



# INSIDE MACINTOSH

---

## Mac OS 8.5 Menu Manager Reference



January 12, 1999  
Technical Publications  
© 1999 Apple Computer, Inc.



Apple Computer, Inc.

© 1998 Apple Computer, 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 Computer, Inc., except to make a backup copy of any documentation provided on CD-ROM.

The Apple logo is a trademark of Apple Computer, 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 book. Apple retains all intellectual property rights associated with the technology described in this book. This book is intended to assist application developers to develop applications only for Apple-labeled or Apple-licensed computers.

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

Apple Computer, Inc.

1 Infinite Loop

Cupertino, CA 95014

408-996-1010

Apple, the Apple logo, Mac, and Macintosh are trademarks of Apple Computer, Inc., registered in the United States and other countries.

Adobe, Acrobat, and PostScript are trademarks of Adobe Systems Incorporated or its subsidiaries and may be registered in certain jurisdictions.

Adobe, the Adobe logo, Acrobat, the Acrobat logo, Distiller, PostScript,

and the PostScript logo are trademarks of Adobe Systems Incorporated.

Helvetica and Palatino are registered trademarks of Linotype-Hell AG and/or its subsidiaries.

ITC Zapf Dingbats is a registered trademark of International Typeface Corporation.

Simultaneously published in the United States and Canada.

**Even though Apple has reviewed this manual, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS MANUAL, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS MANUAL IS SOLD “AS IS,” AND YOU, THE PURCHASER, 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 MANUAL, 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

Chapter 1	Mac OS 8.5 Menu Manager Reference	5
-----------	-----------------------------------	---

---

Gestalt Selector for the Mac OS 8.5 Menu Manager	7
--	---

Functions for the Mac OS 8.5 Menu Manager	8
---	---

Controlling Menu Bar Visibility	8
---------------------------------	---

Enabling Menus and Menu Items	10
-------------------------------	----

Associating Data With Menu Items	15
----------------------------------	----

Adding a Menu Item to a Menu	21
------------------------------	----

Accessing the Menu Font	23
-------------------------	----

Modifying the Menu Width	25
--------------------------	----

Result Codes for the Mac OS 8.5 Menu Manager	27
--	----

Appendix A	Document Version History	29
------------	--------------------------	----

---

Index	31
-------	----

---



# Mac OS 8.5 Menu Manager Reference

---

## Contents

<b>Gestalt Selector for the Mac OS 8.5 Menu Manager</b>	<b>7</b>
<b>Functions for the Mac OS 8.5 Menu Manager</b>	<b>8</b>
<b>Controlling Menu Bar Visibility</b>	<b>8</b>
HideMenuBar	8
IsMenuBarVisible	9
ShowMenuBar	9
<b>Enabling Menus and Menu Items</b>	<b>10</b>
DisableMenuItem	10
DisableMenuItemIcon	11
EnableMenuItem	12
EnableMenuItemIcon	13
IsMenuItemEnabled	14
IsMenuItemIconEnabled	15
<b>Associating Data With Menu Items</b>	<b>15</b>
GetMenuItemProperty	16
GetMenuItemPropertySize	17
RemoveMenuItemProperty	19
SetMenuItemProperty	20
<b>Adding a Menu Item to a Menu</b>	<b>21</b>
AppendMenuItemText	21
InsertMenuItemText	22
<b>Accessing the Menu Font</b>	<b>23</b>
GetMenuFont	24
SetMenuFont	24
<b>Modifying the Menu Width</b>	<b>25</b>
GetMenuExcludesMarkColumn	25
SetMenuExcludesMarkColumn	26

Result Codes for the Mac OS 8.5 Menu Manager	27
--	----

Menus allow users to view or choose from a list of choices and commands that your application provides. You can use the Menu Manager to create, display, and manage the drawing and behavior of pull-down, hierarchical, and contextual menus.

