



Constructor for Palm OS[®]

Palm OS[®] 5 SDK (68K) R3

CONTRIBUTORS

Written by Brian Maas

Engineering contributions by Kevin Cooper, Vivek Magotra, Andrew Stadler, Robert Sirota, and Michael Dilloughery

Copyright © 1996 - 2003, PalmSource, Inc. and its affiliates. All rights reserved. This documentation may be printed and copied solely for use in developing products for Palm OS® software. In addition, two (2) copies of this documentation may be made for archival and backup purposes. Except for the foregoing, no part of this documentation may be reproduced or transmitted in any form or by any means or used to make any derivative work (such as translation, transformation or adaptation) without express written consent from PalmSource, Inc.

PalmSource, Inc. reserves the right to revise this documentation and to make changes in content from time to time without obligation on the part of PalmSource, Inc. to provide notification of such revision or changes.

PALMSOURCE, INC. AND ITS SUPPLIERS MAKE NO REPRESENTATIONS OR WARRANTIES THAT THE DOCUMENTATION IS FREE OF ERRORS OR THAT THE DOCUMENTATION IS SUITABLE FOR YOUR USE. THE DOCUMENTATION IS PROVIDED ON AN "AS IS" BASIS. PALMSOURCE, INC. AND ITS SUPPLIERS MAKE NO WARRANTIES, TERMS OR CONDITIONS, EXPRESS OR IMPLIED, EITHER IN FACT OR BY OPERATION OF LAW, STATUTORY OR OTHERWISE, INCLUDING WARRANTIES, TERMS, OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND SATISFACTORY QUALITY. TO THE FULL EXTENT ALLOWED BY LAW, PALMSOURCE, INC. ALSO EXCLUDES FOR ITSELF AND ITS SUPPLIERS ANY LIABILITY, WHETHER BASED IN CONTRACT OR TORT (INCLUDING NEGLIGENCE), FOR DIRECT, INCIDENTAL, CONSEQUENTIAL, INDIRECT, SPECIAL, OR PUNITIVE DAMAGES OF ANY KIND, OR FOR LOSS OF REVENUE OR PROFITS, LOSS OF BUSINESS, LOSS OF INFORMATION OR DATA, OR OTHER FINANCIAL LOSS ARISING OUT OF OR IN CONNECTION WITH THIS DOCUMENTATION, EVEN IF PALMSOURCE, INC. OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Palm OS, Palm Computing, HandFAX, HandSTAMP, HandWEB, Graffiti, HotSync, iMessenger, MultiMail, Palm.Net, PalmPak, PalmConnect, PalmGlove, PalmModem, PalmPoint, PalmPrint, PalmSource, Palm, the Palm logo, MyPalm, PalmGear, PalmPix, PalmPower, AnyDay, EventClub, HandMAIL, the HotSync logo, Palm Powered, the Palm trade dress, Simply Palm, ThinAir, and WeSync are trademarks of PalmSource, Inc. or its affiliates. All other product and brand names may be trademarks or registered trademarks of their respective owners.

IF THIS DOCUMENTATION IS PROVIDED ON A COMPACT DISC, THE OTHER SOFTWARE AND DOCUMENTATION ON THE COMPACT DISC ARE SUBJECT TO THE LICENSE AGREEMENT ACCOMPANYING THE COMPACT DISC.

Constructor for Palm OS

Document Number 3007-006

July 30, 2003

For the latest version of this document, visit

<http://www.palmos.com/dev/support/docs/>.

PalmSource, Inc.

1240 Crossman Avenue

Sunnyvale, CA 94089

USA

www.palmsource.com

Table of Contents

Introduction	ix
Overview	ix
Read the Release Notes	ix
What's New	x
Version 1.9	x
Version 1.8	x
Version 1.7	x
Version 1.6	xi
System Requirements	xi
Windows Requirements.	xi
Macintosh Requirements	xii
Starting Points	xii
Additional Resources	xiv
 1 Constructor Fundamentals	 1
Constructor Fundamentals Overview	1
What is a Palm OS Resource?.	2
Constructor's Resource Types	2
Application Resources	3
Forms and Controls	3
Menus	4
Character Strings, String Lists, and Category Names	4
Alerts.	5
Icons and Bitmaps	5
Font Families	7
Wave Sound	7
Custom Data	7
Undocumented Resource Types	7
Constructor Project Files.	8
Managing Files with Constructor.	8
Editing PRC Files.	9
Understanding the Resource.frk Folder	9
Moving a Constructor File Between Platforms	10
Moving a File from Macintosh to Windows	11

Moving a File from Windows to Macintosh	11
Header File Generation	12
Project Window	13
Resource Type and Name List	14
File Control Status	15
Project Settings.	16
Editor Windows	21
Property Inspector Window	22
User Interface Features	24
Selection	24
Moving, Copying, and Deleting	25
Lists	26
Text Editing	27
Undo	27
2 Working with Forms	29
Working with Forms Overview.	29
Windows for Forms.	29
Catalog Window	29
Form Layout Window	32
Hierarchy Window	33
Editing Forms	34
Creating a Form	34
Creating an About Box	35
Creating a Dialog Box.	35
Setting Form Properties	36
Adding Interface Items to a Form.	38
Adding Help to a Form	39
Adding a Menu Bar to a Form	40
Arranging Items in a Form	40
Grouping Items in a Form	42
Deleting Items in a Form	42
3 Working with Menu Bars and Menus	43
Working with Menu Bars and Menus Overview	43
Menu Bars and Menus	43

