Jade Robot Panel Driver

Myke Predko

Last Updated: September 11, 2014

License and Warranty

This document and code was written for the Mimetics Jade Robot™ and follow on products.

This document and code is considered Mimetics Proprietary and may not be released outside of Mimetics except by permission of Mimetics, Inc.

Software Compatibility

The panel driver text described in this document was written to be supported by:
Robot Software Release 39 or later, except where noted
Jade Support Version 0.9.7.1 or later
Robot Tokenizer Version 0.11.12 or later
Syscall API 0.0.4 or later
Panel Processor 0.3.1 or later
Bitmap Processor 0.0.1 or later
Wave Processor 0.2.0 or later
_start.s defined as _Header8.script or later

Conventions, Options and Selections

Example code will be put in monospace font like:

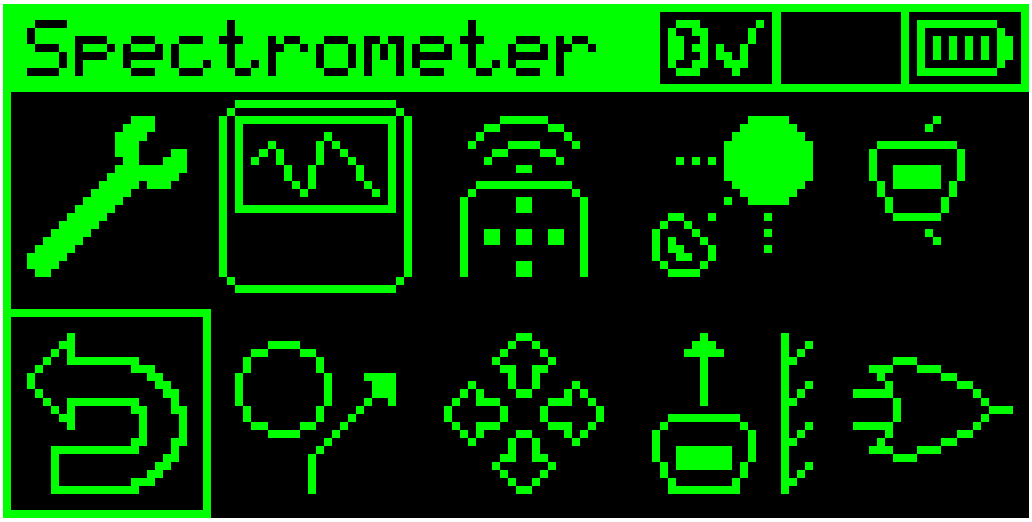
```
A = B + C
```

In the language definition, there are a number of instances where there are optional parameters or multiple parameters for the same task. To make these situations more obvious, the following convention is used:

[] – Optional parameter
| - One parameter or another
... – Previous parameter can be repeated
<none> - indicates that nothing is a possible option

Overview

The Panel Driver was created to allow developers to quickly and easily create user interfaces for the Jade Robot without the need for directly accessing display and input devices directly. The functionality provided here mimics the capabilities of pre-existing tools like GLX, but these tools are woefully over featured for the 128x64 pixel monochrome display and five button user interface of the Jade Robot. The Panel Driver provides all the basic functions required to provide a full function graphical user interface on the Jade Robot.



While the purpose of the Panel Driver and panels is to provide an easy way of developing complex user interfaces for the Jade Robot, this does not mean they are for new programmers. Before attempting programs with panels, the developer should be very familiar with the Jade Programming language and the Syscall API operation.

Support

Along with this document, more comprehensive documentation of the Jade Robot script programming language, syscall APIs is available on the Mimetics web page at:

http://www.mimetics.ca/knowledge-base-2/

Mimetics has a forum for news, questions and sample programs. They can be found here:

http://www.mimetics.ca/forum/

Mimetics is active on social media, look for “MimeticsCanada” on:

- Twitter: http://twitter.com/MimeticsCanada
- Tumblr: http://www.tumblr.com/blog/mimeticscanada
- YouTube: http://www.youtube.com/channel/UCu26ZMBlK9fBgmUOKfwK3Ig

Finally, this document will be undergoing constant updating, please check for the latest version at:

http://www.mimetics.ca/knowledge-base-2/

Jade Robot Panel Driver File Format:

Panels are defined in human readable "panel" files which are normally given the extension ".panel". These files define the size of the panel, the "controls" within it and any graphic files. Each line is a statement and, with the exception of comment lines, they are all terminated in a semicolon (";" – 0x3B). The parameters for each of the statements is listed below and it should be noted that none of them are optional.

