mmsv2gui (version 0.8, 12 jul 2008)

Classes

Control
- ButtonControl
- LabelControl
- ImageControl
- ListControl
- TextBoxControl
- RectangleControl
- TextFieldControl
- ImageListControl
- ProgressControl

ListItem
Window
Action

Functions

redraw
themePath
invokeLater

class **ButtonControl(**Control**)**

```
ButtonControl(int x, int y, int width, int height [, text, font, rgb,
               rgbfocus, focus, nofocus, textoffset, alignment])

text           : string (button text)
font           : string (example, 'Vera')
rgb            : hex string (example, '0xffffff')
rgbfocus       : hex string (example, '0xffffff')
focus          : string (filename for focus texture)
nofocus        : string (filename for no focus texture)
textoffset     : integer (text offset in pixels counting from left)
alignment      : string (text alignment, 'left', 'center' or 'right')
```

Image path for focus and nofocus texture has to be relative script folder or mmsv2 theme folder.

Default values:

```
font='Vera', rgb='0xffffff' (white), rgbfocus='0xffffff' (white),
textoffset=0, alignment='left'
```

Methods defined here:

```
setLabel([text, font, rgb, rgbfocus])
```

text	: string
font	: string (example, 'Vera')
rgb	: hex string (example, '0xffffff')
rgbfocus	: hex string (example, '0xffffff')

Methods inherited from Control:

controlNext(Control) -- Set next control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **LabelControl(**Control**)**

LabelControl(int x, int y, int width, int height [, text, font, rgb, textoffset, alignment])

text	: string (label text)
font	: string (example, 'Vera')
rgb	: hex string (example, '0xffffff')

textoffset	: integer (text offset in pixels counting from left)
alignment	: string (text alignment, 'left', 'center' or 'right')

Default values:

font='Vera', rgb='0xffffff' (white), textoffset=0, alignment='left'

Methods defined here:

setLabel([text, font, rgb])

text	: string
font	: string (example, 'Vera')
rgb	: hex string (example, '0xffffff')

Methods inherited from Control:

controlNext(Control) -- Set next control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **ImageControl(**Control**)**

ImageControl(int x, int y, int width, int height, [path])

```
path           : string (filename for image)
```

Image path has to be relative script folder or mmsv2 theme folder.

Default values:

Methods defined here:

setPath(path)

```
path           : string (filename for image)
```

Sets the image path. Image path has to be relative script folder or mmsv2 theme folder.

Methods inherited from Control:

controlNext(Control) -- Set next control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **ListControl(**Control**)**

```
ListControl(int x, int y, int width, int height [, font, rgb, rgbfocus,
                focus, nofocus, textoffset, alignment, itemheight,
                itemspace])

font           : string (example, 'Vera')
rgb            : hex string (example, '0xffffff')
rgbfocus       : hex string (example, '0xffffff')
focus          : string (filename for focus texture)
nofocus        : string (filename for no focus texture)
textoffset     : integer (text offset in pixels counting from left)
alignment      : string (text alignment, 'left', 'center' or 'right')
itemheight     : integer (height of items)
itemspace      : integer (space between items)
```

Image path for focus and nofocus texture has to be relative script folder or mmsv2 theme folder.

Default values:

font='Vera', rgb='0xffffff' (white), rgbfocus='0xffffff' (white), textoffset=0, alignment='left', itemheight=30, itemspace=0

Methods defined here:

setLabel([font, rgb, rgbfocus])

```
font           : string (example, 'Vera')
rgb            : hex string (example, '0xffffff')
rgbfocus       : hex string (example, '0xffffff')
```

Will change the font and/or rgb of all list items.

addItem(ListItem item) -- Add a new listitem to this list.

```
Throws:     TypeError, if supplied argument is not a ListItem type
            ReferenceError, if item is already in list
```

getSelectedItem()-- Returns the selected listitem.

```
Throws:     RuntimeError, if there are no items in list
```

getSelectedPosition()-- Returns the current position (integer)

clear()-- Clear all listitems in this list.

setItemHeight(int height) -- Set height of items in this list.

setItemSpace(int space) -- Set space between items in this list.

size()-- Returns number of items in list.

