19. LOOPS WINDOWS

This chapter describes the ways to manipulate LOOPS windows.

19.1 The Class Window

The class **Window** is the LOOPS interface to the Medley environment window system, which is used by LOOPS browsers and inspectors.

Window [Class]

Description: This class provides a mechanism for manipulating Lisp windows through an object-oriented interface. See Section 19.4, "Mouse and Menu Functionality," for a discussion of how menus work with LOOPS Windows.

When an instance of a LOOPS window is created, it has an instance variable that points to a Lisp window. This Lisp window is initialized with various window properties:

- The property **LoopsWindow** points to the window object.
- The property **RIGHTBUTTONFN** is set to **WindowRightButtonFn**.
- The property **BUTTONEVENTFN** is set to **WindowButtonEventFn**.
- The property **AFTERMOVEFN** is set to **WindowAfterMoveFn**.
- The property **RESHAPEFN** is set to **WindowReshapeFn**.

MetaClass: Class

Supers: Object

Class Variables:

TitleItems A list that defines the menu that will appear when the left or middle mouse button is pressed and the cursor is in the title bar of the window. The default value is NIL.

LeftButtonItems
A list that defines the menu for the left button in the main window. The default value is ((Update ...)).

ShiftLeftButtonItems
A list that defines the menu for the left button in the main window when the **Meta** key is down. The default value is NIL.

MiddleButtonItems
A list that defines the menu for the middle button in the main window. The default value is NIL.

ShiftMiddleButtonItems
A list that defines the menu for the middle button in the main window when the **Meta** key is down. The default value is NIL.

	RightButtonItems	A list that defines the menu for the right button in the main window. The default value is ((Close ...)).
Instance Variables:	**left**	The location of the left side of the outside of the window in screen coordinates. The default value is NIL.
	bottom	The location of the bottom side of the outside of the window in screen coordinates. The default value is NIL.
	width	The outside width of the window. The default value is 12.
	height	The outside height of the window. The default value is 12.
	window	An active value that contains the Lisp window. The default value is #,($AV LispWindowAV ...).
	title	The title of the window. Default Value: NIL.
	menus	Also has the properties **Title**, **LeftButtonItems**, **MiddleButtonItems**, and **TitleItems**. These properties are caches for menus only if the value of the instance variable is T. Default Value: T.

19.2 Basic Window Methods

This section describes the basic methods to operate on windows.

Name	Type	Description
AfterMove	Method	Updates the instance variables left and bottom.
AfterReshape	Method	Updates the instance variables left, bottom, width, and height.
Blink	Method	Causes the window to blink.
Bury	Method	Buries the window.
Clear	Method	Clears the window.
Close	Method	Closes the window.
CursorInside?	Method	Determines if the cursor is inside a window.
Destroy	Method	Destroys the window instance.
GetProp	Method	Gets a property from a specified window.
Hardcopy	Method	Makes a hardcopy on the default device.
HardcopyToFile	Method	Makes a hardcopy on a file.
HardcopyToPrinter	Method	Makes a hardcopy to a printer.
Invert	Method	Inverts the window; that is, reverses its black-white pattern.
Move	Method	Moves the window.

MousePackage	Method	Returns a package (defined to return the INTERLISP package).
MouseReadtable	Method	Returns a readtable (defined to return the INTERLISP readtable).
Open	Method	Opens the window.
Paint	Method	Calls **PAINTW** on the window.
ScrollWindow	Method	Scrolls the window.
SetProp	Method	Sets the property in the specified window.
Shape	Method	Reshapes the window.
Shape?	Method	Returns the current region for the window.
Shrink	Method	Shrinks the window to an icon.
Snap	Method	Takes a snapshot of the screen.
ToTop	Method	Opens the window and brings it to the top.
Update	Method	Makes the window consistent with the instance variables.
WindowAfterMoveFn	Function	Sends the message **AfterMove**.
WindowReshapeFn	Function	Sends the message **AfterReshape**.

(← *self* **AfterMove**) [Method of Window]

Purpose/Behavior: Updates the instance variables **left** and **bottom** of *self*.

Arguments: *self* An instance of a window.

Returns: Used for side effect only.

Categories: Window

(← *self* **AfterReshape** *oldBitmapImage oldRegion oldScreenRegion*) [Method of Window]

Purpose/Behavior: Updates the instance variables **left**, **bottom**, **width**, and **height** of *self*. Calls **RESHAPEBYREPAINTFN**; see the *Interlisp-D Reference Manual*.

Arguments: *self* An instance of a window.

Returns: Used for side effect only.

Categories: Window

(← *self* **Blink** *numBlinks*) [Method of Window]

Purpose/Behavior: Inverts the window; that is, reverses its black-white pattern, and then returns to normal *numBlinks* times.

Arguments: *self* Pointer to a window instance.

numBlinks Number of times for window to blink.

Returns: NIL

Categories: Window

Example: The command

```
(← self Blink 5)
```

sends a message to *self* to blink five times.

