Data Browser Reference

Carbon > User Experience

2008-04-08

Apple Inc.
© 2003, 2008 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

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

Apple, the Apple logo, Carbon, Mac, Mac OS, and QuickDraw are trademarks of Apple Inc., registered in the United States and other countries.

Finder is a trademark of Apple Inc.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

Data Browser Reference 13

Overview 13
Functions by Task 14
- Creating and Configuring a Data Browser 14
- Manipulating Data Browser Attributes 14
- Setting Up and Installing Callbacks 15
- Formatting Table View 16
- Formatting List View 17
- Formatting Column View 17
- Adding and Removing Data Items 18
- Accessing and Operating on All Items 18
- Accessing and Displaying Individual Items 18
- Selecting and Editing Items 18
- Working With Attributes 19
- Working With Containers 19
- Working With Metrics 19
- Getting and Setting Item Data 19
- Working With Universal Procedure Pointers 21
- Working With AXUIElement References 24

Functions 24
- AddDataBrowserItems 24
- AddDataBrowserListViewColumn 25
- AutoSizeDataBrowserListViewColumns 26
- AXUIElementCreateWithDataBrowserAndItemInfo 27
- AXUIElementGetDataBrowserItemInfo 27
- CloseDataBrowserContainer 28
- CopyDataBrowserEditText 29
- CreateDataBrowserControl 29
- DataBrowserChangeAttributes 31
- DataBrowserGetAttributes 31
- DataBrowserGetMetric 32
- DataBrowserSetMetric 33
- DisposeDataBrowserAcceptDragUPP 33
- DisposeDataBrowserAddDragItemUPP 34
- DisposeDataBrowserDrawItemUPP 34
- DisposeDataBrowserEditItemUPP 35
- DisposeDataBrowserGetContextualMenuUPP 35
- DisposeDataBrowserHitTestUPP 35
- DisposeDataBrowserItemAcceptDragUPP 36
- DisposeDataBrowserItemCompareUPP 36
- DisposeDataBrowserItemDataUPP 37

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DisposeDataBrowserItemDragRgnUPP 37
DisposeDataBrowserItemHelpContentUPP 37
DisposeDataBrowserItemNotificationUPP 38
DisposeDataBrowserItemNotificationWithItemUPP 38
DisposeDataBrowserItemReceiveDragUPP 39
DisposeDataBrowserItemUPP 39
DisposeDataBrowserPostProcessDragUPP 39
DisposeDataBrowserReceiveDragUPP 40
DisposeDataBrowserSelectContextualMenuUPP 40
DisposeDataBrowserTrackingUPP 41
EnableDataBrowserEditCommand 41
ExecuteDataBrowserEditCommand 42
ForEachDataBrowserItem 42
GetDataBrowserActiveItems 43
GetDataBrowserCallbacks 44
GetDataBrowserColumnViewDisplayType 44
GetDataBrowserColumnViewPath 45
GetDataBrowserColumnViewPathLength 46
GetDataBrowserCustomCallbacks 46
GetDataBrowserEditItem 47
GetDataBrowserEditText 47
GetDataBrowserHasScrollBars 48
GetDataBrowserItemCount 49
GetDataBrowserItemDataBooleanValue 49
GetDataBrowserItemDataButtonValue 50
GetDataBrowserItemDataDateTime 51
GetDataBrowserItemDataDrawState 52
GetDataBrowserItemDataIcon 52
GetDataBrowserItemDataIconTransform 53
GetDataBrowserItemDataItemID 54
GetDataBrowserItemDataLongDateTime 54
GetDataBrowserItemDataMaximum 55
GetDataBrowserItemDataMenuRef 56
GetDataBrowserItemDataMinimum 56
GetDataBrowserItemDataProperty 57
GetDataBrowserItemDataRGBColor 58
GetDataBrowserItemDataText 58
GetDataBrowserItemDataValue 59
GetDataBrowserItemPartBounds 60
GetDataBrowserItems 61
GetDataBrowserItemState 62
GetDataBrowserListViewDisclosureColumn 62
GetDataBrowserListViewHeaderBtnHeight 63
GetDataBrowserListViewHeaderDesc 64
GetDataBrowserListViewUsePlainBackground 64
GetDataBrowserPropertyFlags 65

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

GetDataBrowserScrollBarInset 66
GetDataBrowserScrollPosition 66
GetDataBrowserSelectionAnchor 67
GetDataBrowserSelectionFlags 68
GetDataBrowserSortOrder 68
GetDataBrowserSortProperty 69
GetDataBrowserTableViewColumnCount 70
GetDataBrowserTableViewColumnPosition 70
GetDataBrowserTableViewColumnProperty 71
GetDataBrowserTableViewColumnWidth 71
GetDataBrowserTableViewGeometry 72
GetDataBrowserTableViewHiliteStyle 73
GetDataBrowserTableViewItemID 73
GetDataBrowserTableViewItemRow 74
GetDataBrowserTableViewItemRowHeight 74
GetDataBrowserTableViewNamedColumnWidth 75
GetDataBrowserTableViewRowHeight 76
GetDataBrowserTarget 76
GetDataBrowserUserState 77
GetDataBrowserViewStyle 78
InitDataBrowserCallbacks 79
InitDataBrowserCustomCallbacks 80
InvokeDataBrowserAcceptDragUPP 81
InvokeDataBrowserAddDragItemUPP 81
InvokeDataBrowserDrawItemUPP 81
InvokeDataBrowserEditItemUPP 82
InvokeDataBrowserGetContextualMenuUPP 82
InvokeDataBrowserHitTestUPP 83
InvokeDataBrowserItemAcceptDragUPP 83
InvokeDataBrowserItemCompareUPP 84
InvokeDataBrowserItemDataUPP 84
InvokeDataBrowserItemDragRgnUPP 85
InvokeDataBrowserItemHelpContentUPP 85
InvokeDataBrowserItemNotificationUPP 86
InvokeDataBrowserItemNotificationWithItemUPP 86
InvokeDataBrowserItemReceiveDragUPP 87
InvokeDataBrowserItemUPP 87
InvokeDataBrowserPostProcessDragUPP 88
InvokeDataBrowserReceiveDragUPP 88
InvokeDataBrowserSelectContextualMenuUPP 89
InvokeDataBrowserTrackingUPP 89
IsDataBrowserItemSelected 90
MoveDataBrowserSelectionAnchor 90
NewDataBrowserAcceptDragUPP 91
NewDataBrowserAddDragItemUPP 92
NewDataBrowserDrawItemUPP 92

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

NewDataBrowserEditItemUPP 92
NewDataBrowserGetContextualMenuUPP 93
NewDataBrowserHitTestUPP 93
NewDataBrowserItemAcceptDragUPP 94
NewDataBrowserItemCompareUPP 94
NewDataBrowserItemDataUPP 95
NewDataBrowserItemDragRgnUPP 95
NewDataBrowserItemHelpContentUPP 96
NewDataBrowserItemNotificationUPP 96
NewDataBrowserItemNotificationWithItemUPP 97
NewDataBrowserItemReceiveDragUPP 97
NewDataBrowserItemUPP 98
NewDataBrowserPostProcessDragUPP 98
NewDataBrowserReceiveDragUPP 99
NewDataBrowserSelectContextualMenuUPP 99
NewDataBrowserTrackingUPP 100
OpenDataBrowserContainer 100
RemoveDataBrowserItems 101
RemoveDataBrowserTableViewColumn 102
RevealDataBrowserItem 103
SetDataBrowserActiveItems 103
SetDataBrowserCallbacks 104
SetDataBrowserColumnViewDisplayType 106
SetDataBrowserColumnViewPath 106
SetDataBrowserCustomCallbacks 107
SetDataBrowserEditItem 108
SetDataBrowserEditText 109
SetDataBrowserHasScrollBars 110
SetDataBrowserItemDataBooleanValue 110
SetDataBrowserItemDataButtonValue 111
SetDataBrowserItemDataDateTime 112
SetDataBrowserItemDataDrawState 112
SetDataBrowserItemDataIcon 113
SetDataBrowserItemDataIconTransform 114
SetDataBrowserItemDataItemID 114
SetDataBrowserItemDataLongDateTime 115
SetDataBrowserItemDataMaximum 116
SetDataBrowserItemDataMenuRef 117
SetDataBrowserItemDataMinimum 117
SetDataBrowserItemDataRGBColor 118
SetDataBrowserItemDataText 118
SetDataBrowserItemDataValue 119
SetDataBrowserListViewDisclosureColumn 120
SetDataBrowserListViewHeaderBtnHeight 121
SetDataBrowserListViewHeaderDesc 122
SetDataBrowserListViewUsePlainBackground 123

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

SetDataBrowserPropertyFlags 123
SetDataBrowserScrollBarInset 124
SetDataBrowserScrollPosition 125
SetDataBrowserSelectedItems 126
SetDataBrowserSelectionFlags 126
SetDataBrowserSortOrder 127
SetDataBrowserSortProperty 127
SetDataBrowserTableViewColumnPosition 128
SetDataBrowserTableViewColumnWidth 129
SetDataBrowserTableViewGeometry 129
SetDataBrowserTableViewHiliteStyle 130
SetDataBrowserTableViewItemRow 130
SetDataBrowserTableViewItemRowHeight 131
SetDataBrowserTableViewNamedColumnWidth 132
SetDataBrowserTableViewRowHeight 132
SetDataBrowserTarget 133
SetDataBrowserUserState 134
SetDataBrowserViewStyle 134
SortDataBrowserContainer 135
UpdateDataBrowserItems 136
Callbacks 137
DataBrowserAcceptDragProcPtr 137
DataBrowserAddDragItemProcPtr 138
DataBrowserDrawItemProcPtr 139
DataBrowserEditItemProcPtr 141
DataBrowserGetContextualMenuProcPtr 142
DataBrowserHitTestProcPtr 144
DataBrowserItemAcceptDragProcPtr 146
DataBrowserItemCompareProcPtr 147
DataBrowserItemDataProcPtr 149
DataBrowserItemDragRgnProcPtr 150
DataBrowserItemHelpContentProcPtr 152
DataBrowserItemNotificationProcPtr 154
DataBrowserItemNotificationWithItemProcPtr 155
DataBrowserItemProcPtr 157
DataBrowserItemReceiveDragProcPtr 158
DataBrowserPostProcessDragProcPtr 159
DataBrowserReceiveDragProcPtr 160
DataBrowserSelectContextualMenuProcPtr 161
DataBrowserTrackingProcPtr 163
Data Types 164
DataBrowserAccessibilityItemInfo 164
DataBrowserAccessibilityItemInfoV0 165
DataBrowserAccessibilityItemInfoV1 166
DataBrowserPropertyDesc 167
DataBrowserCallbacks 168

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DataBrowserCustomCallbacks 169
DataBrowserDragFlags 170
DataBrowserItemDataRef 171
DataBrowserItemID 171
DataBrowserPropertyFlags 171
DataBrowserPropertyID 172
DataBrowserTableViewRowIndex 172
DataBrowserTableViewColumnIndex 173
DataBrowserTableViewColumnID 173
DataBrowserTableViewColumnDesc 173
DataBrowserListViewHeaderDesc 173
DataBrowserListViewColumnDesc 174
kHIDataBrowserClassID 175
Constants 175
Callback Data Structure Version 175
Control Data Tags 176
Custom Callback Data Structure Version 176
Data Browser Attributes 177
Data Browser Control Kind Tag 178
Data Browser Metric Values 178
Display Types 179
Editing Commands 181
Item Notifications 182
Item States 184
List View Header Description Version 185
List View Append Column 185
No Item Constant 185
Properties 186
Property Flags: Universal 188
Property Flags: Modifiers 189
Property Flags: Offset and Mask for List View Properties 192
Property Flags: List View Column Behavior 193
Property Flags: Offset and Mask for Client-Defined Properties 194
Property Parts 195
Reveal Options 196
Selection Anchor Directions 197
Selection State Options 197
Sorting Orders 198
Table View Highlighting Styles 198
Table View Last Column Value 199
Table View Property Flag 199
Tracking Results 200
User Selection Flags 200
View Styles 202
Result Codes 202

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Document Revision History 205

Index 207

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Figures

Data Browser Reference 13

Figure 1 A container can open to more rows or expand to show more information 121
Figure 2 Differentiation between the selectable content and background 145

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Data Browser Reference

Framework:	Carbon/Carbon.h
Declared in	HIDataBrowser.h

Overview

The data browser application programming interface (API) provides a convenient way to present data for browsing and to create easily customized lists whose columns can be sorted, moved, and resized. It supports two presentation styles, each of which is derived from an abstract table-view base class:

- **List view**, which lets you present items in multiple columns with the option to create hierarchical lists whose contents can be disclosed by the user
- **Column view**, which provides in-place browsing using fixed navigation columns

The data browser programming interface has some routines that apply to both views while others are unique to one view. For functions that can be called for either, there may be differences in how the functions operate. Such differences are noted in the documentation for individual functions.

These terms are essential to understanding the reference:

- An **item** in a data browser refers to the data displayed at a particular row and column intersection. In list view, two values identify each item—an item ID and a property ID. In column view, one value—the item ID—uniquely identifies an item.
- An **item ID** is a unique 32-bit ID number that your application uses to refer to data. When you ask the data browser to display one or more items, you provide an item ID for each data item. You can store the actual data in memory, on disk, or across a network. Item IDs must be greater than `0`, which is used internally by the data browser. Item IDs can be values such as pointer values, data file offsets, and 32-bit TCP/IP host addresses.
- A **Property ID** is a non-zero, 32-bit unsigned integer value that uniquely identifies a list view column. Property IDs do not need to be ordered or sequential, but they cannot be values 0 through 1023 because those values are reserved by Apple. A property ID is typically defined as a four-character sequence. For example, a column that displays dates could be assigned the property ID `DATE`.

  Columns in column view don't use application-defined property IDs. Instead, columns in column view have the predefined property `kDataBrowserItemSelfIdentityProperty`.

After you've created, formatted, and configured a data browser, most of the work of keeping the data browser updated and responsive to user interaction happens through callbacks you provide. For example, all of the functions that get and set item data are called from within an item-data callback provided by your application. Your application has a wide latitude in what it can choose to handle through callbacks and the tasks it lets the system perform. At the very least, your application must provide an item-data callback. Otherwise no

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

data will ever be written to the data browser user interface. Depending on the nature of your application, you may also want to provide callbacks to handle drag-and-drop behavior, to support contextual menus, and to perform custom drawing or some other custom behavior.

The data browser is available with CarbonLib 1.1 and later and in Mac OS X.

For conceptual information and instructions on how to write code that uses a data browser to display data, see *Data Browser Programming Guide*.

Functions by Task

Creating and Configuring a Data Browser

`CreateDataBrowserControl` (page 29)
Creates a data browser programmatically.

`SetDataBrowserViewStyle` (page 134)
Sets the view style of the specified data browser.

`GetDataBrowserViewStyle` (page 78)
Obtains the current view style for the specified data browser.

Manipulating Data Browser Attributes

`GetDataBrowserUserState` (page 77)
Obtains the current view style settings for a list view.

`SetDataBrowserUserState` (page 134)
Restores the view-style settings in list view to a previous state set by the user.

`SetDataBrowserActiveItems` (page 103)
Sets what determines the active state of the items in a data browser.

`GetDataBrowserActiveItems` (page 43)
Obtains what determines the active state of the items in a data browser.

`SetDataBrowserScrollBarInset` (page 124)
Sets the inset values to use for the scroll bars of a data browser.

`GetDataBrowserScrollBarInset` (page 66)
Obtains the inset rectangle used by a data browser to position the scroll bar.

`SetDataBrowserTarget` (page 133)
Sets the target for a data browser.

`GetDataBrowserTarget` (page 76)
Obtains the target for the data browser

`SetDataBrowserSortOrder` (page 127)
Sets the sorting order for a list in list view.

`GetDataBrowserSortOrder` (page 68)
Gets the sorting order of the list view column that's currently set for sorting.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`SetDataBrowserScrollPosition` (page 125)
Scrolls a list to the specified position.

`GetDataBrowserScrollPosition` (page 66)
Obtains the scrolling position of a list.

`SetDataBrowserHasScrollBars` (page 110)
Sets the display state of horizontal and vertical scroll bars for a list view data browser.

`GetDataBrowserHasScrollBars` (page 48)
Obtains the display state of horizontal and vertical scroll bars for a list view data browser.

`SetDataBrowserSortProperty` (page 127)
Designates the list view column to use for sorting.

`GetDataBrowserSortProperty` (page 69)
Obtains the property ID of the column currently used for sorting in list view.

`SetDataBrowserSelectionFlags` (page 126)
Sets allowable selection behavior for a data browser.

`GetDataBrowserSelectionFlags` (page 68)
Obtains the current selection behavior for a data browser.

`SetDataBrowserPropertyFlags` (page 123)
Sets the appearance and behavior attributes for a column in list view.

`GetDataBrowserPropertyFlags` (page 65)
Obtains the appearance and behavior attributes for a column.

`SetDataBrowserEditText` (page 109)
Modifies the displayed contents of a text item while it is being edited.

`CopyDataBrowserEditText` (page 29)
Copies the text being edited by the user.

`GetDataBrowserEditText` (page 47)
Obtains the text being edited by the user.

`SetDataBrowserEditItem` (page 108)
Programmatically starts or ends an editing session.

`GetDataBrowserEditItem` (page 47)
Obtains the item ID and property ID values of the current editing session.

`GetDataBrowserItemPartBounds` (page 60)
Obtains the bounds of a visual part of an item.

Setting Up and Installing Callbacks

`InitDataBrowserCallbacks` (page 79)
Initializes a data browser callback structure in preparation for adding your own callbacks to the structure.

`SetDataBrowserCallbacks` (page 104)
Sets the callback routines to use with a data browser, replacing any previously installed callbacks.

`GetDataBrowserCallbacks` (page 44)
Obtains the callback routines installed for notifying your application of changes to a data browser and for providing the data to be displayed by the data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`InitDataBrowserCustomCallbacks` (page 80)
: Initializes the data browser custom callback structure in preparation for adding your own callbacks for custom drawing or custom behavior to the structure.

`SetDataBrowserCustomCallbacks` (page 107)
: Sets the custom callback routines to use with a data browser, replacing any previously installed custom callbacks.

`GetDataBrowserCustomCallbacks` (page 46)
: Obtains the callbacks installed to implement custom drawing and behavior for the content in a data browser.

Formatting Table View

Table view is a base class from which list and column views are derived. Some functions in this group can be used with both list and column views, while others are useful only in list view.

`RemoveDataBrowserTableViewColumn` (page 102)
: Removes a column from a list view data browser.

`GetDataBrowserTableViewColumnCount` (page 70)
: Obtains the number of columns in a data browser.

`SetDataBrowserTableViewHiliteStyle` (page 130)
: Sets the highlighting style to use for a list view data browser.

`GetDataBrowserTableViewHiliteStyle` (page 73)
: Obtains the highlighting style used for a list view data browser.

`SetDataBrowserTableViewRowHeight` (page 132)
: Sets the default row height for all rows in a data browser.

`GetDataBrowserTableViewRowHeight` (page 76)
: Obtains the default row height used for all rows in a data browser.

`SetDataBrowserTableViewColumnWidth` (page 129)
: Sets the default column width for all columns in a data browser.

`GetDataBrowserTableViewColumnWidth` (page 71)
: Obtains the default column width used for all columns in a data browser.

`SetDataBrowserTableViewItemRowHeight` (page 131)
: Sets the row height for a single row in a list view data browser.

`GetDataBrowserTableViewItemRowHeight` (page 74)
: Obtains the row height for a single row in a list view data browser.

`SetDataBrowserTableViewNamedColumnWidth` (page 132)
: Sets the column width for a single column in a list view data browser.

`GetDataBrowserTableViewNamedColumnWidth` (page 75)
: Obtains the column width for a single column in a data browser.

`SetDataBrowserTableViewGeometry` (page 129)
: Sets whether columns and rows can have variable widths in list view.

`GetDataBrowserTableViewGeometry` (page 72)
: Determines whether columns and rows are set to have variable widths.

`GetDataBrowserTableViewItemID` (page 73)
: Obtains the item ID for the item displayed in the specified row.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`SetDataBrowserTableViewItemRow` (page 130)
Changes the visual position for an item in a list view data browser.

`GetDataBrowserTableViewItemRow` (page 74)
Obtains the visual position for the specified item in list view.

`SetDataBrowserTableViewColumnPosition` (page 128)
Changes the visual position of a column in list view.

`GetDataBrowserTableViewColumnPosition` (page 70)
Obtains the column position for an item in a data browser.

`GetDataBrowserTableViewColumnProperty` (page 71)
Obtains the property ID for a column in a data browser.

Formatting List View

`AutoSizeDataBrowserListViewColumns` (page 26)
Adjusts the size of columns displayed in list view to take best advantage of the available space.

`AddDataBrowserListViewColumn` (page 25)
Adds a column to a data browser that uses list view.

`GetDataBrowserListViewHeaderDesc` (page 64)
Obtains a header description for a column in list view.

`SetDataBrowserListViewHeaderDesc` (page 122)
Provides a description for a column title in list view.

`SetDataBrowserListViewHeaderBtnHeight` (page 121)
Sets the height of the rectangular area where the column title appears.

`GetDataBrowserListViewHeaderBtnHeight` (page 63)
Obtains the height of the rectangular area where the column title appears.

`SetDataBrowserListViewUsePlainBackground` (page 123)
Specifies whether list view uses a plain white background.

`GetDataBrowserListViewUsePlainBackground` (page 64)
Determines whether list view is set to use a plain white background.

`SetDataBrowserListViewDisclosureColumn` (page 120)
Specifies whether there is a column that has disclosure triangles and, if so, which column.

`GetDataBrowserListViewDisclosureColumn` (page 62)
Obtains the property ID of the column whose items can display a disclosure triangle, and tells whether a disclosed item expands the row or adds rows.

Formatting Column View

`SetDataBrowserColumnViewPath` (page 106)
Sets a path for a column view.

`GetDataBrowserColumnViewPath` (page 45)
Obtains the current path for a selection in column view.

`GetDataBrowserColumnViewPathLength` (page 46)
Obtains the length of the current path for a column view.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`SetDataBrowserColumnViewDisplayType` (page 106)
Sets the display type for a data browser in column view.

`GetDataBrowserColumnViewDisplayType` (page 44)
Obtains the display type for a column view.

Adding and Removing Data Items

`AddDataBrowserItems` (page 24)
Adds one or more items to a data browser.

`RemoveDataBrowserItems` (page 101)
Removes one or more items from a data browser.

`UpdateDataBrowserItems` (page 136)
Requests a redraw of one or more items in a data browser.

Accessing and Operating on All Items

`GetDataBrowserItems` (page 61)
Obtains a list of the items that match a specified state; operates on items in the root container or traverses items in the data hierarchy.

`GetDataBrowserItemCount` (page 49)
Obtains the number of items whose state matches the specified state.

`ForEachDataBrowserItem` (page 42)
Applies an item-iterator callback routine to each data item that meets the specified criteria.

Accessing and Displaying Individual Items

`IsDataBrowserItemSelected` (page 90)
Checks to see if a data item is selected.

`GetDataBrowserItemState` (page 62)
Obtains the state of an item.

`RevealDataBrowserItem` (page 103)
Scrolls an item into view, optionally bringing a particular part of that item into view.

Selecting and Editing Items

`EnableDataBrowserEditCommand` (page 41)
Determines whether the data browser is currently able to process a given editing command.

`ExecuteDataBrowserEditCommand` (page 42)
Executes an editing command.

`GetDataBrowserSelectionAnchor` (page 67)
Obtains the first and last items in a selection.

`MoveDataBrowserSelectionAnchor` (page 90)
Moves or extends the current selection.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`SetDataBrowserSelectedItems` (page 126)
Modifies the current selection by adding items, removing items, or toggling the selection state of items.

Working With Attributes

`DataBrowserGetAttributes` (page 31)
Gets the attributes of a data browser.

`DataBrowserChangeAttributes` (page 31)
Sets the attributes for a data browser.

Working With Containers

`OpenDataBrowserContainer` (page 100)
Opens a data browser container.

`CloseDataBrowserContainer` (page 28)
Closes a data browser container.

`SortDataBrowserContainer` (page 135)
Sorts a hierarchical list of items.

Working With Metrics

`DataBrowserGetMetric` (page 32)
Gets the value of a specified data browser metric.

`DataBrowserSetMetric` (page 33)
Sets the value of a specified data browser metric.

Getting and Setting Item Data

The functions in this section are called from within an item-data callback routine (`DataBrowserItemDataProcPtr`) provided by your application. The data browser invokes your item-data callback each time your application needs to provide data for the display. Your callback responds by calling the appropriate function from this section.

`SetDataBrowserItemDataIcon` (page 113)
Specifies the icon to draw.

`GetDataBrowserItemDataIcon` (page 52)
Obtains the icon drawn for an item.

`SetDataBrowserItemDataText` (page 118)
Specifies the text to draw.

`GetDataBrowserItemDataText` (page 58)
Obtains the text entered by the user.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`SetDataBrowserItemDataValue` (page 119)
Sets the value of an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

`GetDataBrowserItemDataValue` (page 59)
Obtains the value of an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

`SetDataBrowserItemDataMinimum` (page 117)
Specifies the minimum integer value that can be displayed for an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

`GetDataBrowserItemDataMinimum` (page 56)
Obtains the minimum integer value that can be displayed for an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

`SetDataBrowserItemDataMaximum` (page 116)
Specifies the maximum integer value that can be displayed for an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

`GetDataBrowserItemDataMaximum` (page 55)
Obtains the maximum integer value that can be displayed; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

`SetDataBrowserItemDataBooleanValue` (page 110)
Specifies a Boolean value for an item.

`GetDataBrowserItemDataBooleanValue` (page 49)
Obtains the Boolean value for an item.

`SetDataBrowserItemDataMenuRef` (page 117)
Sets the pop-up menu to display.

`GetDataBrowserItemDataMenuRef` (page 56)
Obtains the pop-up menu displayed.

`SetDataBrowserItemDataRGBColor` (page 118)
Specifies a color to use when drawing an item.

`GetDataBrowserItemDataRGBColor` (page 58)
Obtains the color used to draw an item.

`SetDataBrowserItemDataDrawState` (page 112)
Specifies whether to draw a checkbox in the active or inactive state.

`GetDataBrowserItemDataDrawState` (page 52)
Determines whether a checkbox is in the active or inactive state.

`SetDataBrowserItemDataButtonValue` (page 111)
Specifies a checkbox value.

`GetDataBrowserItemDataButtonValue` (page 50)
Obtains the value for a checkbox.

`SetDataBrowserItemDataIconTransform` (page 114)
Specifies a transformation to apply to an icon when it is drawn.

`GetDataBrowserItemDataIconTransform` (page 53)
Obtains the transformation currently used to display an icon.

`SetDataBrowserItemDataDateTime` (page 112)
Specifies, as a 32-bit value, a date and time value to display.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`GetDataBrowserItemDataDateTime` (page 51)
Obtains, as a 32-bit value, the date and time value displayed.

`SetDataBrowserItemDataLongDateTime` (page 115)
Specifies, as a 64-bit value, a date and time value to display.

`GetDataBrowserItemDataLongDateTime` (page 54)
Obtains, as a 64-bit value, the date and time value displayed.

`SetDataBrowserItemDataItemID` (page 114)
Communicates a property of an item when that property is another item's ID.

`GetDataBrowserItemDataItemID` (page 54)
Obtains the item ID for an item whose property is another item's ID.

`GetDataBrowserItemDataProperty` (page 57)
Obtains the column property ID for the column in which an item resides.

Working With Universal Procedure Pointers

The functions in this section create and dispose of universal procedure pointers (UPPs) to the callbacks you provide to the data browser. For each callback, there is a New, Dispose, and Invoke function. You don't need to use an Invoke function, because the data browser invokes callbacks for you.

The documentation for the UPP functions in this section is boilerplate text—quite repetitive and you can likely skip over it. The more interesting documentation is for the callbacks themselves, which you can find in the section "Data Browser Callbacks" (page 137).

`NewDataBrowserItemUPP` (page 98)
Creates a universal procedure pointer to an item-iterator callback function.

`InvokeDataBrowserItemUPP` (page 87)
Calls an item-iterator callback function.

`DisposeDataBrowserItemUPP` (page 39)
Disposes of a universal procedure pointer to an item-iterator callback function.

`NewDataBrowserItemDataUPP` (page 95)
Creates a universal procedure pointer to an item-data callback function.

`InvokeDataBrowserItemDataUPP` (page 84)
Calls an item-data callback function.

`DisposeDataBrowserItemDataUPP` (page 37)
Disposes of a universal procedure pointer to an item-data callback function.

`NewDataBrowserItemCompareUPP` (page 94)
Creates a universal procedure pointer to an item-comparison callback function.

`InvokeDataBrowserItemCompareUPP` (page 84)
Calls an item-comparison callback function.

`DisposeDataBrowserItemCompareUPP` (page 36)
Disposes of a universal procedure pointer to an item-comparison callback function.

`NewDataBrowserItemNotificationUPP` (page 96)
Creates a universal procedure pointer to an item-notification callback function.

`InvokeDataBrowserItemNotificationUPP` (page 86)
Calls an item-notification callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DisposeDataBrowserItemNotificationUPP (page 38)
Disposes of a universal procedure pointer to an item-notification callback function.

NewDataBrowserItemNotificationWithItemUPP (page 97)
Creates a universal procedure pointer to an item-notification-with-data callback function.

InvokeDataBrowserItemNotificationWithItemUPP (page 86)
Calls an item-notification-with-data callback function.

DisposeDataBrowserItemNotificationWithItemUPP (page 38)
Disposes of a universal procedure pointer to an item-notification-with-data callback function.

NewDataBrowserAddDragItemUPP (page 92)
Creates a universal procedure pointer to an add-drag-item callback function.

InvokeDataBrowserAddDragItemUPP (page 81)
Calls an add-drag-item callback function.

DisposeDataBrowserAddDragItemUPP (page 34)
Disposes of a universal procedure pointer to an add-drag-item callback function.

NewDataBrowserAcceptDragUPP (page 91)
Creates a universal procedure pointer to an accept-drag callback function.

InvokeDataBrowserAcceptDragUPP (page 81)
Calls an accept-drag callback function.

DisposeDataBrowserAcceptDragUPP (page 33)
Disposes of a universal procedure pointer to an accept-drag callback function.

NewDataBrowserReceiveDragUPP (page 99)
Creates a universal procedure pointer to a receive-drag callback function.

InvokeDataBrowserReceiveDragUPP (page 88)
Calls a receive-drag callback function.

DisposeDataBrowserReceiveDragUPP (page 40)
Disposes of a universal procedure pointer to a receive-drag callback function.

NewDataBrowserPostProcessDragUPP (page 98)
Creates a universal procedure pointer to a postprocess-drag callback function.

InvokeDataBrowserPostProcessDragUPP (page 88)
Calls a postprocess-drag callback function.

DisposeDataBrowserPostProcessDragUPP (page 39)
Disposes of a universal procedure pointer to a postprocess-drag callback function.

NewDataBrowserGetContextualMenuUPP (page 93)
Creates a universal procedure pointer to a get-contextual-menu callback function.

InvokeDataBrowserGetContextualMenuUPP (page 82)
Calls a get-contextual-menu callback function.

DisposeDataBrowserGetContextualMenuUPP (page 35)
Disposes of a universal procedure pointer to a get-contextual-menu callback function.

NewDataBrowserSelectContextualMenuUPP (page 99)
Creates a universal procedure pointer to a select-contextual-menu callback function.

InvokeDataBrowserSelectContextualMenuUPP (page 89)
Calls a select-contextual-menu callback function.

DisposeDataBrowserSelectContextualMenuUPP (page 40)
Disposes of a universal procedure pointer to a select-contextual-menu callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`NewDataBrowserItemHelpContentUPP` (page 96)
Creates a universal procedure pointer to an item-help-content callback function.

`InvokeDataBrowserItemHelpContentUPP` (page 85)
Calls an item-help-content callback function.

`DisposeDataBrowserItemHelpContentUPP` (page 37)
Disposes of a universal procedure pointer to an item-help-content callback function.

`NewDataBrowserDrawItemUPP` (page 92)
Creates a universal procedure pointer to a draw-item callback function.

`InvokeDataBrowserDrawItemUPP` (page 81)
Calls a draw-item callback function.

`DisposeDataBrowserDrawItemUPP` (page 34)
Disposes of a universal procedure pointer to a draw-item callback function.

`NewDataBrowserEditItemUPP` (page 92)
Creates a universal procedure pointer to an edit-item callback function.

`InvokeDataBrowserEditItemUPP` (page 82)
Calls an edit-item callback function.

`DisposeDataBrowserEditItemUPP` (page 35)
Disposes of a universal procedure pointer to an edit-item callback function.

`NewDataBrowserHitTestUPP` (page 93)
Creates a universal procedure pointer to a hit-test callback function.

`InvokeDataBrowserHitTestUPP` (page 83)
Calls a hit-test callback function.

`DisposeDataBrowserHitTestUPP` (page 35)
Disposes of a universal procedure pointer to a hit-test callback function.

`NewDataBrowserTrackingUPP` (page 100)
Creates a universal procedure pointer to a tracking callback function.

`InvokeDataBrowserTrackingUPP` (page 89)
Calls a tracking callback function.

`DisposeDataBrowserTrackingUPP` (page 41)
Disposes of a universal procedure pointer to a tracking callback function.

