a display.
HelpDisplay	A simple, prepackaged "help" text user interface object.
IntVector	A simple expandable vector of integers, similar to `java.util.Vector`.
List	A class representing a list of Objects.
RadioButton	A two-state button meant as part of a group, only one of which can be "on" at one time.
RadioGroup	An object representing a group of RadioButtons.
ScrollTextArea	
ScrollTextBox	A scrolling TextBox object.
SelectScrollTextBox	
Slider	Slider: A graphical valuator object.
Spotlet	This class provides callbacks for event handling.
TextBox	A box displaying text on the screen.
TextField	This class provides a simple TextField.
VerticalScrollBar	A vertical scroll bar user interface object.

com.sun.kjava
Bitmap

Syntax
```
public class Bitmap

java.lang.Object
  |
  +--com.sun.kjava.Bitmap
```

Description
An object of this class represents a black and white bitmap.

Member Summary	
Constructors	
`Bitmap(short[])`	Constructor to create a bitmap.
`Bitmap(short, byte[])`	Constructor defines the bitmap.
`Bitmap(String, int)`	Constructor defines the bitmap.
Methods	
`getRows()`	Return the number of rows in the bitmap.
`getWidth()`	Return the width of the space in pixels used to display the bitmap.

Inherited Member Summary
Methods inherited from class java.lang.Object
`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Constructors

Bitmap(short[])

```
public Bitmap(short[] data)
```

Constructor to create a bitmap. The array is the exact representation of a bitmap in the Palm OS including the headers and flags.

Parameters:
`data` - The Palm OS representation of a bitmap.

Bitmap(short, byte[])

```
public Bitmap(short width, byte[] pixels)
```

Constructor defines the bitmap. The bits of a bitmap are given as an array of bytes, each byte defining 8 bits of the bitmap.

On the Palm OS, the width (in bytes) must be even. If a bitmap is constructed with an odd width, padding is automatically added. It is padded width that is given by a call to getWidth. The maximum width for a bitmap on this platform is currently 32.

Parameters:

`width` - the width of the bitmap in bytes.

`pixels` - the bits of the object.

Bitmap(String, int)

```
public Bitmap(java.lang.String pattern, int scanline)
```

Constructor defines the bitmap. The bits of a bitmap are given as a string pattern, each character define the bitmap.

Create an Image object from the bitmap specified in 'pattern'. Pixels may be spec:ified as transparent, or opaque, using the character '_' for transparency, and any other character for opaqueness. Scanline must be a multiple of 8.

Parameters:

`pattern` - the string pattern for the bitmap.

`scanline` - the scanline for the pattern. Must be a multiple of 8.

Methods

---

getRows()

```
public int getRows()
```

Return the number of rows in the bitmap.

Returns: the number of rows in the bitmap

---

getWidth()

```
public int getWidth()
```

Return the width of the space in pixels used to display the bitmap. This will be a multiple of 16 and so may not correspond with the width specified when constructing the bitmap.

Returns: the width of the space in pixels used to display the bitmap.

com.sun.kjava
Button

Syntax

```
public class Button

java.lang.Object
  |
  +--com.sun.kjava.Button
```

Description

Button: a simple button user interface object. Note that this button causes actions to occur when it is pressed, not when it is released. Therefore it is currently impossible for a user to cancel a button selection once it has started! Bitmap buttons do not have a border drawn around them. If you want your bitmap button to have a border, include the border in the bitmap.

Member Summary

Fields

`minWidth`	

Constructors

`Button(Bitmap, int, int)`	Create a new Button object with graphical label.
`Button(String, int, int)`	Create a new Button object with a text label.

Methods

`isEnabled()`	Is the Button enabled?
`paint()`	Paint the Button on the global Graphics context.
`pressed(int, int)`	Was the button pressed? If the coordinates are within the Button, give the user some feedback.
`setEnabled(boolean)`	Set whether the Button allows input (is "enabled").
`setText(String)`	Set the Button's text label.

Inherited Member Summary

Methods inherited from class java.lang.Object

`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Fields

minWidth

```
public static final int minWidth
```

Constructors

Button(Bitmap, int, int)

```
public Button(Bitmap bitmap, int x, int y)
```

Create a new Button object with graphical label.

Parameters:

s - the button's text label

x - the x coordinate of the button's location

y - the y coordinate of the button's location

Button(String, int, int)

```
public Button(java.lang.String s, int x, int y)
```

Create a new Button object with a text label.

Parameters:

s - the button's text label

x - the x coordinate of the button's location

y - the y coordinate of the button's location

Methods

isEnabled()

```
public boolean isEnabled()
```

Is the Button enabled?

Returns: true if the Button accepts input, false if not.

paint()

```
public void paint()
```

Paint the Button on the global Graphics context. If the Button is not enabled, it draws in a "grayed out" style.

pressed(int, int)

```
public boolean pressed(int x, int y)
```

Was the button pressed? If the coordinates are within the Button, give the user some feedback.

Returns: true if the coordinates were within the bounds of the Button.

---

setEnabled(boolean)

```
public void setEnabled(boolean state)
```

Set whether the Button allows input (is "enabled").

Parameters:

`state` - if true, Button allows input.

---

setText(String)

```
public void setText(java.lang.String s)
```

Set the Button's text label.

Parameters:

`s` - the new label for the button.

com.sun.kjava
Caret

Syntax

```
public class Caret extends java.lang.Thread
```

```
java.lang.Object
  |
  +--java.lang.Thread
        |
        +--com.sun.kjava.Caret
```

All Implemented Interfaces: java.lang.Runnable

Description

Class Caret implements a caret ("|") for use as a marker for the current insertion point in a TextField. (Caret should probably be a private class, since it has no use independent of TextField.)

Member Summary

Fields

`blinking`

`stop`

Constructors

`Caret(int, int, int)`	Create a Caret at a position, blinking at a given rate.

Methods

`drawCaret(int)`	Draw the Caret at its current position.
`eraseCaret()`	
`run()`	Run: flash the Caret at the prescribed rate.
`setPosition(int, int)`	Set the Caret's position.

Inherited Member Summary

Fields inherited from class java.lang.Thread

`MIN_PRIORITY, NORM_PRIORITY, MAX_PRIORITY`

Methods inherited from class java.lang.Thread

`currentThread, yield, sleep, start, isAlive, setPriority, getPriority, activeCount, join, toString`

Methods inherited from class java.lang.Object

`getClass, hashCode, equals, notify, notifyAll, wait, wait, wait`

blinking

Fields

blinking

```
public boolean blinking
```

stop

```
public boolean stop
```

Constructors

Caret(int, int, int)

```
public Caret(int delay, int x, int y)
```

Create a Caret at a position, blinking at a given rate.

Parameters:

`x` - X coordinate of position

`y` - Y coordinate of position

`delay` - delay between blinks, in milliseconds

Methods

drawCaret(int)

```
public void drawCaret(int drawMode)
```

Draw the Caret at its current position.

Parameters:

`drawMode` - mode in which to draw

eraseCaret()

```
public void eraseCaret()
```

run()

```
public void run()
```

Run: flash the Caret at the prescribed rate.

Overrides: java.lang.Thread.run() in class java.lang.Thread

setPosition(int, int)

```
public void setPosition(int x, int y)
```

Set the Caret's position.

Parameters:

x - new X coordinate

y - new Y coordinate

com.sun.kjava
CheckBox

Syntax
```
public class CheckBox

java.lang.Object
  |
  +--com.sun.kjava.CheckBox
```

Description
A checkbox user interface object. A CheckBox object displays a check box next to a text label. It has two states, checked and unchecked.

Member Summary	
Constructors	
CheckBox(int, int, String)	Create a new checkbox at a given position with a text label.
Methods	
handlePenDown(int, int)	The user selected the CheckBox; invert its state.
paint()	Paint the CheckBox.
pressed(int, int)	Did the user's "press" fall within the CheckBox?
setLocation(int, int)	Set the CheckBox's position.
setState(boolean)	Set the state and redraw to reflect it.
setText(String)	Set the CheckBox's label.

Inherited Member Summary
Methods inherited from class java.lang.Object
getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Constructors

CheckBox(int, int, String)

```
public CheckBox(int x, int y, java.lang.String text)
```

Create a new checkbox at a given position with a text label.

Parameters:

`x` - the X coordinate of position.

`y` - the Y coordinate of position.