Methods inherited from Control:

controlNext(Control) -- Set next control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **TextBoxControl(**Control**)**

TextBoxControl(int x, int y, int width, int height [, font, rgb,
textheight, textspace])

```
font           : string (example, 'Vera')
```

```
rgb            : hex string (example, '0xffffff')
textheight     : integer (height of text lines)
textspace      : integer (space between text lines)
```

Default values:

font='Vera', rgb='0xffffff' (white), textheight=30, textspace=0

Methods defined here:

setText([text, font, rgb])

```
text           : string (text to show)
font           : string (example, 'Vera')
rgb            : hex string (example, '0xffffff')
```

clear()-- Clear all text.

setTextHeight(int height) -- Set height of text lines.

setTextSpace(int space) -- Set space between text lines.

Methods inherited from Control:

controlNext(Control) -- Set next control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **RectangleControl(**Control**)**

RectangleControl(int x, int y, int width, int height [, alpha, rgb])

```
alpha          : integer
rgb            : hex string (example, '0xffffff')
```

Default values:

alpha=100, rgb='0xffffff' (white)

Methods defined here:

setColor([alpha, rgb])

```
alpha          : integer
rgb            : hex string (example, '0xffffff')
```

Methods inherited from Control:

controlNext(Control) -- Set next control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **TextFieldControl(**Control**)**

TextFieldControl(int x, int y, int width, int height [, text, font, rgb, textoffset, alignment])

```
text          : string (label text)
font          : string (example, 'Vera')
rgb           : hex string (example, '0xffffff')
textoffset    : integer (text offset in pixels counting from left)
alignment     : string (text alignment, 'left', 'center' or 'right')
```

Default values:

font='Vera', rgb='0xffffff' (white), textoffset=0, alignment='left'

Methods defined here:

setText([text, font, rgb])

```
text          : string
font          : string (example, 'Vera')
rgb           : hex string (example, '0xffffff')
```

getText() -- Return text.

setEditable(editable) -- Make this control editable or not :-)

```
editable      : bool
```

getEditable() -- Return 0 or 1 (not editable, editable).

Methods inherited from Control:

controlNext(Control) -- Set next control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **ImageListControl(**Control**)**

ImageListControl(int x, int y, int width, int height [, font, rgb, rgbfocus, textalignment, itemheight, itemwidth, textheight, itemspacex, itemspacey, listcenterx, listcentery])

font	: string (example, 'Vera')
rgb	: hex string (example, '0xffffff')
rgbfocus	: hex string (example, '0xffffff')
textalignment	: string (image text alignment, 'left', 'center' or 'right')
itemheight	: integer (height of items)
itemwidth	: integer (width of items)
textheight	: integer (height of text)
itemspacex	: integer (horizontel space between items, in pixels)
itemspacey	: integer (vertical space between items, in pixels)
listcenterx	: bool (center the list items horizontal or not)
listcentery	: bool (center the list items vertical or not)

Default values:

```
font='Vera', rgb='0x7f7f7f' (grey), rgbfocus='0xffffff' (white),
textalignment='left', itemheight=30, itemwidth=30, textheight=20,
itemspacex=10, itemspacey=1, listcenterx=False, listcentery=False
```

Methods defined here:

setLabel([font, rgb, rgbfocus])

```
font          : string (example, 'Vera')
rgb           : hex string (example, '0xffffff')
rgbfocus      : hex string (example, '0xffffff')
```

Will change the font and/or rgb of all list items.

addItem(ListItem item) -- Add a new listitem to this list.

```
Throws:     TypeError, if supplied argument is not a ListItem type
            ReferenceError, if item is already in list
```

getSelectedItem()-- Returns the selected listitem.

```
Throws:     RuntimeError, if there are no items in list
```

getSelectedPosition()-- Returns the current position (integer)

clear()-- Clear all listitems in this list.

setItemHeight(int height) -- Set height of items in this list.

setItemWidth(int width) -- Set width of items in this list.

setItemSpaceX(int space) -- Set horizontal space between items in list.

setItemSpaceY(int space) -- Set vertical space between items in list.

size()-- Returns number of items in list.

setTextHeight(int height) -- Set height of text in this list.