`NewDataBrowserItemDragRgnUPP` (page 95)
Creates a universal procedure pointer to an item-drag-region callback function.

`InvokeDataBrowserItemDragRgnUPP` (page 85)
Calls an item-drag-region callback function.

`DisposeDataBrowserItemDragRgnUPP` (page 37)
Disposes of a universal procedure pointer to an item-drag-region callback function.

`NewDataBrowserItemAcceptDragUPP` (page 94)
Creates a universal procedure pointer to an item-accept-drag callback function.

`InvokeDataBrowserItemAcceptDragUPP` (page 83)
Calls an item-accept-drag callback function.

`DisposeDataBrowserItemAcceptDragUPP` (page 36)
Disposes of a universal procedure pointer to an item-accept-drag callback function.

`NewDataBrowserItemReceiveDragUPP` (page 97)
Creates a universal procedure pointer to an item-receive-drag callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`InvokeDataBrowserItemReceiveDragUPP` (page 87)
Calls an item-receive-drag callback function.

`DisposeDataBrowserItemReceiveDragUPP` (page 39)
Disposes of a universal procedure pointer to an item-receive-drag callback function.

Working With AXUIElement References

`AXUIElementCreateWithDataBrowserAndItemInfo` (page 27)
Creates an `AXUIElementRef` that represents some part of a data browser accessibility hierarchy.

`AXUIElementGetDataBrowserItemInfo` (page 27)
Obtains a description of the part of a data browser represented by an `AXUIElementRef`.

Functions

AddDataBrowserItems

Adds one or more items to a data browser.

```
OSStatus AddDataBrowserItems (
   ControlRef browser,
   DataBrowserItemID container,
   ItemCount numItems,
   const DataBrowserItemID *items,
   DataBrowserPropertyID preSortProperty
);
```

Parameters

browser
A data browser.

container
An item ID or the constant `kDataBrowserNoItem`. Pass the item ID that uniquely identifies the container to which you want to add items. Adding one or more items to an existing container opens the container. If you a pass `kDataBrowserNoItem`, the items are added to the root container.

numItems
The number of items in the array pointed to by the `items` parameter.

items
A pointer to an array of item ID values for the items you want to add to the data browser. You supply item ID values based on your own identification scheme. If you pass `NULL`, each time you call `AddDataBrowserItems` the data browser generates item ID values starting at `1`. Calling the function in this way clears whatever items are in the container. Because of this clearing behavior, passing `NULL` is not recommended unless your application uses a data browser to display a simple list that is populated only once with data.

preSortProperty
The property ID of the column whose sorting order matches the sorting order of the `items` array. A property ID is a four-character sequence that you assign to represent a column in list view. Pass `kDataBrowserItemNoProperty` if the `items` array is not sorted or if you don't know the sorting order of your data. You'll get the best performance from this function if you provide a sorting order.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Return Value
A result code. If the item ID specified by the `container` parameter is not classified as a container, returns `errDataBrowserItemNotAdded` if you attempt to add subitems to it. See "Data Browser Result Codes" (page 202).

Discussion
Hierarchical lists are constructed in a top-down fashion. Your application must install all the top-level, or parent, item IDs in the data browser before it associates a list of item ID values as subitems. You can add items to a parent item only after the parent item is classified as a container. A container is an item for which the property `kDataBrowserItemIsContainerProperty` is set to `true`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

AddDataBrowserListViewColumn

Adds a column to a data browser that uses list view.

```
OSStatus AddDataBrowserListViewColumn (
   ControlRef browser,
   DataBrowserListViewColumnDesc *columnDesc,
   DataBrowserTableViewColumnIndex position
);
```

Parameters

browser
A data browser.

columnDesc
A pointer to the list view column description data structure that you have filled out with data that specifies the column property and display information for the column heading.

position
The position, among the columns already installed in the data browser, to insert this column. To insert this column to the right of all other columns, pass `kDataBrowserListViewAppendColumn`. The value `0` means the leftmost column.

Return Value
A result code; `paramErr` is returned if the `columnDesc` parameter is not properly initialized. See "Data Browser Result Codes" (page 202).

Discussion
Typically you use the function `AddDataBrowserListViewColumn` in these cases:

- When you create a data browser programmatically. If you use Interface Builder to design and lay out the data browser, you do not need to call the function `AddDataBrowserListViewColumn`. Interface Builder lets you position a column graphically and then specify the column description in the column pane of the Info window.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

- When you switch from column view to list view. Regardless of how you first create a data browser, if your application allows the user to switch between views, you need to add list view columns each time the view switches from column to list view.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

AutoSizeDataBrowserListViewColumns

Adjusts the size of columns displayed in list view to take best advantage of the available space.

```
OSStatus AutoSizeDataBrowserListViewColumns (
   ControlRef browser
);
```

Parameters

browser
A data browser.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
When you call the function `AutoSizeDataBrowserListViewColumns`, it first calculates whether there is extra space or not enough space in the data browser. Then, the columns are resized using the following rules:

- If there is extra space, the data browser gives as much of the extra space as possible to the first column (that is, the leftmost column) without exceeding the maximum width for the column. If there is still space available, the data browser gives as much of the remaining space as possible to the second column without exceeding the maximum width for the column. The data browser continues to disburse space in this manner until there is no more extra space. Thus, it is possible for the first column to get all the extra space.
- If space is needed to fit all the columns, the data browser takes as much of the needed space as possible from the rightmost column (that is, the last column) without letting the column width fall below the minimum width for the column. If more space is needed, the data browser takes as much of the needed space as possible from the next-to-the-last column without letting the column width fall below the minimum width for the column. The data browser continues to adjust space in this manner until all the columns fit within the data browser.

The function `AutoSizeDataBrowserListViewColumns` resizes only if the horizontal scroll bar is turned off. Your application can call the function `SetDataBrowserHasScrollBars` (page 110) to turn off the horizontal scroll bar.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

AXUIElementCreateWithDataBrowserAndItemInfo

Creates an AXUIElementRef that represents some part of a data browser accessibility hierarchy.

```
AXUIElementRef AXUIElementCreateWithDataBrowserAndItemInfo (
   ControlRef inDataBrowser,
   const DataBrowserAccessibilityItemInfo *inInfo
);
```

Parameters

inDataBrowser
A data browser.

inInfo
A DataBrowserAccessibilityItemInfo (page 164) structure describing the part of the data browser for which you want to create an AXUIElementRef.

Return Value
An AXUIElementRef representing the part, or NULL if an AXUIElementRef cannot be created to represent the part you specified.

Availability
Available in Mac OS X v10.4 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

AXUIElementGetDataBrowserItemInfo

Obtains a description of the part of a data browser represented by an AXUIElementRef.

```
OSStatus AXUIElementGetDataBrowserItemInfo (
   AXUIElementRef inElement,
   ControlRef inDataBrowser,
   UInt32 inDesiredInfoVersion,
   DataBrowserAccessibilityItemInfo *outInfo
);
```

Parameters

inElement
An AXUIElementRef representing part of a data browser.

inDataBrowser
A data browser.

inDesiredInfoVersion
The version of DataBrowserAccessibilityItemInfo (page 164) structure you want to get. Currently, the only supported version is zero, so you must pass 0 or 1 as the value of this parameter.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

outInfo

On input, a pointer to a `DataBrowserAccessibilityItemInfo` (page 164) structure. On return, the structure is filled in with a description of the part of the data browser that the `AXUIElementRef` specified by `inElement` represents.

Return Value

A result code. See "Data Browser Result Codes" (page 202). The function returns `noErr` if it was able to generate a description of the `AXUIElementRef`. If the `AXUIElementRef` does not represent the data browser you passed in, the function returns `paramErr`. If the `AXUIElementRef` represents some non-item part of the data browser, the function returns `errDataBrowserItemNotFound`.

Availability

Available in Mac OS X v10.4 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

CloseDataBrowserContainer

Closes a data browser container.

```
OSStatus CloseDataBrowserContainer (
   ControlRef browser,
   DataBrowserItemID container
);
```

Parameters

browser

A data browser.

container

The item ID of the container you want to close.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Normally the user navigates through a data hierarchy by clicking the disclosure triangle next to a container item in list view, or the container item (such as a folder icon) in column view. In either of these cases, the system automatically opens or closes the container. Under some circumstances your application may need to open or close a container programmatically, such as when you are restoring a display to its last known state. In such cases, you can call the function `OpenDataBrowserContainer` (page 100) to disclose items in a container or the function `CloseDataBrowserContainer` to hide items in a container.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

CopyDataBrowserEditText

Copies the text being edited by the user.

```
OSStatus CopyDataBrowserEditText (
   ControlRef browser,
   CFStringRef *text
);
```

Parameters

browser

A data browser.

text

On input, a `CFStringRef` variable initialized to anything other than `NULL`. See the Special Considerations for details. On return, a CFString object that contains a copy of the text edited by the user. You are responsible for releasing the string when you no longer need it.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

This function is useful only if an edit session is in progress for an item. You can check whether an edit session is open by calling the function `GetDataBrowserEditItem` (page 47).

Special Considerations

For versions of Mac OS X prior to v10.4, the `text` parameter must be set to any value other than `NULL`. Do not allocate the CFStringRef, otherwise your application will leak memory. Instead provide code similar to the following to initialize the variable:

```
CFStringRef myText = 0XFFFFFFFF;
```

Availability

Available in CarbonLib 1.5 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

CreateDataBrowserControl

Creates a data browser programmatically.

```
OSStatus CreateDataBrowserControl (
   WindowRef window,
   const Rect *boundsRect,
   DataBrowserViewStyle style,
   ControlRef *outControl
);
```

Parameters

window

The window in which to place the data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

boundsRect

A pointer to a rectangle that specifies the location where you want the control to appear in the window.

style

The view style to use. Pass the constant `kDataBrowserListView` to draw the data browser using list view or `kDataBrowserColumnView` draw the data browser using column view. See "View Styles" (page 202) for more information on these constants.

outControl

On input, a pointer to a control reference. On return, this is set to the newly created data browser. When you no longer need the data browser, call the Control Manager function `DisposeControl` to release it. When you dispose of the control, deallocate any universal procedure pointers you allocated for use with the control.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function creates a data browser programmatically. If you create a data browser using Interface Builder, you don't need to call `CreateDataBrowserControl`. Instead, you call the function `GetControlByID` to obtain a control reference that points to your data browser.

After you create a data browser by calling `CreateDataBrowserControl`, you can set such attributes as sorting order, scroll bars, and scroll position. See "Manipulating Data Browser Attributes" (page 14) for the functions you can use to set data browser attributes.

You need to set up the display characteristics of the data browser by calling the appropriate functions. See "Formatting Table View" (page 16), "Formatting List View" (page 17), and "Formatting Column View" (page 17) for information on the formatting functions you can call.

You need to call the functions `InitDataBrowserCallbacks` (page 79) and `SetDataBrowserCallbacks` (page 104) to install the callbacks needed for your data browser. At the very least, you must provide an item-data callback to add or change data items; you must do so regardless of the content your data browser displays—noncustom or custom. Otherwise, your data browser will be empty. See `DataBrowserItemDataProcPtr` (page 149) for more information. If you present hierarchical data in list view, or use column view for browsing data, you must provide a callback to handle item notifications. See `DataBrowserItemNotificationProcPtr` (page 154) and `DataBrowserItemNotificationWithItemProcPtr` (page 155).

You can optionally provide callbacks to:

- Perform sorting. See `DataBrowserItemCompareProcPtr` (page 147).
- Handle drag-and-drop behavior. See `DataBrowserAddDragItemProcPtr` (page 138), `DataBrowserAcceptDragProcPtr` (page 137), `DataBrowserReceiveDragProcPtr` (page 160), and `DataBrowserPostProcessDragProcPtr` (page 159).
- Provide contextual menus. See `DataBrowserGetContextualMenuProcPtr` (page 142) and `DataBrowserSelectContextualMenuProcPtr` (page 161).
- Display help tags. See `DataBrowserItemHelpContentProcPtr` (page 152).

If your data browser uses a list whose columns require custom drawing or behavior, you must also provide callbacks to handle the custom tasks. See `InitDataBrowserCustomCallbacks` (page 80) and `SetDataBrowserCustomCallbacks` (page 107) for more information on initializing and installing callbacks for custom behavior. The custom tasks you can handle in list view include:

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

- Drawing custom content. See `DataBrowserDrawItemProcPtr` (page 139).
- Supporting editing of custom content. See `DataBrowserEditItemProcPtr` (page 141). Note that editing is built-in for noncustom content.
- Performing hit-testing and tracking. See `DataBrowserHitTestProcPtr` (page 144) and `DataBrowserTrackingProcPtr` (page 163).
- Handling drag-and-drop behavior. See `DataBrowserItemDragRgnProcPtr` (page 150), `DataBrowserItemAcceptDragProcPtr` (page 146), and `DataBrowserItemReceiveDragProcPtr` (page 158).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

DataBrowserChangeAttributes

Sets the attributes for a data browser.

```
OSStatus DataBrowserChangeAttributes (
   ControlRef inDataBrowser,
   OptionBits inAttributesToSet,
   OptionBits inAttributesToClear
);
```

Parameters

inDataBrowser
A data browser.

inAttributesToSet
The attributes to set. For possible values, see "Data Browser Attributes" (page 177).

inAttributesToClear
The attributes to clear. For possible values, see "Data Browser Attributes" (page 177).

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in Mac OS X v10.4 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

DataBrowserGetAttributes

Gets the attributes of a data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus DataBrowserGetAttributes (
   ControlRef inDataBrowser,
   OptionBits *outAttributes
);
```

Parameters

inDataBrowser

A data browser.

outAttributes

The attributes to get. This parameter cannot be `NULL`. For possible values, see "Data Browser Attributes" (page 177).

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in Mac OS X v10.4 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

DataBrowserGetMetric

Gets the value of a specified data browser metric.

```
OSStatus DataBrowserGetMetric (
   ControlRef inDataBrowser,
   DataBrowserMetric inMetric,
   Boolean *outUsingDefaultValue,
   CGFloat *outValue
);
```

Parameters

inDataBrowser

A data browser.

inMetric

The data browser metric value to get. For possible values, see"Data Browser Metric Values" (page 178).

outUsingDefaultValue

On return, a Boolean whose value indicates whether the metric's value is determined by the data browser's default values. Pass `NULL` if you don't want this information.

outValue

On return, the value of the metric.

Return Value

A result code. See "Data Browser Result Codes" (page 202). If the `inDataBrowser` is not an instance of a data browser or if the value specified by `inMetric` is not known, `DataBrowserGetMetric` returns `paramErr`.

Availability

Available in Mac OS X v10.4 and later.

Not available to 64-bit applications.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

DataBrowserSetMetric

Sets the value of a specified data browser metric.

```
OSStatus DataBrowserSetMetric (
   ControlRef inDataBrowser,
   DataBrowserMetric inMetric,
   Boolean inUseDefaultValue,
   CGFloat inValue
);
```

Parameters

inDataBrowser
A data browser.

inMetric
The data browser metric whose value is to be set. For possible values, see "Data Browser Metric Values" (page 178).

inUsingDefaultValue
A Boolean whose value indicates whether you want the data browser to revert to the default value for the metric. If you pass `true`, `inValue` is ignored and a suitable default value is used. If you pass `false`, `inValue` is set as the value of the metric.

inValue
The value to set for the metric (if the value of `inUsingDefaultValue` is `false`).

Return Value
A result code. See "Data Browser Result Codes" (page 202). If the `inDataBrowser` is not an instance of a data browser or if the value specified by `inMetric` is not known, `DataBrowserSetMetric` returns `paramErr`.

Availability
Available in Mac OS X v10.4 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

DisposeDataBrowserAcceptDragUPP

Disposes of a universal procedure pointer to an accept-drag callback function.

```
void DisposeDataBrowserAcceptDragUPP (
   DataBrowserAcceptDragUPP userUPP
);
```

Parameters

userUPP
The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserAcceptDragProcPtr` (page 137) callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserAddDragItemUPP

Disposes of a universal procedure pointer to an add-drag-item callback function.

```
void DisposeDataBrowserAddDragItemUPP (
   DataBrowserAddDragItemUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserAddDragItemProcPtr` (page 138) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserDrawItemUPP

Disposes of a universal procedure pointer to a draw-item callback function.

```
void DisposeDataBrowserDrawItemUPP (
   DataBrowserDrawItemUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserDrawItemProcPtr` (page 139) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DisposeDataBrowserEditItemUPP

Disposes of a universal procedure pointer to an edit-item callback function.

```
void DisposeDataBrowserEditItemUPP (
   DataBrowserEditItemUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion

See the `DataBrowserEditItemProcPtr` (page 141) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

DisposeDataBrowserGetContextualMenuUPP

Disposes of a universal procedure pointer to a get-contextual-menu callback function.

```
void DisposeDataBrowserGetContextualMenuUPP (
   DataBrowserGetContextualMenuUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion

See the `DataBrowserGetContextualMenuProcPtr` (page 142) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

DisposeDataBrowserHitTestUPP

Disposes of a universal procedure pointer to a hit-test callback function.

```
void DisposeDataBrowserHitTestUPP (
   DataBrowserHitTestUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
See the `DataBrowserHitTestProcPtr` (page 144) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserItemAcceptDragUPP

Disposes of a universal procedure pointer to an item-accept-drag callback function.

```
void DisposeDataBrowserItemAcceptDragUPP (
   DataBrowserItemAcceptDragUPP userUPP
);
```

Parameters
userUPP
The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserItemAcceptDragProcPtr` (page 146) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserItemCompareUPP

Disposes of a universal procedure pointer to an item-comparison callback function.

```
void DisposeDataBrowserItemCompareUPP (
   DataBrowserItemCompareUPP userUPP
);
```

Parameters
userUPP
The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserItemCompareProcPtr` (page 147) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DisposeDataBrowserItemDataUPP

Disposes of a universal procedure pointer to an item-data callback function.

```
void DisposeDataBrowserItemDataUPP (
   DataBrowserItemDataUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion

See the `DataBrowserItemDataProcPtr` (page 149) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

DisposeDataBrowserItemDragRgnUPP

Disposes of a universal procedure pointer to an item-drag-region callback function.

```
void DisposeDataBrowserItemDragRgnUPP (
   DataBrowserItemDragRgnUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion

See the `DataBrowserItemDragRgnProcPtr` (page 150) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

DisposeDataBrowserItemHelpContentUPP

Disposes of a universal procedure pointer to an item-help-content callback function.

```
void DisposeDataBrowserItemHelpContentUPP (
   DataBrowserItemHelpContentUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
See the `DataBrowserItemHelpContentProcPtr` (page 152) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserItemNotificationUPP

Disposes of a universal procedure pointer to an item-notification callback function.

```
void DisposeDataBrowserItemNotificationUPP (
   DataBrowserItemNotificationUPP userUPP
);
```

Parameters
userUPP
The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserItemNotificationProcPtr` (page 154) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserItemNotificationWithItemUPP

Disposes of a universal procedure pointer to an item-notification-with-data callback function.

```
void DisposeDataBrowserItemNotificationWithItemUPP (
   DataBrowserItemNotificationWithItemUPP userUPP
);
```

Parameters
userUPP
The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserItemNotificationWithItemProcPtr` (page 155) callback function.

Availability
Available in CarbonLib 1.5 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DisposeDataBrowserItemReceiveDragUPP

Disposes of a universal procedure pointer to an item-receive-drag callback function.

```
void DisposeDataBrowserItemReceiveDragUPP (
   DataBrowserItemReceiveDragUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion

See the `DataBrowserItemReceiveDragProcPtr` (page 158) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

DisposeDataBrowserItemUPP

Disposes of a universal procedure pointer to an item-iterator callback function.

```
void DisposeDataBrowserItemUPP (
   DataBrowserItemUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion

See the `DataBrowserItemProcPtr` (page 157) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

DisposeDataBrowserPostProcessDragUPP

Disposes of a universal procedure pointer to a postprocess-drag callback function.

```
void DisposeDataBrowserPostProcessDragUPP (
   DataBrowserPostProcessDragUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
See the `DataBrowserPostProcessDragProcPtr` (page 159) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserReceiveDragUPP

Disposes of a universal procedure pointer to a receive-drag callback function.

```
void DisposeDataBrowserReceiveDragUPP (
   DataBrowserReceiveDragUPP userUPP
);
```

Parameters
userUPP
The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserReceiveDragProcPtr` (page 160) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

DisposeDataBrowserSelectContextualMenuUPP

Disposes of a universal procedure pointer to a select-contextual-menu callback function.

```
void DisposeDataBrowserSelectContextualMenuUPP (
   DataBrowserSelectContextualMenuUPP userUPP
);
```

Parameters
userUPP
The universal procedure pointer to dispose of.

Discussion
See the `DataBrowserSelectContextualMenuProcPtr` (page 161) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DisposeDataBrowserTrackingUPP

Disposes of a universal procedure pointer to a tracking callback function.

```
void DisposeDataBrowserTrackingUPP (
   DataBrowserTrackingUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer to dispose of.

Discussion

See the `DataBrowserTrackingProcPtr` (page 163) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

EnableDataBrowserEditCommand

Determines whether the data browser is currently able to process a given editing command.

```
Boolean EnableDataBrowserEditCommand (
   ControlRef browser,
   DataBrowserEditCommand command
);
```

Parameters

browser

A data browser.

command

The editing command you want to enable. You can pass any of the constants described in "Editing Commands" (page 181).

Return Value

A value of `true` if the requested editing command can be performed by the data browser at this time.

Discussion

Editing commands (Cut, Paste, Copy, and so on) can be enabled for an editable text field that is open and selected and for which the data browser is currently able to process the given command. For example, the data browser can process a Paste command only if there is text available on the Clipboard.

Editing commands are also available for a custom display type when the callbacks you install for the custom display indicate editing is available. Your application can call the function `EnableDataBrowserEditCommand` to discover if a specific editing command can be enabled. To execute an editing command, call the function `ExecuteDataBrowserEditCommand` (page 42).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

ExecuteDataBrowserEditCommand

Executes an editing command.

```
OSStatus ExecuteDataBrowserEditCommand (
   ControlRef browser,
   DataBrowserEditCommand command
);
```

Parameters

browser

A data browser.

command

The editing command you want to execute. You can pass any of the constants described in "Editing Commands" (page 181).

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Editing commands can be executed for an editable text field that is open and selected. Your application can check to see if the editing command is enabled by first calling the function `EnableDataBrowserEditCommand` (page 41).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

ForEachDataBrowserItem

Applies an item-iterator callback routine to each data item that meets the specified criteria.

```
OSStatus ForEachDataBrowserItem (
   ControlRef browser,
   DataBrowserItemID container,
   Boolean recurse,
   DataBrowserItemState state,
   DataBrowserItemUPP callback,
   void *clientData
);
```

Parameters

browser

A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

container

An item ID or the constant `kDataBrowserNoItem`. To iterate through items that are organized as subitems of a container item, pass the item ID for the container. To iterate through all items displayed at the root of the data browser, pass the constant `kDataBrowserNoItem`.

recurse

A value that indicates whether or not to traverse the entire item hierarchy when applying the callback specified by the `callback` parameter. Pass `true` to apply the callback to all items in the hierarchy. Pass `false` if you want to apply the callback only to those items at the top level of the container or data browser.

state

A value that specifies the state of the items to which to apply the callback. Pass `0` if you want to apply the callback to all items, regardless of state. Otherwise, pass one of the constants described in "Item States" (page 184).

callback

A universal procedure pointer to your item-iterator callback routine. This routine is called for every item ID that matches the specified criteria. See `DataBrowserItemProcPtr` (page 157) for more information on the callback routine to supply.

clientData

A pointer to a buffer, local variable, or other storage location created and disposed of by your application and needed by your callback routine.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

The function `ForEachDataBrowserItem` is useful for enumerating and performing an operation on a set of item IDs.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserActiveItems

Obtains what determines the active state of the items in a data browser.

```
OSStatus GetDataBrowserActiveItems (
   ControlRef browser,
   Boolean *active
);
```

Parameters

browser

A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

active

On input, a pointer to a Boolean variable. On return, the variable is set to `true` if the active state of each item in the list is determined by the item property `kDataBrowserItemIsActiveProperty`. Otherwise, the variable is set to `false` to indicate that all items are inactive.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserCallbacks

Obtains the callback routines installed for notifying your application of changes to a data browser and for providing the data to be displayed by the data browser.

```
OSStatus GetDataBrowserCallbacks (
   ControlRef browser,
   DataBrowserCallbacks *callbacks
);
```

Parameters

browser

The data browser whose callback routines you want to obtain.

callbacks

On input, a pointer to a `DataBrowserCallbacks` structure. On return, the structure contains universal procedure pointers to the callback routines installed for the data browser.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
When `GetDataBrowserCallacks` is used in conjunction with the function `SetDataBrowserCallbacks` (page 104), your application can override or replace one or more callbacks used by a data browser to notify your application of changes.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserColumnViewDisplayType

Obtains the display type for a column view.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserColumnViewDisplayType (
   ControlRef browser,
   DataBrowserPropertyType *propertyType
);
```

Parameters

browser

A data browser.

propertyType

On input, a pointer to a display type variable. On return, the variable is set to the data type or control that is displayed in the data browser. No display types other than `kDataBrowserIconAndTextType` are currently supported in column view. See "Display Types" (page 179) for more information.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserColumnViewPath

Obtains the current path for a selection in column view.

```
OSStatus GetDataBrowserColumnViewPath (
   ControlRef browser,
   Handle path
);
```

Parameters

browser

A data browser.

path

On input, a handle. On return, the handle contains an array of item ID values that specify the current path. Array element `0` is the root; array element `N-1` is the target. You must allocate the handle before calling this function, and you are responsible for disposing of it.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

GetDataBrowserColumnViewPathLength

Obtains the length of the current path for a column view.

```
OSStatus GetDataBrowserColumnViewPathLength (
   ControlRef browser,
   UInt32 *pathLength
);
```

Parameters

browser

A data browser.

pathLength

On input, a pointer to an unsigned 32-bit integer. On return, this value is set to the number of levels in the path for the currently selected item.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserCustomCallbacks

Obtains the callbacks installed to implement custom drawing and behavior for the content in a data browser.

```
OSStatus GetDataBrowserCustomCallbacks (
   ControlRef browser,
   DataBrowserCustomCallbacks *callbacks
);
```

Parameters

browser

A data browser.

callbacks

On input, a pointer to a `DataBrowserCustomCallbacks` structure. On return, the structure contains universal procedure pointers to the custom callback routines installed for the data browser.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

When `GetDataBrowserCustomCallbacks` is used in conjunction with the function `SetDataBrowserCustomCallbacks` (page 107), your application can temporarily override or replace one or more callbacks used by a data browser to support custom drawing and custom behavior.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserEditItem

Obtains the item ID and property ID values of the current editing session.

```
OSStatus GetDataBrowserEditItem (
   ControlRef browser,
   DataBrowserItemID *item,
   DataBrowserPropertyID *property
);
```

Parameters

browser

A data browser.

item

On input, a pointer to an item ID variable. On return, the variable is set to the item ID of the item that is being edited. If there is no editing session in progress, this parameter is set to `kDataBrowserNoItem`.

property

On input, a pointer to a property ID variable. On return, the variable is set to the property ID of the item that is being edited. If there is no editing session in progress, this parameter is set to `0`.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserEditText

Obtains the text being edited by the user.

Not Recommended

```
OSStatus GetDataBrowserEditText (
   ControlRef browser,
   CFMutableStringRef text
);
```

Parameters

browser

A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

text

On return, the CFMutableString object is set to the text being edited by the user. Your application must allocate this object and pass it to the data browser. The data browser sets its contents to the current contents of the edit session text field. You must release this object when you no longer need it.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function does not work. Instead use `CopyDataBrowserEditText` (page 29).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserHasScrollBars

Obtains the display state of horizontal and vertical scroll bars for a list view data browser.

```
OSStatus GetDataBrowserHasScrollBars (
   ControlRef browser,
   Boolean *horiz,
   Boolean *vert
);
```

Parameters

browser

A list view data browser.

horiz

On input, a pointer to a Boolean variable. On return, the variable is set to `true` if the browser control has a horizontal scroll bar.

vert

On input, a pointer to a Boolean variable. On return, the variable is set to `true` if the browser control has a vertical scroll bar.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
The function `GetDataBrowserHasScrollBars` is useful for determining if the browser control currently has scroll bars. For example, you would call the function `AutoSizeDataBrowserListViewColumns` (page 26) only after you have determined the data browser does not have a horizontal scroll bar.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

GetDataBrowserItemCount

Obtains the number of items whose state matches the specified state.

```
OSStatus GetDataBrowserItemCount (
   ControlRef browser,
   DataBrowserItemID container,
   Boolean recurse,
   DataBrowserItemState state,
   ItemCount *numItems
);
```

Parameters

browser

A data browser.

container

An item ID or the constant `kDataBrowserNoItem`. To obtain the number of items that are organized as subitems of a container item, pass the item ID for the container. To obtain the number of items displayed at the root of the data browser, provide the constant `kDataBrowserNoItem`.

recurse

A value that indicates whether or not to traverse the entire item hierarchy when counting. Pass `true` to obtain a count for all items in the hierarchy. Pass `false` if you want to count only those items at the top level of the container or data browser.

state

A value that specifies the state of the items to obtain. Only items that have this state are counted. Pass `kDataBrowserItemAnyState` if you want to count all items regardless of state. Otherwise, pass one of the constants described in "Item States" (page 184).

numItems

On input, a pointer to an unsigned 32-bit integer. On return, this value is set to the number of items in the container that have the specified state.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

GetDataBrowserItemDataBooleanValue

Obtains the Boolean value for an item.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserItemDataBooleanValue (
   DataBrowserItemDataRef itemData,
   Boolean *theData
);
```

Parameters

itemData

The item data reference for the item whose Boolean value you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataBooleanValue`.

theData

On input, a pointer to a Boolean variable. On return, the variable is set to the Boolean value.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

You can obtain Boolean values for the following properties:

- `kDataBrowserItemIsActiveProperty`
- `kDataBrowserItemIsSelectableProperty`
- `kDataBrowserItemIsEditableProperty`
- `kDataBrowserItemIsContainerProperty`
- `kDataBrowserItemIsOpenableProperty`
- `kDataBrowserItemIsClosableProperty`
- `kDataBrowserItemIsSortableProperty`

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserItemDataButtonValue

Obtains the value for a checkbox.

```
OSStatus GetDataBrowserItemDataButtonValue (
   DataBrowserItemDataRef itemData,
   ThemeButtonValue *theData
);
```

Parameters

itemData

The item data reference for the item whose checkbox setting you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataButtonValue`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

theData

On input, a pointer to a theme button value variable. On return, the variable is set to the checkbox setting. The value can be one of the following theme button value constants defined by the Appearance Manager,:

- `kThemeButtonOff` indicates a checkbox that is not selected.
- `kThemeButtonOn` indicates a checkbox that is selected.
- `kThemeButtonMixed` draws a checkbox that in a mixed state, indicating that a setting is on for some items in a selection and off for others.

See *Appearance Manager Reference* for more information.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your item-data callback calls this function in response to a set-data request for items that have the display type `kDataBrowserCheckboxType`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataDateTime

Obtains, as a 32-bit value, the date and time value displayed.

```
OSStatus GetDataBrowserItemDataDateTime (
   DataBrowserItemDataRef itemData,
   SInt32 *theData
);
```

Parameters

itemData

The item data reference for the item whose date and time value you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataDateTime`.

theData

On input, a 32-bit value. On return, the value is set to the number of elapsed seconds since midnight, January 1, 1904. For more information about date and time encodings used in the Mac OS, see *Date, Time, and Measurement Utilities Reference* in Carbon Text & International Documentation.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function works only with items that have the property `kDataBrowserDateTimeType`. If the column has the property `kDataBrowserRelativeDateTime`, the date is displayed relative to the current time for the computer. For example, a time 24 hours prior to the current time is displayed as "Yesterday." Other examples of relative date and time values are "Today, 1:45 PM" and "Yesterday, 7:30 AM."

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataDrawState

Determines whether a checkbox is in the active or inactive state.

```
OSStatus GetDataBrowserItemDataDrawState (
   DataBrowserItemDataRef itemData,
   ThemeDrawState *theData
);
```

Parameters

itemData

The item data reference for the checkbox whose drawing state you want to obtain. This value is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataDrawState`.

theData

On input, a pointer to a theme draw state variable. On return, the variable is set to the drawing state for the item, either `kThemeStateInactive` or `kThemeStateActive`. See *Appearance Manager Reference* for more information on these constants.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your item-data callback calls this function in response to a get-data request for items that have display type `kDataBrowserCheckboxType`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataIcon

Obtains the icon drawn for an item.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserItemDataIcon (
   DataBrowserItemDataRef itemData,
   IconRef *theData
);
```

Parameters

itemData

The item data reference for the item whose icon you want to obtain. This value is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataIcon`.

theData

On input, a pointer to an `IconRef` variable. On return, the variable is set to the icon that is displayed. You are responsible for disposing of the `IconRef`.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

You call the function `GetDataBrowserItemDataIcon` from within a `DataBrowserItemDataProcPtr` (page 149) callback routine to obtain the icon drawn in a column that has the `kDataBrowserIconType` display type or the `kDataBrowserIconAndTextType` display type.