While statements can be placed anywhere on the line, by convention, they are placed in the leftmost column.

Objects on the panel (or "subpanel", as will be discussed below) are set at pixel coordinates defined as top left being 0, 0 and the first number being the "X" coordinate and the second being the "Y" coordinate. The maximum size of a panel is the size of the OLED display, 128 ("X") by 64 ("Y") pixels. Objects on the panel cannot overlap other objects and they cannot extend beyond the outline of the panel or subpanel.

Panel "Controls" are objects on the panel which can be optionally selected using the Navigation (white) buttons to highlight (put a line around the control) and then select using the Enter (red) button. Every user input control is given an identifier so it can be accessed using the Syscall APIs – no two controls in the same panel can have the same identifiers.

The statement types are:

Blank line – Used for adding white space to the .panel file and make it more readable.

The comment line is to save information for the developer.

```
#  Text to the right of the "#" is ignored by the Panel Processor.
```

Set the dimensions of the panel. This is normally used by subpanels to define their size. If used, this MUST be the first non blank or comment line in a .panel file. If "dimensions" is not present, the Panel Processor assumes the panel size is 128x64 pixels.

```
dimenstions=X, Y;
"X" value for width of the panel
"Y" value for height of the panel
```

Specify the bitmap to be used as the background for the panel or subpanel. This bitmap MUST have the same dimensions as the panel. This statement is either the first or the second (after "dimensions") statement in the .panel file:

```
background=fileName.bmp;
First Parameter – Filename of the bitmap file
```

Specify the position of the subpanel on the panel. The subpanel cannot overlap any other controls on the panel or extend beyond the boundaries of the OLED.

subpanel= X,Y, W,H;

X, Y – top left hand corner of the subpanel on the OLED
W, H – dimensions of the subpanel

Icon showing the charge level of the Jade Robot's battery. This non-input control displays one of five different bitmaps depending on the charge level of the battery.

batticon=empty.bmp,1quart.bmp,1half.bmp,3quart.bmp,full.bmp, X,Y, W,H;

empty.bmp, 1quart.bmp, 1half.bmp, 3quart.bmp, full.bmp are bitmaps showing the battery state
X, Y – top left hand corner of the control
W, H – dimensions of the control

Icon showing the current Bluetooth Status of the Jade Robot. This non-input control displays one of three different bitmaps depending on the Bluetooth state.

bticon=btOFF.bmp,btON.bmp,btCONNECTED.bmp, X,Y, W,H;

btOFF.bmp, btON.bmp, btCONNECTED.bmp are bitmaps showing Bluetooth state
X, Y – top left hand corner of the control
W, H – dimensions of the control

Icon showing the current USB Connection Status of the Jade Robot. This non-input control displays one of two different bitmaps depending on whether or not there is a USB connection.

usbicon=usbNOCONNECT.bmp,usbCONNECTED.bmp, X,Y, W,H;

usbNOCONNECT.bmp, usbCONNECTED.bmp are bitmaps showing USB Connection state
X, Y – top left hand corner of the control
W, H – dimensions of the control

Application Icon allowing the user to select for a specific action. .

icon=ID, icon.bmp, Value, X,Y, W,H, stop, show, rounded;

ID is the control ID for access using the Syscall APIs
icon.bmp is a bitmap displayed for the control
Value is the numeric value assigned to the control. This "Value" can be used as a variable
X, Y – top left hand corner of the control
W, H – dimensions of the control
stop/nostop – indicate whether or not the user can stop at the control (which makes it an input). "nostop" means it is not an input
show/noshow – indicate whether or not the icon is displayed. If the icon is not displayed, the user cannot stop at it (it is a "nostop")
rounded/square – indicate the outline around the icon. "rounded" is recommended