This document describes the Menu Manager application programming interface (API) introduced with Mac OS 8.5 and Appearance Manager 1.1, as follows:

- “Gestalt Selector for the Mac OS 8.5 Menu Manager” (page 7)
- “Functions for the Mac OS 8.5 Menu Manager” (page 8)
- “Result Codes for the Mac OS 8.5 Menu Manager” (page 27)

Note that pre-existing Menu Manager functions, types, and constants are not discussed in this document. For a description of the Mac OS 8 Menu Manager API, see *Mac OS 8 Menu Manager Reference*. For descriptions of the pre-Mac OS 8 Menu Manager API, see *Inside Macintosh: Macintosh Toolbox Essentials*.

## Gestalt Selector for the Mac OS 8.5 Menu Manager

---

Before calling any Menu Manager functions, your application should pass the selector `gestaltMenuMgrAttr` to the `Gestalt` function to determine which Menu Manager functions are available.

```
enum {
    gestaltMenuMgrAttr      = 'menu',
    gestaltMenuMgrPresent   = 1
};
```

### Constant descriptions

`gestaltMenuMgrAttr`

The `Gestalt` selector passed to determine what features of the Menu Manager are present. This selector is available with Mac OS 8.5 and later. Passing `gestaltMenuMgrAttr` produces a 32-bit value whose bits you should test to determine what Menu Manager functionality is available.

`gestaltMenuMgrPresent`

If the bit specified by this mask is set, the Menu Manager

functionality for Appearance Manager 1.1 is available. This bit is set for Mac OS 8.5 and later.

## Functions for the Mac OS 8.5 Menu Manager

---

The Mac OS 8.5 Menu Manager provides new functions in the following areas:

- “Controlling Menu Bar Visibility” (page 8)
- “Enabling Menus and Menu Items” (page 10)
- “Associating Data With Menu Items” (page 15)
- “Adding a Menu Item to a Menu” (page 21)
- “Accessing the Menu Font” (page 23)
- “Modifying the Menu Width” (page 25)

### Controlling Menu Bar Visibility

---

The Mac OS 8.5 Menu Manager provides the following functions to control the visibility of the menu bar:

- `HideMenuBar` (page 8) conceals the menu bar.
- `IsMenuBarVisible` (page 9) reports whether the menu bar is currently visible.
- `ShowMenuBar` (page 9) displays the menu bar.

### HideMenuBar

---

Conceals the menu bar.

```
pascal void HideMenuBar (void);
```

#### DISCUSSION

The `HideMenuBar` function makes the menu bar invisible and unselectable by the user.



**VERSION NOTES**

Available with Mac OS 8.5 and later.

**SEE ALSO**

The function `ShowMenuBar` (page 9).

The function `IsMenuBarVisible` (page 9).

## **IsMenuBarVisible**

---

Reports whether the menu bar is currently visible.

```
pascal Boolean IsMenuBarVisible (void);
```

*function result* A value of type `Boolean`. The function returns `true` if the menu bar is currently visible; otherwise, `false`.

**VERSION NOTES**

Available with Mac OS 8.5 and later.

**SEE ALSO**

The function `HideMenuBar` (page 8).

The function `ShowMenuBar` (page 9).

## **ShowMenuBar**

---

Displays the menu bar.

```
pascal void ShowMenuBar (void);
```

**DISCUSSION**

The `ShowMenuBar` function makes the menu bar visible and selectable by the user.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The function `HideMenuBar` (page 8).

The function `IsMenuBarVisible` (page 9).

## Enabling Menus and Menu Items

---

The Mac OS 8.5 Menu Manager provides the following functions to control the enabling of menus, menu items, and menu item icons:

- `DisableMenuItem` (page 10) disables a menu item or a menu.
- `DisableMenuItemIcon` (page 11) disables the icon associated with a menu item.
- `EnableMenuItem` (page 12) enables a menu item or a menu.
- `EnableMenuItemIcon` (page 13) enables the icon associated with a menu item.
- `IsMenuItemEnabled` (page 14) reports whether a given menu or menu item is enabled.
- `IsMenuItemIconEnabled` (page 15) reports whether a given menu item icon is enabled.