Creating a Menu Bar Resource	43
Menu Bar Editor Window	45
Creating a Menu Resource.	46
Menu Editor Window.	47
Editing Menu Bar and Menu Resources	48
Setting Menu Bar Properties	49
Arranging Menus	49
Deleting Menus	50
Creating Menu Commands	50
Setting Menu Properties.	51
Adding a Command to a Menu	51
Modifying a Command in a Menu	52
Arranging Commands in a Menu.	52
Deleting Commands from a Menu	52
4 Working with Character Strings	53
Working with Character Strings Overview	53
Windows for Character Strings	53
String Editor Window	54
String List Editor Window	54
App Info String List Editor Window	55
Character Map Window.	56
Editing Character Strings	58
Creating a Character String	58
Creating a Character String List	59
Creating a Category String List.	59
Creating Help Text	60
Setting String Properties.	60
Modifying a Character String	61
Setting the String Font	61
Deleting Character Strings.	62
5 Working with Alerts	63
Working With Alerts Overview.	63
Windows for Alerts.	63
Alert Layout Window.	63

Editing Alerts	64
Creating an Alert.	65
Setting Alert Properties	65
Adding Help to an Alert	67
Setting the Alert's Message	68
Adding Buttons to an Alert	68
Deleting Buttons from an Alert.	68

6 Working with Icons and Bitmap Images 71

Working with Icons and Bitmaps Overview	71
Bitmap Families	72
Application Icon Families	75
Icons in Palm OS 3.5 and Later	77
Updating Older Icons.	78
Important Icon Numbering and Size Conventions	80
Bitmap Images in Palm OS 3.5 and Later.	80
Bitmap Editor Window	82
Tool Palette	82
Drawing on the Canvas	83
Viewing Sample Views	84
Using the Bitmap Editor Tools	85
Lasso Tool.	85
Marquee Tool	86
Text Tool	86
Pencil Tool	87
Eraser Tool	88
Paint Bucket Tool.	88
Dropper Tool	89
Line Tool	89
Rectangle Tools	90
Oval Tools.	90
Pattern Tool	91
Color Tool.	92
Editing Icons and Bitmaps	93
Selecting Pixels	94
Nudging Pixels	94

Copying Pixels.	94
Rotating and Inverting Pixels	95
Setting Colors and Patterns	95
Drawing Shapes	96
Drag and Drop.	96
7 Working with Fonts	99
Working with Font Families Overview.	99
Fonts and Font Families	99
Creating a Font Family	100
Windows for Font Families.	102
Font Family Editor Window	102
8 Working with Sound Resources	103
Working with Sound Resources Overview	103
Creating a Sound Resource.	104
Windows for Sound Resources	104
Wave Sound Editor Window.	104
9 Working with Custom Data Resources	107
Working with Custom Data Resources Overview	107
Creating a Custom Data Resource.	108
Windows for Custom Data Resources	110
Custom Data Editor Window	110
10 Menu Reference	113
Constructor Menu Reference Overview	113
File	114
Edit.	115
Window.	118
Arrange	118
Layout	120
Options	121
Colors.	122
Font	123
Style	123

A Catalog Resources	125
Button Resource	127
Check Box Resource.	128
Feedback Slider Resource	129
Field Resource	130
Form Bitmap Resource	132
Gadget Resource	132
Graffiti Shift Indicator Resource	133
Graphic Button Resource	134
Graphic Push Button Resource	135
Graphic Repeating Button Resource.	136
Label Resource	138
List Resource.	138
Popup Trigger Resource	140
Push Button Resource	141
Repeating Button Resource	142
Scroll Bar Resource	143
Selector Trigger Resource	144
Slider Resource.	145
Table Resource	147
 B Palm OS Resource Types	 149
 Index	 153

Introduction

Welcome to Constructor for Palm OS® software. Constructor for Palm OS software is used to build visual interfaces for Palm OS software applications, used by devices like Palm Powered handhelds.

In this book, Constructor is sometimes used as a short name for Constructor for Palm OS software.

Overview

To learn about using Constructor for Palm OS software:

- “[Read the Release Notes](#)” - The release notes provide important information about last-minute changes to Constructor for Palm OS software.
- “[What’s New](#)” - Includes additions and changes to this manual.
- “[System Requirements](#)” on page xi - Provides information about the hardware and software you need to use Constructor for Palm OS software.
- “[Starting Points](#)” on page xii - Provides an overview of the chapters in this manual.

Read the Release Notes

Before using Constructor, read the Constructor release notes in the CW Release Notes folder for important information about new features, bug fixes, and last-minute changes not included in this documentation.

What's New

This book has been updated for the following releases.

Version 1.9

The new features in Constructor for Palm OS Version 1.9 include:

- Support for Simplified Chinese as a Palm OS target setting and as a default text encoding for editing PRC files. See [“Palm OS Target option”](#) on page 19 and [“Preferences”](#) on page 117 for more information.
- Support for one-and-one-half (1.5X) density screens. Your application icon families, bitmap families, and font families can use the new density One and One Half. For more information, see [“Bitmap Families”](#) on page 72, [“Application Icon Families”](#) on page 75, and [“Fonts and Font Families”](#) on page 99.
- Support for including custom resource data. You can include resource data from a binary data file or resource data entered manually in hex binary format. See [Chapter 9, “Working with Custom Data Resources,”](#) on page 107 for more information.