`text` - label of the CheckBox

Methods

---

handlePenDown(int, int)

```
public void handlePenDown(int x, int y)
```

The user selected the CheckBox; invert its state. If it was checked, set the state to unchecked, and *vice-versa*. This will cause the CheckBox to redraw itself.

---

paint()

```
public void paint()
```

Paint the CheckBox.

---

pressed(int, int)

```
public boolean pressed(int x, int y)
```

Did the user's "press" fall within the CheckBox?

Parameters:

`x` - the X coordinate of the user's press

`y` - the Y coordinate of the user's press

Returns: true if `(x, y)` fall within bounds

---

setLocation(int, int)

```
public void setLocation(int x, int y)
```

Set the CheckBox's position.

Parameters:

`x` - the X coordinate of position.

`y` - the Y coordinate of position.

---

setState(boolean)

```
public void setState(boolean state)
```

Set the state and redraw to reflect it.

Parameters:

`state` - the new state

setText(String)

```
public void setText(java.lang.String text)
```

Set the CheckBox's label.

com.sun.kjava
Database

Syntax

```
public class Database
```

```
java.lang.Object
  |
  +--com.sun.kjava.Database
```

Description

This class serves as an interface to the PalmOS database manager. It allows the user to create and access PalmOS databases from KJava.

Member Summary

Fields

ENDOFDATABASE	End of database (last record indicator).
READONLY	Read-only mode.
READWRITE	Read and write mode.
WRITEONLY	Write-only mode.

Constructors

Database(int, int, int)	Open a database.

Methods

addRecord(byte[])	Add a new record to the end of the database.
close()	Close the current database.
create(int, String, int, int, boolean)	Create a new database.
deleteRecord(int)	Delete an existing record.
getNumberOfRecords()	Get the number of records in the database.
getRecord(int)	Read a database record into a Java byte array object.
isOpen()	Check if the database is open.
readRecordTo-Buffer(int, int, int, byte[], int)	Read record to a pre-allocated buffer instead of allocating a new bytearray each time.
setRecord(int, byte[])	Set the contents of a PalmOS database record.
writeRecordFrom-Buffer(int, int, int, byte[], int)	Set the contents of a database record.

Inherited Member Summary

Methods inherited from class java.lang.Object

Inherited Member Summary
`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Fields

ENDOFDATABASE

`public static final int ENDOFDATABASE`

End of database (last record indicator).

READONLY

`public static final int READONLY`

Read-only mode.

READWRITE

`public static final int READWRITE`

Read and write mode.

WRITEONLY

`public static final int WRITEONLY`

Write-only mode.

Constructors

Database(int, int, int)

`public Database(int typeID, int creatorID, int mode)`

Open a database.

Methods

addRecord(byte[])

`public boolean addRecord(byte[] data)`

Add a new record to the end of the database.

---

close()

```
public native void close()
```

Close the current database.

---

create(int, String, int, int, boolean)

```
public static native boolean create(int cardNo, java.lang.String name, int creatorID,
            int typeID, boolean resDB)
```

Create a new database.

---

deleteRecord(int)

```
public native boolean deleteRecord(int recordNumber)
```

Delete an existing record.

---

getNumberOfRecords()

```
public native int getNumberOfRecords()
```

Get the number of records in the database.

---

getRecord(int)

```
public native byte[] getRecord(int recordNumber)
```

Read a database record into a Java byte array object. Remember that PalmOS database record numbers start from 0.

---

isOpen()

```
public boolean isOpen()
```

Check if the database is open.

---

readRecordToBuffer(int, int, int, byte[], int)

```
public native int readRecordToBuffer(int recordNumber, int readOffset, int length,
            byte[] buffer, int writeOffset)
```

Read record to a pre-allocated buffer instead of allocating a new bytearray each time. Also allow a record to be read partially if necessary. Currently unimplemented.

---

setRecord(int, byte[])

```
public native boolean setRecord(int recordNumber, byte[] data)
```

Set the contents of a PalmOS database record.

writeRecordFromBuffer(int, int, int, byte[], int)

```
public native int writeRecordFromBuffer(int recordNumber, int writeOffset, int length,
            byte[] buffer, int readOffset)
```

Set the contents of a database record. Allows more complex data manipulation than setRecord. Currently unimplemented.

com.sun.kjava
Dialog

Syntax

```
public class Dialog extends Spotlet
```

```
java.lang.Object
  |
  +--Spotlet
        |
        +--com.sun.kjava.Dialog
```

Description

A pop-up modal dialog that displays a title string, text box full of text, and a dismiss button.

Member Summary

Fields

`button`
`g`
`haveScroll`
`height`
`owner`
`tb`
`text`
`title`
`width`
`x`
`y`

Constructors

`Dialog(DialogOwner, String, String, String)`	Create a new Dialog of a fixed size.
`Dialog(int, int, int, int, DialogOwner, String, String, String)`	Create a new Dialog with a arbitrary size.

Methods

`dismissDialog()`	Dismiss the Dialog.
`keyDown(int)`	If we have a ScrollTextBox, then allow scrolling.
`paint()`	Paint the Dialog.
`penDown(int, int)`	If the user pressed the dismiss button, dismiss the Dialog.
`penMove(int, int)`	If we have a ScrollTextBox, then allow scrolling.
`showDialog()`	Show the Dialog: register it and paint it.

Inherited Member Summary

Fields inherited from class `Spotlet`

PAGEUP, PAGEDOWN, KEY_HARD1, KEY_HARD2, KEY_HARD3, KEY_HARD4, KEY_POWER, CALCICON, MENUICON, NO_EVENT_OPTIONS, WANT_SYSTEM_KEYS

Methods inherited from class `Spotlet`

dispatch(int, DataInput), unknownEvent(int, DataInput), register(int), setPalmEvenTOptions(int), unregister(), penUp(int, int), beamReceive(byte[]), beamSend(byte[]), getFlashID()

Methods inherited from class java.lang.Object

getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Fields

button

```
protected Button button
```

g

```
protected Graphics g
```

haveScroll

```
protected boolean haveScroll
```

height

```
protected int height
```

owner

```
protected DialogOwner owner
```

tb

```
protected TextBox tb
```

text

```
protected java.lang.String text
```

title

```
protected java.lang.String title
```

---

width

```
protected int width
```

---

x

```
protected int x
```

---

y

```
protected int y
```

Constructors

Dialog(DialogOwner, String, String, String)

```
public Dialog(DialogOwner o, java.lang.String t, java.lang.String str,
              java.lang.String buttonText)
```

Create a new Dialog of a fixed size. Creates a TextBox 140x120 at position 10,10. The contents of the box is passed in the str parameter. A button is created which allows for dismissal of the Dialog. The text for the button is passed in buttonText. If the text overflows the text box, a ScrollTextBox is used to display it. The owner of the Dialog gets called through the DialogOwner interface dialogDismissed() method when the dialog is dismissed. The owner must then re-register the Spotlet that was running when the Dialog was created. It must also re-paint the screen as appropriate.

Parameters:

`t` - the title of this Dialog - used when the Dialog is dismissed

`str` - the contents of the TextBox

`buttonText` - the label of the button

Dialog(int, int, int, int, DialogOwner, String, String, String)

```
public Dialog(int x, int y, int width, int height, DialogOwner o, java.lang.String t,
              java.lang.String str, java.lang.String buttonText)
```

Create a new Dialog with a arbitrary size. Note: MAXSIZE is 160 X 140 The contents of the box is passed in the str parameter. A button is created which allows for dismissal of the Dialog. The text for the button is passed in buttonText. If the text overflows the text box, a ScrollTextBox is used to display it. The owner of the Dialog gets called through the DialogOwner interface dialogDismissed() method when the dialog is dismissed. The owner must then re-register the Spotlet that was running when the Dialog was created. It must also re-paint the screen as appropriate.

Parameters:

`x` - the x location of this Dialog

`y` - the y location of this Dialog

`width` - the width of this Dialog

`height` - the height of this Dialog

`t` - the title of this Dialog - used when the Dialog is dismissed

`str` - the contents of the TextBox

`buttonText` - the label of the button

Methods

---

dismissDialog()

```
public void dismissDialog()
```

Dismiss the Dialog. Unregister it and alert the owner.

---

keyDown(int)

```
public void keyDown(int key)
```

If we have a ScrollTextBox, then allow scrolling.