## DisableMenuItem

---

Disables a menu item or a menu.

```
pascal void DisableMenuItem (
                                MenuHandle theMenu,
                                UInt16 item);
```

`theMenu`      A value of type `MenuHandle`. Pass a handle to the menu containing the item to be disabled.

`item`          An unsigned 16-bit integer. Pass a value specifying the item number of the menu item that you wish to disable. Pass 0 to specify the menu title and thereby disable the entire menu.

**DISCUSSION**

The `DisableMenuItem` function disables a menu item (and any associated icon) so that the user cannot choose the item from the menu. Unlike the pre-Mac OS 8.5 Menu Manager function `DisableItem`, the `DisableMenuItem` function can disable individual menu items with item numbers greater than 31.

If your application disables a menu title using `DisableMenuItem`, it should then call either the `DrawMenuBar` function or the `InvalidMenuBar` function to update the menu bar's appearance.

**VERSION NOTES**

Available with Mac OS 8.5 and later.

**SEE ALSO**

The function `EnableMenuItem` (page 12).

The function `IsMenuItemEnabled` (page 14).

**DisableMenuItemIcon**


---

Disables the icon associated with a menu item.

```
pascal void DisableMenuItemIcon (
                                MenuHandle theMenu,
                                UInt16 item);
```

`theMenu`      A value of type `MenuHandle`. Pass a handle to the menu containing the icon to be disabled.

`item`          An unsigned 16-bit integer. Pass a value specifying the item number of the menu item containing the icon.

**DISCUSSION**

Your application can use the `DisableMenuItemIcon` function to dim individual menu item icons. The menu item that contains the icon is unaffected by calling `DisableMenuItemIcon`. That is, if `DisableMenuItemIcon` disables an enabled menu item's icon, the menu item itself will remain enabled. Calling

`DisableMenuItemIcon` on the icon of a menu item that is currently disabled will cause the icon to be disabled once the menu item is re-enabled.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The function `EnableMenuItemIcon` (page 13).

The function `IsMenuItemIconEnabled` (page 15).

## EnableMenuItem

---

Enables a menu item or a menu.

```
pascal void EnableMenuItem (
                                MenuHandle theMenu,
                                UInt16 item);
```

`theMenu`      A value of type `MenuHandle`. Pass a handle to the menu containing the item to be enabled.

`item`            An unsigned 16-bit integer. Pass a value specifying the item number of the menu item that you wish to enable. If you pass 0, `EnableMenuItem` enables the menu title and all items in the menu that were not previously individually disabled.

## DISCUSSION

The `EnableMenuItem` function enables a menu item so that the user can choose the item from the menu. Unlike the pre-Mac OS 8.5 Menu Manager function `EnableItem`, the `EnableMenuItem` function can enable individual menu items with item numbers greater than 31. If the menu item has an associated icon, that icon is also enabled, unless the icon was previously individually disabled with the function `DisableMenuItemIcon` (page 11).

If your application enables a menu title using `EnableMenuItem`, it should then call either the `DrawMenuBar` function or the `InvalidMenuBar` function to update the menu bar's appearance.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The function `DisableMenuItem` (page 10).

The function `IsMenuItemEnabled` (page 14).

## EnableMenuItemIcon

---

Enables the icon associated with a menu item.

```
pascal void EnableMenuItemIcon (
    MenuHandle theMenu,
    UInt16 item);
```

`theMenu`      A value of type `MenuHandle`. Pass a handle to the menu containing the icon to be enabled.

`item`          An unsigned 16-bit integer. Pass a value specifying the item number of the menu item containing the icon.

## DISCUSSION

Your application can use the `EnableMenuItemIcon` function to enable individual menu item icons that have been previously disabled by a call to the function `DisableMenuItemIcon` (page 11). The menu item that contains the icon is unaffected by calling `EnableMenuItemIcon`. Note that enabling the icon of a currently disabled menu item has no visual effect; however, once the menu item is enabled, the icon is also enabled.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The function `DisableMenuItemIcon` (page 11).