Availability

Available in CarbonLib 1.5 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserItemDataIconTransform

Obtains the transformation currently used to display an icon.

```
OSStatus GetDataBrowserItemDataIconTransform (
   DataBrowserItemDataRef itemData,
   IconTransformType *theData
);
```

Parameters

itemData

The item data reference for the item whose icon transformation you want to obtain. This value is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataIconTransform`.

theData

On input, an icon transformation type variable. On return, the variable is set to an icon transformation type. This value can be any of the icon transformation constants defined by Icon Services and Utilities. See *Icon Services and Utilities Reference* for more information.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

This function works only with items that have either the property `kDataBrowserIconAndTextType` or `kDataBrowserIconType`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataItemID

Obtains the item ID for an item whose property is another item's ID.

```
OSStatus GetDataBrowserItemDataItemID (
   DataBrowserItemDataRef itemData,
   DataBrowserItemID *theData
);
```

Parameters

itemData

The item data reference passed to your item-data callback.

theData

On input, a pointer to an item ID variable. On return, the variable is set to the item ID associated with the `itemData` parameter.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Typically you do not need to call this function. This function is used for item properties `kDataBrowserParentContainerProperty` or `kDataBrowserContainerAliasIDProperty`.

Availability
Available in CarbonLib 1.5 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataLongDateTime

Obtains, as a 64-bit value, the date and time value displayed.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserItemDataLongDateTime (
   DataBrowserItemDataRef itemData,
   LongDateTime *theData
);
```

Parameters

itemData

The item data reference for the item whose long date and time value you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataLongDateTime`.

theData

On input, a 64-bit value. On return, the value is set to the number of seconds elapsed since midnight, January 1, 1904. For more information about date and time encodings used in the Mac OS, see *Date, Time, and Measurement Utilities Reference* in Carbon Text & International Documentation.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

This function works only with items that have the property `kDataBrowserDateTimeType`. If the column has the property `kDataBrowserRelativeDateTime`, the date is displayed relative to the current time for the computer. For example, a time 24 hours prior to the current time is displayed as "Yesterday." Other examples of relative date and time values are "Today, 1:45 PM" and "Yesterday, 7:30 AM."

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserItemDataMaximum

Obtains the maximum integer value that can be displayed; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

```
OSStatus GetDataBrowserItemDataMaximum (
   DataBrowserItemDataRef itemData,
   SInt32 *theData
);
```

Parameters

itemData

The item data reference for the item whose maximum value you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataMaximum`.

theData

On input, a pointer to a signed 32-bit integer. On return, this value is set to the maximum value that can be displayed for the item.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataMenuRef

Obtains the pop-up menu displayed.

```
OSStatus GetDataBrowserItemDataMenuRef (
   DataBrowserItemDataRef itemData,
   MenuRef *theData
);
```

Parameters

itemData

The item data reference for the item whose pop-up menu you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataMenuRef`.

theData

On input, a pointer to a menu reference. On return, this is set to the currently displayed pop-up menu. The system retains the menu reference that you pass; you must release it when you no longer need it.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your item-data callback calls this function in response to a get-data request for items that have the display type `kDataBrowserPopupMenuType`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataMinimum

Obtains the minimum integer value that can be displayed for an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserItemDataMinimum (
   DataBrowserItemDataRef itemData,
   SInt32 *theData
);
```

Parameters

itemData

The item data reference for the item whose minimum value you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataMinimum`.

theData

On input, a pointer to a signed 32-bit integer. On return, this value is set to the minimum value that can be displayed for the item.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserItemDataProperty

Obtains the column property ID for the column in which an item resides.

```
OSStatus GetDataBrowserItemDataProperty (
   DataBrowserItemDataRef itemData,
   DataBrowserPropertyID *theData
);
```

Parameters

itemData

The item data reference for the item whose property ID you want to obtain. This value is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataProperty`.

theData

On input, a pointer to a property ID variable. On return, the variable is set to the property ID for the item.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

When your item-data callback is invoked for an item that has the property `kDataBrowserItemIsEditableProperty`, you call the function `GetDataBrowserItemDataProperty` to obtain the column property ID. Then, your callback can use the column property ID to determine whether the item is in a column whose data can be edited.

For example, consider a list view data browser whose columns are titled "Name" and "Date Modified." Let's say Name can be modified by the user, but the Date Modified column cannot. If the user clicks an item in one of the columns, your item-data callback is called to find out whether the clicked column is editable. Your

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

callback needs to find out which column the "is editable" request is being made for by calling the function `GetDataBrowserItemDataProperty`. In this example, after you obtain the property ID, you would check whether the column is the Date Modified column or the Name column. You'd allow editing only if the item is in the Name column.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataRGBColor

Obtains the color used to draw an item.

```
OSStatus GetDataBrowserItemDataRGBColor (
   DataBrowserItemDataRef itemData,
   RGBColor *theData
);
```

Parameters

itemData
The item data reference for the item whose color you want to obtain. This value is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataRGBColor`.

theData
On input, an RGB color variable. On return, the variable is set to the RGB values that specify the color of the item.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your item-data callback calls this function in response to a get-data request for items that have the display type `kDataBrowserIconType` or `kDataBrowserIconAndTextType`.

As of Mac OS X 10.3, this function does nothing.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataText

Obtains the text entered by the user.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserItemDataText (
   DataBrowserItemDataRef itemData,
   CFStringRef *theData
);
```

Parameters

itemData

The item data reference for the item whose text you want to obtain. This value is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataText`.

theData

On input, a `CFStringRef` variable. On return, a CFString object that contains the text. Your application must release the CFString object when it is no longer needed.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
You can call the function `GetDataBrowserItemDataText` from inside an item-data callback routine when the callback's `setValue` parameter is `true`. A value of `true` indicates that the displayed text has been modified by the user. In that case, your application calls `GetDataBrowserItemDataText` to retrieve the modified text.

Note that a column is editable only if the `kDataBrowserPropertyIsEditable` flag is set for the column.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemDataValue

Obtains the value of an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

```
OSStatus GetDataBrowserItemDataValue (
   DataBrowserItemDataRef itemData,
   SInt32 *theData
);
```

Parameters

itemData

The item data reference for the item whose integer value you want to obtain. The item data reference is passed to the callback routine from which you are calling the function `GetDataBrowserItemDataValue`.

theData

On input, a pointer to a signed 32-bit integer. On return, it is set to the displayed value.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
Your application calls the function `GetDataBrowserItemDataValue` to obtain a new value for a display type when your item-data callback routine is called with the `setValue` parameter set to `true`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserItemPartBounds

Obtains the bounds of a visual part of an item.

```
OSStatus GetDataBrowserItemPartBounds (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserPropertyID property,
   DataBrowserPropertyPart part,
   Rect *bounds
);
```

Parameters

browser
A data browser.

item
The item ID that identifies the row.

property
The property ID that identifies the column.

part
The part for which you want to obtain information. The information requested depends on the type of information displayed in the column. It is up to your application to ensure it requests the appropriate information. See "Property Parts" (page 195) for a list of the constants you can provide in this parameter.

bounds
On input, a pointer to a rectangle. On return, the rectangle contains the bounds for the specified part.

Return Value
A result code. If the item is not visible (scrolled off the screen), returns the result `ItemNotFound`. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

GetDataBrowserItems

Obtains a list of the items that match a specified state; operates on items in the root container or traverses items in the data hierarchy.

```
OSStatus GetDataBrowserItems (
   ControlRef browser,
   DataBrowserItemID container,
   Boolean recurse,
   DataBrowserItemState state,
   Handle items
);
```

Parameters

browser

A data browser.

container

An item ID or the constant `kDataBrowserNoItem`. To obtain a list of items that are organized as subitems of a container, pass the item ID of the container item. To obtain a list of items displayed in the root container, pass the constant `kDataBrowserNoItem`.

recurse

A value that indicates whether or not to traverse the entire item hierarchy when obtaining item IDs. Pass `true` to obtain item IDs for all items in the hierarchy. Pass `false` if you want to count only those item IDs at the top level of the container. If you pass `true`, you obtain a flattened list of item IDs. The list reflects the hierarchy maintained internally by the data browser and might not reflect the order of the items as they appear onscreen to the user.

state

The state of the items to obtain. Only items that have this state are returned in the `items` parameter. Pass `0` if you want to obtain all items regardless of state. Otherwise, pass one of the constants described in "Item States" (page 184).

items

On return, the contents of the handle contain an array of item ID values for the matching items. You must allocate and dispose of the handle. To determine the number of items in the array, call the function `GetHandleSize` and divide by the size of `DataBrowserItemID`. Note that the handle contents are completely replaced by the returned array.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

The function `GetDataBrowserItems` is a powerful routine for gathering information about the items displayed in a data browser. For example, to obtain a list of all the items the user has selected in a list, call the function with the `state` parameter set to `kDataBrowserItemIsSelected`. If your application is interested only in determining the number of items in a selection (and not the item IDs of those items), call the function `GetDataBrowserItemCount` (page 49).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

GetDataBrowserItemState

Obtains the state of an item.

```
OSStatus GetDataBrowserItemState (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserItemState *state
);
```

Parameters

browser

A data browser.

item

The item ID of the item whose state you want to check.

state

On input, a pointer to an item state variable. On return, the variable is set to a value that specifies the state of the item. See "Item States" (page 184) for a description of the values that can be returned.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserListViewDisclosureColumn

Obtains the property ID of the column whose items can display a disclosure triangle, and tells whether a disclosed item expands the row or adds rows.

```
OSStatus GetDataBrowserListViewDisclosureColumn (
   ControlRef browser,
   DataBrowserTableViewColumnID *column,
   Boolean *expandableRows
);
```

Parameters

browser

A data browser.

column

On input, a pointer to a column ID variable. On return, the variable is set to the property ID of the currently selected column. If there is no disclosure column, the variable is set to `kDataBrowserItemNoProperty`. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

expandableRows

On input, a pointer to a Boolean variable. On return, the variable specifies how a disclosed row behaves. The value `true` means that a container opens as a single row with an expanded height. The value `false` means a container opens to expose individual rows. See the Discussion for more details on expandable rows.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
When the `expandableRows` variable is set to `true`:

- Disclosure triangles are drawn top-justified in the row.
- Custom row height, if any, for that row is respected only while the row is disclosed. At other times, the default row height is used.

When the `expandableRows` variable is set to `false`:

- Disclosure triangles are centered vertically in the row.
- Custom row height, if any, for that row is always respected.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserListViewHeaderBtnHeight

Obtains the height of the rectangular area where the column title appears.

```
OSStatus GetDataBrowserListViewHeaderBtnHeight (
   ControlRef browser,
   UInt16 *height
);
```

Parameters

browser

A data browser.

height

On input, a pointer to an unsigned 16-bit integer. On return, this value is set to the height of the rectangular area where the column title appears. You can save this value if you plan to call the function `SetDataBrowserListViewHeaderBtnHeight` (page 121) to turn off header button display. You can then use the value later to turn on header button display.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserListViewHeaderDesc

Obtains a header description for a column in list view.

```
OSStatus GetDataBrowserListViewHeaderDesc (
   ControlRef browser,
   DataBrowserTableViewColumnID column,
   DataBrowserListViewHeaderDesc *desc
);
```

Parameters

browser

A data browser.

column

The property ID for the column whose list view header description you want to obtain. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

desc

On input, a pointer to a list view header description data structure. On return, the structure contains the header description for the specified column in list view.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
If your application allows the user to switch between column and list views, you can call this function to obtain the current header description and then save the description before you switch from list to column view. When you switch from column to list view, you can restore the list view header information by calling the function `SetDataBrowserListViewHeaderDesc` (page 122) passing the header information you saved.

Availability
Available in CarbonLib 1.5 and later.
Available in Mac OS X version 10.2 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserListViewUsePlainBackground

Determines whether list view is set to use a plain white background.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserListViewUsePlainBackground (
   ControlRef browser,
   Boolean *usePlainBackground
);
```

Parameters

browser

A data browser.

usePlainBackground

On input, a pointer to a Boolean variable. On return, the variable is `true` if list view is set to use a plain white background. Regardless of the value that is returned, Mac OS X supports only a plain white background. Mac OS 9 supports a plain white background as well as a shaded background.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserPropertyFlags

Obtains the appearance and behavior attributes for a column.

```
OSStatus GetDataBrowserPropertyFlags (
   ControlRef browser,
   DataBrowserPropertyID property,
   DataBrowserPropertyFlags *flags
);
```

Parameters

browser

A data browser.

property

The property ID of the column whose properties you want to obtain.

flags

On input, a data browser property flags variable. On return, the variable is set to the property flags that specify the appearance and behavior attributes for a column. A `DataBrowserPropertyFlags` value is a 32-bit value that is divided into four parts as follows:

- Bits 0–7 specify properties applied to the data browser as a whole—see "Property Flags: Universal" (page 188)
- Bits 8–15 modify display behavior—see "Property Flags: Modifiers" (page 189)
- Bits 16–23 are properties specific to list view—see "Property Flags: Offset and Mask for List View Properties" (page 192) and "Property Flags: List View Column Behavior" (page 193)
- Bits 24–31 can be defined by your application—see "Property Flags: Offset and Mask for Client-Defined Properties" (page 194)

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserScrollBarInset

Obtains the inset rectangle used by a data browser to position the scroll bar.

```
OSStatus GetDataBrowserScrollBarInset (
   ControlRef browser,
   Rect *insetRect
);
```

Parameters

browser
A data browser.

insetRect
On input, a pointer to a rectangle structure. On return, the rectangle contains the current inset settings for the data browser scroll bars. The `left` and `right` fields contain the horizontal inset values for the horizontal scroll bar, and the `top` and `bottom` fields contain the vertical inset values for the vertical scroll bar.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your application can call the functions `GetDataBrowserScrollBarInset` and `SetDataBrowserScrollBarInset` (page 124) if you want to place placards or controls beside the horizontal scroll bars or above the vertical ones. To do so, first call `GetDataBrowserScrollBarInset` to obtain the current settings. After modifying the current inset settings to provide space for the placard or control, call `SetDataBrowserScrollBarInset` with the new values.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserScrollPosition

Obtains the scrolling position of a list.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserScrollPosition (
   ControlRef browser,
   UInt32 *top,
   UInt32 *left
);
```

Parameters

browser

A data browser.

top

On input, a pointer to an unsigned 32-bit integer. On return, this value is set to the current vertical scrolling position.

left

On input, a pointer to an unsigned 32-bit integer. On return, this value is set to the current horizontal scrolling position.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Normally, you use the function `GetDataBrowserScrollPosition` in conjunction with `SetDataBrowserScrollPosition` (page 125) to save and restore the scrolling position of a list to the user's last scrolling position. These functions should not be used to scroll particular cells into the view. For that, call the function `RevealDataBrowserItem`.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserSelectionAnchor

Obtains the first and last items in a selection.

```
OSStatus GetDataBrowserSelectionAnchor (
   ControlRef browser,
   DataBrowserItemID *first,
   DataBrowserItemID *last
);
```

Parameters

browser

A data browser.

first

On input, a pointer to an item ID variable. On return, the variable is set to the item ID of the first item in the selection.

last

On input, a pointer to an item ID variable. On return, the variable is set to the item ID of the last item in the selection.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserSelectionFlags

Obtains the current selection behavior for a data browser.

```
OSStatus GetDataBrowserSelectionFlags (
   ControlRef browser,
   DataBrowserSelectionFlags *selectionFlags
);
```

Parameters

browser
A data browser.

selectionFlags
On input, a data browser selection flags variable. On return, the variable is set to the current selection flags. See "User Selection Flags" (page 200) for a list of the flags that can be returned.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Selection flags specify the selection behavior available to the user, such as whether the user can select discontinuous items.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserSortOrder

Gets the sorting order of the list view column that's currently set for sorting.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserSortOrder (
   ControlRef browser,
   DataBrowserSortOrder *order
);
```

Parameters

browser

A data browser.

order

On input, a pointer to a sorting order variable. On return, the variable is set to the sorting order of the current sort column in list view. See "Sorting Orders" (page 198) for a list of the values that can be returned.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserSortProperty

Obtains the property ID of the column currently used for sorting in list view.

```
OSStatus GetDataBrowserSortProperty (
   ControlRef browser,
   DataBrowserPropertyID *property
);
```

Parameters

browser

A data browser.

property

On input, a pointer to a property ID variable. On return, the variable is set to the property ID of the column used for sorting.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

You can call the function `GetDataBrowserSortProperty` to discover the property ID of the column currently used for sorting. To designate another column for the sorting operation, call the function `SetDataBrowserSortProperty` (page 127).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

GetDataBrowserTableViewColumnCount

Obtains the number of columns in a data browser.

```
OSStatus GetDataBrowserTableViewColumnCount (
   ControlRef browser,
   UInt32 *numColumns
);
```

Parameters

browser

A data browser.

numColumns

On input, a pointer to an unsigned 32-bit integer. On return, this value is set to the number of columns in the data browser.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

GetDataBrowserTableViewColumnPosition

Obtains the column position for an item in a data browser.

```
OSStatus GetDataBrowserTableViewColumnPosition (
   ControlRef browser,
   DataBrowserTableViewColumnID column,
   DataBrowserTableViewColumnIndex *position
);
```

Parameters

browser

A data browser.

column

The property ID for the list view column for which you want to obtain the position. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

position

On input, a pointer to a column index variable. On return, the variable is set to the column index for the item; `0` is the leftmost column.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserTableViewColumnProperty

Obtains the property ID for a column in a data browser.

```
OSStatus GetDataBrowserTableViewColumnProperty (
   ControlRef browser,
   DataBrowserTableViewColumnIndex column,
   DataBrowserTableViewColumnID *property
);
```

Parameters

browser
A data browser.

column
The column index of the column whose property ID you want to obtain.

property
On input, a pointer to a column ID variable. On return, the variable is set to the property ID for the column. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserTableViewColumnWidth

Obtains the default column width used for all columns in a data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserTableViewColumnWidth (
   ControlRef browser,
   UInt16 *width
);
```

Parameters

browser

A data browser.

width

On input, a pointer to an unsigned 16-bit integer. On return, this value is set to the column width, in pixels.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserTableViewGeometry

Determines whether columns and rows are set to have variable widths.

```
OSStatus GetDataBrowserTableViewGeometry (
   ControlRef browser,
   Boolean *variableWidthColumns,
   Boolean *variableHeightRows
);
```

Parameters

browser

A data browser.

variableWidthColumns

On input, a pointer to a Boolean variable. On return, the variable is set to `true` if column widths can be changed or `false` if they cannot be changed.

variableHeightRows

On input, a pointer to a Boolean variable. On return, the variable is set to `true` if row heights can be changed or `false` if they cannot be changed.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

GetDataBrowserTableViewHiliteStyle

Obtains the highlighting style used for a list view data browser.

```
OSStatus GetDataBrowserTableViewHiliteStyle (
   ControlRef browser,
   DataBrowserTableViewHiliteStyle *hiliteStyle
);
```

Parameters

browser

A list view data browser.

hiliteStyle

On input, a pointer to a highlighting style variable. On return, the variable is set to the highlighting style in use. See "Table View Highlighting Styles" (page 198) for a description of the values that can be returned.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserTableViewItemID

Obtains the item ID for the item displayed in the specified row.

```
OSStatus GetDataBrowserTableViewItemID (
   ControlRef browser,
   DataBrowserTableViewRowIndex row,
   DataBrowserItemID *item
);
```

Parameters

browser

A data browser.

row

The row index for the item. The row index is the visual order of rows in the table onscreen. Rows are numbered starting at the top of the table, with the value `0`, and proceeding sequentially to the bottom of the table.

item

On input, a pointer to an item ID variable. On return, the variable is set to the item ID of the data displayed in the row.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
This function is useful only in edge case situations for which you need to know what item ID is displayed in a particular row. For example, if you are performing some fairly involved and complex custom hit-testing, you might need to call this function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserTableViewItemRow

Obtains the visual position for the specified item in list view.

```
OSStatus GetDataBrowserTableViewItemRow (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserTableViewRowIndex *row
);
```

Parameters

browser
A data browser.

item
The item ID for the item whose row index you want to obtain.

row
On input, a pointer to a row index variable. On return, the variable is set to the row index for the item.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
You can use this function for items in list view. It is the inverse of the function `GetDataBrowserTableViewItemID` (page 73).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserTableViewItemRowHeight

Obtains the row height for a single row in a list view data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus GetDataBrowserTableViewItemRowHeight (
   ControlRef browser,
   DataBrowserItemID item,
   UInt16 *height
);
```

Parameters

browser

A data browser.

item

The item ID of the item whose row height you want to obtain.

height

On input, a pointer to an unsigned 16-bit integer. On return, this value is set to the row height, in pixels.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

GetDataBrowserTableViewNamedColumnWidth

Obtains the column width for a single column in a data browser.

```
OSStatus GetDataBrowserTableViewNamedColumnWidth (
   ControlRef browser,
   DataBrowserTableViewColumnID column,
   UInt16 *width
);
```

Parameters

browser

A data browser.

column

The property ID for the list view column whose width you want to obtain. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

width

On input, a pointer to an unsigned 16-bit integer. On return, this value is set to the width of the column, in pixels.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Not available to 64-bit applications.

Declared In
HIDataBrowser.h

GetDataBrowserTableViewRowHeight

Obtains the default row height used for all rows in a data browser.

```
OSStatus GetDataBrowserTableViewRowHeight (
   ControlRef browser,
   UInt16 *height
);
```

Parameters

browser

A data browser.

height

On input, a pointer to an unsigned 16-bit integer. On return, this value is set to the row height, in pixels.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

GetDataBrowserTarget

Obtains the target for the data browser

```
OSStatus GetDataBrowserTarget (
   ControlRef browser,
   DataBrowserItemID *target
);
```

Parameters

browser

A data browser.

target

On input, a pointer to an item ID variable. On return, the variable is set to the item ID for the currently assigned target.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
In column view, the target is the rightmost column. In list view, the target can be thought of as the root container.

Your application can call the function `SetDataBrowserTarget` (page 133) to set an item ID to use as a target if you do not want to use the default target set by the data browser.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserUserState

Obtains the current view style settings for a list view.

```
OSStatus GetDataBrowserUserState (
   ControlRef browser,
   CFDictionaryRef *stateInfo
);
```

Parameters

browser
A data browser.

stateInfo
On input, a pointer to a `CFDictionaryRef`. On return, a CFDictionary object that contains the current view style settings. You must release the object when you no longer need it by calling the function `CFRelease`. Note that although this parameter is typed as a CFData object, you must treat the result as a CFDictionary object because that is what the system fills out and returns to you.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
You typically use this function to obtain the current user settings for the data browser so that you can save the settings to a preferences file. User settings include data such as sorting order, sorting column, and column widths. Later, you can restore the settings by calling the function `SetDataBrowserUserState` (page 134).

If you want to save the user settings to disk, you need to determine the length of the user-settings data in bytes. The following code shows how to calculate this length. First you need to convert the CFDictionary object you obtain from the function `GetDataBrowserUserState` to a property list. Then you can call the function `CFDataGetLength` to obtain the length, in bytes, of the property list.

```
CFDataRef myUserState = NULL;
OSStatus status;

ControlRef browser = GetDataBrowserFromWindow (window);
status = GetDataBrowserUserState (browser, &myUserState);

if (noErr == status)
{
```


2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
  CFDataRef myTempDataRef = NULL;
  CFIndex index;

  if (myUserState != NULL)
  {
    // Convert the user state dictionary to a property list.
    myTempDataRef = CFPropertyListCreateXMLData (NULL,
                        (CFPropertyListRef) myUserState);
    if (myTempDataRef != NULL)
    {
        // Get the length, in bytes
      index = CFDataGetLength (myTempDataRef);
        // Call your function to save the data
        // You need to release the CFDataRef you created
      CFRelease (myTempDataRef);
    }
  }
 CFRelease (myUserState);
}
```

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

GetDataBrowserViewStyle

Obtains the current view style for the specified data browser.

```
OSStatus GetDataBrowserViewStyle (
   ControlRef browser,
   DataBrowserViewStyle *style
);
```

Parameters

browser
A data browser.

style
On input, a pointer to a view style variable. On return, the variable is set to the current view style for the specified data browser; can be either list view (`kDataBrowserListView`) or column view (`kDataBrowserColumnView`). See "View Styles" (page 202) for more information on these constants.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

InitDataBrowserCallbacks

Initializes a data browser callback structure in preparation for adding your own callbacks to the structure.

```
OSStatus InitDataBrowserCallbacks (
   DataBrowserCallbacks *callbacks
);
```

Parameters

callbacks

A pointer to a `DataBrowserCallbacks` structure. Before calling the function `InitDataBrowserCallbacks`, set the `version` field of this structure to `kDataBrowserLatestCallbacks`. On return, the fields in this structure are set to `NULL`.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

After you call this function, set the appropriate fields in the `DataBrowserCallbacks` structure to your callbacks. The `DataBrowserCallbacks` structure contains fields for the following:

- `DataBrowserItemDataProcPtr` (page 149)
- `DataBrowserItemCompareProcPtr` (page 147)
- `DataBrowserItemNotificationProcPtr` (page 154) or `DataBrowserItemNotificationWithItemProcPtr` (page 155) (Mac OS X only)
- `DataBrowserAddDragItemProcPtr` (page 138)
- `DataBrowserAcceptDragProcPtr` (page 137)
- `DataBrowserReceiveDragProcPtr` (page 160)
- `DataBrowserPostProcessDragProcPtr` (page 159)
- `DataBrowserGetContextualMenuProcPtr` (page 142)
- `DataBrowserSelectContextualMenuProcPtr` (page 161)
- `DataBrowserItemHelpContentProcPtr` (page 152)

After you assign your callbacks to the appropriate field, call the function `SetDataBrowserCallbacks` (page 104).

Note that this is a different set of callbacks from those that are assigned to fields in the `DataBrowserCustomCallbacks` data structure.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

InitDataBrowserCustomCallbacks

Initializes the data browser custom callback structure in preparation for adding your own callbacks for custom drawing or custom behavior to the structure.

```
OSStatus InitDataBrowserCustomCallbacks (
   DataBrowserCustomCallbacks *callbacks
);
```

Parameters

callbacks

A pointer to a `DataBrowserCustomCallbacks` structure. Before calling the function `InitDataBrowserCustomCallbacks`, set the `version` field of this structure to `kDataBrowserLatestCustomCallbacks`. On return, the fields in this structure are set to `NULL`.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Custom callbacks refer to those callback routines that are used to implement custom drawing or custom behavior in your data browser. The data browser API supports a limited set of built-in display types: text, icon and text, checkboxes, and so forth. If you want to display something else, you install custom callbacks to perform drawing and handle user interaction.

After you call the function `InitDataBrowserCustomCallbacks`, set the appropriate fields in the `DataBrowserCustomCallbacks` structure to your callbacks. The `DataBrowserCustomCallbacks` structure contains fields for the following:

- `DataBrowserDrawItemProcPtr` (page 139)
- `DataBrowserEditItemProcPtr` (page 141)
- `DataBrowserHitTestProcPtr` (page 144)
- `DataBrowserTrackingProcPtr` (page 163)
- `DataBrowserItemDragRgnProcPtr` (page 150)
- `DataBrowserItemAcceptDragProcPtr` (page 146)
- `DataBrowserItemReceiveDragProcPtr` (page 158)

After you assign your custom callbacks to the appropriate field, call the function `SetDataBrowserCustomCallbacks` (page 107).

Note that this is a different set of callbacks from those that are assigned to fields in the `DataBrowserCallbacks` data structure.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

InvokeDataBrowserAcceptDragUPP

Calls an accept-drag callback function.

```
Boolean InvokeDataBrowserAcceptDragUPP (
   ControlRef browser,
   DragReference theDrag,
   DataBrowserItemID item,
   DataBrowserAcceptDragUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserAcceptDragProcPtr` (page 137) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserAddDragItemUPP

Calls an add-drag-item callback function.

```
Boolean InvokeDataBrowserAddDragItemUPP (
   ControlRef browser,
   DragReference theDrag,
   DataBrowserItemID item,
   ItemReference *itemRef,
   DataBrowserAddDragItemUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserAddDragItemProcPtr` (page 138) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserDrawItemUPP

Calls a draw-item callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
void InvokeDataBrowserDrawItemUPP (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserPropertyID property,
   DataBrowserItemState itemState,
   const Rect *theRect,
   SInt16 gdDepth,
   Boolean colorDevice,
   DataBrowserDrawItemUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserDrawItemProcPtr` (page 139) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserEditItemUPP

Calls an edit-item callback function.

```
Boolean InvokeDataBrowserEditItemUPP (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserPropertyID property,
   CFStringRef theString,
   Rect *maxEditTextRect,
   Boolean *shrinkToFit,
   DataBrowserEditItemUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserEditItemProcPtr` (page 141) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserGetContextualMenuUPP

Calls a get-contextual-menu callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
void InvokeDataBrowserGetContextualMenuUPP (
   ControlRef browser,
   MenuRef *menu,
   UInt32 *helpType,
   CFStringRef *helpItemString,
   AEDesc *selection,
   DataBrowserGetContextualMenuUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserGetContextualMenuProcPtr` (page 142) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserHitTestUPP

Calls a hit-test callback function.

```
Boolean InvokeDataBrowserHitTestUPP (
   ControlRef browser,
   DataBrowserItemID itemID,
   DataBrowserPropertyID property,
   const Rect *theRect,
   const Rect *mouseRect,
   DataBrowserHitTestUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserHitTestProcPtr` (page 144) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemAcceptDragUPP

Calls an item-accept-drag callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserDragFlags InvokeDataBrowserItemAcceptDragUPP (
   ControlRef browser,
   DataBrowserItemID itemID,
   DataBrowserPropertyID property,
   const Rect *theRect,
   DragReference theDrag,
   DataBrowserItemAcceptDragUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemAcceptDragProcPtr` (page 146) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemCompareUPP

Calls an item-comparison callback function.

```
Boolean InvokeDataBrowserItemCompareUPP (
   ControlRef browser,
   DataBrowserItemID itemOne,
   DataBrowserItemID itemTwo,
   DataBrowserPropertyID sortProperty,
   DataBrowserItemCompareUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemCompareProcPtr` (page 147) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemDataUPP

Calls an item-data callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus InvokeDataBrowserItemDataUPP (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserPropertyID property,
   DataBrowserItemDataRef itemData,
   Boolean setValue,
   DataBrowserItemDataUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemDataProcPtr` (page 149) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemDragRgnUPP

Calls an item-drag-region callback function.

```
void InvokeDataBrowserItemDragRgnUPP (
   ControlRef browser,
   DataBrowserItemID itemID,
   DataBrowserPropertyID property,
   const Rect *theRect,
   RgnHandle dragRgn,
   DataBrowserItemDragRgnUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemDragRgnProcPtr` (page 150) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemHelpContentUPP

Calls an item-help-content callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
void InvokeDataBrowserItemHelpContentUPP (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserPropertyID property,
   HMContentRequest inRequest,
   HMContentProvidedType *outContentProvided,
   HMHelpContentRec *ioHelpContent,
   DataBrowserItemHelpContentUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemHelpContentProcPtr` (page 152) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemNotificationUPP

Calls an item-notification callback function.

```
void InvokeDataBrowserItemNotificationUPP (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserItemNotification message,
   DataBrowserItemNotificationUPP userUPP
);
```

Discussion
In most cases you don't need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemNotificationProcPtr` (page 154) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemNotificationWithItemUPP

Calls an item-notification-with-data callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
void InvokeDataBrowserItemNotificationWithItemUPP (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserItemNotification message,
   DataBrowserItemDataRef itemData,
   DataBrowserItemNotificationWithItemUPP userUPP
);
```

Discussion
In most cases you don't need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemNotificationWithItemProcPtr` (page 155) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.5 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemReceiveDragUPP

Calls an item-receive-drag callback function.

```
Boolean InvokeDataBrowserItemReceiveDragUPP (
   ControlRef browser,
   DataBrowserItemID itemID,
   DataBrowserPropertyID property,
   DataBrowserDragFlags dragFlags,
   DragReference theDrag,
   DataBrowserItemReceiveDragUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemReceiveDragProcPtr` (page 158) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserItemUPP

Calls an item-iterator callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
void InvokeDataBrowserItemUPP (
   DataBrowserItemID item,
   DataBrowserItemState state,
   void *clientData,
   DataBrowserItemUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserItemProcPtr` (page 157) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserPostProcessDragUPP

Calls a postprocess-drag callback function.

```
void InvokeDataBrowserPostProcessDragUPP (
   ControlRef browser,
   DragReference theDrag,
   OSStatus trackDragResult,
   DataBrowserPostProcessDragUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserPostProcessDragProcPtr` (page 159) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserReceiveDragUPP

Calls a receive-drag callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
Boolean InvokeDataBrowserReceiveDragUPP (
   ControlRef browser,
   DragReference theDrag,
   DataBrowserItemID item,
   DataBrowserReceiveDragUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserReceiveDragProcPtr` (page 160) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserSelectContextualMenuUPP

Calls a select-contextual-menu callback function.

```
void InvokeDataBrowserSelectContextualMenuUPP (
   ControlRef browser,
   MenuRef menu,
   UInt32 selectionType,
   SInt16 menuID,
   MenuItemIndex menuItem,
   DataBrowserSelectContextualMenuUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserSelectContextualMenuProcPtr` (page 161) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

InvokeDataBrowserTrackingUPP

Calls a tracking callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserTrackingResult InvokeDataBrowserTrackingUPP (
   ControlRef browser,
   DataBrowserItemID itemID,
   DataBrowserPropertyID property,
   const Rect *theRect,
   Point startPt,
   EventModifiers modifiers,
   DataBrowserTrackingUPP userUPP
);
```

Discussion
In most cases you do not need to use this function, because the system invokes your callback function for you. See the `DataBrowserTrackingProcPtr` (page 163) callback function for more information and for a description of the parameters.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.1 and later.