Overrides: `keyDown(int)` in class `Spotlet`

Parameters:

`key` - the key pressed/entered by the user

---

paint()

```
public void paint()
```

Paint the Dialog.

---

penDown(int, int)

```
public void penDown(int x, int y)
```

If the user pressed the dismiss button, dismiss the Dialog. If we have a ScrollTextBox, then allow scrolling.

Overrides: `penDown(int, int)` in class `Spotlet`

Parameters:

`x` - the X coordinate of the user's press.

`y` - the Y coordinate of the user's press.

---

penMove(int, int)

```
public void penMove(int x, int y)
```

If we have a ScrollTextBox, then allow scrolling.

Overrides: `penMove(int, int)` in class `Spotlet`

Parameters:

x - the X coordinate of the user's press.

y - the Y coordinate of the user's press.

---

showDialog()

```
public void showDialog()
```

Show the Dialog: register it and paint it.

com.sun.kjava
DialogOwner

Syntax
```
public abstract interface DialogOwner
```

Description
A simple interface to be used by anything wishing to display a modal dialog.

See Also: `Dialog`

Member Summary	
Methods	
`dialogDis-missed(String)`	The Dialog with title `title` has been dismissed.

Methods

---

dialogDismissed(String)

```
public void dialogDismissed(java.lang.String title)
```

The Dialog with title `title` has been dismissed.

Parameters:

`title` - title of the Dialog that was dismissed.

com.sun.kjava
Graphics

Syntax

```
public class Graphics

java.lang.Object
  |
  +--com.sun.kjava.Graphics
```

Description

This class contains various methods for drawing on a display. The coordinate system is such that the points along horizontal axis increase in value from left to right, and points along the vertical axis increase in value from top to bottom.

Member Summary

Fields

AND	Region copy mode: The copied region is AND'ed with the destination.
AND_NOT	Region copy mode: The copied region is AND'ed with the inverted destination region.
ERASE	Erase mode.
GRAY	Gray drawing mode.
INVERT	Invert mode.
NOT	Region copy mode: The copied region is inverted and overwrites the destination.
OFFSCREEN_WINDOW	
ONSCREEN_WINDOW	
OR	Region copy mode: The copied region is OR'ed with the destination.
OVERWRITE	Region copy mode: The copied region overwrites the destination.
PLAIN	Plain drawing mode.
RAISED	Constant for a slightly raised border.
SIMPLE	Constant for a plain rectangle border.
SOUND_ALARM	System sound for the alarm.
SOUND_CLICK	System sound for a click.
SOUND_CONFIRMATION	System sound for confirmation.
SOUND_ERROR	System sound for error.
SOUND_INFO	System sound for info.
SOUND_STARTUP	System sound for startup.
SOUND_WARNING	System sound for warning.
XOR	Region copy mode: The copied region is XOR'ed with the destination.

Methods

borderType(int, int, int)	Constructs a border type.
clearScreen()	Clear the screen.
copyOffScreenRegion(int, int, int, int, int, int, int, int)	Copy a rectangular region from one place to another, possibly in different windows.

Member Summary	
copyRegion(int, int, int, int, int, int, int)	Copy a rectangular region from one place to another.
drawBitmap(int, int, Bitmap)	Draw a bitmap.
drawBorder(int, int, int, int, int, int)	Draw a rectangular border.
drawLine(int, int, int, int, int)	Draw a line.
drawRectangle(int, int, int, int, int, int)	Draw a solid rectangle.
drawString(String, int, int)	Draw a string at a given position.
drawString(String, int, int, int)	Draw a string at a given position.
getGraphics()	There is only ever one Graphics object in the system, and this function returns it.
getHeight(String)	Returns the height of a string in pixels.
getWidth(String)	Returns the width of a string in pixels.
playSMF(byte[])	Plays a MIDI file.
playSound(int)	Play a system sound.
playSoundHz(int, int, int)	Play a sound.
resetDrawRegion()	Reset the region in which drawing can be performed to be the whole screen.
setDrawRegion(int, int, int, int)	Set the region in which drawing can be performed.

Inherited Member Summary
Methods inherited from class java.lang.Object
`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Fields

AND

```
public static final int AND
```

Region copy mode: The copied region is AND'ed with the destination.

AND_NOT

```
public static final int AND_NOT
```

Region copy mode: The copied region is AND'ed with the inverted destination region.

ERASE

```
public static final int ERASE
```

Erase mode.

GRAY

```
public static final int GRAY
```

Gray drawing mode.

INVERT

```
public static final int INVERT
```

Invert mode.

NOT

```
public static final int NOT
```

Region copy mode: The copied region is inverted and overwrites the destination.

OFFSCREEN_WINDOW

```
public static final int OFFSCREEN_WINDOW
```

ONSCREEN_WINDOW

```
public static final int ONSCREEN_WINDOW
```

OR

```
public static final int OR
```

Region copy mode: The copied region is OR'ed with the destination.

OVERWRITE

```
public static final int OVERWRITE
```

Region copy mode: The copied region overwrites the destination.

PLAIN

```
public static final int PLAIN
```

Plain drawing mode.

RAISED

`public static final int RAISED`

Constant for a slightly raised border.

SIMPLE

`public static final int SIMPLE`

Constant for a plain rectangle border.

SOUND_ALARM

`public static final int SOUND_ALARM`

System sound for the alarm.

SOUND_CLICK

`public static final int SOUND_CLICK`

System sound for a click.

SOUND_CONFIRMATION

`public static final int SOUND_CONFIRMATION`

System sound for confirmation.

SOUND_ERROR

`public static final int SOUND_ERROR`

System sound for error.

SOUND_INFO

`public static final int SOUND_INFO`

System sound for info.

SOUND_STARTUP

`public static final int SOUND_STARTUP`

System sound for startup.

SOUND_WARNING

`public static final int SOUND_WARNING`

System sound for warning.

XOR

```
public static final int XOR
```

Region copy mode: The copied region is XOR'ed with the destination.

Methods

borderType(int, int, int)

```
public static int borderType(int cornerDiam, int shadow, int width)
```

Constructs a border type.

Parameters:

`cornerDiam` - the diameter of four imaginary circles used to form rounded corners. Must be in the range 0..38.

`shadow` - the width of a shadow. Must be in the range 0..3.

`width` - width of the border.Must be in the range 0..3.

Returns: a value representing the specified type

clearScreen()

```
public static void clearScreen()
```

Clear the screen.

copyOffScreenRegion(int, int, int, int, int, int, int, int, int)

```
public static native void copyOffScreenRegion(int left, int top, int width, int height,
            int dstX, int dstY, int mode, int srcWind, int dstWind)
```

Copy a rectangular region from one place to another, possibly in different windows. There is the usual ONSCREEN_WINDOW and a hidden OFFSCREEN_WINDOW of the same size. The OFFSCREEN_WINDOW is handy for storing bitmaps in game programs.

Parameters:

`left` - the x coordinate of the source region's top left corner

`top` - the y coordinate of the source region's top left corner

`width` - the width of the source region

`height` - the height of the source region

`dstX` - the x coordinate of the point to which the region should be copied in the destination

`dstY` - the y coordinate of the point to which the region should be copied in the destination

`mode` - the copy mode (one of `OVERWRITE, AND, AND_NOT, XOR, OR, INVERT`)

`srcWind` - either ONSCREEN_WINDOW or OFFSCREEN_WINDOW

`dstWind` - either ONSCREEN_WINDOW or OFFSCREEN_WINDOW

---

copyRegion(int, int, int, int, int, int, int)

```
public static native void copyRegion(int left, int top, int width, int height, int dstX,
            int dstY, int mode)
```

Copy a rectangular region from one place to another.

Parameters:

`left` - the x coordinate of the region's top left corner

`top` - the y coordinate of the region's top left corner

`width` - the width of the region

`height` - the height of the region

`dstX` - the x coordinate of the point to which the region should be copied

`dstY` - the y coordinate of the point to which the region should be copied

`mode` - the copy mode (one of `OVERWRITE, AND, AND_NOT, XOR, OR, INVERT`)

---

drawBitmap(int, int, Bitmap)

```
public static native void drawBitmap(int left, int top, Bitmap bitmap)
```

Draw a bitmap.

Parameters:

`left` - the x coordinate of the bitmap's top left corner

`top` - the y coordinate of the bitmap's top left corner

`bitmap` - the bitmap to be drawn

---

drawBorder(int, int, int, int, int, int)

```
public static native void drawBorder(int left, int top, int width, int height, int mode,
            int frameType)
```

Draw a rectangular border. The border is drawn around the rectangle specified by the given dimensions.