Version 1.8

The new features in Constructor for Palm OS Version 1.8 include:

- Support for sound resources with the new Wave Sound Editor. See [Chapter 8, “Working with Sound Resources,”](#) on page 103 for more information.

Version 1.7

The new features in Constructor for Palm OS Version 1.7 include:

- Support for high density bitmaps in the App Icon Family Editor. See [Chapter 6, “Application Icon Families,”](#) on page 75 for more information.
- Support for high density bitmaps in the Bitmap Family Editor. See [Chapter 6, “Bitmap Families,”](#) on page 72 for more information.

- Support for high density fonts with the new Font Family Editor. See [Chapter 7, “Working with Fonts,”](#) on page 99 for more information.

Version 1.6

The new features in Constructor for Palm OS Version 1.6 include:

- Support for editing resources within PRC files. See [Chapter 1, “Editing PRC Files,”](#) on page 9 for more information.
- Updates to the Bitmap Editor to support 16-bit color images. See [Chapter 6, “Working with Icons and Bitmaps Overview,”](#) on page 71 for more information.
- An enhancement to allow up to five bitmap images rendered in different bit depths in a single bitmap resource. See [Chapter 6, “Bitmap Families,”](#) on page 72 for more information.
- The addition of new user interface objects: feedback slider control, graphic button, graphic push button, graphic repeating button, and slider control. See [Chapter 2, “Catalog Window,”](#) on page 29 for more information.

System Requirements

If you currently use other CodeWarrior software, you can use Constructor for Palm OS software.

- [Windows Requirements](#)
- [Macintosh Requirements](#)

Windows Requirements

[Table 1](#) describes hardware and software requirements for using Constructor on Microsoft Windows.

Table 1 Microsoft Windows Requirements

Hardware	Palm Powered organizer with HotSync® cradle (optional, but recommended); Intel Pentium-class processor (recommended) or 80486-class processor; 24 megabytes of RAM; CD-ROM drive; 90 MB free hard disk space.
Software	CodeWarrior for Palm OS software; Windows NT 4.0 or higher (recommended); Windows 95 or higher.

Macintosh Requirements

[Table 2](#) lists hardware and software requirements for using Constructor on a Macintosh computer.

Table 2 Macintosh Requirements

Hardware	Palm Powered organizer with HotSync cradle (optional but recommended); PowerPC (recommended) or MC68040 processor; 24 megabytes of RAM (32 megabytes recommended); a CD-ROM drive, 90 megabytes free hard disk space.
Software	CodeWarrior for Palm OS software, Macintosh System 8.x or higher.

Constructor for Palm OS software is part of a complete package that includes the CodeWarrior IDE and other Palm OS software development tools. Note that you need all of these elements to use Constructor for Palm OS effectively.

Starting Points

Topics for new Constructor users

If you have not used Constructor before, see the [Chapter 1, “Constructor Fundamentals,”](#) on page 1 for an overview of Constructor for Palm OS applications.

Topics for experienced Constructor users

If you've already used other Constructor for other operating systems or frameworks, refer to these topics to find out how to use Constructor for Palm OS:

- [Chapter 2, “Working with Forms Overview,”](#) on page 29 for information about how to create and edit forms, including dialog boxes and about boxes.
- [Chapter 3, “Working with Menu Bars and Menus Overview,”](#) on page 43 for information about how to create and edit menu bars and menus.
- [Chapter 4, “Working with Character Strings Overview,”](#) on page 53 for information about how to create and edit character strings and help text.
- [Chapter 5, “Working With Alerts Overview,”](#) on page 63 for information about how to create and edit alerts.
- [Chapter 6, “Working with Icons and Bitmaps Overview,”](#) on page 71 for information about how to create and edit icons, icon families, bitmaps, and bitmap families.
- [Chapter 7, “Working with Fonts,”](#) on page 99 for information on how to create and use font families.
- [Chapter 8, “Working with Sound Resources,”](#) on page 103 for information on how to add sound resources to your application.
- [Chapter 10, “Menu Reference,”](#) on page 113 for descriptions of menus and menu items in the Constructor for Palm OS application.
- [Appendix A, “Catalog Resources,”](#) on page 125 for detailed descriptions of the attributes you can set for the catalog resources in Constructor for Palm OS.
- [Appendix B, “Palm OS Resource Types,”](#) on page 149 for a list of all the resources used by Palm OS.

Additional Resources

- Documentation

PalmSource publishes its latest versions of this and other documents for Palm OS developers at

<http://www.palmos.com/dev/support/docs/>

- Training

PalmSource and its partners host training classes for Palm OS developers. For topics and schedules, check

<http://www.palmos.com/dev/training>

- Knowledge Base

The Knowledge Base is a fast, web-based database of technical information. Search for frequently asked questions (FAQs), sample code, white papers, and the development documentation at

<http://www.palmos.com/dev/support/kb/>

Constructor Fundamentals

This chapter introduces you to and provides a high-level overview of Constructor for Palm OS® software.

Constructor Fundamentals Overview

Constructor is a visual resource editor used to specify how your application appears to the end user. Using Constructor, you create and edit several types of resources for Palm OS software applications.