(← *self* **Bury**) [Method of Window]

Purpose/Behavior: Calls **BURYW** to bury the specified window.

Arguments: *self* Pointer to a window instance.

Returns: The LOOPS window.

Categories: Window

(← *self* **Clear**) [Method of Window]

Purpose/Behavior: Calls **CLEARW** to clear the specified window.

Arguments: *self* Pointer to a window instance.

Returns: NIL

Categories: Window

Specializations: LatticeBrowser

(← *self* **Close**) [Method of Window]

Purpose/Behavior: Closes the specified window and prompt window, if there is one.

Arguments: *self* Pointer to a window instance.

Returns: NIL

Categories: Window

(← *self* **CursorInside?**) [Method of Window]

Purpose/Behavior: Determines if the cursor is inside the window.

Arguments: *self* Pointer to a LOOPS window.

Returns: Returns T if the cursor is inside the window, otherwise returns NIL.

Categories: Window

(← *self* **Destroy**) [Method of Window]

Purpose/Behavior: Destroys the calling instance, removes all **ButtonFns**, and closes the window.

Arguments: *self* Pointer to a window instance.

Returns: NIL

Categories: Object

Specializes: Object

(← *self* **GetProp** *prop*) [Method of Window]

Purpose/Behavior:	Gets a property from a window.
Arguments:	*self* Pointer to a window instance.
	prop Property to get.
Returns:	The value of the specified property, if it exists; else NIL.
Categories:	Window
Example:	To determine the value of the window property **BUTTONEVENTFN**, enter

```
(← ($ window)  GetProp  'BUTTONEVENTFN)
```

(← *self* **Hardcopy**) [Method of Window]

Purpose/Behavior:	Makes a hardcopy of the window on the default printer.
Arguments:	*self* Pointer to a window instance.
Categories:	Window

(← *self* **HardcopyToFile**) [Method of Window]

Purpose/Behavior:	Makes a hardcopy of the window to a file. You are prompted for the file name.
Arguments:	*self* Pointer to a window instance.
Categories:	Window

(← *self* **HardcopyToPrinter**) [Method of Window]

Purpose/Behavior:	Makes a hardcopy of the window on a printer. You are prompted for the name of the printer.
Arguments:	*self* Pointer to a window instance.
Categories:	Window

(← *self* **Invert**) [Method of Window]

Purpose/Behavior:	Inverts the window; that is, reverses its black-white pattern.
Arguments:	*self* Pointer to a window instance.
Returns:	T if successful.
Categories:	Window
Specializations:	NonRectangularWindow

(← *self* **Move** *xOrPos* *y*) [Method of Window]

Purpose/Behavior:	Moves the specified window. If no arguments are supplied, you will be prompted to position the window.
Arguments:	*self* Pointer to a window instance.

	x	New **left** in screen coordinates or a new position for **left** and **bottom**. If *x* is a position, *y* is ignored.
	y	New **bottom** in screen coordinates.
Returns:	(*x* . *y*)	
Categories:	Window	

Example: The command

```
(← ($ window) Move)
```

causes window to become attached to the cursor, prompting for new location.

The command

```
(← self Move 200 100)
```

moves the lower left corner of the window to (200 . 100).

(←*self* **MousePackage**) [Method of Window]

Purpose/Behavior: Returns the package used during mouse interactions with the window *self*. (LOOPS now uses the INTERLISP package exclusively.) The **MousePackage** method protects LOOPS windows from the new packages. To remove this protection, specialize this method to return *PACKAGE*.

Arguments: *self* An instance of a window.

Returns: The INTERLISP package.

(←*self* **MouseReadtable**) [Method of Window]

Purpose/Behavior: Returns the readtable used during mouse interactions with the window *self*. Medley now has many different readtables; some readtables do not work well with LOOPS. (The Common Lisp readtables are not case-sensitive.) This method protects LOOPS windows from the new readtables. To remove this protection, specialize this method to return *READTABLE*.

Arguments: *self* An instance of a window.

Returns: The INTERLISP readtable.

(← *self* **Open**) [Method of Window]

Purpose/Behavior: Opens the specified window instance.

Arguments: *self* Pointer to a window instance.

Returns: NIL

Categories: Window

(← *self* **Paint**) [Method of Window]

Purpose/Behavior: Calls **PAINTW** on the specified window. You are prompted for instructions in the prompt window.

Arguments: *self* Pointer to a window instance.