Parameters:

`left` - the x coordinate of the rectangle's top left corner

`top` - the y coordinate of the rectangle's top left corner

`width` - the width of the rectangle

`height` - the height of the rectangle

`mode` - the drawing mode to use (one of `PLAIN, GRAY, ERASE` or `INSERT`.

`frameType` - one of `SIMPLE`, `RAISED` or a type constructed by a call to `borderType`.

---

drawLine(int, int, int, int, int)

```
public static native void drawLine(int srcX, int srcY, int dstX, int dstY, int mode)
```

Draw a line.

Parameters:

`srcX` - the X coordinate of the starting point

`srcY` - the Y coordinate of the starting point

`dstX` - the X coordinate of the destination point

`dstY` - the Y coordinate of the destination point

`mode` - the drawing mode to use (one of `PLAIN, GRAY, ERASE` or `INSERT`.

---

drawRectangle(int, int, int, int, int, int)

```
public static native void drawRectangle(int left, int top, int width, int height,
            int mode, int cornerDiam)
```

Draw a solid rectangle.

Parameters:

`left` - the x coordinate of the rectangle's top left corner

`top` - the y coordinate of the rectangle's top left corner

`width` - the width of the rectangle

`height` - the height of the rectangle

`mode` - the drawing mode to use (one of `PLAIN, GRAY, ERASE` or `INSERT`.

`cornerDiam` - the diameter of four imaginary circles used to form the rounded corners. An imaginary circle is placed within each corner tangent to the rectangle on two sides.

---

drawString(String, int, int)

```
public static int drawString(java.lang.String text, int left, int top)
```

Draw a string at a given position. This method is equivalent to `drawString(text, left, top, PLAIN)`.

Parameters:

`text` - the String to draw

`left` - the x coordinate of the top left bound of first character.

`top` - the y coordinate of the top left bound of first character.

Returns: the x coordinate of the right bound of last character drawn

---

drawString(String, int, int, int)

```
public static native int drawString(java.lang.String text, int left, int top, int mode)
```

Draw a string at a given position. Will draw `"null"` if `text` is `null`.

Parameters:

`text` - the String to draw

`left` - the x coordinate of the top left bound of first character.

`top` - the y coordinate of the top left bound of first character.

`mode` - the drawing mode to use (one of `PLAIN, RAY, ERASE` or `INVERT`.

Returns: right bound of last character drawn

---

getGraphics()

```
public static Graphics getGraphics()
```

There is only ever one Graphics object in the system, and this function returns it.

Returns: the single global Graphics context.

---

getHeight(String)

```
public static native int getHeight(java.lang.String s)
```

Returns the height of a string in pixels.

Parameters:
`s` - the String to measure

Returns: the height of the given String in pixels

---

getWidth(String)

```
public static native int getWidth(java.lang.String s)
```

Returns the width of a string in pixels.

Parameters:
`s` - the String to measure

Returns: the width of the given String in pixels

---

playSMF(byte[])

```
public static native void playSMF(byte[] midifile)
```

Plays a MIDI file.

Parameters:
`midifile` - is byte sequence of MIDI data

---

playSound(int)

```
public static native void playSound(int sound)
```

Play a system sound.

Parameters:
`sound` - one of the SOUND_xxx constants

---

playSoundHz(int, int, int)

```
public static native void playSoundHz(int freq, int duration, int volume)
```

Play a sound.

Parameters:
`freq` - is a frequency in Hz

`duration` - is a duration in millisecs

`volume` - is a volume in 0..127

---

resetDrawRegion()

```
public static native void resetDrawRegion()
```

Reset the region in which drawing can be performed to be the whole screen.

---

setDrawRegion(int, int, int, int)

```
public static native void setDrawRegion(int left, int top, int width, int height)
```

Set the region in which drawing can be performed. If the specified region is `null` then the region is set to be the entire window.

Parameters:

`left` - the x coordinate of the top left position of the region

`left` - the y coordinate of the top left position of the region

`width` - the width of the region

`height` - the height of the region

com.sun.kjava
HelpDisplay

Syntax
```
public class HelpDisplay extends Spotlet
```

```
java.lang.Object
  |
  +--Spotlet
        |
        +--com.sun.kjava.HelpDisplay
```

Description
A simple, prepackaged "help" text user interface object.

Member Summary	
Constructors	
HelpDisplay(String, String, int)	Create a new HelpDisplay.
Methods	
keyDown(int)	The user has pressed a key.
penDown(int, int)	The pen has gone down.
penMove(int, int)	The pen moved.

Inherited Member Summary
Fields inherited from class Spotlet
PAGEUP, PAGEDOWN, KEY_HARD1, KEY_HARD2, KEY_HARD3, KEY_HARD4, KEY_POWER, CALCICON, MENUICON, NO_EVENT_OPTIONS, WANT_SYSTEM_KEYS
Methods inherited from class Spotlet
dispatch(int, DataInput), unknownEvent(int, DataInput), register(int), setPalmEvenTOptions(int), unregister(), penUp(int, int), beamReceive(byte[]), beamSend(byte[]), getFlashID()
Methods inherited from class java.lang.Object
getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Constructors

HelpDisplay(String, String, int)

```
public HelpDisplay(java.lang.String hText, java.lang.String className, int eventOptions)
```

Create a new HelpDisplay.

Parameters:

`hText` - the text that's going to help the user

`className` - the exact name of the class to create and run

`eventOptions` - the event options we're interested in

Methods

keyDown(int)

```
public void keyDown(int keyCode)
```

The user has pressed a key.

Overrides: `keyDown(int)` in class `Spotlet`

penDown(int, int)

```
public void penDown(int x, int y)
```

The pen has gone down. If the user pressed the "done" button, create and register the application named by `className`.

Overrides: `penDown(int, int)` in class `Spotlet`

penMove(int, int)

```
public void penMove(int x, int y)
```

The pen moved.

Overrides: `penMove(int, int)` in class `Spotlet`

com.sun.kjava
IntVector

Syntax
```
public class IntVector
```

```
java.lang.Object
  |
  +--com.sun.kjava.IntVector
```

Description
A simple expandable vector of integers, similar to `java.util.Vector`.

Member Summary	
Constructors	
`IntVector()`	Create a new IntVector, and make it small to start.
`IntVector(int)`	Create a new IntVector.
Methods	
`append(int)`	Append an integer to the end, expanding the vector if necessary.
`capacity()`	What is the total capacity of this IntVector?
`ensureCapacity(int)`	Ensure there's room for some number of entries by any means necessary.
`removeAllElements()`	Mark the vector as containing no integers.
`size()`	What is the size of this IntVector?
`valueAt(int)`	What is the value at a given index? N.B.

Inherited Member Summary
Methods inherited from class java.lang.Object
`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Constructors

IntVector()
```
public IntVector()
```
Create a new IntVector, and make it small to start.

IntVector(int)

```
public IntVector(int initSize)
```

Create a new IntVector.

Parameters:
`initSize` - the number of initial elements to allocate

Methods

---

append(int)

```
public void append(int i)
```

Append an integer to the end, expanding the vector if necessary.

Parameters:
`i` - the value of the new datum

---

capacity()

```
public int capacity()
```

What is the total capacity of this IntVector?

Returns: the number of entries currently allocated space, not all of which may be occupied.

See Also: `size()`

---

ensureCapacity(int)

```
public void ensureCapacity(int newCap)
```

Ensure there's room for some number of entries by any means necessary.

Parameters:
`newCap` - the desired new capacity

---

removeAllElements()

```
public void removeAllElements()
```

Mark the vector as containing no integers.

---

size()

```
public int size()
```

What is the size of this IntVector?

Returns: the number of integers stored

---

valueAt(int)

valueAt(int)

```
public int valueAt(int i)
```

What is the value at a given index? N.B. This does no bounds checking.

Parameters:

`i` - the index of the entry

Returns: the integer at that index.

com.sun.kjava
List

Syntax
```
public class List
```

```
java.lang.Object
  |
  +--com.sun.kjava.List
```

Description
A class representing a list of Objects. Resembles `java.util.Vector`.