Declared In
`HIDataBrowser.h`

IsDataBrowserItemSelected

Checks to see if a data item is selected.

```
Boolean IsDataBrowserItemSelected (
   ControlRef browser,
   DataBrowserItemID item
);
```

Parameters

browser
A data browser.

item
The item ID of the item to check.

Return Value
A value of `true` if the item is a member of the current selection.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

MoveDataBrowserSelectionAnchor

Moves or extends the current selection.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus MoveDataBrowserSelectionAnchor (
   ControlRef browser,
   DataBrowserSelectionAnchorDirection direction,
   Boolean extendSelection
);
```

Parameters

browser

A data browser.

direction

The direction to move or extend the current selection. You can pass any one of the constants described in "Selection Anchor Directions" (page 197).

extendSelection

On input, a value that specifies whether to extend the current selection (`true`) or move the selection (`false`).

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

NewDataBrowserAcceptDragUPP

Creates a universal procedure pointer to an accept-drag callback function.

```
DataBrowserAcceptDragUPP NewDataBrowserAcceptDragUPP (
   DataBrowserAcceptDragProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your accept-drag callback function.

Return Value
The universal procedure pointer.

Discussion
See the `DataBrowserAcceptDragProcPtr` (page 137) callback function.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

NewDataBrowserAddDragItemUPP

Creates a universal procedure pointer to an add-drag-item callback function.

```
DataBrowserAddDragItemUPP NewDataBrowserAddDragItemUPP (
   DataBrowserAddDragItemProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your add-drag-item callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserAddDragItemProcPtr` (page 138) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserDrawItemUPP

Creates a universal procedure pointer to a draw-item callback function.

```
DataBrowserDrawItemUPP NewDataBrowserDrawItemUPP (
   DataBrowserDrawItemProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your draw-item callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserDrawItemProcPtr` (page 139) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserEditItemUPP

Creates a universal procedure pointer to an edit-item callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserEditItemUPP NewDataBrowserEditItemUPP (
   DataBrowserEditItemProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your edit-item callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserEditItemProcPtr` (page 141) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserGetContextualMenuUPP

Creates a universal procedure pointer to a get-contextual-menu callback function.

```
DataBrowserGetContextualMenuUPP NewDataBrowserGetContextualMenuUPP (
   DataBrowserGetContextualMenuProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your get-contextual-menu callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserGetContextualMenuProcPtr` (page 142) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserHitTestUPP

Creates a universal procedure pointer to a hit-test callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserHitTestUPP NewDataBrowserHitTestUPP (
   DataBrowserHitTestProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your hit-test callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserHitTestProcPtr` (page 144) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemAcceptDragUPP

Creates a universal procedure pointer to an item-accept-drag callback function.

```
DataBrowserItemAcceptDragUPP NewDataBrowserItemAcceptDragUPP (
   DataBrowserItemAcceptDragProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-accept-drag callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemAcceptDragProcPtr` (page 146) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemCompareUPP

Creates a universal procedure pointer to an item-comparison callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserItemCompareUPP NewDataBrowserItemCompareUPP (
   DataBrowserItemCompareProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-comparison callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemCompareProcPtr` (page 147) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemDataUPP

Creates a universal procedure pointer to an item-data callback function.

```
DataBrowserItemDataUPP NewDataBrowserItemDataUPP (
   DataBrowserItemDataProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-data callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemDataProcPtr` (page 149) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemDragRgnUPP

Creates a universal procedure pointer to an item-drag-region callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserItemDragRgnUPP NewDataBrowserItemDragRgnUPP (
   DataBrowserItemDragRgnProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-drag-region callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemDragRgnProcPtr` (page 150) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemHelpContentUPP

Creates a universal procedure pointer to an item-help-content callback function.

```
DataBrowserItemHelpContentUPP NewDataBrowserItemHelpContentUPP (
   DataBrowserItemHelpContentProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-help-content callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemHelpContentProcPtr` (page 152) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemNotificationUPP

Creates a universal procedure pointer to an item-notification callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserItemNotificationUPP NewDataBrowserItemNotificationUPP (
   DataBrowserItemNotificationProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-notification callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemNotificationProcPtr` (page 154) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemNotificationWithItemUPP

Creates a universal procedure pointer to an item-notification-with-data callback function.

```
DataBrowserItemNotificationWithItemUPP NewDataBrowserItemNotificationWithItemUPP
(
   DataBrowserItemNotificationWithItemProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-notification-with-data callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemNotificationWithItemProcPtr` (page 155) callback function.

Availability

Available in CarbonLib 1.5 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemReceiveDragUPP

Creates a universal procedure pointer to an item-receive-drag callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserItemReceiveDragUPP NewDataBrowserItemReceiveDragUPP (
   DataBrowserItemReceiveDragProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-receive-drag callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemReceiveDragProcPtr` (page 158) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserItemUPP

Creates a universal procedure pointer to an item-iterator callback function.

```
DataBrowserItemUPP NewDataBrowserItemUPP (
   DataBrowserItemProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your item-iterator callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserItemProcPtr` (page 157) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserPostProcessDragUPP

Creates a universal procedure pointer to a postprocess-drag callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserPostProcessDragUPP NewDataBrowserPostProcessDragUPP (
   DataBrowserPostProcessDragProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your postprocess-drag callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserPostProcessDragProcPtr` (page 159) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserReceiveDragUPP

Creates a universal procedure pointer to a receive-drag callback function.

```
DataBrowserReceiveDragUPP NewDataBrowserReceiveDragUPP (
   DataBrowserReceiveDragProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your receive-drag callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserReceiveDragProcPtr` (page 160) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserSelectContextualMenuUPP

Creates a universal procedure pointer to a select-contextual-menu callback function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
DataBrowserSelectContextualMenuUPP NewDataBrowserSelectContextualMenuUPP (
   DataBrowserSelectContextualMenuProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your select-contextual-menu callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserSelectContextualMenuProcPtr` (page 161) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

NewDataBrowserTrackingUPP

Creates a universal procedure pointer to a tracking callback function.

```
DataBrowserTrackingUPP NewDataBrowserTrackingUPP (
   DataBrowserTrackingProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your tracking callback function.

Return Value

The universal procedure pointer.

Discussion

See the `DataBrowserTrackingProcPtr` (page 163) callback function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.1 and later.

Declared In

`HIDataBrowser.h`

OpenDataBrowserContainer

Opens a data browser container.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus OpenDataBrowserContainer (
   ControlRef browser,
   DataBrowserItemID container
);
```

Parameters

browser

A data browser.

container

The item ID of the container to open.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Normally the user navigates through a data hierarchy by clicking the disclosure triangle next to a container item in list view, or the container item (such as a folder icon) in column view. In either of these cases, the system automatically opens or closes the container. Under some circumstances your application may need to open or close a container programmatically, such as when you are restoring a display to its last known state. In such cases, you can call the function `OpenDataBrowserContainer` to disclose items in a container or the function `CloseDataBrowserContainer` (page 28) to hide items in a container.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

RemoveDataBrowserItems

Removes one or more items from a data browser.

```
OSStatus RemoveDataBrowserItems (
   ControlRef browser,
   DataBrowserItemID container,
   ItemCount numItems,
   const DataBrowserItemID *items,
   DataBrowserPropertyID preSortProperty
);
```

Parameters

browser

A data browser.

container

An item ID or the constant `kDataBrowserNoItem`. Pass the item ID that uniquely identifies the container from which you want to remove items. Pass `kDataBrowserNoItem` to remove items from the root container.

numItems

The number of items in the array pointed to by the `items` parameter. To remove all items pass `0` and also pass `NULL` in the items parameter.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

items

A pointer to an array of item ID values for the items you want to remove from the data browser. You can delete an arbitrary list of items from a container. To remove all items, pass `NULL`, and also pass `0` in the `numItems` parameter.

preSortProperty

The property ID of the column whose sorting order is the same as the sorting order of the `items` array. A property ID is a value that identifies a column independent of its position in a data browser. Pass `kDataBrowserItemNoProperty` if the `items` array is not sorted or if you don't know the sorting order. You'll get the best performance from this function if you provide a sorting order.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

RemoveDataBrowserTableViewColumn

Removes a column from a list view data browser.

```
OSStatus RemoveDataBrowserTableViewColumn (
   ControlRef browser,
   DataBrowserTableViewColumnID column
);
```

Parameters

browser

A data browser.

column

The property ID for the list view column you want to remove. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function works only for list view.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

RevealDataBrowserItem

Scrolls an item into view, optionally bringing a particular part of that item into view.

```
OSStatus RevealDataBrowserItem (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserPropertyID propertyID,
   DataBrowserRevealOptions options
);
```

Parameters

browser

A data browser.

item

The item ID of the item to scroll into view.

propertyID

The property ID of the column to scroll into view. A property ID is a four-character sequence that you assign to represent a column in list view. For column view, pass `kDataBrowserNoItem`.

options

A value that specifies how to position the item in the data browser. See "Reveal Options" (page 196) for a list of the constants you can supply.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

In most cases the system takes care of scrolling for you. However, this function is useful if your application supports type-select and you want to scroll a matching item into view.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserActiveItems

Sets what determines the active state of the items in a data browser.

```
OSStatus SetDataBrowserActiveItems (
   ControlRef browser,
   Boolean active
);
```

Parameters

browser

A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

active

A value that specifies the new active state for the items displayed in the list. Pass `true` to make the active state of each item determined by what your callback reports for each item's `kDataBrowserItemIsActiveProperty` property, or `false` to make all items inactive. Inactive items appear dimmed.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Passing `true` for the `active` parameter does not make all the items active. Instead it sets the active state of each individual item according to the value associated with the `kDataBrowserItemIsActiveProperty` property for that item. This means if the active property for an item is set to `false`, and you pass `true` for the `active` parameter, then the item is inactive.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserCallbacks

Sets the callback routines to use with a data browser, replacing any previously installed callbacks.

```
OSStatus SetDataBrowserCallbacks (
   ControlRef browser,
   const DataBrowserCallbacks *callbacks
);
```

Parameters

browser

A data browser.

callbacks

A pointer to a `DataBrowserCallbacks` structure that is filled out with universal procedure pointers (UPPs) to the callback routines your application provides. At a minimum, you need to provide a UPP to an item-data callback (`DataBrowserItemDataProcPtr` (page 149)).

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Before calling the function `SetDataBrowserCallbacks` you must first call `InitDataBrowserCallbacks` (page 79) to initialize the data browser callback structure. Calling `SetDataBrowserCallbacks` replaces any callback routines you installed previously by calling this function.

You can supply the following callbacks. If you don't supply callbacks in cases for which it's optional, you get the default behavior provided by the data browser API.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

- DataBrowserItemDataProcPtr (page 149). You must provide this callback because communicates the text, icons, or other data to display in list view. It also communicates the metadata that defines how data is displayed, such as whether or not an item is a container or has a parent. If you set up your data browser to allow the user to edit, this callback informs your application when the user makes a change.
- DataBrowserItemCompareProcPtr (page 147). You must provide a sorting callback if you want users to be able to sort the items in a column. If you want containers in a hierarchical list to be sorted independently, then you must provide a sorting callback that handles the hierarchical lists appropriately.
- DataBrowserItemNotificationProcPtr (page 154). You must provide this (or the next) callback if you have hierarchical data in a list, or if you use column view.
- DataBrowserItemNotificationWithItemProcPtr (page 155) (Mac OS X only)
- DataBrowserAddDragItemProcPtr (page 138). You can provide this callback to allow dragging out of your data browser.
- DataBrowserAcceptDragProcPtr (page 137). You can provide this callback to allow dragging into your data browser; use this to accept a drag item.
- DataBrowserReceiveDragProcPtr (page 160). You can provide this callback to allow dragging into your data browser; use this to receive a drag item.
- DataBrowserPostProcessDragProcPtr (page 159). If you provide callbacks to allow dragging into your data browser, you can optionally provide a postprocess-drag callback to perform cleanup tasks.
- DataBrowserGetContextualMenuProcPtr (page 142). You can optionally support a contextual menu. If so, you'll need to provide the next callback too.
- DataBrowserSelectContextualMenuProcPtr (page 161)
- DataBrowserItemHelpContentProcPtr (page 152). You can optionally provide help tags.

Note that this function sets a different set of callbacks from those that are set by calling the function SetDataBrowserCustomCallbacks (page 107).

To replace a callback, you first need to get the current set of callbacks by calling the function GetDataBrowserCallbacks (page 44). Set the appropriate fields in the DataBrowserCallbacks structure to your callback. Then you call the function SetDataBrowserCallbacks. Your application can set as many callbacks as appropriate.

The following code shows how to assign UPPs to the callbacks structure and then call the function SetDataBrowserCallbacks. The code assumes you have already called the function InitDataBrowserCallbacks (page 79) to initialize the data browser callback structure.

```
myCallbacks.u.v1.itemNotificationCallback =
        NewDataBrowserItemNotificationUPP (MyItemNotificationCallback);

myCallbacks.u.v1.acceptDragCallback =
        NewDataBrowserAcceptDragUPP (MyAcceptDragCallback);
myCallbacks.u.v1.receiveDragCallback =
        NewDataBrowserReceiveDragUPP (MyReceiveDragCallback);
myCallbacks.u.v1.addDragItemCallback =
        NewDataBrowserAddDragItemUPP (MyAddDragItemCallback);
myCallbacks.u.v1.itemHelpContentCallback =
        NewDataBrowserItemHelpContentUPP (MyItemHelpContentCallback);
myCallbacks.u.v1.getContextualMenuCallback =
        NewDataBrowserGetContextualMenuUPP (MyGetContextualMenuCallback);
myCallbacks.u.v1.selectContextualMenuCallback =
```


2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
        NewDataBrowserSelectContextualMenuUPP (
                    MySelectContextualMenuCallback);
SetDataBrowserCallbacks (browser, &myCallbacks);
```

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserColumnViewDisplayType

Sets the display type for a data browser in column view.

```
OSStatus SetDataBrowserColumnViewDisplayType (
   ControlRef browser,
   DataBrowserPropertyType propertyType
);
```

Parameters

`browser`
A data browser.

`propertyType`
The data type to be displayed in the data browser. The default is `kDataBrowserIconAndTextType`. Currently this is the only value you can supply for column view.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function is effectively not functional because no display types other than `kDataBrowserIconAndTextType` are currently supported. Note that the rightmost column can have the attribute `kDataBrowserColumnViewPreviewProperty` as long as you provide a callback to display the appropriate icon and text information in the preview column.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserColumnViewPath

Sets a path for a column view.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserColumnViewPath (
   ControlRef browser,
   UInt32 length,
   const DataBrowserItemID *path
);
```

Parameters

browser

A data browser.

length

The number of items in the array passed in the `path` parameter.

path

The address to the first item in the array of item ID values that specifies the path. Array element `0` is the root; array element `N - 1` is the target.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserCustomCallbacks

Sets the custom callback routines to use with a data browser, replacing any previously installed custom callbacks.

```
OSStatus SetDataBrowserCustomCallbacks (
   ControlRef browser,
   const DataBrowserCustomCallbacks *callbacks
);
```

Parameters

browser

A data browser.

callbacks

A pointer to a `DataBrowserCustomCallbacks` structure that is filled out with universal procedure pointers (UPPs) to the custom callback routines your application provides.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Before calling the function `SetDataBrowserCustomCallbacks` you must first call `InitDataBrowserCustomCallbacks` (page 80) to initialize the data browser custom callback structure. Calling `SetDataBrowserCustomCallbacks` replaces any callback routines you installed previously by calling this function.

You can supply the following custom callback routines.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

- DataBrowserDrawItemProcPtr (page 139). This callback is invoked by the data browser whenever your content needs to be drawn. You must supply this callback for data whose display type is kDataBrowserCustomType.
- DataBrowserEditItemProcPtr (page 141). Supply this callback when you want to support editing of your content.
- DataBrowserHitTestProcPtr (page 144). You can provide this callback to determine if the pointer is over content that can be selected or dragged.
- DataBrowserTrackingProcPtr (page 163). This callback implements custom tracking behavior.
- DataBrowserItemDragRgnProcPtr (page 150). You can supply this callback when you need to determine which part of an item to use to create a transparent image for a dragged item.
- DataBrowserItemAcceptDragProcPtr (page 146). This callback determines if an item can accept a drag object.
- DataBrowserItemReceiveDragProcPtr (page 158). This callback receives a drop over an item.

Note that this is a different set of callbacks from those that are installed by calling the function SetDataBrowserCallbacks (page 104).

To replace a callback, you first need to set the appropriate fields in the DataBrowserCustomCallbacks structure to your callbacks. Then you call the function SetDataBrowserCustomCallbacks. The following code shows how to set custom callbacks. It assumes you have already called the function InitDataBrowserCustomCallbacks (page 80) to initialize the data browser custom callback structure. Your application can set as many callbacks as appropriate.

```
myCustomCallbacks.u.v1.drawItemCallback =
        NewDataBrowserDrawItemUPP (MyDataBrowserDrawItemCallback);
myCustomCallbacks.u.v1.editItemCallback =
        NewDataBrowserEditItemUPP (MyDataBrowserEditItemCallback);
SetDataBrowserCustomCallbacks (browser,&myCustomCallbacks);
```

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

SetDataBrowserEditItem

Programmatically starts or ends an editing session.

```
OSStatus SetDataBrowserEditItem (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserPropertyID property
);
```

Parameters

browser
A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

item
The item ID of the item to make editable.

property
The property ID of the item to make editable.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
You can call the function `SetDataBrowserEditItem` to begin or end an editing session programmatically for a text item. To begin an editing session for a text item, specify its item ID and property ID. To end an editing session, provide the constant `kDataBrowserNoItem` as the item ID number.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserEditText

Modifies the displayed contents of a text item while it is being edited.

```
OSStatus SetDataBrowserEditText (
   ControlRef browser,
   CFStringRef text
);
```

Parameters

browser
A data browser.

text
A CFString object that specifies the text to edit. The data browser makes its own copy of this object so it is safe to release your own reference after you call this function.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
You can use this function to programmatically change the text. For example, to paste data in response to a Paste command. This function is useful only if an edit session is in progress for an item. You can check whether can get session is open by calling the function `GetDataBrowserEditItem` (page 47).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

SetDataBrowserHasScrollBars

Sets the display state of horizontal and vertical scroll bars for a list view data browser.

```
OSStatus SetDataBrowserHasScrollBars (
   ControlRef browser,
   Boolean horiz,
   Boolean vert
);
```

Parameters

browser

A list view data browser.

horiz

A value that specifies whether to display the browser control with (`true`) or without (`false`) a horizontal scroll bar.

vert

A value that specifies whether to display the browser control with (`true`) or without (`false`) a vertical scroll bar.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

If the list your application displays is small and its coordinates do not extend beyond the bounds of the area used to display the list, then you can call `SetDataBrowserHasScrollBars` to turn off the display of scroll bars.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserItemDataBooleanValue

Specifies a Boolean value for an item.

```
OSStatus SetDataBrowserItemDataBooleanValue (
   DataBrowserItemDataRef itemData,
   Boolean theData
);
```

Parameters

itemData

The item data reference for the item whose Boolean value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataBooleanValue`.

theData

The value to display.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your item-data callback calls this function in response to an inquiry for the following properties:

- `kDataBrowserItemIsActiveProperty`
- `kDataBrowserItemIsSelectableProperty`
- `kDataBrowserItemIsEditableProperty`
- `kDataBrowserItemIsContainerProperty`
- `kDataBrowserItemIsOpenableProperty`
- `kDataBrowserItemIsClosableProperty`
- `kDataBrowserItemIsSortableProperty`

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataButtonValue

Specifies a checkbox value.

```
OSStatus SetDataBrowserItemDataButtonValue (
   DataBrowserItemDataRef itemData,
   ThemeButtonValue theData
);
```

Parameters

itemData

The item data reference for the item whose checkbox value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataButtonValue`.

theData

The checkbox setting. You can supply any of the following theme button value constants defined by the Appearance Manager:

- `kThemeButtonOff` draws a checkbox that is not selected.
- `kThemeButtonOn` draws a checkbox that is selected.
- `kThemeButtonMixed` draws a checkbox that in a mixed state, indicating that a setting is on for some items in a selection and off for others.

See *Appearance Manager Reference* for more information.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
Your item-data callback calls this function in response to a set-data request for items that have the display type `kDataBrowserCheckboxType`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataDateTime

Specifies, as a 32-bit value, a date and time value to display.

```
OSStatus SetDataBrowserItemDataDateTime (
   DataBrowserItemDataRef itemData,
   SInt32 theData
);
```

Parameters

itemData
The item data reference for the item whose date and time value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataDateTime`.

theData
A 32-bit value that represents the number of elapsed seconds since midnight, January 1, 1904. For more information about date and time encodings used in the Mac OS, see *Date, Time, and Measurement Utilities Reference*.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function works only with items that have the property `kDataBrowserDateTimeType`. If the column has the property `kDataBrowserRelativeDateTime`, the date is displayed relative to the current time for the computer. For example, a time 24 hours before the current time is displayed as "Yesterday." Other examples of relative date and time values are "Today, 1:45 PM" and "Yesterday, 7:30 AM."

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataDrawState

Specifies whether to draw a checkbox in the active or inactive state.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserItemDataDrawState (
   DataBrowserItemDataRef itemData,
   ThemeDrawState theData
);
```

Parameters

itemData

The item data reference for the item whose drawing state you want to set. This value is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataDrawState`.

theData

The drawing state to use for the checkbox item. You can supply the following theme drawing state constants:

- `kThemeStateInactive` draws the item in the inactive state.
- `kThemeStateActive` draws the item in the active state.

See *Appearance Manager Reference* for more information on these constants.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your item-data callback calls this function in response to a set-data request for items that have the display type `kDataBrowserCheckboxType`.

Availability
Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataIcon

Specifies the icon to draw.

```
OSStatus SetDataBrowserItemDataIcon (
   DataBrowserItemDataRef itemData,
   IconRef theData
);
```

Parameters

itemData

The item data reference for the item whose icon you want to set. This value is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataIcon`.

theData

The icon to display. The data browser retains the icon, so you may release the `IconRef` after the function returns.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
You call the function `SetDataBrowserItemDataIcon` from within a `DataBrowserItemDataProcPtr` (page 149) callback routine to specify an icon to draw. You can specify an icon for any column that has the `kDataBrowserIconType` display type or the `kDataBrowserIconAndTextType` display type.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataIconTransform

Specifies a transformation to apply to an icon when it is drawn.

```
OSStatus SetDataBrowserItemDataIconTransform (
   DataBrowserItemDataRef itemData,
   IconTransformType theData
);
```

Parameters

itemData

The item data reference for the item whose icon transformation you want to set. This value is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataIconTransform`.

theData

An icon transformation type that specifies how to modify the appearance of the icon. You can pass any of the icon transformation constants defined by Icon Services and Utilities. See *Icon Services and Utilities Reference* for more information.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function works only with items that either have the property `kDataBrowserIconAndTextType` or `kDataBrowserIconType`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataItemID

Communicates a property of an item when that property is another item's ID.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserItemDataItemID (
   DataBrowserItemDataRef itemData,
   DataBrowserItemID theData
);
```

Parameters

itemData

The item data reference passed to your item-data callback.

theData

The item ID to set.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

To display hierarchical data correctly the data browser needs to know whether an item is a container and whether the item is in a container (has a parent). So it sends a get-data request for the properties `kDataBrowserParentContainerProperty` and `kDataBrowserContainerAliasIDProperty` to your item-data callback.

The property `kDataBrowserContainerAliasIDProperty` is sent to your item-data callback to provide your application with a chance to follow an alias that the item might represent. If the incoming item is an alias to another item, you can call `SetDataBrowserItemDataItemID` to inform the data browser which other item the incoming item points to.

The property `kDataBrowserParentContainerProperty` is sent to your item-data callback to check whether an item has a parent. If it does, you call `SetDataBrowserItemDataItemID`, supplying the item ID of the parent in the parameter `theData`. If the item has no parent, set `theData` to `0`.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserItemDataLongDateTime

Specifies, as a 64-bit value, a date and time value to display.

```
OSStatus SetDataBrowserItemDataLongDateTime (
   DataBrowserItemDataRef itemData,
   const LongDateTime *theData
);
```

Parameters

itemData

The item data reference for the item whose long date and time value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataLongDateTime`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

theData

A pointer to a 64-bit value that represents the time as the number of elapsed seconds since midnight, January 1, 1904. For more information about date and time encodings used in the Mac OS, see *Date, Time, and Measurement Utilities Reference* in Carbon Text & International Documentation.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function works only with items that have the property `kDataBrowserDateTimeType`. If the column has the property `kDataBrowserRelativeDateTime`, the date is displayed relative to the current time for the computer. For example, a time 24 hours before the current time is displayed as "Yesterday." Other examples of relative date and time values are "Today, 1:45 PM" and "Yesterday, 7:30 AM."

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataMaximum

Specifies the maximum integer value that can be displayed for an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

```
OSStatus SetDataBrowserItemDataMaximum (
   DataBrowserItemDataRef itemData,
   SInt32 theData
);
```

Parameters

itemData

The item data reference for the item whose maximum value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataMaximum`.

theData

The maximum setting for content displayed for the item.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

SetDataBrowserItemDataMenuRef

Sets the pop-up menu to display.

```
OSStatus SetDataBrowserItemDataMenuRef (
   DataBrowserItemDataRef itemData,
   MenuRef theData
);
```

Parameters

itemData

The item data reference for the item whose pop-up menu value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataMenuRef`.

theData

The pop-up menu set to the value you want to display. The system retains the menu reference that you pass; you must release it when you no longer need it. Pass `NULL` if you no longer want a menu.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Your item-data callback calls this function in response to a set-data request for an item whose display type is `kDataBrowserPopupMenuType`.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserItemDataMinimum

Specifies the minimum integer value that can be displayed for an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

```
OSStatus SetDataBrowserItemDataMinimum (
   DataBrowserItemDataRef itemData,
   SInt32 theData
);
```

Parameters

itemData

The item data reference for the item whose minimum value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataMinimum`.

theData

The minimum setting for the displayed content.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataRGBColor

Specifies a color to use when drawing an item.

```
OSStatus SetDataBrowserItemDataRGBColor (
   DataBrowserItemDataRef itemData,
   const RGBColor *theData
);
```

Parameters

itemData

The item data reference for the item whose color you want to set. This value is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataRGBColor`.

theData

A pointer to the RGB values that specify the color to use.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Typically this function is used to set the color for an item that is an icon type. Your item-data callback calls this function in response to a set-data request for items that have display type `kDataBrowserIconType` or `kDataBrowserIconAndTextType`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserItemDataText

Specifies the text to draw.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserItemDataText (
   DataBrowserItemDataRef itemData,
   CFStringRef theData
);
```

Parameters

itemData

The item data reference for the item whose text you want to set. This value is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataText`.

theData

The CFString object that contains the text you want to draw. You are responsible for releasing the CFString object.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

You call the function `SetDataBrowserItemDataText` from inside a data callback routine when the item being drawn is inside a column that has the `kDataBrowserTextType` display type or the `kDataBrowserIconAndTextType` display type.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserItemDataValue

Sets the value of an item; useful for such display types as sliders, progress bars, relevance indicators, and pop-up menus.

```
OSStatus SetDataBrowserItemDataValue (
   DataBrowserItemDataRef itemData,
   SInt32 theData
);
```

Parameters

itemData

The item data reference for the item whose integer value you want to set. The item data reference is passed to the callback routine from which you are calling the function `SetDataBrowserItemDataValue`.

theData

The value to display. The value must be between the minimum and maximum values specified by calling the functions `SetDataBrowserItemDataMinimum` (page 117) and `SetDataBrowserItemDataMaximum` (page 116). Values displayed by a progress bar can vary between the minimum and maximum values, inclusive.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
Your application calls the function `SetDataBrowserItemDataValue` to set a new value for a display type when your item-data callback routine is called with the `setValue` parameter set to `false`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserListViewDisclosureColumn

Specifies whether there is a column that has disclosure triangles and, if so, which column.

```
OSStatus SetDataBrowserListViewDisclosureColumn (
   ControlRef browser,
   DataBrowserTableViewColumnID column,
   Boolean expandableRows
);
```

Parameters

browser
A data browser.

column
The property ID for the column for which you want to set as disclosure column. Only one column in list view can be designated as a disclosure column. Pass `kDataBrowserNoItemProperty` if you do not want a disclosure column. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

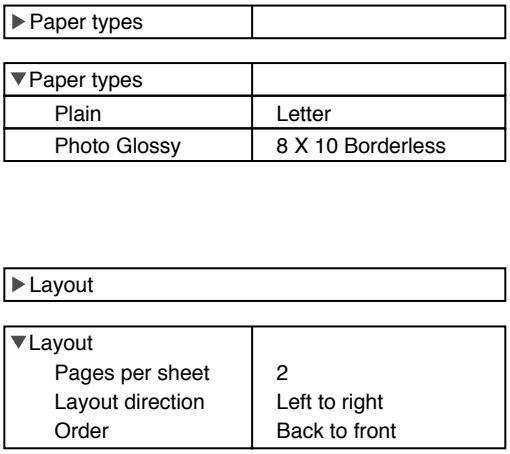
expandableRows
A value that specifies how a disclosed row behaves. Pass `true` to have a container open as a single row with an expanded height. Pass `false` to have a container opens to expose other rows. See the Discussion for more details on expandable rows.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
A disclosure triangle next to an item denotes the item is a container. You can use the `expandableRows` parameter to specify whether an opened container displays its items in individual rows, as shown in the top of Figure 1 or increases its row height to accommodate the contained information, as shown in the bottom of the figure.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Figure 1 A container can open to more rows or expand to show more information



When the `expandableRows` parameter is set to `true`:

- Disclosure triangles are drawn top-justified in the row.
- Custom row height, if any, for that row is respected only while the row is disclosed. At other times, the default row height is used.

When the `expandableRows` parameter is set to `false`:

- Disclosure triangles are centered vertically in the row.
- Custom row height, if any, for that row is always respected.

When a disclosure triangle is clicked by the user, your application receives the same notifications regardless of whether `expandableRows` is set to `true` or `false`. When your application receives a notification that an expandable row is toggled to open, call the function `SetDataBrowserTableViewItemRowHeight` (page 131) to set the row to the appropriate height.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserListViewHeaderBtnHeight

Sets the height of the rectangular area where the column title appears.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserListViewHeaderBtnHeight (
   ControlRef browser,
   UInt16 height
);
```

Parameters

browser

A data browser.

height

The height, in pixels, to use for the rectangular area where the column title appears. Pass `0` to turn off header button display. To turn on header button display, pass the value previously obtained from the function `GetDataBrowserListViewHeaderBtnHeight` (page 63). The default height is currently 17 pixels.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserListViewHeaderDesc

Provides a description for a column title in list view.

```
OSStatus SetDataBrowserListViewHeaderDesc (
   ControlRef browser,
   DataBrowserTableViewColumnID column,
   DataBrowserListViewHeaderDesc *desc
);
```

Parameters

browser

A data browser.

column

The property ID for the column in list view whose title description you want to set. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

desc

The list view header description structure that you have filled out with data that describes the appearance of a column title in list view.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

This functions allows you to change the behavior or appearance of a column title. Typically you call this function if your application:

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

- Supports switching between list and column views, and you need to restore previously saved list view title information.
- Creates a list view data browser programmatically and the columns have titles.

Availability
Available in CarbonLib 1.5 and later.
Available in Mac OS X version 10.2 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserListViewUsePlainBackground

Specifies whether list view uses a plain white background.

```
OSStatus SetDataBrowserListViewUsePlainBackground (
   ControlRef browser,
   Boolean usePlainBackground
);
```

Parameters

`browser`
A data browser.

`usePlainBackground`
A value that specifies whether to use a plain background (`true`) or not to use a plain background (`false`). A plain background is an all-white background. In Mac OS X, passing `false` currently does nothing, as Mac OS X supports only a plain white background. However, pass `true` if you want a plain white background just in case the API changes in the future. In Mac OS 9, passing `false` causes the data browser to use a shaded background.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
A list view that does not use a plain background can use colors or patterns to distinguish one column from another. For example, you could specify a color to designate a column as the sorted column.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserPropertyFlags

Sets the appearance and behavior attributes for a column in list view.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserPropertyFlags (
   ControlRef browser,
   DataBrowserPropertyID property,
   DataBrowserPropertyFlags flags
);
```

Parameters

browser

A data browser.

property

The property ID of the column whose appearance and behavior you want to set.

flags

The property flags to apply. A `DataBrowserPropertyFlags` value is a 32-bit value that is divided into four parts as follows:

- Bits 0–7 specify properties applied to the data browser as a whole—see "Property Flags: Universal" (page 188)
- Bits 8–15 modify display behavior—see "Property Flags: Modifiers" (page 189)
- Bits 16–23 are properties specific to list view—see "Property Flags: Offset and Mask for List View Properties" (page 192) and "Property Flags: List View Column Behavior" (page 193)
- Bits 24–31 can be defined by your application—see "Property Flags: Offset and Mask for Client-Defined Properties" (page 194)

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserScrollBarInset

Sets the inset values to use for the scroll bars of a data browser.

```
OSStatus SetDataBrowserScrollBarInset (
   ControlRef browser,
   Rect *insetRect
);
```

Parameters

browser

A data browser.

insetRect

A pointer to a rectangle that specifies the inset values you want the data browser to use. The left and right fields contain the horizontal inset values for the horizontal scroll bar, and the top and bottom fields contain the vertical inset values for the vertical scroll bar.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your application can call the functions `GetDataBrowserScrollBarInset` (page 66) and `SetDataBrowserScrollBarInset` if you want to place placards or controls beside the horizontal scroll bars or above the vertical ones. To do so, first call `GetDataBrowserScrollBarInset` to obtain the current settings. After modifying the current inset settings to provide space for the placard or control, call `SetDataBrowserScrollBarInset` with the new values.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserScrollPosition

Scrolls a list to the specified position.

```
OSStatus SetDataBrowserScrollPosition (
   ControlRef browser,
   UInt32 top,
   UInt32 left
);
```

Parameters

browser
A data browser.

top
The vertical scrolling position to use.

left
The horizontal scrolling position to use.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
The scrolling position `(0,0)` represents the home position, and is located at the top left of the data browser. Horizontal and vertical units are relative to the home position.