Returns: NIL

Categories: Window

(← *self* **ScrollWindow** *dspX dspY windowX windowY*) [Method of Window]

Purpose/Behavior:	Scrolls the window to move the point *dspX, dspY* to *windowX, windowY*. If *windowX* and *windowY* are NIL, the default is to scroll so that the point *dspx, dspy* appears in the lower left corner of the window. Any of the arguments can be **FIXP** or **FLOATP**. If the value is **FIXP**, then it is treated as an absolute coordinate. If the value is **FLOATP**, then it is treated as a relative position.	
Arguments:	*self*	Pointer to a window instance.
	dspX	The x point in the given window to move; x is in window coordinates if **FIXP**. If **FLOATP**, the value to move is based upon the width of the **EXTENT** property of the window; see the *Interlisp-D Reference Manual*.
	dspY	The y point in the given window to move; y is in window coordinates if **FIXP**. If **FLOATP**, the value to move is based upon the height of the **EXTENT** property of the window; see the *Interlisp-D Reference Manual*.
	windowX	The x point to scroll to in window coordinates if **FIXP**. If **FLOATP**, the value to move is based upon the width of the window.
	windowY	The x point to scroll to in window coordinates if **FIXP**. If **FLOATP**, the value to move is based upon the height of the window.
Returns:	The lower left corner of the new **DSPCLIPPINGREGION**; see the *Interlisp-D Reference Manual*.	
Categories:	Window	

(← *self* **SetProp** *prop value*) [Method of Window]

Purpose/Behavior:	Sets the Interlisp window property of the specified LOOPS window, passing its *prop* and *value* arguments through Interlisp function **WINDOWPROP**.	
Arguments:	*self*	Pointer to a window instance.
	prop	Property to set.
	value	New value for property.
Returns:	Previous value of *prop* if it existed; else NIL.	
Categories:	Window	

(← *self* **Shape** *newRegion noUpdateFlg*) [Method of Window]

Purpose/Behavior:	Reshapes the specified window. If *newRegion* is not specified, you are prompted to reshape the window with the cursor.	
Arguments:	*self*	Pointer to a window instance.
	newRegion	A list specifying the new outer dimensions; the format for the list is (left bottom height width).
	noUpdateFlg	If NIL, reshapes the window.
Returns:	A list specifying the new region.	
Categories:	Window	

Specializations:	NonRectangularWindow
Example:	The command

```
(← ($ window1) Shape '(100 200 300 400))
```

returns

```
(100 200 300 400)
```

(← *self* **Shape?**) [Method of Window]

Purpose/Behavior:	Returns the current region for the window.
Arguments:	*self* Pointer to a window instance.
Returns:	A list specifying outer dimensions of the window.
Categories:	Window
Example:	The command

```
(← ($ window1) Shape?)
```

returns

```
(100 200 300 400)
```

(← *self* **Shrink** *toWhat iconPos expandFn*) [Method of Window]

Purpose/Behavior:	Shrinks the window to a given icon.	
Arguments:	*self*	Pointer to a window instance.
	toWhat	The icon to shrink to; if NIL, an icon is created.
	iconPos	Position of icon on screen.
	expandFn	Function to be called on expansion**.**
Returns:	The icon.	
Categories:	Window	
Specializations:	LatticeBrowser	

(← *self* **Snap**) [Method of Window]

Purpose/Behavior:	Calls **SnapW** to take a snapshot of the window.
Arguments:	*self* Pointer to a window instance.
Returns:	The window.
Categories:	Window

(← *self* **ToTop**) [Method of Window]

Purpose/Behavior:	Opens the window and brings it to the top of the screen.
Arguments:	*self* Pointer to a window instance.
Returns:	The window.

Categories: Window

(← *self* **Update**) [Method of Window]

Purpose/Behavior: Makes the window consistent with the instance variables.

Arguments: *self* Pointer to a window instance.

Returns: NIL

Categories: Window

Specializations: NonRectangularWindow

(**WindowAfterMoveFn** *window*) [Function]

Purpose/Behavior: This function is installed as the **AFTERMOVEFN** property of the Lisp window pointed to by a window object. This function extracts the window object from the property **LoopsWindow** and sends it the message **AfterMove.**

This **AFTERMOVEFN** is installed automatically by the system.

Arguments: *window* The window just moved.

Returns: Used for side effect only.

(**WindowShapeFn** *window oldBitmapImage oldRegion*) [Function]

Purpose/Behavior: This function is installed as the **RESHAPEFN** property of the Lisp window pointed to by a window. This function extracts the window object from the property **LoopsWindow** and sends it the message **AfterReshape** with the arguments *oldBitmapImage oldRegion*.

This **RESHAPEFN** is installed automatically by the system.

Arguments: *window* The window just reshaped.

oldBitmapImage See the *Lisp Release Notes* and the *Interlisp-D Reference Manual* for a discussion of window **ReShapeFns**.

oldRegion See the *Lisp Release Notes* and the *Interlisp-D Reference Manual* for a discussion of window **ReShapeFns**.

Returns: Used for side effect only.

19.3 PROMPT WINDOWS

19.3 Prompt Windows

Prompt windows are windows attached to other windows and are used for displaying messages and for getting input. In LOOPS, these operate similarly to prompt windows in Lisp. Prompt windows are not instances of the class Window; they are only instances of the Interlisp data type **Window**.

The following table lists the methods and functions described in this section.