Member Summary	
Constructors	
List()	Create a new List, and make it small to start.
List(int)	Create a new List.
Methods	
append(Object)	Append an Object to the end, expanding the vector if necessary.
capacity()	Return the total capacity of this List.
elementAt(int)	Return the Object at a given index.
ensureCapacity(int)	Ensure there's room for some number of entries by any means necessary.
removeAllElements()	Mark the vector as containing no Objects, and drop all references to the Objects previously contained.
setElementAt(Object, int)	Set the indexed element to an Object.
size()	Return the size of this List.

Inherited Member Summary
Methods inherited from class java.lang.Object
`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Constructors

List()
```
public List()
```
Create a new List, and make it small to start.

List(int)

```
public List(int initSize)
```

Create a new List.

Parameters:
`initSize` - the number of initial elements to allocate

Methods

append(Object)

```
public void append(java.lang.Object obj)
```

Append an Object to the end, expanding the vector if necessary.

Parameters:
`i` - the value of the new datum

capacity()

```
public int capacity()
```

Return the total capacity of this List.

Returns: the number of entries currently allocated space, not all of which may be occupied.

See Also: `size()`

elementAt(int)

```
public java.lang.Object elementAt(int i)
```

Return the Object at a given index. N.B. This does no bounds checking.

Parameters:
`i` - the index of the entry

Returns: the Object at that index.

ensureCapacity(int)

```
public void ensureCapacity(int newCap)
```

Ensure there's room for some number of entries by any means necessary.

Parameters:
`newCap` - the desired new capacity

removeAllElements()

```
public void removeAllElements()
```

Mark the vector as containing no Objects, and drop all references to the Objects previously contained.

---

setElementAt(Object, int)

```
public boolean setElementAt(java.lang.Object o, int pos)
```

Set the indexed element to an Object.

Note: this is a replacement operation - it is not an insertion into the list!

Parameters:

`o` - the Object to place in the List

`pos` - the index at which to place it.

---

size()

```
public int size()
```

Return the size of this List.

Returns: the number of Objects stored

com.sun.kjava
RadioButton

Syntax
```
public class RadioButton

java.lang.Object
  |
  +--com.sun.kjava.RadioButton
```

Description
A two-state button meant as part of a group, only one of which can be "on" at one time.

See Also: `RadioGroup`

Member Summary	
Constructors	
`RadioButton()`	Create a new RadioButton.
`RadioButton(int, int, String)`	Create a new RadioButton.
Methods	
`getText()`	Get the label of the button.
`handlePenDown(int, int)`	The pen has gone down in the button.
`isSelected()`	Is this RadioButton currently selected?
`paint()`	Paint the RadioButton on the screen.
`pressed(int, int)`	Did the user press inside the RadioButton?
`setLocation(int, int)`	Set the position of the RadioButton.
`setParent(RadioGroup)`	Set the parent RadioGroup of this button.
`setState(boolean)`	Set the state of the button.
`setText(String)`	Set the label of the button.

Inherited Member Summary
Methods inherited from class java.lang.Object
`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Constructors

RadioButton()

```
public RadioButton()
```

Create a new RadioButton.

RadioButton(int, int, String)

```
public RadioButton(int x, int y, java.lang.String text)
```

Create a new RadioButton.

Parameters:

`x` - the X coordinate of the RadioButton's position

`y` - the Y coordinate of the RadioButton's position

`text` - the label for the button

Methods

getText()

```
public java.lang.String getText()
```

Get the label of the button.

Returns: the text of the label

handlePenDown(int, int)

```
public void handlePenDown(int x, int y)
```

The pen has gone down in the button. Handle making or removing the selection.

Parameters:

`x` - the X coordinate of the RadioButton's position

`y` - the Y coordinate of the RadioButton's position

isSelected()

```
public boolean isSelected()
```

Is this RadioButton currently selected?

Returns: true if selected, false if not

paint()

```
public void paint()
```

Paint the RadioButton on the screen.

---

pressed(int, int)

```
public boolean pressed(int x, int y)
```

Did the user press inside the RadioButton?

Parameters:

`x` - the X coordinate of the RadioButton's position

`y` - the Y coordinate of the RadioButton's position

Returns: true if the coordinates are within the area, false otherwise.

---

setLocation(int, int)

```
public void setLocation(int x, int y)
```

Set the position of the RadioButton.

Parameters:

`x` - the X coordinate of the RadioButton's position

`y` - the Y coordinate of the RadioButton's position

---

setParent(RadioGroup)

```
public void setParent(RadioGroup rg)
```

Set the parent RadioGroup of this button.

Parameters:

`rg` - the parental RadioGroup

---

setState(boolean)

```
public void setState(boolean state)
```

Set the state of the button.

Parameters:

`state` - the new state; `true` means "selected"

---

setText(String)

```
public void setText(java.lang.String text)
```

Set the label of the button.

Parameters:

`text` - the new text of the label

com.sun.kjava
RadioGroup

Syntax
```
public class RadioGroup

java.lang.Object
  |
  +--com.sun.kjava.RadioGroup
```

Description
An object representing a group of RadioButtons. At most one RadioButton in a RadioGroup can be selected at one time.

See Also: RadioButton

Member Summary	
Constructors	
RadioGroup(int)	Create a new RadioGroup.
Methods	
add(RadioButton)	Add a RadioButton to the RadioGroup.
buttonAt(int)	Get the RadioButton at an index.
getSelected()	Get the currently selected RadioButton.
hasSelection()	Is any one of the RadioButtons in the group selected?
setSelected(RadioButton)	Set the currently-selected RadioButton.
size()	How many RadioButtons in this group?

Inherited Member Summary
Methods inherited from class java.lang.Object
getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Constructors

RadioGroup(int)
```
public RadioGroup(int numButtons)
```
Create a new RadioGroup.

add(RadioButton)

Parameters:
`numButtons` - the number of RadioButtons it will contain

Methods

---

add(RadioButton)

```
public void add(RadioButton theButton)
```

Add a RadioButton to the RadioGroup.

Parameters:
`theButton` - the RadioButton to add

---

buttonAt(int)

```
public RadioButton buttonAt(int i)
```

Get the RadioButton at an index.

Parameters:
`i` - the index of the RadioButton to return

Returns: the requested RadioButton

---

getSelected()

```
public RadioButton getSelected()
```

Get the currently selected RadioButton.

Returns: the currently selected RadioButton

---

hasSelection()

```
public boolean hasSelection()
```

Is any one of the RadioButtons in the group selected?

Returns: true if one of the RadioButtons in the group is selected.

---

setSelected(RadioButton)

```
public void setSelected(RadioButton theButton)
```

Set the currently-selected RadioButton. Clear the old selection.

Parameters:
`theButton` - the RadioButton to select

---

size()

```
public int size()
```

How many RadioButtons in this group?

Returns: the number of RadioButtons in the group

com.sun.kjava
ScrollOwner

Syntax
```
public abstract interface ScrollOwner
```

All Known Implementing Classes: ScrollTextBox

Description
Interface between something that scrolls and something that cares about that something that scrolls.

Member Summary	
Methods	
setScrollValue(int)	Tell our owner where we've scrolled to.

Methods

---

setScrollValue(int)

```
public void setScrollValue(int value)
```

Tell our owner where we've scrolled to.

com.sun.kjava
ScrollTextArea

Syntax

```
public class ScrollTextArea extends ScrollTextBox
```

```
java.lang.Object
  |
  +--TextBox
        |
        +--ScrollTextBox
              |
              +--com.sun.kjava.ScrollTextArea
```

All Implemented Interfaces: ScrollOwner

Member Summary

Constructors

ScrollTextArea(String, int, int, int, int)	Create a new ScrollTextArea object.

Methods

getText()	getText() Returns the text.
handleKeyDown(int)	The user pressed a key.
handlePenDown(int, int)	The pen has gone down at (x, y).
kill()	kill() Stops the caret from blinking.
paint()	paint() Paints the text onto the screen.
setCaret()	setCaret Set the caret on the screen
setCursor(int, int)	setCursor set the cursor on the screen by determining the closest x and y positions.

Inherited Member Summary

Fields inherited from class TextBox

text, lineStarts, lineEnds, xPos, yPos, width, height, g, widthM, heightM

Methods inherited from class ScrollTextBox

setBounds(int, int, int, int), setText(String), init(), contains(int, int), handlePenMove(int, int), setScrollValue(int)

Methods inherited from class TextBox

getNumLines()

Methods inherited from class java.lang.Object

getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Constructors

ScrollTextArea(String, int, int, int, int)

```
public ScrollTextArea(java.lang.String t, int x, int y, int w, int h)
```

Create a new ScrollTextArea object.