You can call this function to scroll a list to any arbitrary scrolling position. Normally, you use the function `GetDataBrowserScrollPosition` (page 66) in conjunction with `SetDataBrowserScrollPosition` to save and restore the scrolling position of a list to the user's last scrolling position. These functions should not be used to scroll particular cells into the view. For that, call the function `RevealDataBrowserItem`.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

SetDataBrowserSelectedItems

Modifies the current selection by adding items, removing items, or toggling the selection state of items.

```
OSStatus SetDataBrowserSelectedItems (
   ControlRef browser,
   ItemCount numItems,
   const DataBrowserItemID *items,
   DataBrowserSetOption operation
);
```

Parameters

browser
A data browser.

numItems
The number of item ID values stored in the array pointed to by the `items` parameter.

items
A pointer to an array of the item IDs to modify the selection with.

operation
The operation you want to perform on the current selection. You can add, assign, toggle, or remove the items specified by the `items` parameter. See "Selection State Options" (page 197) for a list of the constants you can supply and a complete description of what each constant does.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

SetDataBrowserSelectionFlags

Sets allowable selection behavior for a data browser.

```
OSStatus SetDataBrowserSelectionFlags (
   ControlRef browser,
   DataBrowserSelectionFlags selectionFlags
);
```

Parameters

browser
A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

selectionFlags

Flags that specify the selection behavior you want to allow in the data browser. The flags control such things as whether discontinuous selections are allowed by the user. See "User Selection Flags" (page 200) for detailed descriptions of these flags.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserSortOrder

Sets the sorting order for a list in list view.

```
OSStatus SetDataBrowserSortOrder (
   ControlRef browser,
   DataBrowserSortOrder order
);
```

Parameters

browser

A data browser.

order

The sorting order. See "Sorting Orders" (page 198) for a list of the constants you can supply.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
List view tracks the sorting order by column. In Mac OS X, setting the sorting order only affects the sorting order of the column currently set for sorting.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserSortProperty

Designates the list view column to use for sorting.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserSortProperty (
   ControlRef browser,
   DataBrowserPropertyID property
);
```

Parameters

browser

A data browser.

property

The property ID of the column to use for sorting.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

If the list is not currently sorted, or if the list is currently sorted with a different column, then the list is sorted and redrawn. You can all the function `GetDataBrowserSortProperty` to obtain the property ID of the column currently used for sorting.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserTableViewColumnPosition

Changes the visual position of a column in list view.

```
OSStatus SetDataBrowserTableViewColumnPosition (
   ControlRef browser,
   DataBrowserTableViewColumnID column,
   DataBrowserTableViewColumnIndex position
);
```

Parameters

browser

A data browser.

column

The property ID for the list view column you want to move. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

position

The position you want to move the column to.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

If you set your list to have the property `kDataBrowserListViewMovableColumn`, the user can rearrange columns by dragging. The function `SetDataBrowserTableViewColumnPosition` provides a way for your application to rearrange columns programmatically.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserTableViewColumnWidth

Sets the default column width for all columns in a data browser.

```
OSStatus SetDataBrowserTableViewColumnWidth (
   ControlRef browser,
   UInt16 width
);
```

Parameters

browser
A data browser.

width
The column width, in pixels.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
You can override the default width for an individual list view column by calling the function `SetDataBrowserTableViewNamedColumnWidth` (page 132).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserTableViewGeometry

Sets whether columns and rows can have variable widths in list view.

```
OSStatus SetDataBrowserTableViewGeometry (
   ControlRef browser,
   Boolean variableWidthColumns,
   Boolean variableHeightRows
);
```

Parameters

browser
A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

variableWidthColumns

A Boolean value that specifies whether column widths can be variable (`true`) or not (`false`).

variableHeightRows

A Boolean value that specifies whether row heights can be variable (`true`) or not (`false`).

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
After you call the function `SetDataBrowserTableViewGeometry` to set up variable row heights or columns widths in list view, you can modify individual row heights or columns widths in list view by calling the appropriate function—either `SetDataBrowserTableViewItemRowHeight` (page 131) or `SetDataBrowserTableViewNamedColumnWidth` (page 132).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserTableViewHiliteStyle

Sets the highlighting style to use for a list view data browser.

```
OSStatus SetDataBrowserTableViewHiliteStyle (
   ControlRef browser,
   DataBrowserTableViewHiliteStyle hiliteStyle
);
```

Parameters

browser

A list view data browser.

hiliteStyle

The highlighting style you want to use. See "Table View Highlighting Styles" (page 198) for a description of the constants you can supply.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserTableViewItemRow

Changes the visual position for an item in a list view data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserTableViewItemRow (
   ControlRef browser,
   DataBrowserItemID item,
   DataBrowserTableViewRowIndex row
);
```

Parameters

browser

A data browser.

item

The item ID for the item whose row you want to set.

row

The row index for the row you want to move the item to.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SetDataBrowserTableViewItemRowHeight

Sets the row height for a single row in a list view data browser.

```
OSStatus SetDataBrowserTableViewItemRowHeight (
   ControlRef browser,
   DataBrowserItemID item,
   UInt16 height
);
```

Parameters

browser

A data browser.

item

The item ID for the item whose row height you want to set.

height

The row height, in pixels.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Before calling this function, you must call the function `SetDataBrowserTableViewGeometry` (page 129) to set up variable row heights.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Not available to 64-bit applications.

Declared In
HIDataBrowser.h

SetDataBrowserTableViewNamedColumnWidth

Sets the column width for a single column in a list view data browser.

```
OSStatus SetDataBrowserTableViewNamedColumnWidth (
   ControlRef browser,
   DataBrowserTableViewColumnID column,
   UInt16 width
);
```

Parameters

browser
A data browser.

column
The property ID for the list view column whose width you want to set. The `DataBrowserTableViewColumnID` data type is the same as the `DataBrowserPropertyID` data type.

width
The width of the column, in pixels.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Before calling this function, you must call the function `SetDataBrowserTableViewGeometry` (page 129) to set up variable column widths.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
HIDataBrowser.h

SetDataBrowserTableViewRowHeight

Sets the default row height for all rows in a data browser.

```
OSStatus SetDataBrowserTableViewRowHeight (
   ControlRef browser,
   UInt16 height
);
```

Parameters

browser
A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

height
The row height, in pixels.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
This function sets the default row height for all rows. You override the default row height for an individual row by calling the function `SetDataBrowserTableViewItemRowHeight` (page 131).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserTarget

Sets the target for a data browser.

```
OSStatus SetDataBrowserTarget (
   ControlRef browser,
   DataBrowserItemID target
);
```

Parameters

browser
A data browser.

target
The item ID to assign as the target for the browser control.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Your application can set an item ID to use as a target if you do not want to use the default target set by the data browser. By default, the target is a container whose ID is `kDataBrowserNoItem`. For the list view, the target can be thought of as the root container. For the column view, the target is the rightmost column. When an item is dragged over a data browser but not dropped over any particular item, the target becomes the destination.

`SetDataBrowserTarget` changes the container that the data browser displays, thereby populating the data browser with items. If you use the function in column view, you must make sure your item-data callback responds to the property `kDataBrowserItemParentContainerProperty` by providing the item ID of the target's parent. This allows the function `SetDataBrowserColumnViewPath` (page 106) to process the data properly. The target is the leaf node item whose contents you want to display. However, unlike `GetDataBrowserColumnViewPathLength` (page 46), the function `SetDataBrowserTarget` doesn't offer a way for you to communicate the item IDs of the rest of the column containers, so `SetDataBrowserTarget` asks for them explicitly by requesting the item's parent, then the parent of the item's parent, and so on.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

You can pass a noncontainer item to this function in either list or column views. If you do, you must also respond to the property `kDataBrowserItemParentContainerProperty`. The data browser requests the parent of the target so it knows which container to display the contents of in the list view.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserUserState

Restores the view-style settings in list view to a previous state set by the user.

```
OSStatus SetDataBrowserUserState (
   ControlRef browser,
   CFDictionaryRef stateInfo
);
```

Parameters

browser
A data browser.

stateInfo
A CFDictionary object that specifies the view-style settings that you want to restore. Note that although this parameter is typed as a CFData object, you must supply a CFDictionary object because that is the form of the data the system expects.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
Typically you use this function to restore the user state you previously obtained by calling the function `GetDataBrowserUserState` (page 77).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

SetDataBrowserViewStyle

Sets the view style of the specified data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
OSStatus SetDataBrowserViewStyle (
   ControlRef browser,
   DataBrowserViewStyle style
);
```

Parameters

browser

A data browser.

style

The view style to use. Pass the constant `kDataBrowserListView` to draw the data browser using list view or `kDataBrowserColumnView` to use column view. See "View Styles" (page 202) for more information on these constants.

Return Value

A result code. See "Data Browser Result Codes" (page 202).

Discussion

Although you specify a view style when you call the function `CreateDataBrowserControl` (page 29), you can call `SetDataBrowserViewStyle` to change the style. Use `SetDataBrowserViewStyle` when you provide users the option of changing between list and column views.

After calling `SetDataBrowserViewStyle`, you need to perform the necessary tasks to configure the data browser for the view style you switched to. If you switch to list view, you need to set up list view header and column descriptions and call the function `AddDataBrowserListViewColumn` (page 25). You might also need to call other functions such as `SetDataBrowserListViewDisclosureColumn` (page 120) (for hierarchical lists).

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Not available to 64-bit applications.

Declared In

`HIDataBrowser.h`

SortDataBrowserContainer

Sorts a hierarchical list of items.

```
OSStatus SortDataBrowserContainer (
   ControlRef browser,
   DataBrowserItemID container,
   Boolean sortChildren
);
```

Parameters

browser

A data browser.

container

An item ID or the constant `kDataBrowserNoItem`. To sort all of the items that are organized as subitems of a container item, pass the item ID for the container item. To sort all of the items displayed at the top level of the data browser, pass the constant `kDataBrowserNoItem`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

sortChildren

A value that indicates whether to sort all items in the container hierarchy. Pass `true` to sort all items in the container hierarchy and `false` to sort just the immediate children of the container.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

UpdateDataBrowserItems

Requests a redraw of one or more items in a data browser.

```
OSStatus UpdateDataBrowserItems (
   ControlRef browser,
   DataBrowserItemID container,
   ItemCount numItems,
   const DataBrowserItemID *items,
   DataBrowserPropertyID preSortProperty,
   DataBrowserPropertyID propertyID
);
```

Parameters

browser

A data browser.

container

An item ID or the constant `kDataBrowserNoItem`. Pass the item ID that uniquely identifies the container. If you are updating one or more items that are in the root container, pass `kDataBrowserNoItem`.

numItems

The number of items in the array pointed to by the `items` parameter.

items

A pointer to an array of item ID values for the items you want to update. If you pass `NULL` or if the value `kDataBrowserNoItem` is an element in this array, then all rows are updated.

preSortProperty

The property ID of the column whose sorting order is the same as the sorting order of the `items` array. A property ID is a four-character sequence that you assign to represent a column in list view. Pass `kDataBrowserItemNoProperty` if the `items` array is not sorted of if you don't know the sorting order.

propertyID

The property ID of the column that must be updated. To update all columns associated with the items in the `items` array, pass `kDataBrowserNoItem`.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
After your application makes changes to any of the data items in a data browser you must update the display by calling the function `UpdateDataBrowserItems`. Calling this function also updates any internal caches allocated by the data browser.

Availability
Available in CarbonLib 1.1 and later.
Available in Mac OS X version 10.0 and later.
Not available to 64-bit applications.

Declared In
`HIDataBrowser.h`

Callbacks

DataBrowserAcceptDragProcPtr

Defines a pointer to an accept-drag callback function that determines whether your application can accept a drag object in the specified location.

```
typedef Boolean (*DataBrowserAcceptDragProcPtr) (
    ControlRef browser,
    DragRef theDrag,
    DataBrowserItemID item
);
```

You would declare an accept-drag callback function named `MyDataBrowserAcceptDragCallback` like this:

```
Boolean MyDataBrowserAcceptDragCallback (
    ControlRef browser,
    DragRef theDrag,
    DataBrowserItemID item
);
```

Parameters

browser
A data browser.

theDrag
The drag reference provided by the data browser to your callback.

item
The item ID of the item the drag object is held over. If the drag object is over the data browser but not over any specific item, the `item` parameter contains the item ID that represents one of the following:

- In list view, the target. (See `SetDataBrowserTarget` (page 133).)
- In column view, the item ID of the column the drag object is over

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Return Value
Your callback returns `true` if it is capable of accepting the drag object in the location designated by the item parameter. Otherwise, your callback returns `false`.

Discussion
The accept-drag callback is called by the data browser when your application needs to determine if it can accept a drag object in a particular location.

To provide a pointer to your accept-drag callback, you create a universal procedure pointer (UPP) of type `DataBrowserAcceptDragUPP`, using the function `NewDataBrowserAcceptDragUPP` (page 91). You can do so with code similar to the following:

```
DataBrowserAcceptDragUPP MyDataBrowserAcceptDragUPP;
MyDataBrowserAcceptDragUPP = NewDataBrowserAcceptDragUPP
                        (&MyDataBrowserAcceptDragCallback);
```

You can then assign `MyDataBrowserAcceptDragUPP` to the `acceptDragCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserAcceptDragUPP` (page 33) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserAddDragItemProcPtr

Defines a pointer to an add-drag-item callback function that adds an item to a drag reference.

```
typedef Boolean (*DataBrowserAddDragItemProcPtr) (
    ControlRef browser,
    DragRef theDrag,
    DataBrowserItemID item,
    ItemReference *itemRef
);
```

You would declare an add-drag-item callback function named `MyDataBrowserAddDragItemCallback` like this:

```
Boolean MyDataBrowserAddDragItemCallback (
    ControlRef browser,
    DragRef theDrag,
    DataBrowserItemID item,
    ItemRef *itemRef
);
```

Parameters
browser
A data browser.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

theDrag

The drag reference provided by the data browser to your callback.

item

The item ID of the item to add to the drag object.

itemRef

A pointer to a drag item reference variable. Your callback must set this to the `DragItemRef` value that it passes to the Drag Manager function `AddDragItemFlavor`.

Return Value

Your callback returns `true` to indicate the item should be or is part of the drag object. Your callback returns `false` if the item isn't part of the drag object.

Discussion

The add-drag-item callback is called by the data browser when a drag operation needs to be started. The data browser iterates through the selected items, invoking your callback for each item. Your callback is called after the drag reference is created by the data browser but before the function `TrackDrag` is called by the system.

Your callback adds an item to the drag reference calling the function `AddDragItemFlavor`. When you call `AddDragItemFlavor`, you must provide a unique drag item reference (`DragItemRef`) for each data browser item that you add to the drag. You must also provide the data type of the added item (drag flavor type) and set the appropriate drag flavor flags.

The data browser handles imaging and adds transparency for you. As a result, you do not need to create or add your own transparency information to the drag reference.

To provide a pointer to your add-drag-item callback, you create a universal procedure pointer (UPP) of type `DataBrowserAddDragItemUPP`, using the function `NewDataBrowserAddDragItemUPP` (page 92). You can do so with code similar to the following:

```
DataBrowserAddDragItemUPP MyDataBrowserAddDragItemUPP;
MyDataBrowserAddDragItemUPP = NewDataBrowserAddDragItemUPP
                        (&MyDataBrowserAddDragItemCallback);
```

You can then assign `MyDataBrowserAddDragItemUPP` to the `addDragItemCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserAddDragItemUPP` (page 34) function.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`HIDataBrowser.h`

DataBrowserDrawItemProcPtr

Defines a pointer to a draw-item callback function that draws a custom item.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
typedef void (*DataBrowserDrawItemProcPtr) (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    DataBrowserItemState itemState,
    const Rect *theRect,
    SInt16 gdDepth,
    Boolean colorDevice
);
```

You would declare a draw-item callback function named `MyDataBrowserDrawItemCallback` like this:

```
void MyDataBrowserDrawItemCallback (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    DataBrowserItemState itemState,
    const Rect *theRect,
    SInt16 gdDepth,
    Boolean colorDevice
);
```

Parameters

browser

A data browser.

item

The item ID for the item to draw.

property

The property ID for the item. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

itemState

The state to use when drawing the item. See "Item States" (page 184) for a description of the constants that can be provided to your callback.

theRect

A pointer to the bounding rectangle (in local coordinates, relative to the port) that specifies where to draw the item. This rectangle is the content rectangle, not the enclosing rectangle.

gdDepth

The bit depth of the current QuickDraw graphics port. The data browser sets the current QuickDraw port to the port that you draw into. This may not always be the port of the data browser's own window.

colorDevice

A value that specifies whether the current QuickDraw port is a color device (`true`) or is not (`false`).

Discussion

The draw-item callback is called by the data browser when an item whose display type is `kDataBrowserCustomType` needs to be drawn. Your application draws the item so it reflects the state specified by the `itemState` parameter.

To provide a pointer to your draw-item callback, you create a universal procedure pointer (UPP) of type `DataBrowserDrawItemUPP`, using the function `NewDataBrowserDrawItemUPP` (page 92). You can do so with code similar to the following:

```
DataBrowserDrawItemUPP MyDataBrowserDrawItemUPP;
```


2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
MyDataBrowserDrawItemUPP = NewDataBrowserDrawItemUPP
                        (&MyDataBrowserDrawItemCallback);
```

You can then assign `MyDataBrowserDrawItemUPP` to the `drawItemCallback` field of the structure `DataBrowserCustomCallbacks` (page 169). You install your data browser custom callbacks using the function `SetDataBrowserCustomCallbacks` (page 107).

When you no longer need the UPP, remove it using the `DisposeDataBrowserDrawItemUPP` (page 34) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserEditItemProcPtr

Defines a pointer to an edit-item callback function that determines if the data browser should start an edit session for a custom item.

Not Recommended

```
typedef Boolean (*DataBrowserEditItemProcPtr) (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    CFStringRef theString,
    Rect *maxEditTextRect,
    Boolean *shrinkToFit
);
```

You would declare an edit-item callback function named `MyDataBrowserEditItemCallback` like this:

```
Boolean MyDataBrowserEditItemCallback (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    CFStringRef theString,
    Rect *maxEditTextRect,
    Boolean *shrinkToFit
);
```

Parameters

browser
A data browser.

item
The item ID number for the item.

property
The property ID for the item. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

theString
The string to be edited. See Special Considerations for more information.

maxEditTextRect
On input, a pointer to a rectangle structure. On return, set the rectangle to the largest size the edit field can grow to. If the text grows beyond the size of the edit field, the text scrolls as the user types. This parameter is used only if the parameter `shrinkToFit` is `true`. Otherwise, the current size of the text editing field is used.

shrinkToFit
On input, a pointer to a Boolean variable. On return, set this variable to `true` if you want the data browser to expand or shrink the text editing field to match the width of the text in the edit field. Note that this parameter is currently ignored; `shrinkToFit` is always `true` by default.

Return Value
A value that indicates whether or not you want to start an edit operation for the given property of the item. If your application performs the editing operation, your callback returns `true`. Otherwise, your callback returns `false`.

Discussion
The edit-item callback is called by the data browser for an item whose property is `kDataBrowserCustomType`. Your callback must determine whether an editing session should be started and, if so, set the string to be edited, set the size of the edit rectangle, and specify whether the text editing field can adjust to match the width of the text in the edit field.

To provide a pointer to your edit-item callback, you create a universal procedure pointer (UPP) of type `DataBrowserEditItemUPP`, using the function `NewDataBrowserEditItemUPP` (page 92). You can do so with code similar to the following:

```
DataBrowserEditItemUPP MyDataBrowserEditItemUPP;
MyDataBrowserEditItemUPP = NewDataBrowserEditItemUPP
                        (&MyDataBrowserEditItemCallback);
```

You can then assign `MyDataBrowserEditItemUPP` to the `editItemCallback` field of the structure `DataBrowserCustomCallbacks` (page 169). You install your data browser custom callbacks using the function `SetDataBrowserCustomCallbacks` (page 107).

When you no longer need the UPP, remove it using the `DisposeDataBrowserEditItemUPP` (page 35) function.

Special Considerations

This callback does not work properly. The `theString` parameter is an immutable sting, which means it is not possible for the callback to set the string to the text that is to be edited.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserGetContextualMenuProcPtr

Defines a pointer to a get-contextual-menu callback function that obtains a menu and information about the menu.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
typedef void (*DataBrowserGetContextualMenuProcPtr) (
    ControlRef browser,
    MenuRef *menu,
    UInt32 *helpType,
    CFStringRef *helpItemString,
    AEDesc *selection
);
```

You would declare a get-contextual-menu callback function named `MyDataBrowserGetContextualMenuCallback` like this:

```
void MyDataBrowserGetContextualMenuCallback (
    ControlRef browser,
    MenuRef *menu,
    UInt32 *helpType,
    CFStringRef *helpItemString,
    AEDesc *selection
);
```

Parameters

browser

A data browser.

menu

On input, a pointer to a menu. Your callback must set the pointer to the menu that you want to have displayed for the given item. Your application is responsible for disposing of the menu; you typically perform this task in the callback `DataBrowserSelectContextualMenuProcPtr` (page 161).

helpType

On input, a pointer to an unsigned 32-bit integer. On return, this value specifies the type of help available for this item. This value is then passed by the data browser to the Menu Manager function `ContextualMenuSelect`. You can provide one of the following values:

- `kCMHelpItemNoHelp` if your application does not support help. The Menu Manager inserts an appropriate string into the menu and then disables the associated help item.
- `kCMHelpItemAppleGuide` if your application supports Apple Guide help. The Menu Manager inserts the name of the main Apple Guide file into the menu and enables the associated help item. You can pass this in Mac OS 9. Apple Guide is not supported in Mac OS X. In Mac OS X, this value is ignored; a generic, but inactive, help item is displayed.
- `kCMHelpItemOtherHelp` if your application supports some other form of help. In this case, your application must pass a valid string in the `helpItemString` parameter. The Menu Manager inserts the string in the menu and enables the associated help item.

See *Menu Manager Reference* for more information about these constants.

helpItemString

On input, a `CFStringRef` variable. On return, a CFString object that contains the name of the item to display in the contextual menu. This is the first item that appears in the contextual menu. If you pass `NULL`, the default string (“Help”) is displayed. Data Browser does not retain the string; it releases it.

selection

On input, a pointer to an empty `AEDesc` data structure. On return, the structure contains the data supplied by your callback. The data browser passes this structure to the function `ContextualMenuSelect`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
The get-contextual-menu callback is called by the data browser when the user Control-clicks in the data browser. You application provides a menu and information about help that is (or is not) available for the data browser. You can determine what to provide for the content of the menu and what information to put in the `AEDesc` structure by calling the function `GetDataBrowserItems` with the `state` parameter set to `kDataBrowserItemIsSelected`. This tells you what items are selected, which you can then use to choose the appropriate information to supply.

To provide a pointer to your get-contextual-menu callback, you create a universal procedure pointer (UPP) of type `DataBrowserGetContextualMenuUPP`, using the function `NewDataBrowserGetContextualMenuUPP` (page 93). You can do so with code similar to the following:

```
DataBrowserGetContextualMenuUPP MyDataBrowserGetContextualMenuUPP;
MyDataBrowserGetContextualMenuUPP = NewDataBrowserGetContextualMenuUPP
                        (&MyDataBrowserGetContextualMenuCallback);
```

You can then assign `MyDataBrowserGetContextualMenuUPP` to the `getContextualMenuCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserGetContextualMenuUPP` (page 35) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserHitTestProcPtr

Defines a pointer to a hit-test callback function that determines if the pointer is over content that can be selected or dragged.

```
typedef Boolean (*DataBrowserHitTestProcPtr) (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    const Rect *mouseRect
);
```

You would declare a hit-test callback function named `MyDataBrowserHitTestCallback` like this:

```
Boolean MyDataBrowserHitTestCallback (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    const Rect *mouseRect
);
```


2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Parameters

browser

A data browser.

itemID

The item ID number for the item over which the pointer is located.

property

The property ID for the column in which the pointer is located. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

theRect

A pointer to the bounding rectangle, in local coordinates, of the item.

mouseRect

A pointer to a rectangle structure that contains the local coordinates of the selection. If the top-left and bottom-right coordinates of this rectangle are identical, then a single point is being tested. If they differ, then the data browser is testing to see whether your custom item is inside of the bounding rectangle of a selection.

Return Value

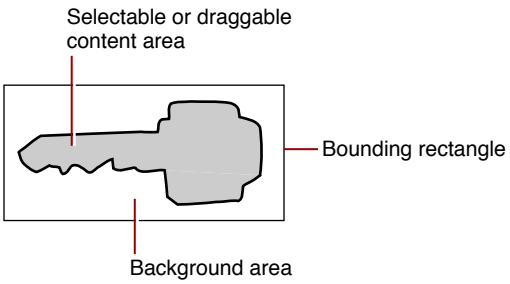
Your application returns a `true` value for either of the following conditions:

- The pointer is located over the part of the item that can be selected or dragged.
- The rectangle provided in the parameter `mouseRect` intersects with the content area of the item that can be selected or dragged.

Discussion

The hit-test callback is called by the data browser when the user hovers the pointer over, clicks the mouse within, or drags within an item whose display type is `kDataBrowserCustomType`. Your callback can use the functions `SetRect` or `SectRgn` to determine if the selectable content of the custom item is part of the selection. Figure 2 illustrates a situation for which the selectable or draggable content area differs from the background area in which the item is displayed.

Figure 2 Differentiation between the selectable content and background



To provide a pointer to your hit-test callback, you create a universal procedure pointer (UPP) of type `DataBrowserHitTestItemUPP`, using the function `NewDataBrowserHitTestUPP` (page 93). You can do so with code similar to the following:

```
DataBrowserHitTestUPP MyDataBrowserHitTestUPP;
MyDataBrowserHitTestUPP = NewDataBrowserHitTestUPP
                        (&MyDataBrowserHitTestCallback);
```


2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

You can then assign `MyDataBrowserHitTestUPP` to the `hitTestCallback` field of the structure `DataBrowserCustomCallbacks` (page 169). You install your data browser custom callbacks using the function `SetDataBrowserCustomCallbacks` (page 107).

When you no longer need the UPP, remove it using the `DisposeDataBrowserHitTestUPP` (page 35) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserItemAcceptDragProcPtr

Defines a pointer to an item-accept-drag callback function that determines if a custom item can accept a drag object.

```
typedef DataBrowserDragFlags (*DataBrowserItemAcceptDragProcPtr)  (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    DragRef theDrag
);
```

You would declare an item-accept-drag callback function named `MyDataBrowserItemAcceptDragCallback` like this:

```
DataBrowserDragFlags MyDataBrowserItemAcceptDragCallback (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    DragRef theDrag
);
```

Parameters

browser
A data browser.

itemID
The item ID number for the item the drag object is held over. If the drag object is over the data browser but not over any specific item, the `item` parameter contains the item ID that represents one of the following:

- The target in list view. (See `SetDataBrowserTarget` (page 133).)
- The column the drag object is over in column view

property
The property ID of the column the dragged object is over. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

theRect

A pointer to the bounding rectangle of the item, in local coordinates relative to the current port.

theDrag

The drag reference provided by the data browser to your callback.

Return Value

If your callback determines the drag object can be accepted, return a nonzero value that has the bit `kDataBrowserItemIsDragTarget` set. Otherwise return `kDataBrowserItemNoState`. The return value is then passed to your item-receive-drag callback in the `dragFlags` parameter.

Discussion

The item-accept-drag callback is called by the data browser for an item whose display type is `kDataBrowserCustomType` when a drag object is moved over the item. Your application determines whether or not the associated item can accept the drag object. If the item cannot accept the drag object, return `0`. Otherwise, if the item is an acceptable drop location for the drag object, return a nonzero value.

If the drag object was acceptable and the drop occurs over that same item ID and property ID pair, the `DataBrowserDragFlags` values you returned from your item-accept-drag callback are passed in the `dragFlags` parameter to your item-receive-drag callback. This allows you to generate state information during drag tracking that can be communicated to you at drop time.

Do not call the function `SetOrigin` in this or any of the other drag processing callbacks.

To provide a pointer to your item-accept-drag callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemAcceptDragUPP`, using the function `NewDataBrowserItemAcceptDragUPP` (page 94). You can do so with code similar to the following:

```
DataBrowserItemAcceptDragUPP MyDataBrowserItemAcceptDragUPP;
MyDataBrowserItemAcceptDragUPP = NewDataBrowserItemAcceptDragUPP
                        (&MyDataBrowserItemAcceptDragCallback);
```

You can then assign `MyDataBrowserItemAcceptDragUPP` to the `itemAcceptDragCallback` field of the structure `DataBrowserCustomCallbacks` (page 169). You install your data browser custom callbacks using the function `SetDataBrowserCustomCallbacks` (page 107).

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemAcceptDragUPP` (page 36) function.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`HIDataBrowser.h`

DataBrowserItemCompareProcPtr

Defines a pointer to an item-comparison callback function that orders the values displayed in a column.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
typedef Boolean (*DataBrowserItemCompareProcPtr) (
    ControlRef browser,
    DataBrowserItemID itemOne,
    DataBrowserItemID itemTwo,
    DataBrowserPropertyID sortProperty
);
```

You would declare an item-comparison callback function named `MyDataBrowserItemCompareCallback` like this:

```
Boolean MyDataBrowserItemCompareCallback (
    ControlRef browser,
    DataBrowserItemID itemOne,
    DataBrowserItemID itemTwo,
    DataBrowserPropertyID sortProperty
);
```

Parameters

browser

A data browser.

itemOne

The item ID of the first item to use in the comparison.

itemTwo

The item ID of the second item to use in the comparison.

sortProperty

The property ID for the column to sort. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

Return Value

Your callback returns `true` if the value of the data referenced by `itemOne` is less than the value of the data referenced by `itemTwo`. It returns `false` if the value of the data referenced by `itemOne` is greater than or equal to the value of the data referenced by `itemTwo`.

Discussion

The item-comparison callback is called by the data browser when it needs to order the values displayed in a column. Your callback determines the display type of the data, then carries out the appropriate comparison for that data.

If you want your callback to use secondary and tertiary sorting, your application must keep track of previous sorting operations. Then you must make sure that each time a user clicks a column, the column is sorted, but the associated sorting orders for secondary and tertiary items in the column are preserved.

To provide a pointer to your item-comparison callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemCompareUPP`, using the function `NewDataBrowserItemCompareUPP` (page 94). You can do so with code similar to the following:

```
DataBrowserItemCompareUPP MyDataBrowserItemCompareUPP;
MyDataBrowserItemCompareUPP = NewDataBrowserItemCompareUPP
                            (&MyDataBrowserItemCompareCallback);
```

You can then assign `MyDataBrowserItemCompareUPP` to the `itemCompareCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemCompareUPP` (page 36) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserItemDataProcPtr

Defines a pointer to an item-data callback function that gets and sets properties for individual items in a data browser.