Application Icon allowing the user to select a specific arrow direction as a command rather than having to click on the red ("Enter") button.
NOTE: Available in Release 41 and later

arrowicon=ID, icon.bmp, Value, X,Y, W,H, stop, show, rounded, north;

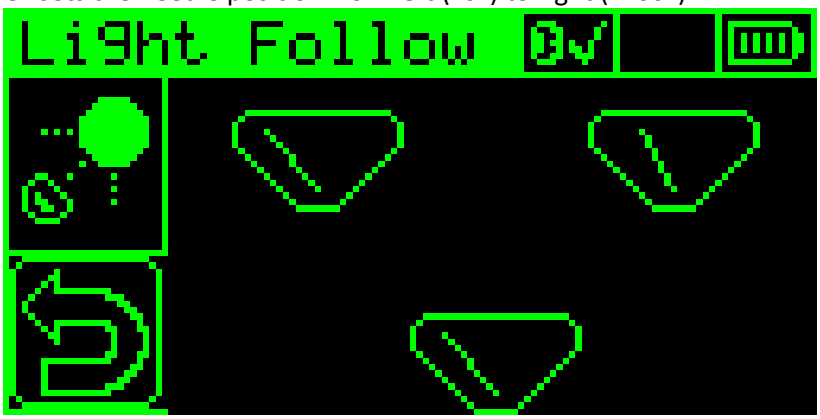
ID is the control ID for access using the Syscall APIs
icon.bmp is a bitmap displayed for the control
Value is the numeric value assigned to the control. This "Value" can be used as a variable
X, Y – top left hand corner of the control
W, H – dimensions of the control
stop/nostop – indicate whether or not the user can stop at the control (which makes it an input). "nostop" means it is not an input
show/noshow – indicate whether or not the icon is displayed. If the icon is not displayed, the user cannot stop at it (it is a "nostop")
rounded/square – indicate the outline around the icon. "rounded" is recommended
north/south/east/west – Indicate the direction of the arrow. When corresponding navigation (white) button is pressed, this will act like an "Enter" command.

Set/Reset and Radio Button for Application allowing the user to select simple options. This is the only case where multiple controls can use the same ID, in this case the buttons act like "radiobuttons" with only one being active at any time with the "Value" indicating which instances (starting from the first) is active. If a "Value" of "0" is written to the radio buttons, none of them are selected. "Value" is initialized to "1".

radiobutton=ID, X,Y, W,H, stop, show;

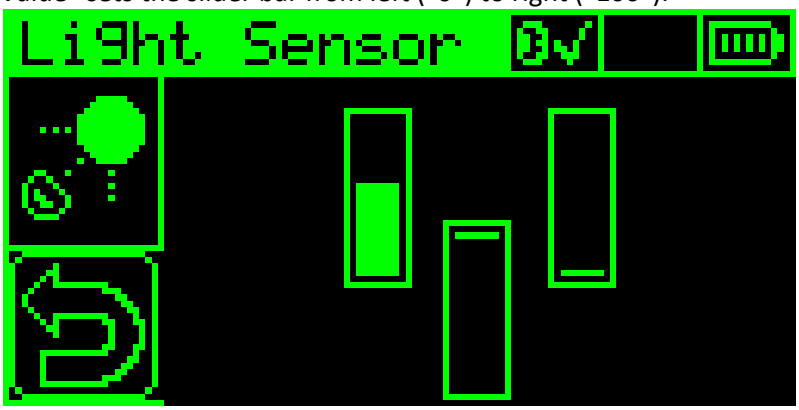
ID is the control ID for access using the Syscall APIs
X, Y – top left hand corner of the control
W, H – dimensions of the control
stop/nostop – indicate whether or not the user can stop at the control (which makes it an input). "nostop" means it is not an input
show/noshow – indicate whether or not the icon is displayed. If the icon is not displayed, the user cannot stop at it (it is a "nostop")

The meter is used to display values (from 0 to 100) in a “Meter” like format (example panel screen shot below). The “Value” sets the needle position from left (“0”) to right (“100”).