Parameters:

`t` - the initial text

`x` - the X coordinate of the ScrollTextArea's position

`y` - the Y coordinate of the ScrollTextArea's position

`w` - the width

`h` - the height

Methods

---

getText()

```
public java.lang.String getText()
```

getText() Returns the text.

Overrides: `getText()` in class `TextBox`

---

handleKeyDown(int)

```
public void handleKeyDown(int keyCode)
```

The user pressed a key. Do the right thing.

Overrides: `handleKeyDown(int)` in class `ScrollTextBox`

Parameters:

`keyCode` - a code representing the key the user pressed

---

handlePenDown(int, int)

```
public void handlePenDown(int x, int y)
```

The pen has gone down at (x, y). Do the right thing.

Overrides: `handlePenDown(int, int)` in class `ScrollTextBox`

Parameters:

`x` - the X coordinate of the pen position

`y` - the Y coordinate of the pen position

---

kill()

```
public void kill()
```

kill() Stops the caret from blinking.

---

paint()

```
public void paint()
```

paint() Paints the text onto the screen.

Overrides: `paint()` in class `ScrollTextBox`

---

setCaret()

```
public boolean setCaret()
```

setCaret Set the caret on the screen

---

setCursor(int, int)

```
public void setCursor(int x, int y)
```

setCursor set the cursor on the screen by determining the closest x and y positions.

com.sun.kjava
ScrollTextBox

Syntax

```
public class ScrollTextBox extends TextBox implements ScrollOwner
```

```
java.lang.Object
  |
  +--TextBox
        |
        +--com.sun.kjava.ScrollTextBox
```

Direct Known Subclasses: ScrollTextArea, SelectScrollTextBox

All Implemented Interfaces: ScrollOwner

Description

A scrolling TextBox object. You need to control this class from a registered Spotlet. In the Spotlet class, implement penDown(), penMove() and keyDown() to call the handlePenDown(), handlePenMove() and handleKeyDown() methods of this class.

Member Summary

Constructors

ScrollTextBox()	
ScrollTextBox(String, int, int, int, int)	Create a new ScrollTextBox object.

Methods

contains(int, int)	Is this point inside the bounds of the object?
handleKeyDown(int)	The user pressed a key.
handlePenDown(int, int)	The pen has gone down at (x, y).
handlePenMove(int, int)	The pen has moved at (x, y).
init()	Initialize the object.
paint()	Paint the ScrollTextBox.
setBounds(int, int, int, int)	Reset the display bounds of the ScrollTextBox.
setScrollValue(int)	Set the current scroll value and repaint.
setText(String)	Set the text.

Inherited Member Summary

Fields inherited from class TextBox

Inherited Member Summary
text, lineStarts, lineEnds, xPos, yPos, width, height, g, widthM, heightM
Methods inherited from class TextBox
getNumLines(), getText()
Methods inherited from class java.lang.Object
getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Constructors

ScrollTextBox()

```
protected ScrollTextBox()
```

ScrollTextBox(String, int, int, int, int)

```
public ScrollTextBox(java.lang.String t, int x, int y, int w, int h)
```

Create a new ScrollTextBox object.

Parameters:

t - the initial text

x - the X coordinate of the ScrollTextBox's position

y - the Y coordinate of the ScrollTextBox's position

w - the width

h - the height

Methods

contains(int, int)

```
public boolean contains(int x, int y)
```

Is this point inside the bounds of the object?

Parameters:

x - the X coordinate of the position to test

y - the Y coordinate of the position to test

Returns: true of the point is inside our bounds

handleKeyDown(int)

```
public void handleKeyDown(int keyCode)
```

The user pressed a key. Do the right thing.

Parameters:

`keyCode` - a code representing the key the user pressed

---

handlePenDown(int, int)

```
public void handlePenDown(int x, int y)
```

The pen has gone down at (x, y). Do the right thing.

Parameters:

`x` - the X coordinate of the pen position

`y` - the Y coordinate of the pen position

---

handlePenMove(int, int)

```
public void handlePenMove(int x, int y)
```

The pen has moved at (x, y). Do the right thing.

Parameters:

`x` - the X coordinate of the pen position

`y` - the Y coordinate of the pen position

---

init()

```
protected void init()
```

Initialize the object.

---

paint()

```
public void paint()
```

Paint the ScrollTextBox.

Overrides: `paint()` in class `TextBox`

---

setBounds(int, int, int, int)

```
public void setBounds(int x, int y, int w, int h)
```

Reset the display bounds of the ScrollTextBox.

Overrides: `setBounds(int, int, int, int)` in class `TextBox`

Parameters:

`x` - the new X coordinate of the ScrollTextBox's position

`y` - the new Y coordinate of the ScrollTextBox's position

`w` - the new width

`h` - the new height

---

setScrollValue(int)

```
public void setScrollValue(int val)
```

Set the current scroll value and repaint.

Specified By: `setScrollValue(int)` in interface `ScrollOwner`

Parameters:

`val` - the new scroll value.

---

setText(String)

```
public void setText(java.lang.String t)
```

Set the text. You need to call paint() on the ScrollTextBox to get the new text/scrollbar to display.

Overrides: `setText(String)` in class `TextBox`

Parameters:

`t` - a String representing the new text.

com.sun.kjava
SelectScrollTextBox

Syntax
```
public class SelectScrollTextBox extends ScrollTextBox
```

```
java.lang.Object
  |
  +--TextBox
        |
        +--ScrollTextBox
              |
              +--com.sun.kjava.SelectScrollTextBox
```

All Implemented Interfaces: ScrollOwner

Member Summary

Fields
LEADING

Constructors
SelectScrollText-
Box(String, int, int,
int, int)

Methods
getSelection(int,
int)
setText(String)

Inherited Member Summary

Fields inherited from class TextBox

text, lineStarts, lineEnds, xPos, yPos, width, height, g, widthM, heightM

Methods inherited from class ScrollTextBox

setBounds(int, int, int, int), init(), contains(int, int), handlePenDown(int, int), handlePenMove(int, int), handleKeyDown(int), paint(), setScrollValue(int)

Methods inherited from class TextBox

getNumLines(), getText()

Methods inherited from class java.lang.Object

getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Fields

LEADING

```
public static final int LEADING
```

Constructors

SelectScrollTextBox(String, int, int, int, int)

```
public SelectScrollTextBox(java.lang.String t, int x, int y, int w, int h)
```

Methods

---

getSelection(int, int)

```
public java.lang.String getSelection(int x, int y)
```

---

setText(String)

```
public void setText(java.lang.String t)
```

Overrides: `setText(String)` in class `ScrollTextBox`

com.sun.kjava
Slider

Syntax
```
public class Slider

java.lang.Object
  |
  +--com.sun.kjava.Slider
```

Description
Slider: A graphical valuator object. Allows user to select a value by sliding a marker on a scale. This class isn't very graceful about handling conditions where the width of the slider is less than the interval of the maximum and minimum values. It calculates a "skip" value in these cases to increment the value for each pixel on the screen, e.g. Slider s1 = new Slider(5, 100, 100, 0, 1000, 0) creates a slider 100 pixels wide to handle the interval 0->1000. It then treats each pixel as being 10 units, and the user can only generate values in multiples of 10.

Member Summary	
Constructors	
Slider()	Create a new Slider object.
Slider(int, int, int, int, int, int)	Create a Slider object.
Methods	
contains(int, int)	Is this point within the Slider's bounds?
drawMarker(int)	Draw the Slider's marker.
handlePenDown(int, int)	Deal with the fact that the pen went down.
handlePenMove(int, int)	Deal with the fact that the pen moved.
paint()	Draw the Slider.
setLocation(int, int)	Set the position of the Slider.
setSizeRange(int, int, int, int)	Reset the width, limits, and value of the Slider.

Inherited Member Summary
Methods inherited from class java.lang.Object
getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Constructors

Slider()

```
public Slider()
```

Create a new Slider object.

Slider(int, int, int, int, int, int)

```
public Slider(int x, int y, int w, int mn, int mx, int initVal)
```

Create a Slider object.