```
typedef OSStatus (*DataBrowserItemDataProcPtr) (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    DataBrowserItemDataRef itemData,
    Boolean setValue
);
```

You would declare an item-data callback function named `MyDataBrowserItemDataCallback` like this:

```
OSStatus MyDataBrowserItemDataCallback (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    DataBrowserItemDataRef itemData,
    Boolean setValue
);
```

Parameters

browser
The data browser.

item
The item ID of the item whose data is set or obtained.

property
A property ID. This value can be any of the following:

- A four-character sequence that you assign to represent a column in list view.
- Any of the API-defined properties, such as `kDataBrowserItemSelfIdentityProperty` for a column in column view or `kDataBrowserItemIsContainerProperty` for an item in list or column view that has children. See "Properties" (page 186) for a complete list and more information on the API-defined properties.

itemData
The data buffer that either contains the data to set or receives the data to obtain.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

setValue

A value that indicates whether data is to be obtained or set. This value is `false` if your application needs to set the value of the item by calling one of the set functions described in the section "Getting and Setting Item Data" (page 19). This value is `true` if the value of the item has changed. In this case, you should call the appropriate get function, passing the item data reference provided to you in the `itemData` parameter.

Return Value
A result code. See "Data Browser Result Codes" (page 202).

Discussion
The item-data callback communicates data between the data browser and your application. When the data browser needs to display a value for an item, it invokes your callback to request the data. If the user changes the value, the data browser invokes your callback with a new copy of the data that you can use to replace your application's internal copy. Your application must supply an item-data callback; otherwise, your data browser will not contain any data.

Your callback determines the kind of data is associated with an item and whether data needs to be obtained or set. Then, your callback takes the appropriate action by calling one of the functions listed in "Getting and Setting Item Data" (page 19).

To provide a pointer to your item-data callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemDataUPP`, using the function `NewDataBrowserItemDataUPP` (page 95). You can do so with code similar to the following:

```
DataBrowserItemDataUPP MyDataBrowserItemDataUPP;
MyDataBrowserItemDataUPP = NewDataBrowserItemDataUPP
                           (&MyDataBrowserItemDataCallback);
```

You can then assign `MyDataBrowserItemDataUPP` to the `itemDataCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemDataUPP` (page 37) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserItemDragRgnProcPtr

Defines a pointer to an item-drag-region callback function that determines which part of the item rectangle to use when creating a transparent image for a dragged custom item.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
typedef void (*DataBrowserItemDragRgnProcPtr) (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    RgnHandle dragRgn
);
```

You would declare an item-drag-region callback function named `MyDataBrowserItemDragRgnCallback` like this:

```
void MyDataBrowserItemDragRgnCallback (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    RgnHandle dragRgn
);
```

Parameters

browser

A data browser.

itemID

The item ID number for the row for which the drag image is generated.

property

The property ID for the column for which the drag image is generated. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

theRect

A pointer to the bounding rectangle of the item, in local coordinates.

dragRgn

On return, the drag region set to the portion of the rectangle to use for the transparent drag image. Typically this is the boundary of the content area inside your custom item. This region is used as a mask when passed to your custom draw-item callback.

Discussion

The item-drag-region callback is called by the data browser for an item whose display type is `kDataBrowserCustomType` when a drag is about to begin. Your application determines which part of the item rectangle to use when creating the transparent image that appears during a drag operation. The data browser uses this area as a clipping region when it invokes your draw-item callback.

Do not call the function `SetOrigin` in this or any of the other drag processing callbacks.

To provide a pointer to your item-drag-region callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemDragRgnUPP`, using the function `NewDataBrowserItemDragRgnUPP` (page 95). You can do so with code similar to the following:

```
DataBrowserItemDragRgnUPP MyDataBrowserItemDragRgnUPP;
MyDataBrowserItemDragRgnUPP = NewDataBrowserItemDragRgnUPP
                        (&MyDataBrowserItemDragRgnCallback);
```

You can then assign `MyDataBrowserItemDragRgnUPP` to the `itemDragRgnCallback` field of the structure `DataBrowserCustomCallbacks` (page 169). You install your data browser custom callbacks using the function `SetDataBrowserCustomCallbacks` (page 107).

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemDragRgnUPP` (page 37) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserItemHelpContentProcPtr

Defines a pointer to an item-help-content callback function that provides help tag content for an item.

```
typedef void (*DataBrowserItemHelpContentProcPtr) (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    HMContentRequest inRequest,
    HMContentProvidedType *outContentProvided,
    HMHelpContentPtr ioHelpContent
);
```

You would declare an item-help-content callback function named `MyDataBrowserItemHelpContentCallback` like this:

```
void DataBrowserItemHelpContentCallback (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserPropertyID property,
    HMContentRequest inRequest,
    HMContentProvidedType *outContentProvided,
    HMHelpContentPtr ioHelpContent
);
```

Parameters

browser
A data browser.

item
The item ID of the item to provide help for.

property
The property ID of the column to provide help for. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

inRequest

On input, a value that indicates the nature of the help tag content request. Your callback is passed one of the following constants:

- `kHMSupplyContent` indicates your callback needs to supply help content.
- `kHMDisposeContent` indicates your callback must dispose of help content.

outContentProvided

On input, a help-content-type variable. On return, the variable is set to one of the following constants that indicate whether your item-help callback was able to fulfill the request specified in the `inRequest` parameter:

- `kHMContentProvided` indicates help content is provided in the `ioHelpContent` parameter.
- `kHMContentNotProvided` indicates help content is not provided. When your callback returns this constant, the Carbon Help Manager consults other help content providers in the hierarchy until the request for help tag content is fulfilled, the top of the hierarchy is reached, or a help tag callback notifies the Carbon Help Manager to stop propagating the request.
- `kHMContentNotProvidedDontPropagate` indicates help content is not provided. When your callback returns this constant, the Carbon Help Manager assumes that there is no help content for the data browser item and does not propagate the request.

See *Carbon Help Manager Reference* for more information on these constants.

ioHelpContent

A help tag structure that describes the help tag for the item. On input, the data browser passes a value in the version field. If the value of the `inRequest` parameter is `kHMSupplyContent`, your callback must fill in the remaining fields of the structure or specify that it is unable to fulfill the help tag content request.

Discussion

The item-help-content callback is called by the data browser when the user hovers the pointer over an item in a data browser for which you've registered a help tag callback. Your application fills in the help tag structure pointed to by the `ioHelpContent` parameter.

When the help tag for the item is no longer needed, the data browser invokes your callback with a `kHMDisposeContent` request. When you receive this request, free any memory allocated for the help tag content and perform any other cleanup tasks that are necessary.

To provide a pointer to your item-help-content callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemHelpContentUPP`, using the function `NewDataBrowserItemHelpContentUPP` (page 96). You can do so with code similar to the following:

```
DataBrowserItemHelpContentUPP MyDataBrowserItemHelpContentUPP;
MyDataBrowserItemHelpContentUPP = NewDataBrowserItemHelpContentUPP
                        (&MyDataBrowserItemHelpContentCallback);
```

You can then assign `MyDataBrowserItemHelpContentUPP` to the `itemHelpContentCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemHelpContentUPP` (page 37) function.

Availability

Available in Mac OS X v10.0 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

DataBrowserItemNotificationProcPtr

Defines a pointer to an item-notification callback function that notifies your application of changes in the data browser.

```
typedef void (*DataBrowserItemNotificationProcPtr) (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserItemNotification message
);
```

You would declare an item-notification callback function named `MyDataBrowserItemNotificationCallback` like this:

```
void MyDataBrowserItemNotificationCallback (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserItemNotification message
);
```

Parameters

browser
A data browser.

item
The item ID of the item that generated the notification.

message
A notification. See "Item Notifications" (page 182) for a description of the values that can be provided to your callback.

Discussion

The item-notification callback is called by the data browser to notify your application of actions taken by the user (such as editing started, container opened, container closed) or any other condition that your application might choose to respond to. Your item-notification callback can evaluate the notification and take appropriate action.

To provide a pointer to your item-notification callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemNotificationUPP`, using the function `NewDataBrowserItemNotificationUPP` (page 96). You can do so with code similar to the following:

```
DataBrowserItemNotificationUPP MyDataBrowserItemNotificationUPP;
MyDataBrowserItemNotificationUPP = NewDataBrowserItemNotificationUPP
                            (&MyDataBrowserItemNotificationCallback);
```

You can then assign `MyDataBrowserItemNotificationUPP` to the `itemNotificationCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemNotificationUPP` (page 38) function.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Special Considerations

In CarbonLib the item-notification callback is invoked with the three parameters shown in `DataBrowserItemNotificationProcPtr` (page 154). The four-parameter version—`DataBrowserItemNotificationWithItemProcPtr` (page 155)—does not provide valid data in the fourth parameter. Any attempt to use the invalid data in a CarbonLib application may result in a crash.

In Mac OS X, the item-notification callback is invoked with the four parameters shown in `DataBrowserItemNotificationWithItemProcPtr` (page 155). In Mac OS X you have the option of using the `DataBrowserItemNotificationProcPtr` (page 154) or the `DataBrowserItemNotificationWithItemProcPtr` (page 155). Which one you choose depends on whether your application needs to use the data passed to your callback in the fourth parameter (the `itemData` parameter).

Availability

Available in Mac OS X v10.1 and later.

Declared In

`HIDataBrowser.h`

DataBrowserItemNotificationWithItemProcPtr

Defines a pointer to an item-notification-with-data callback function that notifies your application of changes in the data browser and supplies any data associated with the changes.

```
typedef void (*DataBrowserItemNotificationWithItemProcPtr) (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserItemNotification message,
    DataBrowserItemDataRef itemData
);
```

You would declare an item-notification-with-data callback function named `MyDataBrowserItemNotificationWithItemCallback` like this:

```
void MyDataBrowserItemNotificationWithItemCallback (
    ControlRef browser,
    DataBrowserItemID item,
    DataBrowserItemNotification message,
    DataBrowserItemDataRef itemData
);
```

Parameters

browser

A data browser.

item

The item ID of the item that generated the notification.

message

A notification. See "Item Notifications" (page 182) for a description of the values that can be provided to your callback.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

itemData

The data associated with the changes in the data browser that caused this callback to be invoked. You pass this data to functions that get and set item data (such as `SetDataBrowserItemDataIcon` (page 113), `SetDataBrowserItemDataText` (page 118), `GetDataBrowserItemDataIcon` (page 52), and `GetDataBrowserItemDataText` (page 58)). See "Getting and Setting Item Data" (page 19) for a list of all the function that can use item data.

Discussion

The item-notification-with-data callback is called by the data browser to notify your application of actions taken by the user (such as editing started, container opened, container closed) or any other condition that your application might choose to respond to. Your item-notification-with-data callback can evaluate the notification and take appropriate action. Unlike the callback `DataBrowserItemNotificationProcPtr` (page 154), the callback `DataBrowserItemNotificationWithItemProcPtr` (page 155) provides a fourth parameter that contains any data associated with the item from which the notification is generated. This data is available only to Mac OS X applications. See Special Considerations for more details.

To provide a pointer to your item-notification-with-data callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemNotificationWithItemUPP`, using the function `NewDataBrowserItemNotificationWithItemUPP` (page 97). You can do so with code similar to the following:

```
DataBrowserItemNotificationWithItemUPP
                    MyDataBrowserItemNotificationWithItemUPP;
MyDataBrowserItemNotificationWithItemUPP =
                NewDataBrowserItemNotificationWithItemUPP
                (&MyDataBrowserItemNotificationWithItemCallback);
```

You can then assign `MyDataBrowserItemNotificationWithItemUPP` to the `itemNotificationCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemNotificationWithItemUPP` (page 38) function.

Special Considerations

In CarbonLib the item-notification callback is invoked with the three parameters shown in `DataBrowserItemNotificationProcPtr` (page 154). The four-parameter version (`DataBrowserItemNotificationWithItemProcPtr` (page 155)) does not provide valid data in the fourth parameter. Any attempt to use the invalid data in a CarbonLib application may result in a crash.

In Mac OS X, the item-notification callback is invoked with the four parameters shown in `DataBrowserItemNotificationWithItemProcPtr` (page 155). In Mac OS X you have the option of using the `DataBrowserItemNotificationProcPtr` (page 154) or the `DataBrowserItemNotificationWithItemProcPtr` (page 155). Which one you choose depends on whether your application needs to use the data passed to your callback in the fourth parameter (`itemData`).

Availability

Available in Mac OS X v10.1 and later.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DataBrowserItemProcPtr

Defines a pointer to an item-iterator callback function that is applied by the function `ForEachDataBrowserItem` to each item in a data browser.

```
typedef void (*DataBrowserItemProcPtr) (
    DataBrowserItemID item,
    DataBrowserItemState state,
    void *clientData
);
```

You would declare an item-iterator callback function named `MyDataBrowserItemCallback` like this:

```
void MyDataBrowserItemCallback (
    DataBrowserItemID item,
    DataBrowserItemState state,
    void *clientData
);
```

Parameters

item

The item ID of the item to operate on.

state

The state of the item. See "Item States" (page 184) for a description of possible states. If the function `ForEachDataBrowserItem` (page 42) is set up to operate on items of a specified state, then the state passed to your callback includes the state specified as a parameter to `ForEachDataBrowserItem`.

clientData

A pointer to a buffer, local variable, or other storage location created and disposed of by your application, and supplied to the data browser with a previous call to `ForEachDataBrowserItem`.

Discussion

An item-iterator callback is supplied as a parameter to the function `ForEachDataBrowserItem`. The function applies your callback to each data item that meets the criteria specified by the function `ForEachDataBrowserItem`.

To provide a pointer to your item-iterator callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemUPP`, using the function `NewDataBrowserItemUPP` (page 98). You can do so with code similar to the following:

```
DataBrowserItemUPP MyDataBrowserItemUPP;
MyDataBrowserItemUPP = NewDataBrowserItemUPP
                            (&MyDataBrowserItemCallback);
```

You can then pass `MyDataBrowserItemUPP` in the `callback` parameter of the function `ForEachDataBrowserItem`. When you no longer need the UPP, remove it using the `DisposeDataBrowserItemUPP` (page 39) function.

Availability

Available in CarbonLib 1.1 and later.

Available in Mac OS X version 10.0 and later.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DataBrowserItemReceiveDragProcPtr

Defines a pointer to an item-receive-drag callback function that receives a drop over a custom item.

```
typedef Boolean (*DataBrowserItemReceiveDragProcPtr) (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    DataBrowserDragFlags dragFlags,
    DragRef theDrag
);
```

You would declare an item-receive-drag callback function named `MyDataBrowserItemReceiveDragCallback` like this:

```
Boolean MyDataBrowserItemReceiveDragCallback (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    DataBrowserDragFlags dragFlags,
    DragRef theDrag
);
```

Parameters

browser
: A data browser.

itemID
: The item ID number for the item over which the drop occurred.

property
: The property ID for the column in which the drop occurred. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

dragFlags
: A drag flag. This value is `kDataBrowserItemIsDragTarget` if your item-accept-drag callback determined the drag object can be accepted.

theDrag
: The drag reference provided by the data browser to your callback.

Return Value

A value that indicates whether the drop is received. Your callback returns `true` if it successfully receives the drag object. If it returns `false`, zoom-back animation occurs.

Discussion

After your item-accept-drag callback has determined that a location can accept a drag object and after a drop operation occurs, the data browser calls your item-receive-drag callback. Your application takes whatever actions necessary to add the dropped data to the data browser.

Do not call the function `SetOrigin` in this or any of the drag processing callbacks.

To provide a pointer to your item-receive-drag callback, you create a universal procedure pointer (UPP) of type `DataBrowserItemReceiveDragUPP`, using the function `NewDataBrowserItemReceiveDragUPP` (page 97). You can do so with code similar to the following:

```
DataBrowserItemReceiveDragUPP MyDataBrowserItemReceiveDragUPP;
```


2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
MyDataBrowserItemReceiveDragUPP = NewDataBrowserItemReceiveDragUPP
                        (&MyDataBrowserItemReceiveDragCallback);
```

You can then assign `MyDataBrowserItemReceiveDragUPP` to the `itemReceiveDragCallback` field of the structure `DataBrowserCustomCallbacks` (page 169). You install your data browser custom callbacks using the function `SetDataBrowserCustomCallbacks` (page 107).

When you no longer need the UPP, remove it using the `DisposeDataBrowserItemReceiveDragUPP` (page 39) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserPostProcessDragProcPtr

Defines a pointer to a postprocess-drag callback function that performs necessary cleanup tasks, such as deallocating resources that were allocated by your other drag processing callbacks.

```
typedef void (*DataBrowserPostProcessDragProcPtr) (
    ControlRef browser,
    DragRef theDrag,
    OSStatus trackDragResult
);
```

You would declare a postprocess-drag callback function named `MyDataBrowserPostProcessDragCallback` like this:

```
void MyDataBrowserPostProcessDragCallback (
    ControlRef browser,
    DragRef theDrag,
    OSStatus trackDragResult
);
```

Parameters

browser
A data browser.

theDrag
The drag reference provided by the data browser to your callback.

trackDragResult
The result returned by the function `TrackDrag` and passed to your callback by the data browser.

Discussion
This callback is called after starting a drag from within the data browser. It is not called if the drag originated from somewhere else.

The postprocess-drag callback is called by the data browser after a drag process is complete and any drag processing callback routines you installed (add drag, accept drag, or receive drag callbacks) were called during the drag operation. Your postprocess-drag callback deallocates any resources that were allocated by

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

your other drag-processing callbacks. Your postprocess-drag callback is called immediately before the drag reference is deallocated by the data browser so your application should not assume the drag reference exists after your callback completes.

To provide a pointer to your postprocess-drag callback, you create a universal procedure pointer (UPP) of type `DataBrowserPostProcessDragUPP`, using the function `NewDataBrowserPostProcessDragUPP` (page 98). You can do so with code similar to the following:

```
DataBrowserPostProcessDragUPP MyDataBrowserPostProcessDragUPP;
MyDataBrowserPostProcessDragUPP = NewDataBrowserPostProcessDragUPP
                        (&MyDataBrowserPostProcessDragCallback);
```

You can then assign `MyDataBrowserPostProcessDragUPP` to the `postProcessDragCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserPostProcessDragUPP` (page 39) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserReceiveDragProcPtr

Defines a pointer to a receive-drag callback function that extract items from a drag object and handles the drag item appropriately.

```
typedef Boolean (*DataBrowserReceiveDragProcPtr) (
    ControlRef browser,
    DragRef theDrag,
    DataBrowserItemID item
);
```

You would declare a receive-drag callback function named `MyDataBrowserReceiveDragCallback` like this:

```
Boolean MyDataBrowserReceiveDragCallback (
    ControlRef browser,
    DragRef theDrag,
    DataBrowserItemID item
);
```

Parameters

browser
A data browser.

theDrag
The drag reference provided by the data browser to your callback.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

item

The item ID of the item over which the drop operation occurred. If the drag object is over the data browser, but not over any specific item, the `item` parameter contains the item ID that represents one of the following:

- In list view, the target. (See `SetDataBrowserTarget` (page 133).)
- In column view, the item ID that represents the column the drag object is over

Return Value
Your callback returns `true` if it successfully processes the information in the drag object. Otherwise, your callback returns `false` to have zoom-back animation occur for the drag object, thereby indicating to the user that the drag operation was not successful.

Discussion
The receive-drag callback is called by the data browser when your application needs to receive a drag object. Your application extracts the items it needs from the drag object and processes them appropriately.

To provide a pointer to your receive-drag callback, you create a universal procedure pointer (UPP) of type `DataBrowserReceiveDragUPP`, using the function `NewDataBrowserReceiveDragUPP` (page 99). You can do so with code similar to the following:

```
DataBrowserReceiveDragUPP MyDataBrowserReceiveDragUPP;
MyDataBrowserReceiveDragUPP = NewDataBrowserReceiveDragUPP
                        (&MyDataBrowserReceiveDragCallback);
```

You can then assign `MyDataBrowserReceiveDragUPP` to the `receiveDragCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserReceiveDragUPP` (page 40) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserSelectContextualMenuProcPtr

Defines a pointer to a select-contextual-menu callback function that processes a contextual menu selection.

```
typedef void (*DataBrowserSelectContextualMenuProcPtr) (
    ControlRef browser,
    MenuRef menu,
    UInt32 selectionType,
    SInt16 menuID,
    MenuItemIndex menuItem
);
```

You would declare a select-contextual-menu callback function named `MyDataBrowserSelectContextualMenuCallback` like this:

```
void MyDataBrowserSelectContextualMenuCallback (
    ControlRef browser,
```


2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
    MenuRef menu,
    UInt32 selectionType,
    SInt16 menuID,
    MenuItemIndex menuItem
);
```

Parameters

browser

A data browser.

menu

On input, the menu reference your application provided to the data browser in the callback `DataBrowserGetContextualMenuProcPtr` (page 142).

selectionType

On input, the selection type provided to the data browser from the Menu Manager function `ContextualMenuSelect`.

menuID

On input, the menu ID of the menu selected. This value is `0` if no selection was made.

menuItem

The menu item index of the item selected.

Discussion

The select-contextual-menu callback is called by the data browser when the user finishes interacting with a contextual menu. Your callback needs to:

- Check whether the user chose an item from the menu. If so, process the selection appropriately. Note that your callback is invoked even if the user does not choose an item from the menu.
- Optionally dispose of the menu you allocated in your get-contextual-menu callback, and that is passed to your select-contextual-menu callback in the `menu` parameter.

To provide a pointer to your select-contextual-menu callback, you create a universal procedure pointer (UPP) of type `DataBrowserSelectContextualMenuUPP`, using the function `NewDataBrowserSelectContextualMenuUPP` (page 99). You can do so with code similar to the following:

```
DataBrowserSelectContextualMenuUPP MyDataBrowserSelectContextualMenuUPP;
MyDataBrowserSelectContextualMenuUPP =
                 NewDataBrowserSelectContextualMenuUPP
                (&MyDataBrowserSelectContextualMenuCallback);
```

You can then assign `MyDataBrowserSelectContextualMenuUPP` to the `selectContextualMenuCallback` field of the structure `DataBrowserCallbacks` (page 168). You install your data browser callbacks using the function `SetDataBrowserCallbacks` (page 104).

When you no longer need the UPP, remove it using the `DisposeDataBrowserSelectContextualMenuUPP` (page 40) function.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`HIDataBrowser.h`

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DataBrowserTrackingProcPtr

Defines a pointer to a tracking callback function that implements tracking behavior for such tasks as highlighting a button or providing animation when the user clicks a custom item.

```
typedef DataBrowserTrackingResult (*DataBrowserTrackingProcPtr)  (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    Point startPt,
    EventModifiers modifiers
);
```

You would declare a tracking callback function named `MyDataBrowserTrackingCallback` like this:

```
DataBrowserTrackingResult MyDataBrowserTrackingCallback (
    ControlRef browser,
    DataBrowserItemID itemID,
    DataBrowserPropertyID property,
    const Rect *theRect,
    Point startPt,
    EventModifiers modifiers
);
```

Parameters

browser
: A data browser.

itemID
: The item ID number for the item the user clicked.

property
: The property ID for the column in which the pointer is located. In list view, this is the four-character sequence that you previously assigned to the column. In column view, this is the property `kDataBrowserItemSelfIdentityProperty`.

theRect
: A pointer to the bounding rectangle of the item, in local coordinates relative to the current port.

startPt
: The location of the pointer at the start of the click.

modifiers
: The state of the modifier keys. See *Carbon Event Manager Reference* in Carbon Events & Other Input Documentation for a list of the constants that can be passed to your callback.

Return Value

A tracking result that indicates whether further processing is required by the data browser. Your callback returns `kDataBrowserStopTracking`, `kDataBrowserContentHit`, or `kDataBrowserNothingHit`. See the Discussion for more details.

Discussion

The tracking callback is called by the data browser for an item whose display type is `kDataBrowserCustomType` when a mouse click is inside the content area of the item. Your tracking callback is called only for mouse-down events and only after your `DataBrowserHitTestProcPtr` (page 144) callback returns `true`. Your tracking callback performs one of following tasks:

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

- Provides its own custom tracking behavior and animation and returns the result `kDataBrowserStopTracking`. This result informs the data browser that your application handled the click and no further processing is required. The data browser does not attempt to display a contextual menu, start a drag operation, process a double-click, or draw a selection rectangle. You are responsible for all facets of click handling if you return `kDataBrowserStopTracking`.
- Returns the value `kDataBrowserNothingHit` to indicate a negative hit and no further processing needs to take place. This result indicates that a nonselectable portion—whitespace—was hit. The data browser won't select the item, but it could, for example, start a selection rectangle.
- Returns the value `kDataBrowserContentHit` to request that the data browser continues to process the click. This result indicates that a selectable portion of the item is hit. The data browser selects the item and takes other appropriate actions.

To provide a pointer to your tracking callback, you create a universal procedure pointer (UPP) of type `DataBrowserTrackingUPP`, using the function `NewDataBrowserTrackingUPP` (page 100). You can do so with code similar to the following:

```
DataBrowserTrackingUPP MyDataBrowserTrackingUPP;
MyDataBrowserTrackingUPP = NewDataBrowserTrackingUPP
                        (&MyDataBrowserTrackingCallback);
```

You can then assign `MyDataBrowserTrackingUPP` to the `trackingCallback` field of the structure `DataBrowserCustomCallbacks` (page 169). You install your data browser custom callbacks using the function `SetDataBrowserCustomCallbacks` (page 107).

When you no longer need the UPP, remove it using the `DisposeDataBrowserTrackingUPP` (page 41) function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

Data Types

DataBrowserAccessibilityItemInfo

Contains a structure that describes data browser accessibility item information.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
struct DataBrowserAccessibilityItemInfo {
    UInt32 version;
    union {
        DataBrowserAccessibilityItemInfoV0 v0;
        DataBrowserAccessibilityItemInfoV1 v1;
    } u;
typedef struct DataBrowserAccessibilityItemInfoV0 v0;
typedef struct DataBrowserAccessibilityItemInfoV1 v1;
```

Fields

`version`

Identifies how to interpret the following union. Set this field to `0` if you fill out the union's data in the form of a `DataBrowserAccessibilityItemInfoV0` structure. Set this field to `1` if you fill out the union's data in the form of a `DataBrowserAccessibilityItemInfoV1` structure.

`u.v0`

A `DataBrowserAccessibilityItemInfoV0` (page 165) structure.

`u.v1`

A `DataBrowserAccessibilityItemInfoV1` (page 166) structure.

DataBrowserAccessibilityItemInfoV0

Contains a description of data browser accessibility item information.

```
struct DataBrowserAccessibilityItemInfoV0 {
    DataBrowserItemID container;
    DataBrowserItemID item;
    DataBrowserPropertyID columnProperty;
    DataBrowserPropertyPart propertyPart;
    }
typedef struct DataBrowserAccessibilityItemInfoV0  DataBrowserAccessibilityInfoV0;
```

Fields

`container`

The `DataBrowserItemID` of the container the `AXUIElementRef` represents or lives within. Even `kDataBrowserNoItem` might be meaningful, since it is the root container ID if you haven't overridden it via `SetDataBrowserTarget` (page 133). In list view, the container helps narrow down the `AXUIElementRef` to either a disclosed child of another row, or the list as a whole. In column view, the container helps narrow down the `AXUIElementRef` to a column. See also the description of the `columnProperty` field.

`item`

The `DataBrowserItemID` of the item the `AXUIElementRef` represents or lives within. If `item` is `kDataBrowserNoItem`, the `AXUIElementRef` represents just the container. In list view, `item` helps narrow down the `AXUIElementRef` to a row or the root container as a whole. In column view, `item` helps narrow down the `AXUIElementRef` to a cell or a column as a whole. See also the description of the `columnProperty` field.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`columnProperty`

The `DataBrowserPropertyID` of the column the `AXUIElementRef` represents or lives within. If `columnProperty` is `kDataBrowserItemNoProperty` and `item` is not `kDataBrowserNoItem`, the `AXUIElementRef` represents a whole row. In list view, this field helps narrow down the `AXUIElementRef` to a cell or a row as a whole. In column view, `columnProperty` must always be set to `kDataBrowserItemNoProperty` unless the `AXUIElementRef` represents the preview column. When the `AXUIElementRef` represents the preview column, `columnProperty` must always be set to `kDataBrowserColumnViewPreviewProperty` and the other fields of this structure must be set to `0` or the equivalent constant.

`propertyPart`

The `DataBrowserPropertyPart` of the sub-cell part the `AXUIElementRef` represents. Examples include the disclosure triangle in a cell, the text in a cell, and the check box in a cell. If `propertyPart` is `kDataBrowserPropertyEnclosingPart` and `columnProperty` is not `kDataBrowserItemNoProperty`, the `AXUIElementRef` represents the cell as a whole. In both list view and column view, this field helps narrow down the `AXUIElementRef` to either a sub-cell part or a cell as a whole. For column view, see also the description of the `columnProperty` field.

DataBrowserAccessibilityItemInfoV1

Contains a description of data browser accessibility item information that includes a row and a column index.

```
struct DataBrowserAccessibilityItemInfoV1 {
    DataBrowserItemID container;
    DataBrowserItemID item;
    DataBrowserPropertyID columnProperty;
    DataBrowserPropertyPart propertyPart;
    DataBrowserTableViewRowIndex rowIndex;
    DataBrowserTableViewColumnIndex columnIndex;
    }
typedef struct DataBrowserAccessibilityItemInfoV1  DataBrowserAccessibilityInfoV1;
```

Fields

`container`

The `DataBrowserItemID` of the container the `AXUIElementRef` represents or lives within. Even `kDataBrowserNoItem` might be meaningful, since it is the root container ID if you haven't overridden it via `SetDataBrowserTarget` (page 133). In list view, the container helps narrow down the `AXUIElementRef` to either a disclosed child of another row, or the list as a whole. In column view, the container helps narrow down the `AXUIElementRef` to a column. See also the description of the `columnProperty` field.

`item`

The `DataBrowserItemID` of the item the `AXUIElementRef` represents or lives within. If `item` is `kDataBrowserNoItem`, the `AXUIElementRef` represents just the container. In list view, `item` helps narrow down the `AXUIElementRef` to a row or the root container as a whole. In column view, `item` helps narrow down the `AXUIElementRef` to a cell or a column as a whole. See also the description of the `columnProperty` field.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`columnProperty`

The `DataBrowserPropertyID` of the column the `AXUIElementRef` represents or lives within. If `columnProperty` is `kDataBrowserItemNoProperty` and `item` is not `kDataBrowserNoItem`, the `AXUIElementRef` represents a whole row. In list view, this field helps narrow down the `AXUIElementRef` to a cell or a row as a whole. In column view, `columnProperty` must always be set to `kDataBrowserItemNoProperty` unless the `AXUIElementRef` represents the preview column. When the `AXUIElementRef` represents the preview column, `columnProperty` must always be set to `kDataBrowserColumnViewPreviewProperty` and the other fields of this structure must be set to `0` or the equivalent constant.

`propertyPart`

The `DataBrowserPropertyPart` of the sub-cell part the `AXUIElementRef` represents. Examples include the disclosure triangle in a cell, the text in a cell, and the check box in a cell. If `propertyPart` is `kDataBrowserPropertyEnclosingPart` and `columnProperty` is not `kDataBrowserItemNoProperty`, the `AXUIElementRef` represents the cell as a whole. In both list view and column view, this field helps narrow down the `AXUIElementRef` to either a sub-cell part or a cell as a whole. For column view, see also the description of the `columnProperty` field.

`rowIndex`

The zero-based `DataBrowserTableViewRowIndex` of the row specified by the other parts of this structure. If the other parts of this structure do not specify a row or a part thereof, this field must be set to 0. Because this field is zero based, you must test the other parts of this structure to see whether this field is meaningful. In list view, when the other parts of this structure specify an item or part thereof, this field must be set to the row index at which the specified item can be found. In column view, when the other parts of this structure specify a cell or part thereof, this field must be set to the row index at which the specified cell can be found.

`propertyPart`

The zero-based `DataBrowserTableViewColumnIndex` of the column specified by the other parts of this structure. If the other parts of this structure do not specify a column or a part thereof, this field must be set to zero. Because this field is zero based, you must test the other parts this structure to see whether this field is meaningful. In list view, when the other parts of this structure specify a cell or part thereof, this field must be set to the column index at which the specified cell can be found. In column view, when the other parts of this structure specify a column or part thereof, this field must be set to the column index at which the specified cell can be found.

DataBrowserPropertyDesc

Contains property and display information for a list view column.

```
struct DataBrowserPropertyDesc {
    DataBrowserPropertyID propertyID;
    DataBrowserPropertyType propertyType;
    DataBrowserPropertyFlags propertyFlags;
};
typedef struct DataBrowserPropertyDesc DataBrowserPropertyDesc;
```

Fields

`propertyID`

A four-character sequence that uniquely identifies the column. If you use Interface Builder to design the data browser, this is the unique value you enter in the Property ID field in the column paned of the Info window for a list view column. (For example, `mTxt` or `BLUE`). The four-character sequence must have at least one uppercase letter in it because sequences that are all lowercase are reserved for Apple.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`propertyType`

The data type or control type to be displayed in the column. See "Display Types" (page 179) for a list of the possible values for this field.

`propertyFlags`

A value that contains property flags that control the display or interaction provided by the column. This is a 32-bit value that is divided into four parts as follows:

- Bits 0–7 specify properties applied to the data browser as a whole—see "Property Flags: Universal" (page 188)
- Bits 8–15 modify display behavior—see "Property Flags: Modifiers" (page 189)
- Bits 16–23 are properties specific to list view—see "Property Flags: Offset and Mask for List View Properties" (page 192) and "Property Flags: List View Column Behavior" (page 193)
- Bits 24–31 can be defined by your application—see "Property Flags: Offset and Mask for Client-Defined Properties" (page 194)

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserCallbacks

Contains universal procedure pointers (UPPs) to callback functions used to obtain information from your application or notify your application of changes to the data browser.