Methods inherited from Control:

controlNext(Control) -- Set next control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

Throws: TypeError, if the supplied argument is not a Control type
ReferenceError, if the control is not added to a window

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() -- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **ProgressControl(**Control**)**

ProgressControl(int x, int y, int width, int height [, text, font, rgb, alignment, rgbprogress, alphaprogress, textoffset])

text	: string (label text)
font	: string (example, 'Vera')
rgb	: hex string (example, '0xffffff')
alignment	: string (text alignment, 'left', 'center' or 'right')
rgbprogress	: hex string (example, '0xffffff')
alphaprogress	: integer (alpha for progressbar)
textoffset	: integer (text offset in pixels counting from left)

Default values:

font='Vera', rgb='0xffffff' (white), textoffset=0, alignment='center', rgbprogress=0x00ff00 (green), alphaprogress=100

Methods defined here:

setText([text, font, rgb])

text	: string
font	: string (example, 'Vera')
rgb	: hex string (example, '0xffffff')

setBackground([rgb, alpha])

```
rgb            : hex string (example, '0xffffff')
alpha          : integer
```

setBounds([min, max])

```
min            : integer
max            : integer
```

update([value])

```
value          : integer
```

Methods inherited from Control:

controlNext(Control) -- Set next control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses next, the supplied Control will receive focus.

controlLeft(Control) -- Set left control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses left, the supplied Control will receive focus.

controlRight(Control) -- Set right control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses right, the supplied Control will receive focus.

controlPrev(Control) -- Set prev control.

```
Throws:     TypeError, if the supplied argument is not a Control type
            ReferenceError, if the control is not added to a window
```

When this control is active and the user presses prev, the supplied Control will receive focus.

setHeight(int height) -- Set's the height of this control.

setPosition(int x, int y) -- Set's the position of this control.

setVisible(bool visible) -- Hide's or Show's this control.

getVisible() –- Return 0 or 1 (not visible, visible).

setWidth(int width) -- Set's the width of this control.

class **ListItem()**

ListItem() -- Creates a new listitem.

Inherit from this class to create a custom listitem. Just remember to set the label.

Methods defined here:

setLabel(string label) -- Set's the listitem label.

getLabel() -- Return's the listitem label.

setLabel2(string label) -- Set's the listitem label2.

getLabel2() -- Return's the listitem label2.

setThumbImage(string thumbimage) -- Set's the listitem thumbimage.

Image path for thumb texture has to be relative script folder or mmsv2 theme folder.

class **Window()**

Window([string xmlfile])

```
xmlfile       : xml window and control definitions to load
```

Deleting this window will activate the old window that was active, remove and reset (not delete) all controls that are associated with this window

Methods defined here:

addControl(Control) -- Add a Control to this window.

```
Throws:     TypeError, if supplied argument is not a Control type
            ReferenceError, if control is already added
            RuntimeError, should not happen
```

The controls below can be added to a window atm

-LabelControl
-TextBoxControl
-ButtonControl
-ListControl
-ImageControl
-RectangleControl

onAction(string action) -- on action method.

This method will receive all actions (key presses) that the main program will send to this window.
By default only the 'back' action is handled.
Overwrite this method to let your script handle all actions. Don't forget to capture some action (key press), else the user can't close this window.

onClick(int id) -- on click method.

This method will be called when a control is clicked.
Overwrite this method to let your script handle all clicks.

onFocus(int id) -- on focus method.

This method will be called when a control is focused.
Overwrite this method to let your script handle the focus event.

doModal() -- Display this window until close() is called.

This method will block and wait for key presses by user. Will in turn call the on_action method.

show() -- Shows/activates the calling window.

This method will **not** block. No key presses can be handled by the window.

close() -- Closes this window.

getFocus() -- Returns the control which is in focus.

Throws: RuntimeError, if no control has focus

getFocusId() -- Returns the id of control which is in focus.

Throws: RuntimeError, if no control has focus

setFocus(Control) -- Give the supplied control focus.

Throws: TypeError, if supplied argument is not a Control type
RuntimeError, if control is not added to this window

setFocusId(int id) -- Give the supplied control (id) focus.

Throws: TypeError, if supplied argument is not a Control type
RuntimeError, if control is not added to this window

getWidth() -- Returns the width of the screen.

getHeight() -- Returns the height of the screen.

removeControl(Control) -- Removes the control from this window.

Throws: TypeError, if supplied argument is not a Control type
RuntimeError, if control is not added to this window

This will not delete the control. It's only removed from the window.

getControl(int id) -- Returns the control with given id.

Throws: RuntimeError, if control id is not found in this window

class **Action()**

Action()

The Action class is used in conjuction with the invokeLater function (see below)

Methods defined here:

run()

Override this method to specify the action taken when called from the event handling thread.

Functions

redraw() -- Updates the active window.

themePath() -- Returns the current theme path.

invokeLater(action)

```
action        : Action object
```

Puts given action in the event queue. Use this function if you want to change the gui from a thread different then the main python thread.

Author

Fredrik