Name	Type	Description
Clear PromptWindow	Method	Clears the prompt window.
ClosePromptWindow	Method	Closes the prompt window.
GetPromptWindow	Method	Associates a prompt window with a LOOPS window.
PromptEval	Function	Prompts for, reads, and evaluates an expression.
PromptForList	Method	Prompts for a list of items.
PromptForString	Method	Prompts for a string.
PromptForWord	Method	Prompts for a word.
PromptPrint	Method	Prints a message in the prompt window.
PromptRead	Function	Prompts for and reads data.
NiceMenu	Function	Creates a menu.
SelectFile	Lambda NoSpread	Prompts for a file name.

Note: The methods **PromptForList**, **PromptForString**, and **PromptForWord**, as well as the functions **PromptRead** and **PromptEval** when called with a prompt window for a LOOPS window, all disable normal mouse button events in the prompting browser and will not allow it to close until the prompt is completed.

(← *self* **ClearPromptWindow**) [Method of Window]

Purpose/Behavior: Clears the prompt window associated with the window *self*.

Arguments: *self* Evaluates to a window instance.

Returns: NIL

Categories: Window

(← *self* **ClosePromptWindow**) [Method of Window]

Purpose/Behavior: Closes the prompt window associated with the window *self*.

Arguments: *self* Evaluates to a window instance.

Returns: The symbol **CLOSED** if a prompt window existed; else NIL.

Categories: Window

(← *self* **GetPromptWindow** *lines fontDef*) [Method of Window]

Purpose/Behavior: Gets a prompt window for window *self*. If one exists, it is returned; else a prompt window is created.

Arguments:

self	Pointer to a window instance.
lines	Number of lines in window; default is 2.
fontDef	Font used in the window; if NIL, this defaults to DEFAULTFONT.

Returns: Pointer to a prompt window.

Categories: Window

(**PromptEval** *promptString window sameLine?*) [Function]

Purpose: Prompts for, reads, and evaluates an expression.

Behavior: Temporarily moves the TTYDISPLAYSTREAM to *window*, if *window* is non-NIL, else to the system prompt window.

The *promptString* is printed followed by a carriage return and the string "The expression read will be EVALuated."

The prompt "> " is printed on the same line as the above if *sameLine?* is non-NIL, else it is printed on a new line. Data entered by the user is evaluated and returned. **LISPX** and **LISPXREAD** are used so that the entered data is placed on the **LISPX** history list. (See the *Lisp Release Notes* and the *Interlisp-D Reference Manual*).

Note: When called with a prompt window for a LOOPS window, **PromptEval** disables normal mouse button events in the prompting browser and will not allow it to close until the prompt is completed.

Arguments:

promptString	A string to be printed.
window	A window where the prompting and reading should occur. Defaults to the system prompt window.
sameLine?	If non-NIL, the data is read from the same line as the string "The expression read will be EVALuated."

Returns: The data entered by the user after it has been evaluated.

Example: The command

```
26←(← ($ Window) New (PromptEval "Specify new window for object name."))
```

causes the following to appear in the Prompt Window:

```
Specify new window for object name.
The expression read will be EVALuated.

>
```

Entering

```
'NewWindow
```

after the > causes the following return in the Executive Window.

```
#,($& Window (NEW0.1Y%:.;h.eN6 . 501))
```

(← *self* **PromptForList** *promptStr initialString*) [Method of Window]

Purpose/Behavior: Prompts you in prompt window for a list of symbols. If prompt window does not exist, one is created. Input is terminated by a carriage return.

TTYIN is used for editing the user's input; see the *Interlisp-D Reference Manual*.

Note: **PromptForList** disables normal mouse button events in the prompting browser and will not allow it to close until the prompt is completed.

Arguments: *self* Pointer to a window instance.

	promptStr	Displayed in prompt window.
	initialString	Can be used as the default or the first item of the list.

Returns: The list of words entered in prompt window.

Categories: Window

Example: If ($ Window1) is a window, then the command

```
27←(← ($ Window1) PromptForList  "ENTER THE CODES ")
```

causes the prompt ENTER THE CODES to be displayed in an attached prompt window. The system waits for user input.

(← *self* **PromptForString** *promptStr initialStr*) [Method of Window]

Purpose/Behavior: Prompts you in prompt window for a string. If a prompt window does not exist, one is created. Input is terminated by a carriage return.

TTYIN is used for editing the user's input; see the *Interlisp-D Reference Manual*.

Note: **PromptForString** disables normal mouse button events in the prompting browser and will not allow it to close until the prompt is completed.

Arguments:	*self*	Pointer to a window instance.
	promptStr	Displayed in prompt window.
	initialStr	Can be used as the default or the prefix to the string.

Returns: The string entered in prompt window.

Categories: Window

Example: If ($ Window1) is a window, then the command

```
28←(← ($ Window1) PromptForString  "ENTER THE CODES ")
```

causes the prompt ENTER THE CODES to be displayed in an attached prompt window . The system waits for user input.

(← *self* **PromptForWord** *promptStr initialWord*) [Method of Window]

Purpose/Behavior: Returns (CAR (← *self* **PromptForList** *promptStr initialWord*))

Arguments:	*self*	Evaluates to a window instance.
	promptStr	Displayed in prompt window.
	initialWord	Can be used as the default.