The function `IsMenuItemIconEnabled` (page 15).

## IsMenuItemEnabled

---

Reports whether a given menu or menu item is enabled.

```
pascal Boolean IsMenuItemEnabled (
    MenuHandle menu,
    UInt16 item);
```

**menu** A value of type `MenuHandle`. Pass a handle to the menu containing the item to be examined.

**item** An unsigned 16-bit integer. Pass a value specifying the item number of the menu item. Pass 0 to specify the menu title and determine whether the menu as a whole is enabled.

**function result** A value of type `Boolean`. The function returns `true` if the menu item is currently enabled; otherwise, `false`.

## DISCUSSION

Your application can use the `IsMenuItemEnabled` function to determine whether specific menu items, even those with item numbers greater than 31, are currently enabled and can therefore be selected by the user.

The `IsMenuItemEnabled` function works for items enabled or disabled with either the Mac OS 8.5 Menu Manager functions `EnableMenuItem` (page 12) and `DisableMenuItem` (page 10) or with the pre-Mac OS 8.5 Menu Manager functions `EnableItem` and `DisableItem`.

## VERSION NOTES

Available with Mac OS 8.5 and later.

**IsMenuItemIconEnabled**

---

Reports whether a given menu item icon is enabled.

```
pascal Boolean IsMenuItemIconEnabled (
    MenuHandle menu,
    UInt16 item);
```

**menu**            A value of type `MenuHandle`. Pass a handle to the menu containing the icon to be examined.

**item**            An unsigned 16-bit integer. Pass a value specifying the item number of the menu item containing the icon.

**function result** A value of type `Boolean`. The function returns `true` if the menu item icon is currently enabled; otherwise, `false`.

**DISCUSSION**

Your application can use the `IsMenuItemIconEnabled` function to determine whether a specific menu item's icon is currently enabled or dimmed.

**VERSION NOTES**

Available with Mac OS 8.5 and later.

**SEE ALSO**

The `DisableMenuItemIcon` (page 11) function.

The `EnableMenuItemIcon` (page 13) function.

**Associating Data With Menu Items**

---

The Mac OS 8.5 Menu Manager provides the following functions for your application to associate data with menu items:

- `GetMenuItemProperty` (page 16) obtains a piece of data that has been previously associated with a menu item.
- `GetMenuItemPropertySize` (page 17) obtains the size of a piece of data that has been previously associated with a menu item.

- `RemoveMenuItemProperty` (page 19) removes a piece of data that has been previously associated with a menu item.
- `SetMenuItemProperty` (page 20) associates data with a menu item.

## GetMenuItemProperty

---

Obtains a piece of data that has been previously associated with a menu item.

```
pascal OSStatus GetMenuItemProperty (
    MenuHandle menu,
    UInt16 item,
    OSType propertyCreator,
    OSType propertyTag,
    UInt32 bufferSize,
    UInt32 *actualSize,
    void *propertyBuffer);
```

`menu`            A value of type `MenuHandle`. Pass a handle to the menu containing the item to be examined for associated data.

`item`            An unsigned 16-bit integer. Pass a value specifying the item number of the menu item or 0 if the data is associated with the menu as a whole.

`propertyCreator`    A four-character code. Pass your program's signature, as registered through Apple Developer Technical Support. If your program is of a type that would not normally have a signature (for example, a plug-in), you should still register and use a signature in this case, even though your program's file may not have the same creator code as the signature that you register. The 'macs' property signature is reserved for the system and should not be used.

`propertyTag`        A four-character code. Pass the application-defined code identifying the data.