meter=ID, Value, X,Y, W,H, show;
ID is the control ID for access using the Syscall APIs
Value is the numeric value used to specify the needle position
X, Y – top left hand corner of the control
W, H – dimensions of the control
stop/nostop – indicate whether or not the user can stop at the control (which makes it an input). “nostop” means it is not an input

Sliders are used to show relative values (from 0 to 100) in a “tape” like format (example panel screen shot below). The “Value” sets the slider bar from left (“0”) to right (“100”).



slider=ID, Value, X,Y, W,H, show, north;
ID is the control ID for access using the Syscall APIs
Value is the numeric value used to specify the needle position
X, Y – top left hand corner of the control
W, H – dimensions of the control
stop/nostop – indicate whether or not the user can stop at the control (which makes it an input). “nostop” means it is not an input
north/south/east/west – indicate the direction the slider is to “grow” in.
NOTE: the Width and Height parameters designed for “north” and “south” and are swapped if the “east” or “west” orientation is used.

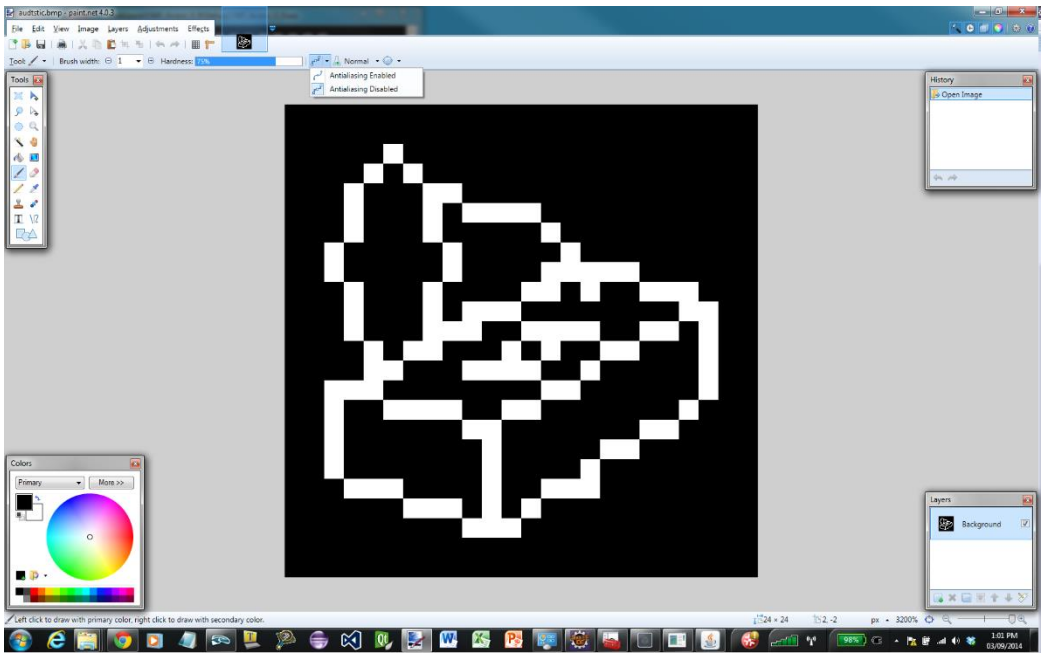
Text can be placed on the panel as a navigatable control as well. ONLY characters from " " (0x20) to "~" (0x7E) can be displayed on the initial text.

text=ID, "initString",Length, Val, X,Y, W,H, 5x7, left, stop, show, normal;

