Browsers

User Experience: Data Presentation

2004-08-31

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

Figures

Introduction to Browsers

A browser provides a user interface for displaying and selecting items from a list of data or from hierarchically organized lists of data such as directory paths.

Organization of This Document

This document discusses how to use browsers. "About Browsers" (page 9) gives basic information on browsers and describes what happens when you select a browser item. "Browser Interface Features" (page 11) describes how to control some aspects of a browser's appearance. "Using a Browser Delegate" (page 13) describes how to use a delegate to fill a browser with data.

About Browsers

A browser provides a user interface for displaying and selecting items from a list of data or from hierarchically organized lists of data such as directory paths. When working with a hierarchy of data, the levels are displayed in columns, which are numbered from left to right, beginning with 0. A browser is implemented by the NSBrowser class, and each of its column consists of an NSScrollView containing an NSMatrix filled with NSBrowserCells. NSBrowser relies on a delegate to provide the data in its NSBrowserCells.

How Browser Selection Works

Each entry in an NSBrowser column is an NSBrowser Cell. This cell can be either a branch cell (such as a directory) or a leaf cell (such as a file). A branch cell displays an image indicating that, when the cell is clicked, the NSBrowser will display a new column of NSBrowserCells. To display the new column, the NSBrowser sends itself the `addColumn` message, which messages its delegate to load the next column. An NSBrowserCell can also be loaded or unloaded; loaded NSBrowserCells have their state set and are ready for display. If your code needs access to a specific NSBrowserCell, you can use the NSBrowser method `loadedCellAtRow:column:`.

The user's selection can be represented as a character string; if the selection is hierarchical (for example, a filename within a directory), each component of the path to the selected node is separated by "/". To use some other character as the delimiter, invoke the NSBrowser method `setPathSeparator:`.

An NSBrowser can be set to allow selection of multiple cells in a column, or to limit selection to a single cell. When set for multiple selection, it can also be set to limit multiple selection to leaf cells only, or to allow selection of both types of cells together.

Since NSBrowser inherits from NSControl, it has a target object and action message. Each time the user selects one or more cells in a column, the NBrowser's action message is sent to its target. NSBrowser also adds an action to be sent when the user double-clicks on an entry, which allows the user to select items without any action being taken, and then double-click to invoke some useful action such as opening a file.

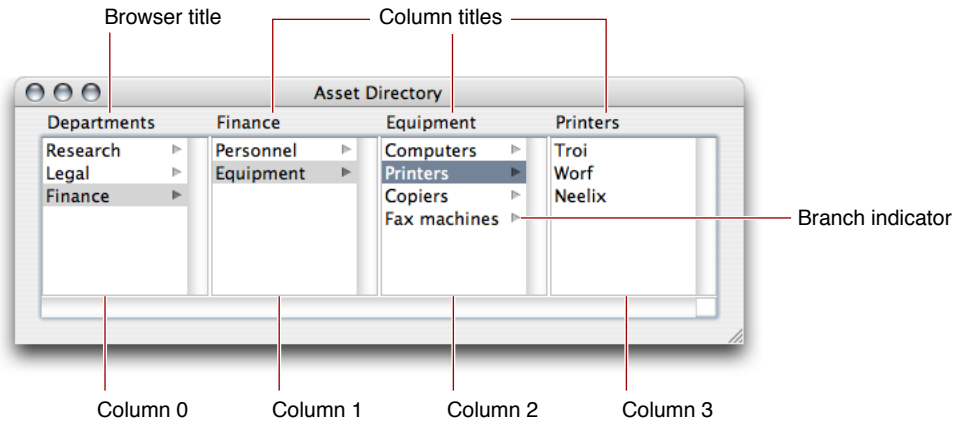
Since NSBrowserCell does not inherit from NSActionCell, it doesn't hold a target and action value and can't directly participate in the target/action paradigm. However, an NSBrowser action method can obtain the last selected NSBrowserCell with the `selectedCell` method.

Browser Interface Features

The user interface features of an NSBrowser can be changed in a number of ways. The NSBrowser may or may not have a horizontal scroller. (The NSBrowser's columns, by contrast, always have vertical scrollers—although a scroller's buttons and knob might be invisible if the column doesn't contain many entries.) You generally shouldn't create an NSBrowser without a horizontal scroller; if you do, you must make sure the bounds rectangle of the NSBrowser is wide enough that all the columns can be displayed. An NSBrowser's columns may be bordered and titled, bordered and untitled, or unbordered and untitled. A column's title may be taken from the selected entry in the column to its left, or may be provided explicitly by the NSBrowser or its delegate.

Figure 1 shows an example of a an NSBrowser.

Figure 1 Browser example



These are some aspects of the user interface shown in Figure 1:

- **Browser title:** You set the NSBrowser's title through its Title attribute in Interface Builder.
- **Column titles:** You can change the title of each column through the `setTitle:ofColumn:` instance method of NSBrowser. Note that if you set the browser's Title attribute in Interface Builder, it's displayed in place of Column 0's title.
- **Branch indicator:** This indicator appears based on the response from the corresponding NSBrowserCell to the `isLeaf` message. The presence of the indicator tells users that when they click the cell, the column to its right displays information that is hierarchically associated under that cell.

Using a Browser Delegate

NSBrowser requires a delegate to provide it with data to display. The delegate is responsible for providing the data and for setting each item as a branch or leaf node, enabled or disabled. It can also receive notification of events like scrolling and requests for validation of columns that may have changed.

You can implement one of two delegate types: Active or passive. An active delegate creates a column's rows (that is, the NSBrowserCells) itself, while a passive one leaves that job to the NSBrowser. Normally, passive delegates are preferable, because they're easier to implement. An active delegate must implement `browser:createRowsForColumn:` to create the rows of the specified column. A passive delegate, on the other hand, must implement `browser:numberOfRowsInColumn:` to let the NSBrowser know how many rows to create. These two methods are mutually exclusive; you can implement one or the other, but not both. (The NSBrowser ascertains what type of delegate it has by which method the delegate responds to.)

Both types of delegate implement `browser:willDisplayCell:atRow:column:` to set up state (such as the cell's string value and whether the cell is a leaf or a branch) before an individual cell is displayed. (This delegate method doesn't need to invoke NSBrowserCell's `setLoaded:` method, because the NSBrowser can determine that state by itself.) An active delegate can instead set all the cells' state at the time the cells are created, in which case it doesn't need to implement `browser:willDisplayCell:atRow:column:`. However, a passive delegate must always implement this method.

Document Revision History

This table describes the changes to *Browsers*.

Date	Notes
2004-08-31	Added figure to illustrate an NSBrowser's user interface.
2003-01-23	Corrected minor typos in overview.
2002-11-12	Revision history was added to existing topic. It will be used to record changes to the content of the topic.