```
struct DataBrowserCallbacks {
    UInt32 version
    union {
        struct {
            DataBrowserItemDataUPP itemDataCallback;
            DataBrowserItemCompareUPP itemCompareCallback;
            DataBrowserItemNotificationUPP itemNotificationCallback;
            DataBrowserAddDragItemUPP addDragItemCallback;
            DataBrowserAcceptDragUPP acceptDragCallback;
            DataBrowserReceiveDragUPP receiveDragCallback;
            DataBrowserPostProcessDragUPP postProcessDragCallback;
            DataBrowserItemHelpContentUPP itemHelpContentCallback;
            DataBrowserGetContextualMenuUPP getContextualMenuCallback;
            DataBrowserSelectContextualMenuUPP selectContextualMenuCallback;
        } v1;
    } u;
};
typedef struct DataBrowserCallbacks DataBrowserCallbacks;
```

Fields

`version`

The version of the custom callbacks structure. Set this field to the constant `kDataBrowserLatestCallbacks`.

`u.v1.itemDataCallback`

A universal procedure pointer to an item-data callback.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`u.v1.itemCompareCallback`
A universal procedure pointer to an item-compare callback.

`u.v1.itemNotificationCallback`
A universal procedure pointer to an item-notification callback or item-notification-with-data callback.

`u.v1.addDragItemCallback`
A universal procedure pointer to an add-drag-item callback.

`u.v1.acceptDragCallback`
A universal procedure pointer to an accept-drag callback.

`u.v1.receiveDragCallback`
A universal procedure pointer to a receive-drag callback.

`u.v1.postProcessDragCallback`
A universal procedure pointer to a postprocess-drag callback.

`u.v1.itemHelpContentCallback`
A universal procedure pointer to an item-help-content callback.

`u.v1.getContextualMenuCallback`
A universal procedure pointer to a get-contextual-menu callback.

`u.v1.selectContextualMenuCallback`
A universal procedure pointer to a select-contextual-menu callback.

Discussion
Your application does not need to fill out the entire data structure. It must provide a UPP to an item-data callback. You provide UPPs only for the other tasks your application wants to handle. For more information on installing callbacks, see the functions `InitDataBrowserCallbacks` (page 79) and `SetDataBrowserCallbacks` (page 104).

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserCustomCallbacks

Contains universal procedure pointers (UPPs) to callback functions that implement custom drawing and user interaction for columns that have the display type `kDataBrowserCustomType`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
struct DataBrowserCustomCallbacks {
    UInt32 version
    union {
        struct {
            DataBrowserDrawItemUPP drawItemCallback;
            DataBrowserEditItemUPP editTextCallback;
            DataBrowserHitTestUPP hitTestCallback;
            DataBrowserTrackingUPP trackingCallback;
            DataBrowserItemDragRgnUPP dragRegionCallback;
            DataBrowserItemAcceptDragUPP acceptDragCallback;
            DataBrowserItemReceiveDragUPP receiveDragCallback;
        } v1;
    } u;
};
typedef struct DataBrowserCustomCallbacks DataBrowserCustomCallbacks;
```

Fields

`version`
: The version of the custom callbacks structure. Use `kDataBrowserLatestCustomCallbacks`.

`u.v1.drawItemCallback`
: A universal procedure pointer to a draw-item callback.

`u.v1.editTextCallback`
: A universal procedure pointer to an edit-text callback.

`u.v1.hitTestCallback`
: A universal procedure pointer to a hit-test callback

`u.v1.trackingCallback`
: A universal procedure pointer to a tracking callback.

`u.v1.dragRegionCallback`
: A universal procedure pointer to an item-drag-region callback.

`u.v1.acceptDragCallback`
: A universal procedure pointer to an item-accept-drag callback.

`u.v1.receiveDragCallback`
: A universal procedure pointer to an item-receive-drag callback.

Discussion

Your application can use the `DataBrowserCustomCallbacks` structure to provide callbacks that control the presentation of user interface elements displayed inside a data browser. Your application does not need to fill out the entire data structure. You need to provide UPPs only for tasks your application wants to handle. For more information on installing callbacks, see the functions `InitDataBrowserCustomCallbacks` (page 80) and `SetDataBrowserCustomCallbacks` (page 107).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`HIDataBrowser.h`

DataBrowserDragFlags

Defines a data type for values used in drag callbacks.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
typedef unsigned long DataBrowserDragFlags;
```

Discussion
This data type is used as a return value for the item-accept-drag callback (`DataBrowserItemAcceptDragProcPtr` (page 146)) and as a parameter to the item-receive-drag callback (`DataBrowserItemReceiveDragProcPtr` (page 158)). The values associated with this data type are `kDataBrowserItemIsDragTarget` and `kDataBrowserItemNoState`. See "Item States" (page 184) for more information on these constants.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserItemDataRef

Defines a data type for a pointer that specifies an item.

```
typedef void *DataBrowserItemDataRef;
```

Discussion
Functions listed in "Getting and Setting Item Data" (page 19) are called from within a data browser item-data callback routine (`DataBrowserItemDataProcPtr` (page 149)). Each of these functions use a `DataBrowserItemDataRef` data type to specify the item to get or set data for.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserItemID

Defines a data type for a value that identifies an item independent of its position in a data browser.

```
typedef UInt32 DataBrowserItemID;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserPropertyFlags

Defines a data type for a value that specifies the behavior and look of a data browser.

```
typedef unsigned long DataBrowserPropertyFlags;
```

Discussion
These constants in the following sections are `DataBrowserPropertyFlags` data types:

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

- "Property Flags: Universal" (page 188). Bits 0–7 modify the appearance or behavior of display properties.
- "Property Flags: Modifiers" (page 189). Bits 8–15 specify how the data associated with a display type is displayed.
- "Property Flags: Offset and Mask for List View Properties" (page 192). Specify an offset and mask for bits 16–23.
- "Property Flags: List View Column Behavior" (page 193). Bits 16–23 specify behaviors for columns in list view.
- "Property Flags: Offset and Mask for Client-Defined Properties" (page 194). Specify an offset and mask for bits 24–31.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserPropertyID

Defines a data type for a value that identifies a column independent of its position in a data browser.

```
typedef UInt32 DataBrowserPropertyID;
```

Discussion
Typically, this is a four-character sequence that you assign to represent a column in list view. For example, a column that displays song titles could be assigned a property ID of `SONG` and you'd assign it to a constant in your application using code similar to the following:

```
kSongColumn = SONG;
```

Then, each time call a function that requires a parameter of type `DataBrowserPropertyID`, supply the appropriate application-defined constant.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`HIDataBrowser.h`

DataBrowserTableViewRowIndex

Defines a data type for a value that specifies a row position in a table view.

```
typedef UInt32 DataBrowserTableViewRowIndex;
```

Discussion
Row indices are zero based.

Availability
Available in Mac OS X v10.0 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Declared In
HIDataBrowser.h

DataBrowserTableViewColumnIndex

Defines a data type for a value that specifies column position in a table view.

```
typedef UInt32 DataBrowserTableViewColumnIndex;
```

Discussion
This data type is typically used for table-view formatting. Column indices are zero based.

Availability
Available in Mac OS X v10.0 and later.

Declared In
HIDataBrowser.h

DataBrowserTableViewColumnID

Defines a data type for a value that identifies a column independent of its position in a data browser.

```
typedef DataBrowserPropertyID DataBrowserTableViewColumnID;
```

Discussion
For details on using this data type, see DataBrowserPropertyID (page 172).

Availability
Available in Mac OS X v10.0 and later.

Declared In
HIDataBrowser.h

DataBrowserTableViewColumnDesc

Defines a data type for a table view column description.

```
typedef DataBrowserPropertyDesc DataBrowserTableViewColumnDesc;
```

Discussion
See the DataBrowserPropertyDesc (page 167) data structure for more information.

Availability
Available in Mac OS X v10.0 and later.

Declared In
HIDataBrowser.h

DataBrowserListViewHeaderDesc

Describes the appearance of a column title in list view.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
struct DataBrowserListViewHeaderDesc {
    UInt32 version;
    UInt16 minimumWidth;
    UInt16 maximumWidth;
    SInt16 titleOffset;
    CFStringRef titleString;
    DataBrowserSortOrder initialOrder;
    ControlFontStyleRec btnFontStyle;
    ControlButtonContentInfo btnContentInfo;
};
typedef struct DataBrowserListViewHeaderDesc DataBrowserListViewHeaderDesc;
```

Fields

`version`
: The format of the structure. Set this field to the value `kDataBrowserListViewLatestHeaderDesc`.

`minimumWidth`
: For resizable columns, the smallest width to which the column can be resized. If the column is not resizable, set this to the same value as the `maximumWidth` field.

`maximumWidth`
: For resizable columns, the largest width to which the column can be resized. If the column is not resizable, set this to the same value as the `minimumWidth` field.

`titleOffset`
: An offset, in pixels, from the left side of the column that specifies where the title text will be drawn. The title alignment (set in the `just` field of the `btnFontStyle` parameter) and the `titleOffset` values dictate the alignment and offset (inset by default) of the content of the column when displaying one of he predefined display types. Typically the title offset is set to `0`.

`titleString`
: The text to use for the column title. Set the string to `NULL` if you do not want to display a title.

`initialOrder`
: The initial sorting order to use for the column when the column is the current sort column. After the data browser is visible, the user can change the sorting order. You can assign one of the following values:

- `kDataBrowserOrderIncreasing` means this column sorts in ascending order.
- `kDataBrowserOrderDecreasing` means this column sorts in descending order.

`btnFontStyle`
: A structure that describes the contents of the column heading and how to draw them. This allows you to customize the font that the column title is drawn with, which is independent of the font used to draw the data in the column.

`btnContentInfo`
: A structure that defines the icon, if any, to use for the column heading.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`HIDataBrowser.h`

DataBrowserListViewColumnDesc

Contains property information for a list view column and specifies display information for the column title.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
struct DataBrowserListViewColumnDesc {
    DataBrowserTableViewColumnDesc propertyDesc;
    DataBrowserListViewHeaderDesc headerBtnDesc;
};
typedef struct DataBrowserListViewColumnDesc DataBrowserListViewColumnDesc;
```

Fields

`propertyDesc`

A structure that contains property and display information for a column. See `DataBrowserTableViewColumnDesc` (page 173) for more information.

`headerBtnDesc`

A structure that contains display information for the column title. See `DataBrowserListViewHeaderDesc` (page 173) for more information.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`HIDataBrowser.h`

kHIDataBrowserClassID

Defines the HIObject class ID for the HIDataBrowser class.

```
#define kHIDataBrowserClassID CFSTR("com.apple.HIDataBrowser");
```

Availability

Available in Mac OS X v10.4 and later.

Constants

Callback Data Structure Version

Specifies the version of the latest standard callback data structure.

```
enum {
    kDataBrowserLatestCallbacks = 0
};
```

Constants

`kDataBrowserLatestCallbacks`

A convenience constant used in the `DataBrowserCallbacks` (page 168) data structure to specify the version of the structure.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Control Data Tags

Define data-browser-specific tags for use with the Control Manager functions `GetControlData` and `SetControlData`.

```
enum {
  kControlDataBrowserIncludesFrameAndFocusTag = 'brdr',
  kControlDataBrowserKeyFilterTag = kControlEditTextKeyFilterTag,
  kControlDataBrowserEditTextKeyFilterTag =
                kControlDataBrowserKeyFilterTag,
  kControlDataBrowserEditTextValidationProcTag =
                kControlEditTextValidationProcTag
};
```

Constants

`kControlDataBrowserIncludesFrameAndFocusTag`
: Include the frame and user focus. The associated data is of type `Boolean`.

  Available in Mac OS X v10.0 and later.

  Declared in `HIDataBrowser.h`.

`kControlDataBrowserKeyFilterTag`
: Use a filter. The associated data is a universal procedure pointer to a `ControlKeyFilterProcPtr` callback. This callback is invoked when the user edits an item.

  Available in Mac OS X v10.0 and later.

  Declared in `HIDataBrowser.h`.

`kControlDataBrowserEditTextKeyFilterTag`
: Use a text filter. This is a duplicate of `kControlDataBrowserKeyFilterTag`.

  Available in Mac OS X v10.0 and later.

  Declared in `HIDataBrowser.h`.

`kControlDataBrowserEditTextValidationProcTag`
: Use a callback to validate the text. The associated data is a universal procedure pointer to a `ControlEditTextValidataionProcPtr` callback. This callback is invoked when the user finishes editing an item.

  Available in Mac OS X v10.0 and later.

  Declared in `HIDataBrowser.h`.

Discussion

You can use these options with the Control Manger functions `GetControlData` and `SetControlData`.

Custom Callback Data Structure Version

Specifies the version of the latest custom callback data structure.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
enum {
    kDataBrowserLatestCustomCallbacks = 0
};
```

Constants

`kDataBrowserLatestCustomCallbacks`
A convenience constant used in the `DataBrowserCustomCallbacks` (page 169) data structure to specify the version of the structure.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Data Browser Attributes

Specifies data browser attribute constants.

```
enum {
kDataBrowserAttributeNone = 0,
kDataBrowserAttributeColumnViewResizeWindow = (1 << 0),
kDataBrowserAttributeListViewAlternatingRowColors = (1 <<
1)
kDataBrowserAttributeListViewDrawColumnDividers = (1 << 2)
};
```

Constants

`kDataBrowserAttributeNone`
The data browser has no attributes.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserAttributeColumnViewResizeWindow`
In column view, this data browser is allowed to resize the owning window whenever necessary. This includes, but is not necessarily limited to, situations where column resize operations need more visible space in the window. If you turn this attribute on, your window must tolerate being resized behind your application's back. If your window needs to react to bounds changes, use a `kEventWindowBoundsChanged` event handler. If you need to constrain your window's minimum and maximum bounds, use the `kEventWindowGetMinimumSize` and `kEventWindowGetMaximumSize` handlers, the `SetWindowResizeLimits` function, or something similar.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserAttributeListViewAlternatingRowColors`
In list view, this data browser should draw alternating row background colors. However, note that this attribute does not work with variable row heights as of Mac OS X v10.4.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserAttributeListViewDrawColumnDividers`
In list view, this data browser should draw a vertical line between the columns.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
Use these constants in conjunction with `DataBrowserGetAttributes` (page 31) and `DataBrowserChangeAttributes` (page 31).

Availability
Available in Mac OS X v10.4 and later.

Data Browser Control Kind Tag

Specifies the control type is a data browser.

```
enum {
kControlKindDataBrowser       = 'datb'
};
```

Constants
`kControlKindDataBrowser`
The data browser control type.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
When you call the function `GetControlKind` with a control reference that represents a data browser, you obtain the data browser tag `kControlKindDataBrowser` as the control type.

Data Browser Metric Values

Specifies constants used by `DataBrowserSetMetric`.

```
enum {
kDataBrowserMetricCellContentInset = 1,
kDataBrowserMetricIconAndTextGap = 2,
kDataBrowserMetricDisclosureColumnEdgeInset = 3,
kDataBrowserMetricDisclosureTriangleAndContentGap = 4,
kDataBrowserMetricDisclosureColumnPerDepthGap = 5,
kDataBrowserMetricLast = kDataBrowserMetricDisclosureColumnPerDepthGap
};
typedef UInt32 DataBrowserMetric;
```

Constants
`kDataBrowserMetricCellContentInset`
The content (icon, text, etc.) within a cell is drawn a certain amount in from the left and right edges of the cell. This metric governs the amount of inset.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserMetricIconAndTextGap`
This metric controls the space between the icon and text within a column of type `kDataBrowserIconAndTextType`.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserMetricDisclosureColumnEdgeInset`
In list view only, this metric is used instead of (not in addition to) `DataBrowserMetricCellContentInset` for the side of the cell in the disclosure column that displays the disclosure triangle.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserMetricDisclosureTriangleAndContentGap`
In list view only, this metric controls the amount of space between the disclosure triangle and the cell's content.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserMetricDisclosureColumnPerDepthGap`
In list view only, this metric controls the amount of space in the disclosure column for each level of indentation in progressively deeper hierarchies of disclosed items.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserMetricLast`
Same as `kDataBrowserMetricDisclosureColumnPerDepthGap`.

Available in Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

Display Types

Specify a data type or control to display in a column.

```
typedef OSType DataBrowserPropertyType;
enum {
    kDataBrowserCustomType = 0x3F3F3F3F,
    kDataBrowserIconType = 'icnr',
    kDataBrowserTextType = 'text',
    kDataBrowserDateTimeType = 'date',
    kDataBrowserSliderType = 'sldr',
    kDataBrowserCheckboxType = 'chbx',
    kDataBrowserProgressBarType = 'prog',
    kDataBrowserRelevanceRankType = 'rank',
    kDataBrowserPopupMenuType = 'menu',
    kDataBrowserIconAndTextType = 'ticn'
};
```

Constants

`kDataBrowserCustomType`
Displays custom data defined by your application. You must install callbacks to handle items of this type. Use custom types with caution. In some cases custom types do not display properly or exhibit the appropriate behavior.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserIconType`
Displays icons. The associated data for the icon can be of type `IconRef`, `IconTransformType`, and `RGBColor`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserTextType`
Displays text. The associated data is a CFString object.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserDateTimeType`
Displays date and time information. The associated data can be of type `DateTime` or `LongDateTime`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserSliderType`
Displays slider controls. The associated data are values that define the minimum and maximum values and the current value for the slider. Avoid using slider controls in a data browser because, in some cases, they do not display properly onscreen.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserCheckboxType`
Displays checkbox controls. The associated data is the current value of the checkbox and a `ThemeButtonValue` value. Avoid using checkbox controls in a data browser because, in some cases, they do not display properly onscreen.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserProgressBarType`
Displays progress bar controls. The associated data are values that define the minimum and maximum values and the current setting for the progress bar.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserRelevanceRankType`
Displays relevance indicators. The associated data are values that define the minimum and maximum values and the current setting for the relevance indicator. Avoid using relevance indicators in a data browser because, in some cases, they do not display properly onscreen.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPopupMenuType`
Displays pop-up menus. The associated data is a `MenuRef` data type, a menu item ID, and the value of the pop-up menu, which indicates the item of the menu to draw in the pop-up menu. Avoid using pop-up menu controls in a data browser because, in some cases, they do not display properly onscreen.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserIconAndTextType`
Displays icon and text data. The associated data can be of any data type used for icons or text, such as an `IconRef` data type and a CFString object.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
You can use these constants to define what is displayed in a column. Each constant defines a presentation style and implies a specific set of primitive types or data structures.

Editing Commands

Specifies editing actions to apply to a data browser item.

```
typedef UInt32 DataBrowserEditCommand;
enum {
    kDataBrowserEditMsgUndo = kHICommandUndo,
    kDataBrowserEditMsgRedo = kHICommandRedo,
    kDataBrowserEditMsgCut = kHICommandCut,
    kDataBrowserEditMsgCopy = kHICommandCopy,
    kDataBrowserEditMsgPaste = kHICommandPaste,
    kDataBrowserEditMsgClear = kHICommandClear,
    kDataBrowserEditMsgSelectAll = kHICommandSelectAll
};
```

Constants
`kDataBrowserEditMsgUndo`
Undo the last editing operation.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserEditMsgRedo`
Redo the last editing operation that was undone.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserEditMsgCut`
Cut the contents of the selection to the Clipboard.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserEditMsgCopy`
Copy the contents of the selection o the Clipboard.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserEditMsgPaste`
Replace the contents of the selection with the contents of the Clipboard.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserEditMsgClear`
Remove the contents of the selection.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserEditMsgSelectAll`
Select all of the text inside of the current edit session.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
These options are for use with the functions `EnableDataBrowserEditCommand` (page 41) and `ExecuteDataBrowserEditCommand` (page 42).

Item Notifications

Specify notifications provided by the data browser to your application.

```
typedef UInt32 DataBrowserItemNotification;
enum {
    kDataBrowserItemAdded = 1,
    kDataBrowserItemRemoved = 2,
    kDataBrowserEditStarted = 3,
    kDataBrowserEditStopped = 4,
    kDataBrowserItemSelected = 5,
    kDataBrowserItemDeselected = 6,
    kDataBrowserItemDoubleClicked = 7,
    kDataBrowserContainerOpened = 8,
    kDataBrowserContainerClosing = 9,
    kDataBrowserContainerClosed = 10,
    kDataBrowserContainerSorting = 11,
    kDataBrowserContainerSorted = 12,
    kDataBrowserUserStateChanged = 13,
    kDataBrowserSelectionSetChanged = 14,
    kDataBrowserTargetChanged = 15,
    kDataBrowserUserToggledContainer = 16
};
```

Constants
`kDataBrowserItemAdded`
The specified item has been added to a container in the data browser.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemRemoved`
The specified item has been removed from a container in the data browser.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserEditStarted`
An text editing session has started for the specified item.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserEditStopped`
An text editing session has stopped for the specified item.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemSelected`
An item has been added to the selection set.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemDeselected`
An item has been removed from the selection set.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemDoubleClicked`
The user double-clicked an item.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerOpened`
A container has been opened.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerClosing`
A container is about to close. This notification is sent at the start of the close operation. The container still contains its child items and it is still valid to pass them to data browser functions. The data browser handles closing automatically, so typically applications do not look for this notification. You'd use this only if you are interested in fetching information on the items before the close actually happens.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerClosed`
A container has been closed. This notification is sent after the close operation, so it is no longer valid to pass child items to data browser functions.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerSorting`
A container is about to be sorted.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerSorted`
A container has been sorted.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserUserStateChanged`
The user has reformatted the view for the target. For example, the user changed a sorting order or a column width.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserSelectionSetChanged`
The selection set has been modified.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserTargetChanged`
The target has changed to the specified item.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserUserToggledContainer`
The user has toggled opened or closed a container by clicking a disclosure triangle.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
These notifications are used with the callbacks `DataBrowserItemNotificationProcPtr` (page 154) and `DataBrowserItemNotificationWithItemProcPtr` (page 155). Notifications sent for containers are the same regardless of whether the container opens to an expandable row or to individual rows for each item in the container.

Item States

Indicate the current state of an item in the data browser.

```
typedef UInt32 DataBrowserItemState;
enum {
    kDataBrowserItemNoState = 0,
    kDataBrowserItemAnyState = (unsigned long) (-1),
    kDataBrowserItemIsSelected = 1 << 0,
    kDataBrowserContainerIsOpen = 1 << 1,
    kDataBrowserItemIsDragTarget = 1 << 2
};
```

Constants
`kDataBrowserItemNoState`
The state is undefined.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemAnyState`
Any state is acceptable.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserItemIsSelected`
The item is selected.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerIsOpen`
The item container is open. This state applies to:

- A parent item in a hierarchical list in list view
- An item in column view if the item's contents are displayed in the next column

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemIsDragTarget`
The item is a drag target.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

List View Header Description Version

Defines the version of the list view header description data structure.

```
enum {
    kDataBrowserListViewLatestHeaderDesc = 0
};
```

Constants

`kDataBrowserListViewLatestHeaderDesc`
A convenience constant that specifies the version of the list view header description data structure.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

List View Append Column

Defines the last column as the column to use for an append operation.

```
enum {
    kDataBrowserListViewAppendColumn = kDataBrowserTableViewLastColumn
};
```

Constants

`kDataBrowserListViewAppendColumn`
The column to use for an append operation.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

No Item Constant

Specifies that there is no item to provide or obtain.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
enum {
    kDataBrowserNoItem = 0L
};
```

Constants

`kDataBrowserNoItem`
A convenience constant used when there is no item to provide or obtain. This value is of type `DataBrowserItemID`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Properties

Specify attributes for items, containers, and columns.

```
enum {
    kDataBrowserItemNoProperty = 0L,
    kDataBrowserItemIsActiveProperty = 1L,
    kDataBrowserItemIsSelectableProperty = 2L,
    kDataBrowserItemIsEditableProperty = 3L,
    kDataBrowserItemIsContainerProperty = 4L,
    kDataBrowserContainerIsOpenableProperty = 5L,
    kDataBrowserContainerIsClosableProperty = 6L,
    kDataBrowserContainerIsSortableProperty = 7L,
    kDataBrowserItemSelfIdentityProperty = 8L,
    kDataBrowserContainerAliasIDProperty = 9L,
    kDataBrowserColumnViewPreviewProperty = 10L,
    kDataBrowserItemParentContainerProperty = 11L
};
```

Constants

`kDataBrowserItemNoProperty`
No property; no associated data.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemIsActiveProperty`
The active state of the item. The associated data is of type `Boolean`. The default value `true` indicates the item is active.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemIsSelectableProperty`
The selection capability of the item. The associated data is of type `Boolean`. The default value `true` indicates the item can be selected.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemIsEditableProperty`
The editing capability of the item. The associated data is of type `Boolean`. The default value `false` indicates the item cannot be edited.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserItemIsContainerProperty`
The container attribute for an item. The associated data is of type `Boolean`. The default value `false` indicates the item cannot contain other items.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerIsOpenableProperty`
The opening capability of a container item. The associated data is of type `Boolean`. The default value `true` indicates the container can be opened.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerIsClosableProperty`
The closing capability of a container item. The associated data is of type `Boolean`. The default value `true` indicates the container can be closed.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerIsSortableProperty`
The sorting capability of container item. The associated data is of type `Boolean`. The default value `true` indicates the items in the container can be sorted.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemSelfIdentityProperty`
This property is used only in column view for the item-data and item-compare callbacks. You should not use this property for anything else. It is passed to your item-data callback when the data browser needs to know the data to draw to represent the item. It is your responsibility to provide the data to display for that item in whatever format would be appropriate for the column view display type, which is `kDataBrowserIconAndTextType`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserContainerAliasIDProperty`
An alias or symbolic link from an item to a container item. The associated data is of type `DataBrowserItemID`. This property is sent to your item-data callback to provide your application with a chance to follow an alias that the item might represent. If the incoming item is an alias to another item, you can call the function `SetDataBrowserItemDataItemID` to inform the data browser which other item the incoming item points to.

This property is sent only from column view. Your support for it is optional. Your response allows the data browser to be a bit more memory efficient with its internal storage. If a given item is an alias to an item whose contents are already displayed in one column of the column view, the contents can be shared between those two columns.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserColumnViewPreviewProperty`

The column displays a preview. There is no associated data. Available only in column view. This property is sent to your draw-item callback to indicate the need for you to draw a preview of the given item. It can be sent to other callbacks to provide an opportunity for your application to draw or track in the preview column.

You can also pass this in the `propertyID` parameter of the function `RevealDataBrowserItem` (along with the appropriate item ID of the item whose preview is displayed) to make sure the preview column is visible to the user.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemParentContainerProperty`

Designates the parent of the specified item. The associated data is of type `DataBrowserItemID`. This is sent to your item-data callback when the data browser needs to know the parent container item for a given item.

In column view, this allows the function `SetDataBrowserTarget` (page 133) to process the data properly. The target is the leaf node item whose contents you want to display, which is the rightmost column in column view. However, unlike `SetDataBrowserColumnViewPath`, the function `SetDataBrowserTarget` doesn't offer a way for you to communicate the item IDs of the rest of the column containers, so `SetDataBrowserTarget` asks for them explicitly by requesting the item's parent, then the parent of the item's parent, and so on. (Your item-data callback might be called with the parent container property at times other than an explicit call to `SetDataBrowserTarget`, so your item-data callback should support this property.)

In list view, this property allows you to pass a non-container to `SetDataBrowserTarget`. In this case, the data browser requests the parent of the target so it knows which container to display the contents of in the list view. (Again, your item-data callback might be called with the parent container property at times other than an explicit call to `SetDataBrowserTarget`, so your item-data callback should be sure to support this property.)

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion

You can use these constants along with the appropriate value to manage the attributes of items in a data browser. Typically your item-data callback responds to inquires about these properties by calling the appropriate accessor function from within the callback. See `DataBrowserItemDataProcPtr` (page 149) for more information on the item-data callback. "Getting and Setting Item Data" (page 19) describes the accessor functions.

Property Flags: Universal

Modify the appearance or behavior of display properties.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
enum {
    kDataBrowserUniversalPropertyFlagsMask = 0xFF,
    kDataBrowserPropertyIsMutable = 1 << 0,
    kDataBrowserDefaultPropertyFlags = 0 << 0,
    kDataBrowserUniversalPropertyFlags =  kDataBrowserUniversalPropertyFlagsMask,
    kDataBrowserPropertyIsEditable = kDataBrowserPropertyIsMutable
};
```

Constants

`kDataBrowserUniversalPropertyFlagsMask`

Test for universal property flags. This constant is used by the data browser; your application doesn't need to use it.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyIsMutable`

The property is mutable. You can assign this flag to the `propertyFlags` field of the `DataBrowserPropertyDesc` structure. You must set this flag is you want to allow editing of the text part of the property.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserDefaultPropertyFlags`

The default properties. You can assign this flag to the `propertyFlags` field of the `DataBrowserPropertyDesc` structure if all you need is the default behavior. The default is for all flags to be off.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserUniversalPropertyFlags`

Universal property flags. This constant is used by the data browser; your application doesn't need to use it.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyIsEditable`

The data can be edited. You must set this flag if you want to allow editing of the text part of the property. This flag can be set if the values displayed in the column can be changed. If your application specifies this flag, then it must also provide a callback that allows the data browser to retrieve and store data values displayed in this column. You can assign this flag to the `propertyFlags` field of the `DataBrowserPropertyDesc` structure.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion

These constants reside in bits 0–7 of the `DataBrowserPropertyFlags` data type.

Property Flags: Modifiers

Specify how to display the data associated with a display type.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
enum {
    kDataBrowserPropertyFlagsOffset = 8,
    kDataBrowserPropertyFlagsMask =
                0xFF << kDataBrowserPropertyFlagsOffset,
    kDataBrowserCheckboxTriState  =
                1 << kDataBrowserPropertyFlagsOffset,
    kDataBrowserDateTimeRelative  =
                1 << (kDataBrowserPropertyFlagsOffset),
    kDataBrowserDateTimeDateOnly  =
                1 << (kDataBrowserPropertyFlagsOffset + 1),
    kDataBrowserDateTimeTimeOnly  =
                1 << (kDataBrowserPropertyFlagsOffset + 2),
    kDataBrowserDateTimeSecondsToo =
                1 << (kDataBrowserPropertyFlagsOffset + 3),
    kDataBrowserSliderPlainThumb  =
            kThemeThumbPlain << kDataBrowserPropertyFlagsOffset,
    kDataBrowserSliderUpwardThumb =
            kThemeThumbUpward << kDataBrowserPropertyFlagsOffset,
    kDataBrowserSliderDownwardThumb =
            kThemeThumbDownward << kDataBrowserPropertyFlagsOffset,
    kDataBrowserDoNotTruncateText =
                3 << kDataBrowserPropertyFlagsOffset,
    kDataBrowserTruncateTextAtEnd =
                2 << kDataBrowserPropertyFlagsOffset,
    kDataBrowserTruncateTextMiddle =
                0 << kDataBrowserPropertyFlagsOffset,
    kDataBrowserTruncateTextAtStart =
                1 << kDataBrowserPropertyFlagsOffset,
    kDataBrowserPopupMenuButtonless =
                1 << kDataBrowserPropertyFlagsOffset,
    kDataBrowserPropertyModificationFlags =
                 kDataBrowserPropertyFlagsMask,
    kDataBrowserRelativeDateTime  = kDataBrowserDateTimeRelative
};
```

Constants

`kDataBrowserPropertyFlagsOffset`

The offset value for this set of property flags.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyFlagsMask`

Use to set or test for property modifier flags.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserCheckboxTriState`

Modifies the `kDataBrowserCheckboxType` display type to display a checkbox that can have on, off, and mixed-mode states instead of just on and off states.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserDateTimeRelative`

Modifies the `kDataBrowserDateTimeType` display type to display relative date and time.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserDateTimeDateOnly`
Modifies the `kDataBrowserDateTimeType` display type to display only the date.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserDateTimeTimeOnly`
Modifies the `kDataBrowserDateTimeType` display type to display only the time.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserDateTimeSecondsToo`
Modifies the `kDataBrowserDateTimeType` display type to display the time with seconds.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserSliderPlainThumb`
Modifies the `kDataBrowserSliderType` display type to display a round thumb.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserSliderUpwardThumb`
Modifies the `kDataBrowserSliderType` display type to display a directional thumb that points up.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserSliderDownwardThumb`
Modifies the `kDataBrowserSliderType` display type to display a directional thumb that points down.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserDoNotTruncateText`
Modifies the `kDataBrowserTextType` and the `kDataBrowserIconAndTextType` display types so they do not truncate text.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserTruncateTextAtEnd`
Modifies the `kDataBrowserTextType` and the `kDataBrowserIconAndTextType` display types so they truncate text, if needed, at the end of the text string.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserTruncateTextMiddle`
Modifies the `kDataBrowserTextType` and the `kDataBrowserIconAndTextType` display types so they truncate text, if needed, in the middle of the text string.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserTruncateTextAtStart`
Modifies the `kDataBrowserTextType` and the `kDataBrowserIconAndTextType` display type so they truncate text, if needed, at the beginning of the text string.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPopupMenuButtonless`
This flag is only for use with columns of type `kDataBrowserPopupMenuType` and indicates that the popup is to be drawn in a sleek buttonless fashion. The text is drawn next to a popup glyph, and the whole cell is clickable.

Available on Mac OS X v10.4 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyModificationFlags`
Old name. Instead use `kDataBrowserPropertyFlagsMask`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserRelativeDateTime`
Old name. Instead use `kDataBrowserDateTimeRelative`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
These constants reside in bits 8–15 of the `DataBrowserPropertyFlags` data type.

Property Flags: Offset and Mask for List View Properties

Specify an offset and mask for bits 16–23 of the `DataBrowserPropertyFlag` data type.

```
enum {
    kDataBrowserViewSpecificFlagsOffset = 16,
    kDataBrowserViewSpecificFlagsMask =
            0xFF << kDataBrowserViewSpecificFlagsOffset,
    kDataBrowserViewSpecificPropertyFlags =
            kDataBrowserViewSpecificFlagsMask
};
```

Constants
`kDataBrowserViewSpecificFlagsOffset`
The offset value for this set of property flags.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserViewSpecificFlagsMask`
Use to set or test for view-specific property flags.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserViewSpecificPropertyFlags`
Old name. Instead use `kDataBrowserViewSpecificFlagsMask`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
See also "Property Flags: List View Column Behavior" (page 193).