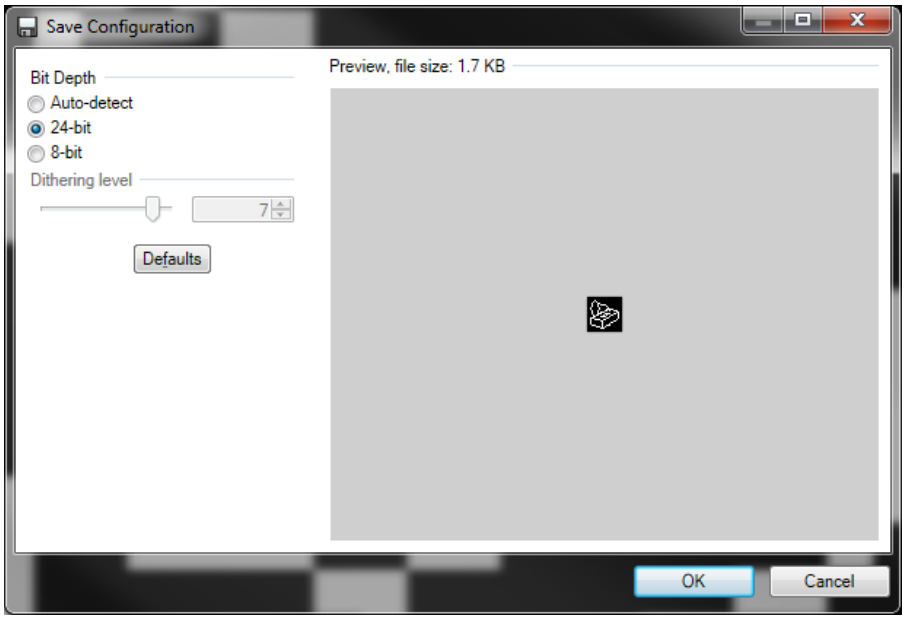
ID is the control ID for access using the Syscall APIs
"initString" – specify the initial string to be displayed. NOTE: this cannot be longer than the number of characters specified in "Length".
Length – specify the maximum number of characters in the text string
Val is the numeric value assigned to the control. This "Value" can be used as a variable
X, Y – top left hand corner of the control
W, H – dimensions of the control
5x7/3x5 – Text sizes. "5x7" for most test, "3x5" for small text. **NOTE:** "3x5" recommended for text strings which just contains numbers as some characters are difficult to decipher
left/center – specifies how the text is to be justified
stop/nostop – indicate whether or not the user can stop at the control (which makes it an input). "nostop" means it is not an input
normal/reverse – specifies whether or not the text is to be green on a black background ("normal") or black on a green background ("reverse").

Bitmap Format for Use with the Panel Driver

Standard Microsoft Windows format bitmaps are used to provide icons and backgrounds in a panel file. The bitmap format is 24 bits with only two colors defined – white and black. The tool normally used for creating the bitmaps is “Paint.Net” (http://www.getpaint.net/) with the “Antialiasing Disabled” as shown in the screen shot below:



After the image is made, it is saved as a bitmap (.bmp) with 24 colors as shown below:



Bitmaps MUST be have the same dimensions as the icon or background they are being loaded into. If the bitmaps is going to be used as an icon, it is recommend that two black pixels surround the image for the highlighting.

Syscall APIs Used with the Panel Driver

The following Syscall APIs are used to control, update and access the Panel Driver and the data on the display.

getpanelstatus – 5	
Return a numeric string with the Active Panel Driver Status: "0" – Ready, no panel loaded "1" – Error encountered with load or execution "2" – Panel is active "3" – "Enter"/Red button pressed and panel driver has stopped **NOTE:** It is recommended that getActivePanelStatus is used instead of this API due to its more complete information.	
`int panelStatus = stoi(syscall(getpanelstatus, ""));`	
Passed Argument: None	**Returned Data:** Numeric String, Defined above
Error: None	

loadrunpanelfile – 29	
Load panel file and start executing	
`syscall(loadrunpanelfile, "anypanel.p"`	
Passed Argument: filename of panel (".p") file	**Returned Data:** none
Error: File not found or invalid	

getactivepanelstatus – 30	
Return the active panel information in the format "Panel Driver Status:Current control". The return values are: "READY" – No panel loaded "ERROR:control" – Error encountered with control "RUN:control" – Panel Driver is active with the specified control. If no controls on the panel, nothing after the colon(:) "END:control" – User has pressed "Enter"/Red button and the Panel Driver has stopped at the specified control. If no controls on the panel, nothing after the colon(:)	
`str panelstatus = syscall(getactivepanelstatus, "");`	
Passed Argument: None	**Returned Data:** String in format shown above
Error: None	