The following topics are included in this chapter:

- “[What is a Palm OS Resource?](#)” on page 2, defines resources and how they are used in a Palm OS software application.
- “[Constructor’s Resource Types](#)” on page 2, identifies the resource types you can create and edit using Constructor.
- “[Constructor Project Files](#)” on page 8, files are used to hold the resources you create and edit using Constructor.
- “[Header File Generation](#)” on page 12, is used to create C preprocessor definitions to refer to resources you have created with Constructor.
- “[Project Window](#)” on page 13, defines the display window for a project file.
- “[Editor Windows](#)” on page 21, display the relationship between individual resource editors and the Project window.
- “[Property Inspector Window](#)” on page 22, is a generic display for resource properties.
- “[User Interface Features](#)” on page 24, provides a quick look at the Constructor user interface.

What is a Palm OS Resource?

A Palm OS software *resource* is a data structure that describes the characteristics of user interface element, for example, a form, a dialog box, or a string of text.

A Palm OS software application uses resources to present something to the user. For example, a database application for Palm OS software might have one form for entering and editing records, and another form for displaying a list of records. Each form is stored in the application as a *form* resource, which contains a description of the buttons, text fields, and other elements that make up the form. In addition, the same database application may use other resources to store help text, a default list of category names for the database, bitmap pictures, and an application icon.

All resources are assigned an integer resource ID. For applications, resource IDs should be less than 9999. Resource IDs of 10000 or above are reserved for use by Palm OS.

Constructor's Resource Types

Constructor manages these types of resources for Palm OS software:

- “[Application Resources](#)”
- “[Forms and Controls](#)” on page 3
- “[Menus](#)” on page 4
- “[Character Strings, String Lists, and Category Names](#)” on page 4
- “[Alerts](#)” on page 5
- “[Icons and Bitmaps](#)” on page 5
- “[Font Families](#)” on page 7
- “[Wave Sound](#)” on page 7
- “[Custom Data](#)” on page 7
- “[Undocumented Resource Types](#)” on page 7

Application Resources

Constructor optionally creates three application resources, shown in [Table 1.1](#).

Table 1.1 Application Resources

Resource	Resource Type	Description
Application icon name	'tAIN'	Application name shown under the application icon. This string cannot be blank. Constructor uses the default value <code>Untitled</code>
Version string	'tver'	Version information displayed by the Info dialog box. This string cannot be blank. Constructor uses the default value <code>1.0</code>
Default application category	'taic'	The category used for the application. Example: Games If this string is blank, then the 'taic' resource will not be generated.

Generation of these application resources is dependent on the **Generate App Resources** option. See “[Generate App Resources option](#)” on page 17, for more information.

Forms and Controls

A typical Palm OS software application has one or more forms. A form is a *window* that contains user interface controls that can only be displayed in a form. These user interface controls are also stored as resources to which a form resource refers. Constructor creates these control resources when you add them to a form.

You use Constructor to create, design, and modify form resources (a resource of type 'tFORM') interactively in a WYSIWYG (What You See Is What You Get) environment. Constructor displays your forms

almost exactly as they will appear on an actual Palm Powered® handheld.

For more information about working with forms, see “[Working with Forms Overview](#)” on page 29.

Menus

A Palm OS application often uses a menu bar with menus. Menu bars and menus are displayed across the top of the Palm Powered handheld when you tap the Menu icon in the input area. You use Constructor to create and edit resources for menu bars and menu items.

Palm OS uses a single, integrated resource type (a resource of type 'MBAR') for menu bars and menu items.

For more information about working with menu bars and menus, see “[Working with Menu Bars and Menus Overview](#)” on page 43.

Character Strings, String Lists, and Category Names

This topic describes the character string resources that you can create and edit with Constructor. For more information about working with character strings, string lists, and application information string lists, see “[Working with Character Strings Overview](#)” on page 53.

Character string resource

A character string resource (a resource of type 'tSTR') holds a character string. Use a character string resource to hold a piece of text like a non-editable text caption in a form or help text.

Use Constructor's string editor to create and edit character string resources.

Character string list resource

A character string list resource (a resource of type 'tSTL') holds a group of character strings. Use a string list resource to hold a group of related strings like country names for a list in a form. To access a

string within a string list resource, use the `SysStringByIndex()` routine in Palm OS.

Use Constructor's string list editor to create and edit character string list resources.

Category name list resource

An application information string list resource (a resource of type `'tAIS'`) holds a group of default category names for a database. Use this resource when calling `CategoryInitialize()`.

Use Constructor's App Info String List editor to create and edit application information string resources.

NOTE: On Palm OS 3.5 and earlier, an error may occur if the default category names list contains fewer than 16 strings. The strings may be blank, but there should be exactly 16 strings.

Alerts

A Palm OS application uses an alert to display information that the user must address immediately. An alert is a modal dialog box with an icon, text, and one or more buttons. Dialog boxes in Palm OS are modal, that is, a dialog box appears in front of other forms on the screen, forcing the user to interact before continuing.

Use Constructor to create and manage alert resources (`'tAlt'`).

For more information about working with alerts, see “[Working With Alerts Overview](#)” on page 63.

Icons and Bitmaps

Icons and bitmap images are often used in Palm OS software applications. You use Constructor's bitmap tools to draw and manage icons and bitmaps:

- Application Icon Bitmap (a resource of type `'tAIB'`)

This bitmap is used by the Palm OS Launcher to display the icon for an application. You use the App Icon Family editor to create application icon bitmaps.

Constructor Fundamentals

Constructor's Resource Types

- Logical Bitmap Image (a resource of type 'Tbmp')

A logical bitmap image, which is also called a *bitmap family*, can contain multiple versions of the same image, each stored at a different pixel bit-depths and/or density levels.