`bufferSize`        An unsigned 32-bit integer. Pass a value specifying the size of the data to be obtained. If the size of the data is unknown, use the function `GetMenuItemPropertySize` (page 17) to get the data's size. If the size specified in the `bufferSize` parameter does not



match the actual size of the property, `GetMenuItemProperty` only retrieves data up to the size specified or up to the actual size of the property, whichever is smaller, and an error is returned.

`actualSize` A pointer to an unsigned 32-bit integer. On return, this value is set to the actual size of the associated data. You may pass `nil` for the `actualSize` parameter if you are not interested in this information.

`propertyBuffer` A pointer to a buffer. On return, this buffer contains a copy of the data that is associated with the specified menu item.

*function result* A result code. See “Result Codes for the Mac OS 8.5 Menu Manager” (page 27).

## DISCUSSION

You may use the function `GetMenuItemProperty` to obtain a copy of data previously set with the function `SetMenuItemProperty` (page 20).

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The `RemoveMenuItemProperty` (page 19) function.

## GetMenuItemPropertySize

---

Obtains the size of a piece of data that has been previously associated with a menu item.

```
pascal OSStatus GetMenuItemPropertySize (
    MenuHandle menu,
    UInt16 item,
    OSType propertyCreator,
    OSType propertyTag,
    UInt32 *size);
```

<code>menu</code>	A value of type <code>MenuHandle</code> . Pass a handle to the menu containing the item to be examined for associated data.
<code>item</code>	An unsigned 16-bit integer. Pass a value specifying the item number of the menu item or 0 if the data is associated with the menu as a whole.
<code>propertyCreator</code>	A four-character code. Pass your program's signature, as registered through Apple Developer Technical Support. If your program is of a type that would not normally have a signature (for example, a plug-in), you should still register and use a signature in this case, even though your program's file may not have the same creator code as the signature that you register. The 'macs' property signature is reserved for the system and should not be used.
<code>propertyTag</code>	A four-character code. Pass the application-defined code identifying the data.
<code>size</code>	A pointer to an unsigned 32-bit integer. On return, this value is set to the actual size of the data.
<i>function result</i>	A result code. See "Result Codes for the Mac OS 8.5 Menu Manager" (page 27).

### DISCUSSION

If you want to retrieve a piece of associated data with the function `GetMenuItemProperty` (page 16), you will typically need to use the `GetMenuItemPropertySize` function beforehand to determine the size of the associated data.

### VERSION NOTES

Available with Mac OS 8.5 and later.

**RemoveMenuItemProperty**


---

Removes a piece of data that has been previously associated with a menu item.

```
pascal OSStatus RemoveMenuItemProperty (
    MenuHandle menu,
    UInt16 item,
    OSType propertyCreator,
    OSType propertyTag);
```

**menu**            A value of type `MenuHandle`. Pass a handle to the menu containing the item whose associated data is to be removed.

**item**            An unsigned 16-bit integer. Pass a value specifying the item number of the menu item or 0 if the data is associated with the menu as a whole.

**propertyCreator**    A four-character code. Pass your program's signature, as registered through Apple Developer Technical Support. If your program is of a type that would not normally have a signature (for example, a plug-in), you should still register and use a signature in this case, even though your program's file may not have the same creator code as the signature that you register. The 'macs' property signature is reserved for the system and should not be used.

**propertyTag**        A four-character code. Pass the application-defined code identifying the associated data.

**function result**    A result code. See "Result Codes for the Mac OS 8.5 Menu Manager" (page 27).

**DISCUSSION**

Your application may remove data set with the `SetMenuItemProperty` (page 20) function by calling the `RemoveMenuItemProperty` function.

**VERSION NOTES**

Available with Mac OS 8.5 and later.

## SEE ALSO

The `GetMenuItemProperty` (page 16) function.

## SetMenuItemProperty

---

Associates data with a menu item.

```
pascal OSStatus SetMenuItemProperty (
    MenuHandle menu,
    UInt16 item,
    OSType propertyCreator,
    OSType propertyTag,
    UInt32 propertySize,
    void *propertyData);
```

**menu** A value of type `MenuHandle`. Pass a handle to the menu containing the item with which you wish to associate data.