Property Flags: List View Column Behavior

Specify behaviors for columns in list view.

```
typedef DataBrowserPropertyFlags DataBrowserListViewPropertyFlags;
enum {
    kDataBrowserListViewSelectionColumn =
                kDataBrowserTableViewSelectionColumn,
    kDataBrowserListViewMovableColumn =
                1 << (kDataBrowserViewSpecificFlagsOffset  + 1),
    kDataBrowserListViewSortableColumn =
                1 << (kDataBrowserViewSpecificFlagsOffset  + 2),
    kDataBrowserListViewTypeSelectColumn =
                1 << (kDataBrowserViewSpecificFlagsOffset  + 3),
    kDataBrowserListViewNoGapForIconInHeaderButton =
                1 << (kDataBrowserViewSpecificFlagsOffset  + 4),
    kDataBrowserListViewDefaultColumnFlags =
              kDataBrowserListViewMovableColumn +
              kDataBrowserListViewSortableColumn
};
```

Constants
`kDataBrowserListViewSelectionColumn`
If you are using a minimally highlighted list, this indicates to draw the contents of this column as highlighted when the item is selected. (Minimal highlighting is the highlighting used by the Finder for list view prior to Mac OS X version 10.3.)

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserListViewMovableColumn`
The column is movable.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserListViewSortableColumn`
The column can be sorted.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserListViewTypeSelectColumn`
: The column is capable of being selected and having text entered. If one or more of your list view columns are marked as type-selectable, the data browser handles type-selection for you automatically. You can use this flag with columns whose display type is `kDataBrowserTextType`, `kDataBrowserIconAndTextType`, or `kDataBrowserDateTimeType`. If you set this flag for a column of another type, the type-select behavior is undefined. Turning on this flag causes the data browser to obtain keyboard input using a Carbon event handler instead of relying on calls to the function `HandleControlKey`.

  Declared in `HIDataBrowser.h`.

  Available in Mac OS X 10.3 and later.

`kDataBrowserListViewNoGapForIconInHeaderButton`
: Normally the text in a header button for a column of type `kDataBrowserIconAndTextType` is aligned as though it has an icon next to it even if no icon is specified for the header button. This flag indicates that space should not be reserved for an icon if no icon is provided for the header button. This flag allows a client to justify the left edge of the text in a header button to the left edge of the icon in the cells beneath it.

  Declared in `HIDataBrowser.h`.

  Available in Mac OS X v10.4 and later.

`kDataBrowserListViewDefaultColumnFlags`
: The default properties. You can assign this flag to the `propertyFlags` field of the `DataBrowserPropertyDesc` structure if all you need is the default behavior for list view.

  Available in Mac OS X v10.0 and later.

  Declared in `HIDataBrowser.h`.

Discussion
These constants reside in bits 16–23 of the `DataBrowserPropertyFlags` data type and are specific to a list view.

Property Flags: Offset and Mask for Client-Defined Properties

Specify an offset and mask for the high 8 bits of the `DataBrowserPropertyFlag` data type.

```
enum {
    kDataBrowserClientPropertyFlagsOffset = 24,
    kDataBrowserClientPropertyFlagsMask = (unsigned long)
            (0xFF << kDataBrowserClientPropertyFlagsOffset)
};
```

Constants
`kDataBrowserClientPropertyFlagsOffset`
: The offset value for this set of property flags.

  Available in Mac OS X v10.0 and later.

  Declared in `HIDataBrowser.h`.

`kDataBrowserClientPropertyFlagsMask`
: The mask value for this set of property flags.

  Available in Mac OS X v10.0 and later.

  Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Discussion
Bits 24–31 of the `DataBrowserPropertyFlags` data type are reserved for use by your application. You can set and query them for your own purposes.

Property Parts

Specify the visual components of a data type or control displayed in a column.

```
typedef OSType DataBrowserPropertyPart;
enum {
    kDataBrowserPropertyEnclosingPart = 0,
    kDataBrowserPropertyContentPart = '----',
    kDataBrowserPropertyDisclosurePart = 'disc',
    kDataBrowserPropertyTextPart = kDataBrowserTextType,
    kDataBrowserPropertyIconPart = kDataBrowserIconType,
    kDataBrowserPropertySliderPart = kDataBrowserSliderType,
    kDataBrowserPropertyCheckboxPart = kDataBrowserCheckboxType,
    kDataBrowserPropertyProgressBarPart = kDataBrowserProgressBarType,
    kDataBrowserPropertyRelevanceRankPart =
                kDataBrowserRelevanceRankType
};
```

Constants

`kDataBrowserPropertyEnclosingPart`
The outer boundary of an item.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyContentPart`
The content of an item.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyDisclosurePart`
The location of a disclosure rectangle.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyTextPart`
The location where text is drawn.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyIconPart`
The location where an icon is displayed.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserPropertySliderPart`
The location of a slider.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserPropertyCheckboxPart`
The location of a checkbox.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyProgressBarPart`
The location of a progress bar.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserPropertyRelevanceRankPart`
The location of a relevance indicator.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
You can pass these constants as a parameter to the function `GetDataBrowserItemPartBounds` (page 60) to obtain a rectangle that contains the bounds for the specified part.

Reveal Options

Specify how to position an item in a data browser.

```
typedef UInt8 DataBrowserRevealOptions;
enum {
    kDataBrowserRevealOnly = 0,
    kDataBrowserRevealAndCenterInView = 1 << 0,
    kDataBrowserRevealWithoutSelecting = 1 << 1
};
```

Constants
`kDataBrowserRevealOnly`
Move the content of the data browser as little as possible to make the item visible, and show the item in a selected state.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserRevealAndCenterInView`
Reveal the item so that, if possible, the item is centered in the data browser.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserRevealWithoutSelecting`
Reveal the item but do not select it.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
You can pass reveal options as a parameter to the function `RevealDataBrowserItem` (page 103). You can pass more than one at a time for combined functionality.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Selection Anchor Directions

Specify an anchor direction to apply when the user drags a data selection.

```
typedef UInt32 DataBrowserSelectionAnchorDirection;
enum {
    kDataBrowserSelectionAnchorUp = 0,
    kDataBrowserSelectionAnchorDown = 1,
    kDataBrowserSelectionAnchorLeft = 2,
    kDataBrowserSelectionAnchorRight = 3
};
```

Constants

`kDataBrowserSelectionAnchorUp`
Apply the anchor direction at the top of the selection set.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserSelectionAnchorDown`
Apply the anchor direction at the bottom of the selection set.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserSelectionAnchorLeft`
Apply the anchor direction at the left of the selection set.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserSelectionAnchorRight`
Apply the anchor direction at the right of the selection set.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

Discussion

These options are for use with the function `MoveDataBrowserSelectionAnchor` (page 90).

Selection State Options

Specify how the selection state should be affected by a given list of items.

```
typedef UInt32 DataBrowserSetOption;
enum {
    kDataBrowserItemsAdd = 0,
    kDataBrowserItemsAssign = 1,
    kDataBrowserItemsToggle = 2,
    kDataBrowserItemsRemove = 3
};
```

Constants

`kDataBrowserItemsAdd`
Add specified items to the existing set or selection.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

`kDataBrowserItemsAssign`
Assign the destination set to the specified item and redraw the list appropriately.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemsToggle`
Toggle the membership state of the specified items. Any of the items in the current selection are removed from the selection, and those items that are not in the selection are added to the selection.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserItemsRemove`
Remove specified items from the existing set and redraw the items so they are not highlighted.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion
These options are for use with the function `SetDataBrowserSelectedItems` (page 126).

Sorting Orders

Specify the order in which to sort data.

```
typedef UInt16 DataBrowserSortOrder;
enum {
    kDataBrowserOrderUndefined = 0,
    kDataBrowserOrderIncreasing = 1,
    kDataBrowserOrderDecreasing = 2
};
```

Constants
`kDataBrowserOrderUndefined`
This constant has no meaning in the context of your application; don't use it.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserOrderIncreasing`
Sort in increasing order.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserOrderDecreasing`
Sort in decreasing order.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Table View Highlighting Styles

Specify a highlighting style to use in list view.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
typedef UInt32 DataBrowserTableViewHiliteStyle;
enum {
    kDataBrowserTableViewMinimalHilite = 0,
    kDataBrowserTableViewFillHilite = 1
};
```

Constants

`kDataBrowserTableViewMinimalHilite`

Use minimal highlighting. This is the highlighting used by the Finder for list view prior to Mac OS X v. 10.3. For this style, the highlight color for active items is `kThemeBrushPrimaryHighlightColor` and for inactive items the color is `kThemeBrushSecondaryHighlightColor`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserTableViewFillHilite`

Use full-row highlighting. This is the highlighting used by the Finder for list view in Mac OS X v. 10.3. For this style, the highlight color in Mac OS X v. 10.3 for active items is `kThemeBrushAlternatePrimaryHighlightColor` and for inactive items the color is `kThemeBrushSecondaryHighlightColor`. Prior to Mac OS X v. 10.3 the highlight color for active and inactive items is `kThemeBrushPrimaryHighlightColor`.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Table View Last Column Value

Specifies the last column position.

```
enum {
    kDataBrowserTableViewLastColumn = -1
};
```

Constants

`kDataBrowserTableViewLastColumn`

The last column position.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Table View Property Flag

Specifies a table view property.

```
typedef UInt32 DataBrowserTableViewPropertyFlags;
enum {
   kDataBrowserTableViewSelectionColumn = 1 <<  kDataBrowserViewSpecificFlagsOffset
};
```

Constants

`kDataBrowserTableViewSelectionColumn`

The column can be selected.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Tracking Results

Specify the outcome of tracking a drag operation.

```
typedef SInt16 DataBrowserTrackingResult;
enum {
    kDataBrowserContentHit = 1,
    kDataBrowserNothingHit = 0,
    kDataBrowserStopTracking = -1
};
```

Constants

`kDataBrowserContentHit`

Indicates that a selectable portion of the item was hit. The data browser will select the item and do other relevant actions as appropriate.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserNothingHit`

Indicates that a nonselectable portion—whitespace—was hit. The data browser won't select the item, but it could, for example, start a selection rectangle.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserStopTracking`

Indicates the callback handled the click completely. The data browser will not attempt to display a contextual menu, start a drag, process a double-click, or draw a selection rectangle. The callback is responsible for all facets of click handling if it returns this value.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion

These constants are returned from the custom tracking callback, `DataBrowserTrackingProcPtr` (page 163).

User Selection Flags

Specify allowable selection behavior.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

```
typedef UInt32 DataBrowserSelectionFlags;
enum {
    kDataBrowserDragSelect = 1 << 0,
    kDataBrowserSelectOnlyOne = 1 << 1,
    kDataBrowserResetSelection = 1 << 2,
    kDataBrowserCmdTogglesSelection = 1 << 3,
    kDataBrowserNoDisjointSelection = 1 << 4,
    kDataBrowserAlwaysExtendSelection = 1 << 5,
    kDataBrowserNeverEmptySelectionSet = 1 << 6
};
```

Constants

`kDataBrowserDragSelect`

Allows items to be selected by dragging. If the user to clicks the mouse to select a nondraggable item and drags the mouse, any item the pointer moves over is selected. This is on by default.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserSelectOnlyOne`

Only one item can be selected at a time. If you use this flag in conjunction with the flag `kDataBrowserDragSelect`, then as the user drags, previous items are deselected as each new item is dragged over.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserResetSelection`

Clears all selected items before processing the next selection. This is off by default.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserCmdTogglesSelection`

Enables use of a command to toggle items in and out of a selection. This is on by default.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserNoDisjointSelection`

Prevents a discontinuous selection.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserAlwaysExtendSelection`

Multiple items can be selected without holding down a modifier key.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

`kDataBrowserNeverEmptySelectionSet`

There must always be at least one selected item; the user cannot deselect the last selected item.

Available in Mac OS X v10.0 and later.

Declared in `HIDataBrowser.h`.

Discussion

You can use one or more of these flags together.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

View Styles

Specify a style to use when displaying data in a data browser.

```
typedef OSType DataBrowserViewStyle;
enum {
    kDataBrowserNoView = '????',
    kDataBrowserListView = 'lstv',
    kDataBrowserColumnView = 'clmv'
};
```

Constants

`kDataBrowserNoView`
There is no view.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserListView`
Display data in a list. Lists can be hierarchical.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

`kDataBrowserColumnView`
Display data in columns that can be browsed.
Available in Mac OS X v10.0 and later.
Declared in `HIDataBrowser.h`.

Result Codes

The most common result codes returned by the data browser are listed here.

Result Code	Value	Description
`errDataBrowserNotConfigured`	-4970	The data browser is not properly configured. Available in Mac OS X v10.0 and later.
`errDataBrowserItemNotFound`	-4971	The item is not in the data browser. Available in Mac OS X v10.0 and later.
`errDataBrowserPropertyNotFound`	-4972	The property is not in the data browser. Available in Mac OS X v10.0 and later.
`errDataBrowserInvalidPropertyPart`	-4973	The property part is not valid. Available in Mac OS X v10.0 and later.
`errDataBrowserInvalidPropertyData`	-4974	The property data is not valid. Available in Mac OS X v10.0 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Result Code	Value	Description
`errDataBrowserItemNotAdded`	-4975	The item was not added to the data browser. Available in Mac OS X v10.0 and later.
`errDataBrowserPropertyNotSupported`	-4979	The property is not supported in the data browser. Available in Mac OS X v10.0 and later.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Document Revision History

This table describes the changes to *Data Browser Reference*.

Date	Notes
2008-04-08	Made minor technical fixes.
	Corrected information about the `helpItemString` parameter in `DataBrowserGetContextualMenuProcPtr` (page 142).
	Added default behavior for some of the constants in "User Selection Flags" (page 200).
2007-06-28	Made minor technical corrections.
2006-07-24	Updated for Mac OS X v10.4.
2004-03-18	Described which colors are used to highlight active and inactive items. See "Table View Highlighting Styles" (page 198).
2003-12-10	First version of this document.

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Document Revision History

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

Index

A

AddDataBrowserItems function 24
AddDataBrowserListViewColumn function 25
AutoSizeDataBrowserListViewColumns function 26
AXUIElementCreateWithDataBrowserAndItemInfo function 27
AXUIElementGetDataBrowserItemInfo function 27

C

Callback Data Structure Version 175
CloseDataBrowserContainer function 28
Control Data Tags 176
CopyDataBrowserEditText function 29
CreateDataBrowserControl function 29
Custom Callback Data Structure Version 176

D

Data Browser Attributes 177
Data Browser Control Kind Tag 178
Data Browser Metric Values 178
DataBrowserAcceptDragProcPtr callback 137
DataBrowserAccessibilityItemInfo structure 164
DataBrowserAccessibilityItemInfoV0 structure 165
DataBrowserAccessibilityItemInfoV1 structure 166
DataBrowserAddDragItemProcPtr callback 138
DataBrowserCallbacks structure 168
DataBrowserChangeAttributes function 31
DataBrowserCustomCallbacks structure 169
DataBrowserDragFlags data type 170
DataBrowserDrawItemProcPtr callback 139
DataBrowserEditItemProcPtr callback 141
DataBrowserGetAttributes function 31
DataBrowserGetContextualMenuProcPtr callback 142
DataBrowserGetMetric function 32
DataBrowserHitTestProcPtr callback 144
DataBrowserItemAcceptDragProcPtr callback 146
DataBrowserItemCompareProcPtr callback 147
DataBrowserItemDataProcPtr callback 149
DataBrowserItemDataRef data type 171
DataBrowserItemDragRgnProcPtr callback 150
DataBrowserItemHelpContentProcPtr callback 152
DataBrowserItemID data type 171
DataBrowserItemNotificationProcPtr callback 154
DataBrowserItemNotificationWithItemProcPtr callback 155
DataBrowserItemProcPtr callback 157
DataBrowserItemReceiveDragProcPtr callback 158
DataBrowserListViewColumnDesc structure 174
DataBrowserListViewHeaderDesc structure 173
DataBrowserPostProcessDragProcPtr callback 159
DataBrowserPropertyDesc structure 167
DataBrowserPropertyFlags data type 171
DataBrowserPropertyID data type 172
DataBrowserReceiveDragProcPtr callback 160
DataBrowserSelectContextualMenuProcPtr callback 161
DataBrowserSetMetric function 33
DataBrowserTableViewColumnDesc data type 173
DataBrowserTableViewColumnID data type 173
DataBrowserTableViewColumnIndex data type 173
DataBrowserTableViewRowIndex data type 172
DataBrowserTrackingProcPtr callback 163
Display Types 179
DisposeDataBrowserAcceptDragUPP function 33
DisposeDataBrowserAddDragItemUPP function 34
DisposeDataBrowserDrawItemUPP function 34
DisposeDataBrowserEditItemUPP function 35
DisposeDataBrowserGetContextualMenuUPP function 35
DisposeDataBrowserHitTestUPP function 35
DisposeDataBrowserItemAcceptDragUPP function 36
DisposeDataBrowserItemCompareUPP function 36

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

DisposeDataBrowserItemDataUPP function 37
DisposeDataBrowserItemDragRgnUPP function 37
DisposeDataBrowserItemHelpContentUPP function 37
DisposeDataBrowserItemNotificationUPP function 38
DisposeDataBrowserItemNotificationWithItemUPP function 38
DisposeDataBrowserItemReceiveDragUPP function 39
DisposeDataBrowserItemUPP function 39
DisposeDataBrowserPostProcessDragUPP function 39
DisposeDataBrowserReceiveDragUPP function 40
DisposeDataBrowserSelectContextualMenuUPP function 40
DisposeDataBrowserTrackingUPP function 41

E

Editing Commands 181
EnableDataBrowserEditCommand function 41
errDataBrowserInvalidPropertyData constant 202
errDataBrowserInvalidPropertyPart constant 202
errDataBrowserItemNotAdded constant 203
errDataBrowserItemNotFound constant 202
errDataBrowserNotConfigured constant 202
errDataBrowserPropertyNotFound constant 202
errDataBrowserPropertyNotSupported constant 203
ExecuteDataBrowserEditCommand function 42

F

ForEachDataBrowserItem function 42

G

GetDataBrowserActiveItems function 43
GetDataBrowserCallbacks function 44
GetDataBrowserColumnViewDisplayType function 44
GetDataBrowserColumnViewPath function 45
GetDataBrowserColumnViewPathLength function 46
GetDataBrowserCustomCallbacks function 46
GetDataBrowserEditItem function 47
GetDataBrowserEditText function 47
GetDataBrowserHasScrollBars function 48
GetDataBrowserItemCount function 49
GetDataBrowserItemDataBooleanValue function 49
GetDataBrowserItemDataButtonValue function 50
GetDataBrowserItemDataDateTime function 51
GetDataBrowserItemDataDrawState function 52
GetDataBrowserItemDataIcon function 52
GetDataBrowserItemDataIconTransform function 53
GetDataBrowserItemDataItemID function 54
GetDataBrowserItemDataLongDateTime function 54
GetDataBrowserItemDataMaximum function 55
GetDataBrowserItemDataMenuRef function 56
GetDataBrowserItemDataMinimum function 56
GetDataBrowserItemDataProperty function 57
GetDataBrowserItemDataRGBColor function 58
GetDataBrowserItemDataText function 58
GetDataBrowserItemDataValue function 59
GetDataBrowserItemPartBounds function 60
GetDataBrowserItems function 61
GetDataBrowserItemState function 62
GetDataBrowserListViewDisclosureColumn function 62
GetDataBrowserListViewHeaderBtnHeight function 63
GetDataBrowserListViewHeaderDesc function 64
GetDataBrowserListViewUsePlainBackground function 64
GetDataBrowserPropertyFlags function 65
GetDataBrowserScrollBarInset function 66
GetDataBrowserScrollPosition function 66
GetDataBrowserSelectionAnchor function 67
GetDataBrowserSelectionFlags function 68
GetDataBrowserSortOrder function 68
GetDataBrowserSortProperty function 69
GetDataBrowserTableViewColumnCount function 70
GetDataBrowserTableViewColumnPosition function 70
GetDataBrowserTableViewColumnProperty function 71
GetDataBrowserTableViewColumnWidth function 71
GetDataBrowserTableViewGeometry function 72
GetDataBrowserTableViewHiliteStyle function 73
GetDataBrowserTableViewItemID function 73
GetDataBrowserTableViewItemRow function 74
GetDataBrowserTableViewItemRowHeight function 74
GetDataBrowserTableViewNamedColumnWidth function 75
GetDataBrowserTableViewRowHeight function 76
GetDataBrowserTarget function 76
GetDataBrowserUserState function 77
GetDataBrowserViewStyle function 78

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

I

InitDataBrowserCallbacks function 79
InitDataBrowserCustomCallbacks function 80
InvokeDataBrowserAcceptDragUPP function 81
InvokeDataBrowserAddDragItemUPP function 81
InvokeDataBrowserDrawItemUPP function 81
InvokeDataBrowserEditItemUPP function 82
InvokeDataBrowserGetContextualMenuUPP function 82
InvokeDataBrowserHitTestUPP function 83
InvokeDataBrowserItemAcceptDragUPP function 83
InvokeDataBrowserItemCompareUPP function 84
InvokeDataBrowserItemDataUPP function 84
InvokeDataBrowserItemDragRgnUPP function 85
InvokeDataBrowserItemHelpContentUPP function 85
InvokeDataBrowserItemNotificationUPP function 86
InvokeDataBrowserItemNotificationWithItemUPP function 86
InvokeDataBrowserItemReceiveDragUPP function 87
InvokeDataBrowserItemUPP function 87
InvokeDataBrowserPostProcessDragUPP function 88
InvokeDataBrowserReceiveDragUPP function 88
InvokeDataBrowserSelectContextualMenuUPP function 89
InvokeDataBrowserTrackingUPP function 89
IsDataBrowserItemSelected function 90
Item Notifications 182
Item States 184

K

kControlDataBrowserEditTextKeyFilterTag constant 176
kControlDataBrowserEditTextValidationProcTag constant 176
kControlDataBrowserIncludesFrameAndFocusTag constant 176
kControlDataBrowserKeyFilterTag constant 176
kControlKindDataBrowser constant 178
kDataBrowserAlwaysExtendSelection constant 201
kDataBrowserAttributeColumnViewResizeWindow constant 177
kDataBrowserAttributeListViewAlternatingRowColors constant 177
kDataBrowserAttributeListViewDrawColumnDividers constant 177
kDataBrowserAttributeNone constant 177
kDataBrowserCheckboxTriState constant 190
kDataBrowserCheckboxType constant 180
kDataBrowserClientPropertyFlagsMask constant 194
kDataBrowserClientPropertyFlagsOffset constant 194
kDataBrowserCmdTogglesSelection constant 201
kDataBrowserColumnView constant 202
kDataBrowserColumnViewPreviewProperty constant 188
kDataBrowserContainerAliasIDProperty constant 187
kDataBrowserContainerClosed constant 183
kDataBrowserContainerClosing constant 183
kDataBrowserContainerIsClosableProperty constant 187
kDataBrowserContainerIsOpen constant 185
kDataBrowserContainerIsOpenableProperty constant 187
kDataBrowserContainerIsSortableProperty constant 187
kDataBrowserContainerOpened constant 183
kDataBrowserContainerSorted constant 183
kDataBrowserContainerSorting constant 183
kDataBrowserContentHit constant 200
kDataBrowserCustomType constant 179
kDataBrowserDateTimeDateOnly constant 191
kDataBrowserDateTimeRelative constant 190
kDataBrowserDateTimeSecondsToo constant 191
kDataBrowserDateTimeTimeOnly constant 191
kDataBrowserDateTimeType constant 180
kDataBrowserDefaultPropertyFlags constant 189
kDataBrowserDoNotTruncateText constant 191
kDataBrowserDragSelect constant 201
kDataBrowserEditMsgClear constant 182
kDataBrowserEditMsgCopy constant 181
kDataBrowserEditMsgCut constant 181
kDataBrowserEditMsgPaste constant 181
kDataBrowserEditMsgRedo constant 181
kDataBrowserEditMsgSelectAll constant 182
kDataBrowserEditMsgUndo constant 181
kDataBrowserEditStarted constant 182
kDataBrowserEditStopped constant 183
kDataBrowserIconAndTextType constant 181
kDataBrowserIconType constant 180
kDataBrowserItemAdded constant 182
kDataBrowserItemAnyState constant 184
kDataBrowserItemDeselected constant 183
kDataBrowserItemDoubleClicked constant 183
kDataBrowserItemIsActiveProperty constant 186
kDataBrowserItemIsContainerProperty constant 187
kDataBrowserItemIsDragTarget constant 185

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

kDataBrowserItemIsEditableProperty constant 186
kDataBrowserItemIsSelectableProperty constant 186
kDataBrowserItemIsSelected constant 185
kDataBrowserItemNoProperty constant 186
kDataBrowserItemNoState constant 184
kDataBrowserItemParentContainerProperty constant 188
kDataBrowserItemRemoved constant 182
kDataBrowserItemsAdd constant 197
kDataBrowserItemsAssign constant 198
kDataBrowserItemSelected constant 183
kDataBrowserItemSelfIdentityProperty constant 187
kDataBrowserItemsRemove constant 198
kDataBrowserItemsToggle constant 198
kDataBrowserLatestCallbacks constant 175
kDataBrowserLatestCustomCallbacks constant 177
kDataBrowserListView constant 202
kDataBrowserListViewAppendColumn constant 185
kDataBrowserListViewDefaultColumnFlags constant 194
kDataBrowserListViewLatestHeaderDesc constant 185
kDataBrowserListViewMovableColumn constant 193
kDataBrowserListViewNoGapForIconInHeaderButton constant 194
kDataBrowserListViewSelectionColumn constant 193
kDataBrowserListViewSortableColumn constant 193
kDataBrowserListViewTypeSelectColumn constant 194
kDataBrowserMetricCellContentInset constant 178
kDataBrowserMetricDisclosureColumnEdgeInset constant 179
kDataBrowserMetricDisclosureColumnPerDepthGap constant 179
kDataBrowserMetricDisclosureTriangleAndContentGap constant 179
kDataBrowserMetricIconAndTextGap constant 178
kDataBrowserMetricLast constant 179
kDataBrowserNeverEmptySelectionSet constant 201
kDataBrowserNoDisjointSelection constant 201
kDataBrowserNoItem constant 186
kDataBrowserNothingHit constant 200
kDataBrowserNoView constant 202
kDataBrowserOrderDecreasing constant 198
kDataBrowserOrderIncreasing constant 198
kDataBrowserOrderUndefined constant 198
kDataBrowserPopupMenuButtonless constant 192
kDataBrowserPopupMenuType constant 180
kDataBrowserProgressBarType constant 180
kDataBrowserPropertyCheckboxPart constant 196
kDataBrowserPropertyContentPart constant 195
kDataBrowserPropertyDisclosurePart constant 195
kDataBrowserPropertyEnclosingPart constant 195
kDataBrowserPropertyFlagsMask constant 190
kDataBrowserPropertyFlagsOffset constant 190
kDataBrowserPropertyIconPart constant 195
kDataBrowserPropertyIsEditable constant 189
kDataBrowserPropertyIsMutable constant 189
kDataBrowserPropertyModificationFlags constant 192
kDataBrowserPropertyProgressBarPart constant 196
kDataBrowserPropertyRelevanceRankPart constant 196
kDataBrowserPropertySliderPart constant 195
kDataBrowserPropertyTextPart constant 195
kDataBrowserRelativeDateTime constant 192
kDataBrowserRelevanceRankType constant 180
kDataBrowserResetSelection constant 201
kDataBrowserRevealAndCenterInView constant 196
kDataBrowserRevealOnly constant 196
kDataBrowserRevealWithoutSelecting constant 196
kDataBrowserSelectionAnchorDown constant 197
kDataBrowserSelectionAnchorLeft constant 197
kDataBrowserSelectionAnchorRight constant 197
kDataBrowserSelectionAnchorUp constant 197
kDataBrowserSelectionSetChanged constant 184
kDataBrowserSelectOnlyOne constant 201
kDataBrowserSliderDownwardThumb constant 191
kDataBrowserSliderPlainThumb constant 191
kDataBrowserSliderType constant 180
kDataBrowserSliderUpwardThumb constant 191
kDataBrowserStopTracking constant 200
kDataBrowserTableViewFillHilite constant 199
kDataBrowserTableViewLastColumn constant 199
kDataBrowserTableViewMinimalHilite constant 199
kDataBrowserTableViewSelectionColumn constant 199
kDataBrowserTargetChanged constant 184
kDataBrowserTextType constant 180
kDataBrowserTruncateTextAtEnd constant 191
kDataBrowserTruncateTextAtStart constant 192
kDataBrowserTruncateTextMiddle constant 191
kDataBrowserUniversalPropertyFlags constant 189

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

kDataBrowserUniversalPropertyFlagsMask constant 189
kDataBrowserUserStateChanged constant 184
kDataBrowserUserToggledContainer constant 184
kDataBrowserViewSpecificFlagsMask constant 192
kDataBrowserViewSpecificFlagsOffset constant 192
kDataBrowserViewSpecificPropertyFlags constant 193
kHIDataBrowserClassID data type 175

L

List View Append Column 185
List View Header Description Version 185

M

MoveDataBrowserSelectionAnchor function 90

N

NewDataBrowserAcceptDragUPP function 91
NewDataBrowserAddDragItemUPP function 92
NewDataBrowserDrawItemUPP function 92
NewDataBrowserEditItemUPP function 92
NewDataBrowserGetContextualMenuUPP function 93
NewDataBrowserHitTestUPP function 93
NewDataBrowserItemAcceptDragUPP function 94
NewDataBrowserItemCompareUPP function 94
NewDataBrowserItemDataUPP function 95
NewDataBrowserItemDragRgnUPP function 95
NewDataBrowserItemHelpContentUPP function 96
NewDataBrowserItemNotificationUPP function 96
NewDataBrowserItemNotificationWithItemUPP function 97
NewDataBrowserItemReceiveDragUPP function 97
NewDataBrowserItemUPP function 98
NewDataBrowserPostProcessDragUPP function 98
NewDataBrowserReceiveDragUPP function 99
NewDataBrowserSelectContextualMenuUPP function 99
NewDataBrowserTrackingUPP function 100
No Item Constant 185

O

OpenDataBrowserContainer function 100

P

Properties 186
Property Flags
  List View Column Behavior 193
  Modifiers 189
  Offset and Mask for Client-Defined Properties 194
  Offset and Mask for List View Properties 192
  Universal 188
Property Parts 195

R

RemoveDataBrowserItems function 101
RemoveDataBrowserTableViewColumn function 102
Reveal Options 196
RevealDataBrowserItem function 103

S

Selection Anchor Directions 197
Selection State Options 197
SetDataBrowserActiveItems function 103
SetDataBrowserCallbacks function 104
SetDataBrowserColumnViewDisplayType function 106
SetDataBrowserColumnViewPath function 106
SetDataBrowserCustomCallbacks function 107
SetDataBrowserEditItem function 108
SetDataBrowserEditText function 109
SetDataBrowserHasScrollBars function 110
SetDataBrowserItemDataBooleanValue function 110
SetDataBrowserItemDataButtonValue function 111
SetDataBrowserItemDataDateTime function 112
SetDataBrowserItemDataDrawState function 112
SetDataBrowserItemDataIcon function 113
SetDataBrowserItemDataIconTransform function 114
SetDataBrowserItemDataItemID function 114
SetDataBrowserItemDataLongDateTime function 115
SetDataBrowserItemDataMaximum function 116
SetDataBrowserItemDataMenuRef function 117
SetDataBrowserItemDataMinimum function 117
SetDataBrowserItemDataRGBColor function 118

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.

SetDataBrowserItemDataText function 118
SetDataBrowserItemDataValue function 119
SetDataBrowserListViewDisclosureColumn function 120
SetDataBrowserListViewHeaderBtnHeight function 121
SetDataBrowserListViewHeaderDesc function 122
SetDataBrowserListViewUsePlainBackground function 123
SetDataBrowserPropertyFlags function 123
SetDataBrowserScrollBarInset function 124
SetDataBrowserScrollPosition function 125
SetDataBrowserSelectedItems function 126
SetDataBrowserSelectionFlags function 126
SetDataBrowserSortOrder function 127
SetDataBrowserSortProperty function 127
SetDataBrowserTableViewColumnPosition function 128
SetDataBrowserTableViewColumnWidth function 129
SetDataBrowserTableViewGeometry function 129
SetDataBrowserTableViewHiliteStyle function 130
SetDataBrowserTableViewItemRow function 130
SetDataBrowserTableViewItemRowHeight function 131
SetDataBrowserTableViewNamedColumnWidth function 132
SetDataBrowserTableViewRowHeight function 132
SetDataBrowserTarget function 133
SetDataBrowserUserState function 134
SetDataBrowserViewStyle function 134
SortDataBrowserContainer function 135
Sorting Orders 198

T

Table View Highlighting Styles 198
Table View Last Column Value 199
Table View Property Flag 199
Tracking Results 200

U

UpdateDataBrowserItems function 136
User Selection Flags 200

V

View Styles 202

2008-04-08 | © 2003, 2008 Apple Inc. All Rights Reserved.