Bit-depths supported:

- 1-bit, black and white
- 2-bit, 4-level grayscale
- 4-bit, 16-level grayscale
- 8-bit, 256 colors
- 16-bit, thousands of colors

Density levels supported:

- Single
- One and one half
- Double

Each version of the bitmap must have the same dimensions. When an application loads the resource and draws it on the screen, Palm OS will pick the best image from the bit-depths and/or density levels available. For example, if the handheld has a 4-bit gray screen and there is a 4-bit image available in the bitmap family, then Palm OS will display the 4-bit image.

You use the Bitmap Family editor to create logical bitmap image resources.

For more information about working with icons and bitmaps, see [“Working with Icons and Bitmaps Overview”](#) on page 71.

NOTE: The 2-bit (4-level grayscale) icons and bitmaps are supported in Palm OS 3.0 and later. The 4-bit (16-level grayscale) and 8-bit color icons and bitmaps are supported in Palm OS 3.5 and later. The 16-bit color icons and bitmaps are supported in Palm OS 4.0 and later.

Font Families

Palm Powered handhelds support standard displays, QVGA displays, and double-density displays. In order to support all of these display types, your application can define fonts with different densities.

Although Constructor does not provide an editor for creating fonts, it does provide support for font families, which allow you to more easily manage multiple font densities. For more information about working with font families, see [Chapter 7](#), “[Working with Fonts](#),” on page 99.

Wave Sound

Constructor allows you to add wave files as resources in your application. For more information about working with sound resources, see [Chapter 8](#), “[Working with Sound Resources](#),” on page 103.

Custom Data

Constructor supports a custom data resource to allow you to create resource types that are not already defined. For more information about using custom data resources, see [Chapter 9](#), “[Working with Custom Data Resources](#),” on page 107.

Undocumented Resource Types

The Create New Resources dialog box includes the following resources in the **Resource Type** selection menu:

- Soft Constant
- Short Integer List
- Locale Setting
- Overlay
- Feature
- Bootscreen Bitmap Family

These resources are undocumented, and are intended for PalmSource and PalmSource licensee use.

Constructor Project Files

This section discusses how Constructor stores information in files, and how to work with those files, including:

- “[Managing Files with Constructor](#)” on page 8
- “[Editing PRC Files](#)” on page 9
- “[Understanding the Resource.frk Folder](#)” on page 9
- “[Moving a Constructor File Between Platforms](#)” on page 10

Managing Files with Constructor

Constructor works on resources within Constructor *project* files. To incorporate your resources into a Palm OS application, you add your Constructor project file to the project file you created in the CodeWarrior IDE.

Constructor recognizes and edits many (but not all) resource types used in Palm OS software development, including PRC, OPRC, and BPRC files. Non-Constructor resource types are retained in the project file but are ignored by Constructor.

Table 1.2 Managing a Constructor Project File

To	Use This Menu Item
Create a new project file	Choose File > New Project File
Open a project file	Choose File > Open Project File...
Save a project file	Choose File > Save
Save a project file with a different name or in a different location	Choose File > Save As...
Close a project file	Choose File > Close

The ability to save changes is affected by the status of the file. If a file is locked or read-only, you cannot save changes. If the file has a modified read-only status from a version control system, you may not be able to check the file back into the system.

Editing PRC Files

To open a PRC, OPRC, or BPRC file in Constructor, choose **File > Open Project File...**. Constructor opens a Project Window and lists the resources contained in the PRC file, with the following considerations:

- Bitmap resources are always converted into bitmap families and bitmaps. Application icons are always converted into application icon families and bitmaps, even if they were originally created using the Icon editor or the Multibit Icon editor.

The bitmap or application icon families will always be numbered with the appropriate resource IDs, but the bitmaps themselves may be given unique IDs to avoid conflicts. For more information, see “[Icons in Palm OS 3.5 and Later](#)” on page 77.

- Constructor inspects character strings to determine the appropriate Palm OS Target option setting for the PRC. If the character strings do not have characters that indicate a specific Palm OS Target option, then Constructor uses the default setting **Palm OS 3.1 or later**. For more information see, “[Palm OS Target option](#)” on page 19.
- If the PRC contains Constructor resources that are not specific to the Palm OS, Constructor may discard these resource.
- Form object flags that were not set in the original PRC are set to the Constructor defaults.

You can move, copy, and delete resources in PRC files. You can also copy and move resources between PRC files and RSRC files. For more information, see “[Moving, Copying, and Deleting](#)” on page 25.

Understanding the Resource.frk Folder

Constructor works with resources. On the Macintosh, resource information is stored in the resource fork. In Windows, resource forks do not exist, so Constructor stores all the resource data in a regular Windows file with the extension RSRC.

However, the Macintosh OS duality between data fork and resource fork affects how Constructor works on Windows. When you create and save a new Constructor project file, Constructor creates the file and (if it doesn't already exist) a folder named `Resource.frk`. In the `Resource.frk` folder, a file with the same name as the file you created is present—the file you created represents the data fork of the file, and the file of the same name in the `Resource.frk` folder represents the resource fork of the file. When you save a file, Constructor completes the following:

- Creates an empty file in the folder you choose.
- Creates a `Resource.frk` folder if necessary.
- Creates another file with the same name inside the `Resource.frk` directory to hold resource information.