**item** An unsigned 16-bit integer. Pass a value specifying the item number of the menu item or 0 if the data is to be associated with the menu as a whole.

**propertyCreator** A four-character code. Pass your program's signature, as registered through Apple Developer Technical Support. If your program is of a type that would not normally have a signature (for example, a plug-in), you should still register and use a signature in this case, even though your program's file may not have the same creator code as the signature that you register. The 'macs' property signature is reserved for the system and should not be used.

**propertyTag** A four-character code. Pass a value identifying the data. You define the tag your application uses to identify the data; this code is not to be confused with the file type for the data, but may coincide if you wish.

**propertySize** An unsigned 32-bit integer. Pass a value specifying the size of the data.

**propertyData** Pass a pointer to the data.

*function result* A result code. See “Result Codes for the Mac OS 8.5 Menu Manager” (page 27).

## DISCUSSION

You may use the `SetMenuItemProperty` function to associate any type of data with a menu item.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The `GetMenuItemProperty` (page 16) function.

The `RemoveMenuItemProperty` (page 19) function.

## Adding a Menu Item to a Menu

---

The Mac OS 8.5 Menu Manager provides the following functions to add individual menu items to menus:

- `AppendMenuItemText` (page 21) appends a menu item to a menu.
- `InsertMenuItemText` (page 22) inserts a menu item into a menu.

## AppendMenuItemText

---

Appends a menu item to a menu.

```
pascal OSStatus AppendMenuItemText (
    MenuHandle menu,
    ConstStr255Param inString);
```

`menu` A value of type `MenuHandle`. Pass a handle to the menu to which the menu item is to be appended.

**inString** A value of type `ConstStr255Param`. Pass a Pascal string containing the text of the menu item to append. You can pass a string containing any characters, and these characters will be presented verbatim in the menu item.

**function result** A result code. See “Result Codes for the Mac OS 8.5 Menu Manager” (page 27).

## DISCUSSION

The `AppendMenuItemText` function appends the menu item containing the specified string to a menu, without evaluating the string for metacharacters, as the pre-Mac OS 8.5 Menu Manager function `AppendMenu` does. You may wish to use `AppendMenuItemText` if you have a need to present non-alphanumeric characters in a menu item.

The appended menu item appears at the end of the menu as an enabled item. If you wish to place the menu item elsewhere than at the end of the menu you should use the function `InsertMenuItemText` (page 22).

## VERSION NOTES

Available with Mac OS 8.5 and later.

## InsertMenuItemText

---

Inserts a menu item into a menu.

```
pascal OSStatus InsertMenuItemText (
    MenuHandle menu,
    ConstStr255Param inString,
    SInt16 afterItem);
```

**menu** A value of type `MenuHandle`. Pass a handle to the menu into which the menu item is to be inserted.

**inString** A value of type `ConstStr255Param`. Pass a Pascal string containing the text of the menu item to insert. You can pass a string containing any characters, and these characters will be presented verbatim in the menu item.

**afterItem** A signed 16-bit integer. Pass a value specifying the item number of the menu item after which the new menu item is to be inserted. Specify 0 to insert the new menu item at the top of the menu, before the first menu item. Specify a value greater than or equal to the last menu item to append the new item to the end of the menu.

**function result** A result code. See “Result Codes for the Mac OS 8.5 Menu Manager” (page 27).

## DISCUSSION

The `InsertMenuItemText` function inserts an enabled menu item containing the specified string into a menu, without evaluating the string for metacharacters, as the pre-Mac OS 8.5 Menu Manager function `InsertMenuItem` does. You may wish to use `InsertMenuItemText` if you have a need to present non-alphanumeric characters in a menu item.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The function `AppendMenuItemText` (page 21).

## Accessing the Menu Font

---

The Mac OS 8.5 Menu Manager provides the following functions to access a menu's font:

- `GetMenuFont` (page 24) obtains the font used in a menu.
- `SetMenuFont` (page 24) sets the font to be used in a menu.