Returns: See Behavior.

Categories: Window

Example: If ($ Window1) is a window, then the command

```
29←(← ($ Window1)  PromptForWord  "NEW WORD " )
```

prompts you with NEW WORD in an attached prompt window.

(← *self* **PromptPrint** *msg*) [Method of Window]

Purpose/Behavior: Displays a message in the prompt window associated with the specified window instance. Creates the prompt window if it does not exist.

Arguments:

self	Evaluates to a window instance.
msg	Message displayed.

Returns: The message printed.

Categories: Window

(**PromptRead** *promptString window sameLine?*) [Function]

Purpose: Prompts for and reads data.

Behavior: Temporarily moves the **TTYDISPLAYSTREAM** to *window*, if *window* is non-NIL, else to the system prompt window.

The *promptString* is printed. The prompt "> " is printed on the same line as the above if *sameLine?* is non-NIL, else it is printed on a new line. Data that you entered is read and returned.

This contrasts with **PromptEval** in that the entered data is not placed on the **LISPX** history list (see the *Lisp Release Notes* and the *Interlisp-D Reference Manual*).

Note: When called with a prompt window for a LOOPS window **PromptRead** disables normal mouse button events in the prompting browser and will not allow it to close until the prompt is completed.

Arguments:

promptString	A string to be printed.
window	A window where the prompting and reading should occur. Defaults to the system prompt window.
sameLine?	If non-NIL, the data is read from the same line as the *promptString*.

Returns: The data entered by the user.

(**NiceMenu** *items title*) [Function]

Purpose: Provides an interface to create a menu and displays the menu.

Behavior: Varies according to the arguments.

- If *items* is NIL, prints "No items for *title*" in the system prompt window and returns NIL.
- If *items* is non-NIL, this builds a menu with the **TITLE** *title*, with the **ITEMS** *items*, and with **CHANGEOFFSETFLG** set to T (see the *Interlisp-D Reference Manual*). If the length of *items* is more than 35, the menu has multiple columns.

Arguments:

items	A form that can be placed in the **ITEMS** field of a menu.
title	A value that will be placed in the **TITLE** field of a menu**.**

Returns: Value depends on the arguments; see Behavior.

(**SelectFile** *prompts*) [Lambda NoSpread Function]

Purpose:	Prompts you for a file name.

Behavior: Takes on unlimited number of arguments and will **PROMPTPRINT** all the arguments.

Builds a menu with the items ***newFile*** and the files found on the variable **FILELST**.

If you select one of the files, that is returned.

If you select ***newFile***, you are prompted to enter a file name. An empty filecoms is built for that file name, and the file name is returned.

newFile has three subitems:

- ***newFile***

  See Behavior.

- ***loadFile***

  You are prompted to enter a file name. A search is performed to try to find and load the compiled file. If that is not found, an attempt is made to load the source file. Returns NIL if the file is not found.

- ***hiddenFile***

A menu is displayed containing files that are on the variable **LOADEDFILELST** but not on **FILELST**.

Arguments:	*prompts*	A number of expressions to be printed in the system prompt window.

Returns: Value depends on the arguments; see Behavior.

19.4   Mouse and Menu Functionality

When a LOOPS window is instantiated, its instance variable **window** points to an instance of a Lisp window. This window has several properties set, among which are the following that are described in this section:

- The property **RIGHTBUTTONFN** that is set to **WindowRightButtonFn**.
- The property **BUTTONEVENTFN** that is set to **WindowButtonEventFn**.

This section will also explain the functionality of the above and how menus associated with LOOPS windows operate. For more information on Medley windows, see the *Lisp Release Notes* and the *Interlisp-D Reference Manual*.

Name	Type	Description
ButtonEventFn	Method	Sends either the message **TitleSelection**, **LeftSelection**, or **MiddleSelection**.
ClearMenuCache	Method	Deletes menus saved on the menus field of a browser.
ItemMenu	Method	Creates a simple one-level menu.
LeftSelection	Method	Triggers functionality when the cursor is in a window and the left button is pressed.

MiddleSelection	Method	Triggers functionality when the cursor is in a window and the middle button is pressed.
RightButtonFn	Method	Sends the message **RightSelection.**
RightSelection	Method	Triggers functionality when the cursor is in a window and the right button is pressed.
TitleSelection	Method	Triggers functionality when the cursor is in a window's title bar and the left or middle button is pressed.
WhenMenuItemHeld	Method	Displays in the prompt window what happens when option is selected.
WindowButtonEventFn	Function	Invokes the method **ButtonEventFn**.
WindowRightButtonFn	Function	Invokes the method **RightButtonFn.**

(← *self* **ButtonEventFn**) [Method of Window]

Purpose/Behavior: If the cursor is not inside of the window pointed to by *self*, this sends the message **TitleSelection** to *self*.