It is important to understand that there are two aspects to each Constructor file: the data aspect, which is a single file, and the resource aspect, which is also a single file. Although these files have the same name, the resource file is and *must be* in the `Resource.frk` folder in order for Constructor to work properly with your data and to move files from one platform to another. See “[Moving a Constructor File Between Platforms](#)” for more information.

Note that PRC files are stored as simple data files, and not as resource files. As a result, PRC files can be moved between platforms without concern about resource aspects.

Moving a Constructor File Between Platforms

To successfully move a file between the Macintosh and Windows platforms (and vice versa) you must consider how Constructor handles the differences between the Macintosh and Windows operating systems. You may move files via a network connection or disk.

NOTE: Before transferring files between Macintosh and Windows on a disk, make sure that PC Exchange software is running on your Macintosh. The PC Exchange system extension is required in order to read MS-DOS-formatted diskettes.

Moving a File from Macintosh to Windows

To transfer a file over a network:

When transferring a file from Macintosh to Windows over a network, only the Macintosh resource file is transferred to the Windows system (because the data fork of the Macintosh Constructor project file is empty). Once the file has been transferred, add an RSRC extension to the transferred file and place it in a `Resource.frk` folder. Next, create a “phantom” file of the same name and place it outside the `Resource.frk` folder. This empty file represents the “data fork” of the file. This is the file you open from Constructor on a Windows machine.

To transfer a file via disk:

Copy the file from the Macintosh computer onto an MS-DOS-formatted disk; the Macintosh OS automatically splits the file and stores the resource fork in an invisible directory named `Resource.frk`, and the data fork of the file appears on the disk.

When you open the disk in Windows, both the actual file (data part) and the `Resource.frk` directory appear. Copy both parts of the file to the final destination on the Windows system—treat the data part of the file as the regular file, and move the resource part of the file into a `Resource.frk` directory.

Moving a File from Windows to Macintosh

Before you move a file from Windows to Macintosh, make sure that the PC Exchange control panel is installed on the Macintosh system.

To transfer a file via disk:

On the Windows system, make sure that a `Resource.frk` folder appears at the top level of the disk (this is a requirement of the Macintosh file system); then copy both parts of the file you want to move to the disk. Place the data part of the file on the top level of the disk, and place the resource part inside the `Resource.frk` folder.

Note that when you open the disk on the Macintosh, two issues occur, which will be automatically handled by PC Exchange: the `Resource.frk` folder is invisible, and the file is in two pieces

(rather than a single file with two forks as is expected on Macintosh OS).

Drag the visible `RSRC` file from the Windows disk onto the Macintosh desktop; then launch Constructor and open the file. (You cannot double-click a file from the desktop to open it in Constructor; you must launch Constructor and open the file within the program.)

Header File Generation

Constructor optionally creates a C header file containing preprocessor definitions for the resources defined in your Constructor for Palm OS project file. You may include this header file in the C source code files in your Palm OS application's project.

To generate a C header file for a Constructor for Palm OS project:

1. Open the Constructor project file.

See “[Managing Files with Constructor](#)” on page 8.

2. Turn on header file generation.

Select the [Auto Generate Header File option](#) in the [Project Settings](#) section of the Constructor project file's window. You may have to expand this part of the window to see the Project Settings section. See “[Project Window](#)” on page 13, for more information.

3. Name the header file.

By default Constructor names the header file after the project file and adds `_res.h`. To specify a different name, click the text field of the [Header file name option](#) entry in the [Project Settings](#) section of the Constructor project file's window; then enter a name. If you do not add a header filename extension, Constructor adds `_res.h`.

4. Set header file options.

For descriptive comments to appear in the generated header file, select the [Include Details in header option](#) in the Project Settings section of the project file window.

Now, each time you save changes to the Constructor project file, Constructor updates the C header file associated with it. If the header file doesn't exist in the same folder as the Constructor project file, Constructor creates a new one.