Parameters:

`x` - the X coordinate of the Slider's position

`y` - the Y coordinate of the Slider's position

`w` - the width

`mn` - the minimum value

`mx` - the maximum value

`initVal` - the initial value

Methods

contains(int, int)

```
public boolean contains(int x, int y)
```

Is this point within the Slider's bounds?

Parameters:

`x` - the X coordinate to test

`y` - the Y coordinate to test

Returns: true if the point is in bounds, false otherwise

drawMarker(int)

```
public void drawMarker(int drawStyle)
```

Draw the Slider's marker.

Parameters:

`drawStyle` - the style in which to draw it.

handlePenDown(int, int)

```
public void handlePenDown(int x, int y)
```

Deal with the fact that the pen went down.

Parameters:

`x` - the X coordinate of the pen's new position

`y` - the Y coordinate of the pen's new position

---

handlePenMove(int, int)

```
public void handlePenMove(int x, int y)
```

Deal with the fact that the pen moved.

Parameters:

`x` - the X coordinate of the pen's new position

`y` - the Y coordinate of the pen's new position

---

paint()

```
public void paint()
```

Draw the Slider.

---

setLocation(int, int)

```
public void setLocation(int x, int y)
```

Set the position of the Slider.

Parameters:

`x` - the new X coordinate

`y` - the new Y coordinate

---

setSizeRange(int, int, int, int)

```
public void setSizeRange(int w, int mn, int mx, int val)
```

Reset the width, limits, and value of the Slider.

Parameters:

`w` - the new width

`mn` - the new minimum value

`mx` - the new maximum value

`val` - the new current value

com.sun.kjava
Spotlet

Syntax

```
public class Spotlet

java.lang.Object
  |
  +--com.sun.kjava.Spotlet
```

Direct Known Subclasses: Dialog, HelpDisplay

Description

This class provides callbacks for event handling. Applications extend this class and override the relevant event handling methods. An application may use more than one Spotlet object, but at most one Spotlet can have the *focus* at any one time. That is, events will only trigger the callbacks of one Spotlet at any given time, the Spotlet with the current focus.

To become the focus, a Spotlet invokes the `register` method which also removes the focus from the previously registered Spotlet (if any).

Member Summary

Fields

CALCICON	Constant for the calculator icon.
KEY_HARD1	Constants for the other Palm system "hard" keys.
KEY_HARD2	
KEY_HARD3	
KEY_HARD4	
KEY_POWER	
MENUICON	Constant for the menu icon.
NO_EVENT_OPTIONS	Constants for the eventOptions of register().
PAGEDOWN	
PAGEUP	Constants for the page up/down "hard" keys.
WANT_SYSTEM_KEYS	

Constructors

Spotlet()	

Methods

beamReceive(byte[])	This method is used for receiving packets of data via infrared from other Palm devices.
beamSend(byte[])	This method is used for beaming data packets via infrared to another Palm device.
dispatch(int, DataInput)	
getFlashID()	This method is used to get the flashID of the Palm device.
keyDown(int)	This method is invoked if the user presses either of the page up or page down hard keys, taps the calculator or menu icon, or enters a character (e.g.
penDown(int, int)	This method is invoked if the user places the pen on the display.
penMove(int, int)	This method is invoked if the user moves the pen over the display.

Member Summary	
penUp(int, int)	This method is invoked if the user removes the pen from the display.
register(int)	Register the event handlers of this object.
setPalmEventOptions(int)	
unknownEvent(int, DataInput)	Catch all routine
unregister()	Unregister the event handlers of this object.

Inherited Member Summary
Methods inherited from class java.lang.Object
`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Fields

CALCICON

```
public static final int CALCICON
```

Constant for the calculator icon.

KEY_HARD1

```
public static final int KEY_HARD1
```

Constants for the other Palm system "hard" keys.

KEY_HARD2

```
public static final int KEY_HARD2
```

KEY_HARD3

```
public static final int KEY_HARD3
```

KEY_HARD4

```
public static final int KEY_HARD4
```

KEY_POWER

```
public static final int KEY_POWER
```

MENUICON

```
public static final int MENUICON
```

Constant for the menu icon.

NO_EVENT_OPTIONS

```
public static final int NO_EVENT_OPTIONS
```

Constants for the eventOptions of register().

PAGEDOWN

```
public static final int PAGEDOWN
```

PAGEUP

```
public static final int PAGEUP
```

Constants for the page up/down "hard" keys.

WANT_SYSTEM_KEYS

```
public static final int WANT_SYSTEM_KEYS
```

Constructors

Spotlet()

```
public Spotlet()
```

Methods

beamReceive(byte[])

```
public void beamReceive(byte[] data)
```

This method is used for receiving packets of data via infrared from other Palm devices. The data that is read is received in a byte array that is allocated automatically by the virtual machine.

beamSend(byte[])

```
public static native boolean beamSend(byte[] data)
```

This method is used for beaming data packets via infrared to another Palm device. IMPORTANT: Unlike the methods above, this method is not an event handler. Rather, you call this method explicitly to beam data to another device. The other device must have registered a beamReceive handler in its current Spotlet to receive data.

Returns: true if beaming succeeded, false otherwise.

---

dispatch(int, DataInput)

```
public void dispatch(int event, java.io.DataInput in)
```

Throws: `IOException`

---

getFlashID()

```
public static native java.lang.String getFlashID()
```

This method is used to get the flashID of the Palm device. IMPORTANT: Unlike the methods above, this method is not an event handler.

Returns: a String containing the flashID.

---

keyDown(int)

```
public void keyDown(int keyCode)
```

This method is invoked if the user presses either of the page up or page down hard keys, taps the calculator or menu icon, or enters a character (e.g. via Graffiti). If it is one of the hard key presses, then it will match one of the corresponding constants defined in this class.

Parameters:

`keyCode` - the code of the key the user entered

---

penDown(int, int)

```
public void penDown(int x, int y)
```

This method is invoked if the user places the pen on the display.

Parameters:

`x` - the x coordinate of the point at which the pen was placed

`y` - the y coordinate of the point at which the pen was placed

---

penMove(int, int)

```
public void penMove(int x, int y)
```

This method is invoked if the user moves the pen over the display.

Parameters:

`x` - the x coordinate of the destination point of the move

`y` - the y coordinate of the destination point of the move

penUp(int, int)

```
public void penUp(int x, int y)
```

This method is invoked if the user removes the pen from the display.

Parameters:

`x` - the x coordinate of the point from which the pen was removed

`y` - the y coordinate of the point from which the pen was removed

register(int)

```
public void register(int eventOptions)
```

Register the event handlers of this object. This effectively makes this Spotlet the *focus* for event handling. A side effect this is that all previously registered handlers (if any) are unregistered and the Spotlet to which they belong loses the focus.

Parameters:

`eventOptions` - one of NO_EVENT_OPTIONS or WANT_SYSTEM_KEYS

setPalmEventOptions(int)

```
public static native void setPalmEventOptions(int eventOptions)
```

unknownEvent(int, DataInput)

```
public void unknownEvent(int event, java.io.DataInput in)
```

Catch all routine

unregister()

```
public void unregister()
```

Unregister the event handlers of this object. It is only necessary to use this method when not transferring the *focus* from this Spotlet to another one via a subsequent call to `register`. If this Spotlet does not currently have the focus, this method does nothing.

com.sun.kjava
TextBox

Syntax

```
public class TextBox

java.lang.Object
  |
  +--com.sun.kjava.TextBox
```

Direct Known Subclasses: ScrollTextBox

Description

A box displaying text on the screen. This class flows the text in the box. It doesn't break words, and therefore isn't graceful handling words larger than the width of the box.

Member Summary

Fields

g
height
heightM
lineEnds
lineStarts
text
width
widthM
xPos
yPos

Constructors

TextBox()	Create a new TextBox object.
TextBox(String, int, int, int, int)	Create a new TextBox object.

Methods

getNumLines()	How many lines of text does the TextBox currently hold?
getText()	Gets the text entered into the textbox
paint()	Paint the TextBox on the screen.
setBounds(int, int, int, int)	Reset the display bounds of the TextBox.
setText(String)	Set the text.

Inherited Member Summary
Methods inherited from class java.lang.Object `getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Fields

---

g

```
protected Graphics g
```

---

height

```
protected int height
```

---

heightM

```
protected static int heightM
```

---

lineEnds

```
protected IntVector lineEnds
```

---

lineStarts

```
protected IntVector lineStarts
```

---

text

```
protected java.lang.String text
```

---

width

```
protected int width
```

---

widthM

```
protected static int widthM
```

---

xPos

```
protected int xPos
```

---

yPos

```
protected int yPos
```

Constructors

TextBox()

```
public TextBox()
```

Create a new TextBox object.