**GetMenuFont**

---

Obtains the font used in a menu.

```

pascal OSStatus GetMenuFont (
    MenuHandle menu,
    SInt16 *outFontID,
    UInt16 *outFontSize);

```

**menu**            A value of type `MenuHandle`. Pass a handle to the menu whose font is to be obtained.

**outFontID**       A pointer to a signed 16-bit integer. On return, this value identifies the font family ID for the menu font.

**outFontSize**     A pointer to an unsigned 16-bit integer. On return, this value identifies the size of the font, in points.

**function result** A result code. See “Result Codes for the Mac OS 8.5 Menu Manager” (page 27).

**DISCUSSION**

Your application may use the `GetMenuFont` function to retrieve the font for an individual menu, such as a pop-up menu.

**VERSION NOTES**

Available with Mac OS 8.5 and later.

**SetMenuFont**

---

Sets the font to be used in a menu.

```

pascal OSStatus SetMenuFont (
    MenuHandle menu,
    SInt16 inFontID,
    UInt16 inFontSize);

```

**menu**            A value of type `MenuHandle`. Pass a handle to the menu whose font is to be set.



<code>inFontID</code>	A signed 16-bit integer. Pass the font family ID for the font to be used. Pass 0 to use the current system font. See <i>Inside Macintosh: Text</i> for a discussion of font family IDs.
<code>inFontSize</code>	An unsigned 16-bit integer. Pass a value identifying the size in points of the font to be used. Valid font size values range from 9 to 24 points, inclusive. Pass 0 to use the font size of the current system font.
<i>function result</i>	A result code. See “Result Codes for the Mac OS 8.5 Menu Manager” (page 27).

## DISCUSSION

Your application may use the `SetMenuFont` function to set the font for an individual menu, such as a pop-up menu.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## Modifying the Menu Width

---

The Mac OS 8.5 Menu Manager provides the following functions to modify a menu’s width:

- `GetMenuExcludesMarkColumn` (page 25) returns whether a menu contains space for mark characters.
- `SetMenuExcludesMarkColumn` (page 26) sets whether a menu contains space for mark characters.

## GetMenuExcludesMarkColumn

---

Returns whether a menu contains space for mark characters.

```
pascal Boolean GetMenuExcludesMarkColumn (
    MenuHandle menu);
```

`menu`                      A value of type `MenuHandle`. Pass a handle to the menu whose width is to be examined.

**function result** A value of type `Boolean`. The `GetMenuExcludesMarkColumn` function returns a value of `true` if the menu currently contains space for mark characters. `GetMenuExcludesMarkColumn` returns a value of `false` if the menu is currently drawn in its full width, with space for mark characters.

## VERSION NOTES

Available with Mac OS 8.5 and later.

## SEE ALSO

The function `SetMenuExcludesMarkColumn` (page 26).

## SetMenuExcludesMarkColumn

---

Sets whether a menu contains space for mark characters.

```
pascal OSStatus SetMenuExcludesMarkColumn (
    MenuHandle menu,
    Boolean excludesMark);
```

**menu** A value of type `MenuHandle`. Pass a handle to the menu whose width is to be set.

**excludesMark** A value of type `Boolean`. Pass a value of `true` to specify that the menu be drawn without space for mark characters. Pass a value of `false` to specify that the menu be drawn in its full width, with space for mark characters.

**function result** A result code. See “Result Codes for the Mac OS 8.5 Menu Manager” (page 27).

## DISCUSSION

Your application may use the `SetMenuExcludesMarkColumn` function to set the width of an individual menu, so that no space is provided for mark characters such as checkmarks, dashes, or notification symbols (diamonds).