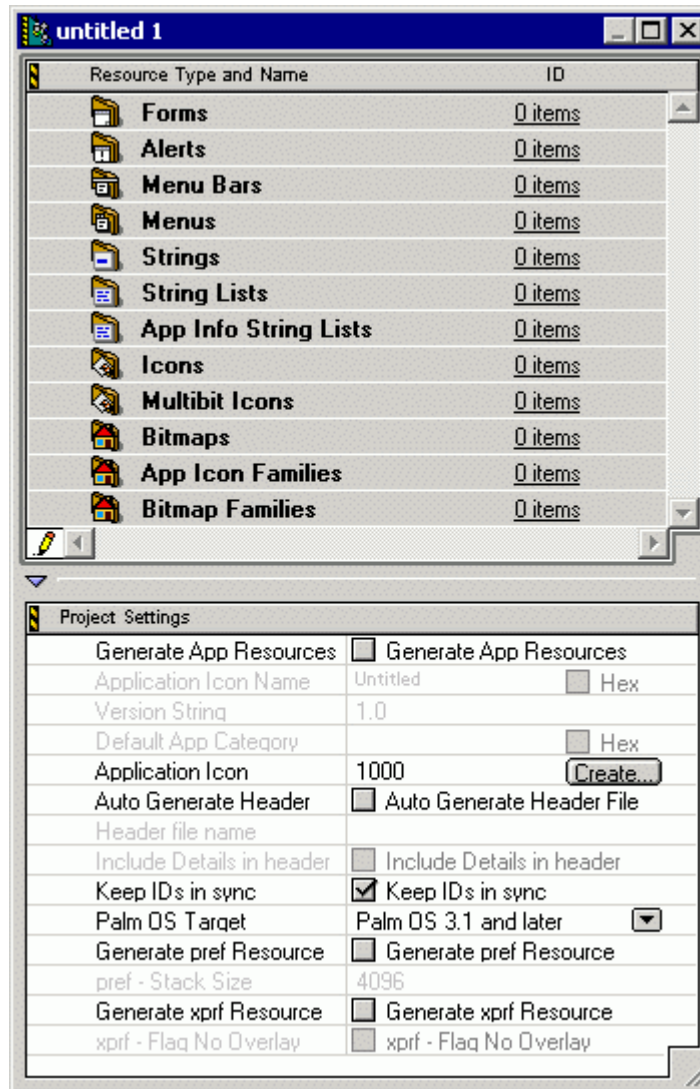
Related Information

See “[Save](#)” on page 114, and “[Generate Header File](#)” on page 115.

Project Window

The Project window is used to create and manage resources. When you open a Constructor project file, the Project window lists each type of resource and the individual resources available for each type.

Figure 1.1 A Constructor for Palm OS Project Window



The Project window is organized into the following sections:

- “[Resource Type and Name List](#)” on page 14
- “[File Control Status](#)” on page 15
- “[Project Settings](#)” on page 16

Resource Type and Name List

Each gray bar in the Project window reflects a resource type. The following types of resources are available:

- Forms
- Alerts
- Menu Bars
- Menus
- String Lists, Strings, App Info String Lists
- Alerts
- Icons, Multibit Icons
- Bitmaps
- Application Icon Families, Bitmap Families

Once you have created at least one resource, a triangle appears to the left of the resource heading. Click the triangle to expand or collapse the list of resources.

When a resource has changed but has not been saved, a dot appears to the left of that resource, as shown in [Figure 1.2](#).

Figure 1.2 A Changed Resource



You may have multiple projects open simultaneously, and copy resources from one to another.

For more information about the resources in a project file, see “[Constructor’s Resource Types](#)” on page 2.

File Control Status

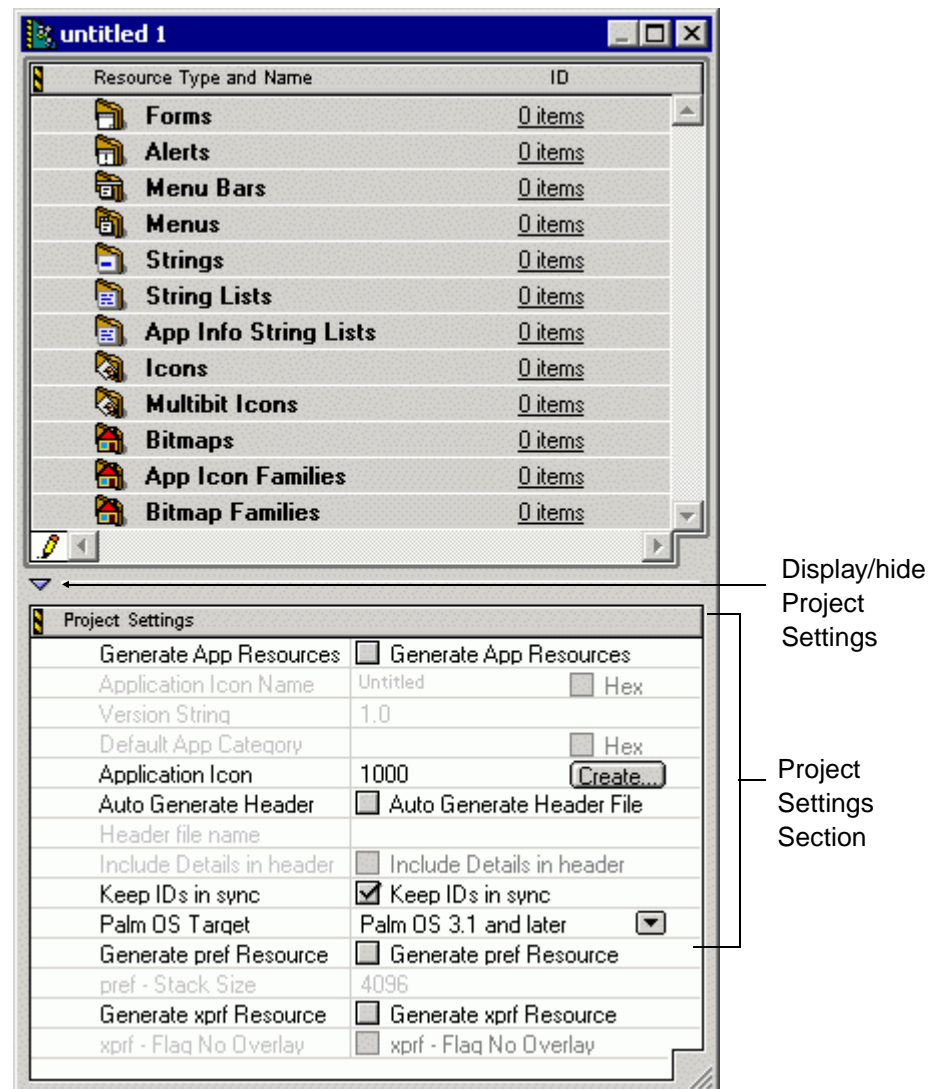
The file privileges icon in the lower left corner of the Project window indicates the read/write status of the file, usually with respect to a source code control system. The file privileges icon works just like the same icon in the CodeWarrior IDE. For information on this topic, see the *IDE User Guide* section on file privileges.