loadpanelfile – 92	
Load the specified .panel (.p) file into the panel driver but do not execute it.	
`syscall(loadpanelfile, "_demo.p");`	
Passed Argument: .Panel (.p) file	**Returned Data:** None
Error: .Panel file not found/bmp (.b) file specified in .panel file not found	

runpanelfile – 93	
Start executing the loaded panel file. This command can be used to re-run the current panel file loaded in the Jade Robot (ie after processing an "Enter"/Red button press).	
`syscall(runpanelfile, "");`	
Passed Argument: None	**Returned Data:** None
Error: No panel file loaded	

stoppanelfile – 94	
Stop execution of the current panel file. For the user, the only indication they will have the panel file is not running is that the display will not change due to button presses.	
`syscall(stoppanelfile, "");`	
Passed Argument: None	**Returned Data:** None
Error: No panel file loaded/active	

setpanelicon – 95	
Change a specific icon control on the panel. The format for the data passed is "i:fileName.b" where "i" is the control name of the icon to be changed and "fileName.b" is the new icon filename.	
`syscall(setpanelicon, "iconControl:newBMP.b");`	
Passed Argument: String as described above	**Returned Data:** None
Error: Panel not loaded/Control not found/Wrong type of control/Invalid bmp file	

getpanelicon – 96	
Return the current bitmap (.bmp/.b) being used for an icon.	
`str currentIcon = syscall(getpanelicon, "iconControl");`	
Passed Argument: String with icon label	**Returned Data:** bitmap being displayed
Error: Panel not loaded/Control not found/Invalid control type	

setpaneltext – 97	
Set the text value for a text control. The passed information consists of "c:newText" where "c" is text control and "newText" is the text string. Valid text consists of characters from ASCII space (' ' 0x20) to '~' (0x7E).	
`syscall(setpaneltext, "textControl:New Text");`	
Passed Argument: String as described above	**Returned Data:** None
Error: Panel not loaded/Control not found/Invalid control type	

getpaneltext – 98	
Return the current text for the specified control.	
`str textValue = syscall(getpaneltext, "");`	
Passed Argument: None	**Returned Data:** Text value for Control
Error: Panel not loaded/Control not found/Invalid control type	

setpanelvalue – 99	
Each control has a value associated with it for information storage, to display a specific value on the control (as in the case of dials and sliders) or indicate that a specific control has been selected (radio buttons). The value is specified using the format "c:#" where "c" is the control name and "#" is the value (ranging from "0" to "100" or the maximum number of radio buttons).	
`syscall(setpanelvalue, "control:55");`	
Passed Argument: String as described above	**Returned Data:** None
Error: Panel not loaded/Control not found/Invalid control type	

getpanelvalue – 100	
Return the value for the specified panel control.	
`str controlValue = syscall(getpanelvalue, "control");`	
Passed Argument: Control name	**Returned Data:** Control Value
Error: Panel not loaded/Control not found	

showpanelcontrol – 101	
Change the operation parameters of the control so it will be displayed on the Jade Robot Display.	
`syscall(showpanelcontrol, "control");`	
Passed Argument: Control name	**Returned Data:** None
Error: Panel not loaded/Control not found	

noshowpanelcontrol – 102	
Change the operation parameters of the control so it will NOT be displayed on the Jade Robot Display.	
`syscall(noshowpanelcontrol, "control");`	
Passed Argument: Control name	**Returned Data:** None
Error: Panel not loaded/Control not found	

stoppanelcontrol – 103	
Change the control parameters so that the user can navigate to the control.	
`Syscall(stoppanelcontrol, "control");`	
Passed Argument: Control name	**Returned Data:** None
Error: Panel not loaded/Control not found	

nostoppanelcontrol – 104	
Change the control parameters so the user can NOT navigate to the control.	
`syscall(nostoppanelcontrol, "Control");`	
Passed Argument: Control name	**Returned Data:** None
Error: Panel not loaded/Control not found	