If the left mouse button is pressed, this sends the message **LeftSelection** to *self*.

If the left middle button is pressed, this sends the message **MiddleSelection** to *self*.

Arguments: *self* An instance of a window.

Returns: Used for side effect only.

Categories: Window

(← *self* **ClearMenuCache**) [Method of Window]

Purpose/Behavior: Deletes menus saved in any properties of the instance variable **menus** of a window. Use this method if you ever change the class variables describing a menu, and you want the new menu to take effect.

Arguments: *self* Pointer to a window instance.

Returns: *self*

Categories: Window

(← *self* **ItemMenu** *items title*) [Method of Window]

Purpose/Behavior: Creates a simple one-level menu guaranteed not to be more than 750 bits high. A large number of menu options will cause a multiple column menu to be formed.

Arguments: *self* Pointer to a window instance.

items The value of this is passed to the **ITEMS** field when the menu is created.

title The title for the menu's window.

Returns: A menu.

Categories: Window

Example: The command

```
32←(← (←New ($ Window)) ItemMenu '(a b c))
```

will create a menu with the three options.

(← *self* **LeftSelection**) [Method of Window]

Purpose/Behavior: Invokes a number of internal methods of **Window**. A menu will pop up. The options in the menu will be defined by the class variable **LeftButtonItems** (or **ShiftLeftButtonItems** if the **Meta** key is also pressed). If an option is selected from the menu, a message will be sent to *self* with a selector as specified by the chosen menu option.

Arguments: *self* Pointer to a window instance.

Returns: Used for side effect only.

Categories: Window

Specializations: LatticeBrowser

(← *self* **MiddleSelection**) [Method of Window]

Purpose/Behavior: Invokes a number of internal methods of **Window**. A menu will pop up. The options in the menu will be defined by the class variable **MiddleButtonItems** (or **ShiftMiddleButtonItems** if the **Meta** key is also pressed). If an option is selected from the menu, a message will be sent to *self* with a selector as specified by the chosen menu option.

Arguments: *self* Pointer to a window instance.

Categories: Window

Specializations: LatticeBrowser

(← *self* **RightSelection**) [Method of Window]

Purpose/Behavior: Invokes a number of internal methods of **Window**. A menu will pop up. The options in the menu will be defined by the class variable **RightButtonItems**. If an option is selected from the menu, a message will be sent to *self* with a selector as specified by the chosen menu option.

Arguments: *self* Pointer to a window instance.

Returns: The menu.

Categories: Window

(← *self* **TitleSelection**) [Method of Window]

Purpose/Behavior: Invokes a number of internal methods of **Window**. A menu will pop up. The options in the menu will be defined by the class variable **TitleItems**. If an option is selected from the menu, a message will be sent to *self* with a selector as specified by the chosen menu option .

Arguments: *self* Pointer to a window instance.

Returns: The choice if selected; else NIL.

Categories: Window

Specializations: LatticeBrowser

(← *self* **WhenMenuItemHeld** *item* - -) [Method of Window]

Purpose/Behavior: Displays in the system prompt window what will happen when the menu item is chosen. The information displayed will either be the help string for the item or the documentation for the method pointed to by the item.

Arguments:

self	Pointer to a window instance.
item	The menu item selector.

Returns: NIL

Categories: Window

WindowButtonEventFn [Function]

Purpose/Behavior: Invokes the method **ButtonEventFn**.

WindowRightButtonFn [Function]

Purpose/Behavior: Invokes the method **RightButtonFn.**

(**WindowButtonEventFn** *window*) [Function]

Purpose/Behavior: This retrieves the value of the Lisp window property **LoopsWindow**. It sends the message **ButtonEventFn** to that window object. If the window object is an instance of **NonRectangularWindow** or one of its subclasses and if the cursor is not within the icon bitmap, nothing occurs.

This is invoked automatically by the system when the cursor is inside of a window object and the left or middle mouse button is pressed.

Arguments:

window	The window that contained the cursor when the mouse button was pressed.

Returns: Used for side effect only.

(**WindowRightButtonFn** *window*) [Function]

Purpose/Behavior: This retrieves the value of the Lisp window property **LoopsWindow**. It sends the message **RightButtonFn** to that window object. If the window object is an instance of **NonRectangularWindow** or one of its subclasses and if the cursor is not within the icon bitmap, nothing occurs.

This is invoked automatically by the system when the cursor is inside of a window object and the left or middle mouse button is pressed.

Arguments:

window	The window that contained the cursor when the mouse button was pressed.

Returns: Used for side effect only.

19.4.1 Menu Item Structure

The default behavior for the methods **LeftSelection**, **MiddleSelection**, and **RightSelection** causes a menu to pop up. The options that will appear in a menu are defined in various class variables of the window object being selected. (**IconWindows** are an exception). When an option is selected from a menu, a message is sent to the window with no arguments.

The value of the various class variables can be an item list, such as (item1....itemn) where each item can be one of:

- *selector*

  In this case, the *selector* appears in the menu, and it is the selector of the message sent to the window.