TextBox(String, int, int, int, int)

```
public TextBox(java.lang.String t, int x, int y, int w, int h)
```

Create a new TextBox object.

Parameters:

`t` - the initial text

`x` - the X coordinate of the ScrollTextBox's position

`y` - the Y coordinate of the ScrollTextBox's position

`w` - the width

`h` - the height

Methods

---

getNumLines()

```
public int getNumLines()
```

How many lines of text does the TextBox currently hold?

Returns: the number of lines of text contained

---

getText()

```
public java.lang.String getText()
```

Gets the text entered into the textbox

Returns: String containing the user's entry

---

paint()

```
public void paint()
```

Paint the TextBox on the screen.

---

setBounds(int, int, int, int)

```
public void setBounds(int x, int y, int w, int h)
```

Reset the display bounds of the TextBox.

Parameters:

`x` - the new X coordinate of the ScrollTextBox's position

`y` - the new Y coordinate of the ScrollTextBox's position

`w` - the new width

`h` - the new height

---

setText(String)

```
public void setText(java.lang.String t)
```

Set the text. You need to call paint() on the TextBox to get the new text displayed.

Parameters:

`t` - a String representing the new text.

com.sun.kjava
TextField

Syntax

```
public class TextField

java.lang.Object
  |
  +--com.sun.kjava.TextField
```

Description

This class provides a simple TextField. It creates a thread for the caret to blink, accepts key input (including delete and backspace) and allows for only upper case entry. At present there is no support for Pen selection at all. It needs to be used in conjunction with a Spotlet, as this class does not extend Spotlet and therefore has no event handling itself. You need to get the Spotlet keyDown() method to call this class's handleKeyDown() method. After construction, to get the field "working" call setFocus() this will start the caret. Call loseFocus() to stop the caret when it's all over.

Member Summary

Constructors

`TextField(String, int, int, int, int)`	Create a new TextField

Methods

`getText()`	Gets the text entered into the textfield
`handleKeyDown(int)`	Should be called by Spotlet.keyDown().
`hasFocus()`	Returns whether or not the textfield has focus
`killCaret()`	Stops the caret thread.
`loseFocus()`	Stops the caret blinking.
`paint()`	
`pressed(int, int)`	Returns whether or not the x,y position is inside the textfield
`setFocus()`	Give the textfield "focus".
`setText(String)`	Sets the text in the textfield.
`setUpperCase(boolean)`	Set whether or not the textfield should convert everything to upper case

Inherited Member Summary

Methods inherited from class java.lang.Object

`getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait`

Constructors

TextField(String, int, int, int, int)

```
public TextField(java.lang.String ttext, int x, int y, int w, int h)
```

Create a new TextField

Parameters:

`ttext` - The title (label) for the text field

`x` - x position (upper left)

`y` - y position (upper left)

`w` - width (including label)

`h` - height

Methods

getText()

```
public java.lang.String getText()
```

Gets the text entered into the textfield

Returns: String containing the user's entry

handleKeyDown(int)

```
public void handleKeyDown(int key)
```

Should be called by Spotlet.keyDown(). Currently this handles backspace (0x08) and delete (0x7f) as backwards delete. Does upper case conversion if necessary.

hasFocus()

```
public boolean hasFocus()
```

Returns whether or not the textfield has focus

See Also: `setFocus()`, `loseFocus()`

killCaret()

```
public void killCaret()
```

Stops the caret thread.

loseFocus()

```
public void loseFocus()
```

Stops the caret blinking.

paint()

See Also: `setFocus()`

---

paint()

```
public void paint()
```

---

pressed(int, int)

```
public boolean pressed(int x, int y)
```

Returns whether or not the x,y position is inside the textfield

See Also: `setFocus()`, `loseFocus()`

---

setFocus()

```
public void setFocus()
```

Give the textfield "focus". The registered Spotlet actually has focus. This method kicks off the caret thread to get the caret to blink.

---

setText(String)

```
public void setText(java.lang.String txt)
```

Sets the text in the textfield. Use this to pre-set (or clear) the value displayed in the textfield. Note: Does not convert the string to upper case, even if the textfield has been set to upper case only.

---

setUpperCase(boolean)

```
public void setUpperCase(boolean flag)
```

Set whether or not the textfield should convert everything to upper case

Parameters:

`flag` - if true then convert chars to upper case

com.sun.kjava
VerticalScrollBar

Syntax
```
public class VerticalScrollBar

java.lang.Object
  |
  +--com.sun.kjava.VerticalScrollBar
```

Description
A vertical scroll bar user interface object.

Member Summary

Fields

SCROLL_BAR_WIDTH	

Constructors

VerticalScroll-Bar(ScrollOwner)	Create a new VerticalScrollBar and associate it with an owner.
VerticalScroll-Bar(ScrollOwner, int, int, int, int, int)	Create a new VerticalScrollBar and associate it with an owner.

Methods

contains(int, int)	Does the scroll bar contain the point in question?
handleKeyDown(int)	The user pressed a key.
handlePenDown(int, int)	The pen went down somewhere.
handlePenMove(int, int)	Deal with the fact that the pen moved.
init(int, int, int, int, int, int)	Initialize the scroll bar.
paint()	Paint the VerticalScrollBar.
setBounds(int, int, int, int, int, int)	Set the scroll bar's bounds.

Inherited Member Summary

Methods inherited from class java.lang.Object

getClass, hashCode, equals, toString, notify, notifyAll, wait, wait, wait

Fields

SCROLL_BAR_WIDTH

```
public static int SCROLL_BAR_WIDTH
```

Constructors

VerticalScrollBar(ScrollOwner)

```
public VerticalScrollBar(ScrollOwner so)
```

Create a new VerticalScrollBar and associate it with an owner.

Parameters:

`so` - the ScrollOwner that owns this scroll bar.

VerticalScrollBar(ScrollOwner, int, int, int, int, int, int)

```
public VerticalScrollBar(ScrollOwner so, int x, int y, int h, int min, int max,
            int initVal)
```

Create a new VerticalScrollBar and associate it with an owner.

Parameters:

`so` - the ScrollOwner that owns this scroll bar.

`x` - the X coordinate of the scroll bar

`y` - the Y coordinate of the scroll bar

`h` - the height of the scroll bar

`min` - the minimum value allowed

`max` - the maximum value allowed

`initVal` - the initial value

Methods

contains(int, int)

```
public boolean contains(int x, int y)
```

Does the scroll bar contain the point in question?

Parameters:

`x` - the X coordinate to test

`y` - the Y coordinate to test

Returns: true if the point is within the scroll bar's bounds

---

handleKeyDown(int)

```
public void handleKeyDown(int keyCode)
```

The user pressed a key. Deal with it.

Parameters:

`keyCode` - the code of the key the user pressed

---

handlePenDown(int, int)

```
public void handlePenDown(int x, int y)
```

The pen went down somewhere. Deal with it.

Parameters:

`x` - the X coordinate of the pen's position

`y` - the Y coordinate of the pen's position

---

handlePenMove(int, int)

```
public void handlePenMove(int x, int y)
```

Deal with the fact that the pen moved.

Parameters:

`x` - the X coordinate of the pen's position

`y` - the Y coordinate of the pen's position

---

init(int, int, int, int, int, int)

```
protected void init(int x, int y, int h, int min, int max, int initVal)
```

Initialize the scroll bar.

Parameters:

`x` - the X coordinate of the scroll bar

`y` - the Y coordinate of the scroll bar

`h` - the height of the scroll bar

`min` - the minimum value allowed

`max` - the maximum value allowed

`initVal` - the initial value

---

paint()

```
public void paint()
```

Paint the VerticalScrollBar.

---

setBounds(int, int, int, int, int, int)

setBounds(int, int, int, int, int, int)

```
public void setBounds(int x, int y, int h, int min, int max, int initVal)
```

Set the scroll bar's bounds.

Parameters:

`x` - the X coordinate of the scroll bar

`y` - the Y coordinate of the scroll bar

`h` - the height of the scroll bar

`min` - the minimum value allowed

`max` - the maximum value allowed

`initVal` - the initial value

Index

A

B

C

D

E

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Y