loadsubpanelfile – 105	
Load a sub-panel into the current panel. A sub-panel file has the same extension (.p) as a panel file and can be used as a panel file, but when used as a sub-panel file, it's dimensions must match those of the primary panel file.	
`syscall(loadsubpanelfile, "subPanel.p");`	
Passed Argument: sub-panel filename	**Returned Data:** None
Error: Panel not loaded/Sub-panel file not found/Sub-Panel dimensions are invalid for panel	

setactivepanelcontrol – 106	
Set the currently active panel control. When the panel driver first starts up, the first control in the .panel file is selected as the active panel control – this API allows the program to select which one is active.	
`syscall(setactivepanelcontrol, "control");`	
Passed Argument: Control name	**Returned Data:** None
Error: Control not found/Control not "stoppable"	

Example Panels and Script Code

To demonstrate how the panel driver is used, the following example uses the code from the “Object Avoidance” application.

The main panel, which contains the common and outline information for the UI is called “_exec2.panel” and is:

```
#  _exec2 - Subpanel equipped Panel

background = _2exec.bmp;

bticon     = _btoff.bmp,_bton.bmp,_btcon.bmp, 82,1, 14,9;
usbicon    = _usbnc.bmp,_usbcon.bmp, 97,1, 15,9;
batticon   = _b0.bmp,_b1.bmp,_b2.bmp,_b3.bmp,_b4.bmp, 113,1, 14,9;

icon       = tools, _2tools.bmp, 0, 1,12, 24,24, nostop,show,rounded;
icon       = return, _3return.bmp, 0, 1,39, 24,24, stop,show,rounded;

subpanel   = 26,11, 102,53;

text       = pgmname, "0123456789012",13, 0, 1,0, 5x7,left,nostop,
show,reverse;
```

The subpanel added to “_exec2.panel” is called “objAv05.panel” and consists only of sliders:

```
#  objAv05 - Add sliders for Object Avoid

dimensions = 102, 53;
background = movBACKG.bmp;

slider     = leftFront,        0,    1, 34,  10, 28,   show,  west;
slider     = leftSide,         0,   29,  5,  10, 28,   show, north;
slider     = rightFront,       0,   62,  5,  10, 28,   show, north;
slider     = rightSide,        0,   73, 34,  10, 28,   show,  east;
slider     = rearMiddle,       0,   45, 23,  10, 28,   show, south;
```

Finally, the code which loads the panel, followed by the sub panel and then polls, waiting for the user to select a control (of which there is only one) is:

```
    syscall(loadpanelfile, "_exec2.p");
    syscall(loadsubpanelfile, "objAv05.p");
    syscall(setpanelicon, "tools:_2objav.b");
    syscall(setpaneltext, "pgmname:Object Avoid");
    syscall(runpanelfile, "");

    for (; "RUN:" == syscall(tostring, "4:" +
            syscall(getactivepanelstatus, ""));) {
        //  Program runs from here...
```

The recommended approach to loading and using a panel is:

1. Using the “loadpanelfile” Syscall API, load the main panel.
2. If there is a subpanel to be used, load it using the “loadsubpanelfile” Syscall API.
3. Load any application specific bitmaps.
4. Set any application specific text.
5. Set the values for any meters or sliders.
6. Start the panel executing with the “runpanelfile” Syscall API.
   :
7. Poll the status of the panel using the “getactivepanelstatus” Syscall API which will indicate whether or not the “Enter” (red) button has been pressed and what is the currently selected control.

Finally, it should be pointed out that the icon bitmap, text string, meter value and slider value can all be updated while the panel is running.

Supporting Documents

Jade Robot™ Support Software Installation and Introduction - TBD
Jade Robot™ Script Language Outline – available at http://www.mimetics.ca/knowledge-base-2/

Glossary

ASCII – Standard 8 bit character set. See http://en.wikipedia.org/wiki/ASCII

Document Updates

Date	Changes	Author
2014.09.03	Initial Document Version	Myke Predko
2014.09.11	Added “arrowicon” control to Panel Driver. Noticed that the “icon” definition was repeated in original document and used second for the “arrowicon” definition.	Myke Predko