- *(prompt selector help-string)*

  In this case, the *prompt* appears in the menu, and *selector* is the selector of the message sent to the window. *help-string* is printed when the cursor is over the item and the mouse is pressed.

- *(prompt subitemStructure help-string)* where *subitemStructure* = *(defaultSelector itemlist)*

  This form allows a menu to contain submenus. In this case, the *prompt* appears in the main menu, and *defaultSelector* is the selector of the message sent to the window if the main menu item is selected. *itemlist* defines the submenu behavior.

For example, in the class **Window**, the class variable **LeftButtonItems** has the following value:

```
((Update (QUOTE Update) "Update window to agree with object IVs"))
```

The class variable **RightButtonItems** has the value:

```
((Close (Close (Close Destroy))) Snap Paint Clear Bury Repaint
(Hardcopy (Hardcopy (HardcopyToFile HardcopyToPrinter))) Move Shape
Shrink)
```

19.4.2 Caching Menus

When a menu is created by pressing a mouse button on a LOOPS window, the menu is cached on a property of the instance variable **menus** if **menus** has the value T. The name of the property where it is stored has the same name as the class variable that describes the menu. The method **ClearMenuCache** will set these properties to NIL, causing the menus to be deleted from the window instance.

19.5 Subclasses of Window

This section describes the classes **NonRectangularWindow**, **IconWindow**, and **LoopsIcon** and functionality associated with them.

Name	Type	Description
NonRectangularWindow	Class	Provides the capability for windows to act as icons.
CreateWindow	Method	Creates a window that acts like an icon.
EditIcon	Method	Edits an icon bitmap.
EditMask	Method	Edits a mask bitmap.
Invert	Method	Inverts the image of an icon; that is, reverses its black-white pattern.
Shape	Method	Prevents the window from being shaped by calling **LoopsHelp**.
IconWindow	Class	Provides some menu options for icon windows.
LoopsIcon	Class	Provides an icon that is part of the LOOPS user interface.
PutSavedValue	Function	Stores a value. This is called from within browser and inspector menu events.
SavedValue	Function	Retrieves a saved value.

NonRectangularWindow [Class]

Description: Provides the capability for windows to act as icons.

MetaClass: Class

Supers: Window

Instance Variables:

icon	Allows a bitmap to be used as an icon to be stored in the instance. If the **bitMap** property is set to a symbol whose value is a bitmap, then that bitmap will be used. The default value is NIL.
mask	Allows a bitmap to be used as an icon mask to be stored in the instance. If the **bitMap** property is set to a symbol whose value is a bitmap, then that bitmap will be used. The default value is NIL.

(← *self* **CreateWindow**) [Method of NonRectangularWindow]

Purpose: Creates a window that acts like an icon.

Behavior: Determines if **icon** and **mask** or the property **bitMap** have values. If so, it uses those within a call to **ICONW**. If not, it sends the messages **EditIcon** and **EditMask**.

This method is invoked by the system if the instance variable **window** is not yet bound to a value and it is accessed.

Arguments: *self* A window instance.

Returns: Value returned from **ICONW**.

Categories: Window

Specializes: Window

(← *self* **EditIcon**) [Method of NonRectangularWindow]

Purpose:	Edits an icon bitmap.
Behavior:	Calls **EDITBM** with the value of the instance variable **icon** and assigns **icon** to the returned value.
Arguments:	*self* A window instance.
Returns:	Value returned from **EDITBM**.
Categories:	NonRectangularWindow

(← *self* **EditMask**) [Method of NonRectangularWindow]

Purpose:	Edits a mask bitmap.
Behavior:	Calls **EDITBM** withthe value of the instance variable **mask** if it is non-NIL, or a copy of **icon,** and assigns **mask** to the returned value.
Arguments:	*self* A window instance.
Returns:	Value returned from **EDITBM**.
Categories:	NonRectangularWindow

(← *self* **Invert**) [Method of NonRectangularWindow]

Purpose:	Inverts the image of an icon; that is, reverses its black-white pattern.
Behavior:	Modifies the bitmap of the **ICONIMAGE** window property of the Lisp window pointed to by *self*.
Arguments:	*self* A window instance.
Returns:	Used for side effect only.
Categories:	Window
Specializes:	Window

(← *self* **Shape**) [Method of NonRectangularWindow]

Purpose/Behavior:	Prevents the window from being shaped by calling **LoopsHelp**. This method is provided to restrict the shaping of this class of window, not to provide additional functionality.
Arguments:	*self* A window instance.
Returns:	Used for side effect only.
Categories:	Window
Specializes:	Window

IconWindow [Class]

Description:	Provides some menu options for icon windows. The menu behavior of this class is different from the class **Window** in that the item lists are stored on instance variables and not class variables.
MetaClass:	Class

Supers: NonRectangularWindow

Instance Variables:

RightButtonItems
A list that defines the menu that will appear when the right mouse button is pressed when the cursor is in the window. The default value is (Move).