**SPECIAL CONSIDERATIONS**

The `SetMenuExcludesMarkColumn` function is only recommended for use with pop-up menus, and then only in special cases. Mac OS human interface guidelines require that all standard (menu bar) menus include space for mark characters, and pop-up menus that present user-selectable attributes or commands should also contain space for marks. If a pop-up menu does not present a list of user-selectable attributes or commands, as is the case with the Mac OS 8.5 Window Manager window proxy pop-up menus that display a standard file system path, then narrowing the menu to exclude space for marks may be appropriate.

**VERSION NOTES**

Available with Mac OS 8.5 and later.

**SEE ALSO**

The function `GetMenuExcludesMarkColumn` (page 25).

## Result Codes for the Mac OS 8.5 Menu Manager

---

The new result codes returned by the Mac OS 8.5 Menu Manager are listed below.

<code>menuPropertyInvalid</code>	<b>-5603</b>	'macs' property signature not allowed
<code>menuPropertyNotFoundErr</code>	<b>-5604</b>	Specified property does not exist

## CHAPTER 1

### Mac OS 8.5 Menu Manager Reference

# Document Version History

---

This document has had the following releases:

**Table A-1**      *Mac OS 8.5 Menu Manager Reference* Revision History

---

Version	Notes
Jan. 12, 1999	<p>Initial public release. The following changes were made from the prior version:</p> <p>Changed “Menu Manager 2.0” to “Mac OS 8.5 Menu Manager” throughout to reflect final versioning.</p> <p>“Gestalt Selector for the Mac OS 8.5 Menu Manager” (page 7). Added description.</p> <p>DisableMenuItem (page 10), DisableMenuItemIcon (page 11), EnableMenuItem (page 12), EnableMenuItemIcon (page 13). Changed function result to void.</p> <p>“Result Codes for the Mac OS 8.5 Menu Manager” (page 27). Removed pre-release result codes menuPropertyNotFoundErr and menuPropertyInternalErr and added descriptions of final result codes menuPropertyInvalid and menuPropertyNotFoundErr.</p>
Apr. 13, 1998	First seed draft release.

## **A P P E N D I X**

### Document Version History

# Index

---

## A

---

accessing the menu font 23  
adding a menu item to a menu 21  
AppendMenuItemText **function** 21  
associating data with menu items 15

---

## C

---

controlling menu bar visibility 8

---

## D

---

DisableMenuItem **function** 10  
DisableMenuItemIcon **function** 11

---

## E

---

EnableMenuItem **function** 12  
EnableMenuItemIcon **function** 13  
enabling menus and menu items 10

---

## G

---

gestaltMenuMgrAttr **constant** 7  
gestaltMenuMgrPresent **constant** 7  
GetMenuExcludesMarkColumn **function** 25  
GetMenuFont **function** 24  
GetMenuItemProperty **function** 16  
GetMenuItemPropertySize **function** 17

---

## H

---

HideMenuBar **function** 8

---

## I

---

InsertMenuItemText **function** 22  
IsMenuBarVisible **function** 9  
IsMenuItemEnabled **function** 14  
IsMenuItemIconEnabled **function** 15

---

## M

---

menuPropertyInvalid **result code** 27  
menuPropertyNotFoundErr **result code** 27  
modifying the menu width 25

---

## R

---

RemoveMenuItemProperty **function** 19

---

## S

---

SetMenuExcludesMarkColumn **function** 26  
SetMenuFont **function** 24  
SetMenuItemProperty **function** 20  
ShowMenuBar **function** 9

---

This Apple manual was written, edited, and composed on a desktop publishing system using Apple Macintosh computers and FrameMaker software. Line art was created using Adobe<sup>™</sup> Illustrator and Adobe Photoshop.

Text type is Palatino<sup>®</sup> and display type is Helvetica<sup>®</sup>. Bullets are ITC Zapf Dingbats<sup>®</sup>. Some elements, such as program listings, are set in Adobe Letter Gothic.

WRITER

Donna S. Lee

PRODUCTION EDITOR

Glen Frank

Acknowledgments to Guy Fullerton, Pete Gontier, Chris Thomas, and Ed Voas.