MiddleButtonItems
A list that defines the menu that will appear when the middle mouse button is pressed when the cursor is in the window. The default value is NIL.

LeftButtonItems
A list that defines the menu that will appear when the left mouse button is pressed when the cursor is in the window. The default value is (Move).

ShiftMiddleButtonItems
A list that defines the menu that will appear when the middle mouse button is pressed when the cursor is in the window and the **Meta** key is pressed. The default value is NIL.

ShiftLeftButtonItems
A list that defines the menu that will appear when the left mouse button is pressed when the cursor is in the window and the **Meta** key is pressed. The default value is (Move).

LoopsIcon [Class]

Description: Implements the LOOPS icon which is part of the LOOPS user interface to LatticeBrowsers (see Chapter 10, Browsers).

MetaClass: Class

Supers: NonRectangularWindow

Class Variables:

RightButtonItems
A list that defines the menu that will appear when the right mouse button is pressed when the cursor is in the window. The default value is (Close Move).

MiddleButtonItems
A list that defines the menu that will appear when the middle mouse button is pressed when the cursor is in the window. The default value is (("Browse File" (...))).

LeftButtonItems
A list that defines the menu that will appear when the left mouse button is pressed when the cursor is in the window. The default value is (("Browse Class" (...))).

Instance Variables:

savedValue	Used by the functions **PutSavedValue** and **SavedValue.** The default value is NIL.
icon	The property **bitMap** has the value **BlackLoopsIconBM.** The default value is NIL.
mask	The property **bitMap** has the value **LoopsIconShadow.** The default value is NIL.

(**PutSavedValue** *value*) [Function]

Purpose: Stores a value. This is called from within browser and inspector menu events.

Behavior: Sets the instance variable **savedValue** of the prototype instance of the class **LoopsIcon** to *value*. Also sets the top level binding of **IT** to *value*; see the *Interlisp-D Reference Manual* for information on **IT**.

Arguments: *value* Any arbitrary data.

Returns: *value*

(**SavedValue**) [Function]

Purpose: Retrieves a saved value.

Behavior/Returns: Gets the value of the instance variable **savedValue** of the prototype instance of the class **LoopsIcon**.

19.6 Lisp Windows

The methods in this section define the interface between LOOPS windows and Lisp windows. These methods are used internally by the system, and will rarely be used or specialized by users.

Name	Type	Description
AttachLispWindow	Method	Gives a LOOPS window a Lisp window.
CreateWindow	Method	Creates a Lisp window.
DetachLispWindow	Method	Forgets about the current Lisp window.
GetWrappedValue	Method	Gets the value wrapped in the active value.
HasLispWindow	Method	Checks if a Lisp window has ever been created for the LOOPS window.
PutWrappedValue	Method	Replaced the value wrapped in the active value.

(← *self* **AttachLispWindow** *window*) [Method of Window]

Purpose/Behavior: Used to associate a LOOPS window with a Medley window. This detaches any currently attached window before attaching a new one, and fills in the instance variables **left**, **bottom**, **width**, **height**, and **title** from the Lisp window.

Arguments: *self* An instance of a LOOPS window.

window Must be a window pointer.

Returns: Used for side effect only.

Categories: Window

(← *self* **CreateWindow**) [Method of Window]

Purpose/Behavior: Creates a Lisp window for a LOOPS window but does not open it.

Arguments: *self* Pointer to a LOOPS window.

Returns: The window.

Categories: Window

Specializations: NonRectangularWindow

(← *self* **DetachLispWindow**) [Method of Window]

Purpose/Behavior: Removes the pointer from the LOOPS window to the Lisp window.

Arguments: *self* Pointer to a LOOPS window.

Returns: Used for side effect only.

Categories: Window

(← *self* **GetWrappedValue** *containingObj varName propName type*) [Method of LispWindowAV]

Purpose/Behavior: Used by the system to fetch the Medley window from a LOOPS window. If the local state of this active value is not a window, it is made a window.

Arguments:

self	An instance of **LispWindowAV**.
containingObj	A LOOPS window.
varName	Variable associated with the wrapped value.
propName	Used internally.
type	Used internally.

Returns: A Lisp window.

Categories: LispWindowAv

Specializes: LocalStateActiveValue

(← *self* **HasLispWindow**) [Method of Window]

Purpose/Behavior: Checks if Lisp window has ever been created for this LOOPS window.

Arguments: *self* A LOOPS window.

Returns: The window pointer, if one exists; else NIL.

Categories: Window

(← *self* **PutWrappedValue** *containingObj varName newvalue propName*) [Method of LispWindowAV]

Purpose/Behavior: Places the window *newvalue* as local state of the active value.

Arguments:

self	An instance of **LispWindowAV.**
containingObj	A LOOPS window.
varName	Variable associated with the wrapped value.
propName	Used internally.
type	Used internally.

Returns:	The window pointer, if one exists.
Categories:	LispWindowAV
Specializes:	LocalStateActiveValue

[This page intentionally left blank]