Project Settings

The Project Settings section of a project window contains options that apply to an entire project.

By default, the Project Settings section is displayed; to hide the Project Settings section, click the small triangle at the left corner above the section.

Figure 1.3 Display/Hide Project Settings



Generate App Resources option

When selected, this option instructs Constructor for Palm OS to include a version resource ('tver'), an application icon name resource ('tAIN'), and a default application category resource ('taic') in the project's resources. Note that this option is cleared by default.

When cleared, the [Application Icon Name option](#) and [Version String option](#) items are dimmed; Constructor does not generate a version resource ('tver') or an application icon category ('taic') resource.

Application Icon Name option

Contains the text to place in the project's application icon name resource ('tAIN'). Constructor places the text in this field into a application icon name resource. This field is limited to 31 characters.

If an application has a 'tAIN' resource, Palm OS software displays the text in the resource under the application icon when the user taps the Applications icon in the input area. If an application doesn't have this resource, the application's filename is used.

The Application Icon Name option is dimmed if [Generate App Resources option](#) is not selected.

Version String option

Contains the text to place in the project's version resource ('tver'). The format of the text in this field should be:

majorVersion.minorVersion

where *majorVersion* represents the major version number and *minorVersion* represents a minor version number. These two numbers are separated by a period (.). This field is limited to 15 characters.

This option is dimmed if [Generate App Resources option](#) is cleared.

Default App Category option

Specifies a default application category for an application when it is installed on a handheld. The default application category is stored

in an application icon category resource ('taic'). If you do not make an entry in the **Default App Category** field, the application is assigned to the **Unfiled** application category by default.

Application Icon option

Specifies the icon resource ID of the icon that represents the application when the user taps the Applications icon in the input area.

The resource ID 1000 is defined as the Large Application Icon Bitmap and should be 22 pixels wide and 22 pixels tall. The resource ID 1001 is defined as the Small Application Icon Bitmap and should be 15 pixels wide and 9 pixels tall. You should use the App Icon Family editor to make resources 1000 and 1001.

If the resource ID that you enter doesn't match the ID of an icon resource in the project, a **Create** button appears next to the **Application Icon** field. Click **Create** to draw a new icon for the application. After creating the new icon, its resource ID appears in the **Application Icon** field.

If the resource ID that you enter matches the ID of a icon resource already in the project, an **Edit** button appears next to the **Application Icon** field. Click **Edit** to open the icon resource in a bitmap editor window.

Entering an ID of 0 specifies that the application does not have an icon.

Resource IDs should be less than 9999. Resource IDs of 10000 or above are reserved for use by Palm OS.

See also "[Working with Icons and Bitmaps Overview](#)" on page 71.

Auto Generate Header File option

When selected, this option instructs Constructor to generate a C header file (an H file) that contains preprocessor definitions of the resources in the Constructor project file whenever the project file is saved. Use the [Header file name option](#) and [Include Details in header option](#) to customize the header file.

To generate a header file when **Auto Generate Header File** is cleared, choose **File > [Generate Header File](#)**.

Header file name option

Specifies the filename of the C header file that Constructor for Palm OS generates. If left empty, Constructor fills this field with the name of the project file with `_res.h` added to the filename.

If the filename doesn't have a header filename extension, Constructor adds `_res.h` to the filename.

This option is dimmed if [Auto Generate Header File option](#) is cleared.

Include Details in header option

Instructs Constructor to add descriptive comments for the items listed in the C header file.

This option is dimmed if [Auto Generate Header File option](#) is cleared.

Keep IDs in sync option

Instructs Constructor to maintain resource ID consistency. When this option is selected, Constructor:

- keeps form resource IDs and internal form IDs consistent
- renumbers object IDs in a form when the form ID changes

See also “[Adding Interface Items to a Form](#)” on page 38.

Palm OS Target option

The **Palm OS Target** option is used to select a target operating system. These options are available:

- Palm OS 3.0.x and earlier. Select this option for 3.0.x or earlier systems.
- Palm OS 3.1 and later. Select this option for Latin-encoded Palm OS 3.1 and later systems.
- Palm OS for Japan: Choosing the **Palm OS for Japan** option displays the input area that appears on Japanese Palm

Powered handhelds. Note that this input area is only for preview display purposes.

The option also sets the text encoding and fonts to be used in resources that contain text, such as forms, alerts, menus, and strings.

Note that you can enter Japanese text in Constructor if you are using a Japanese system or if your computer has a Japanese Language Kit (Macintosh only).

Note that if you enter Japanese text and later select another Palm OS target, the text is not translated into Latin characters. You must reenter the desired text once you select another option.

- **Palm OS Chinese (Simp.):** Choosing the **Palm OS Chinese (Simp.)** option displays the input area that appears on Simplified Chinese Palm Powered handhelds. Note that this input area is only for preview display purposes.

The option also sets the text encoding and fonts to be used in resources that contain text, such as forms, alerts, menus, and strings.

Generate Pref Resource option

When selected, instructs Constructor to generate the `pref` resource.

Pref - Stack Size option

Specifies the requested stack size for your application. For Palm OS 3.0 and later, when the operating system launches your application, it will attempt to allocate at least this much memory for the stack. If this option is not specified, Palm OS will use a default value of 3 . 25 K. (This default stack size is determined by the Palm OS and is subject to change.)

This option is dimmed if [Generate Pref Resource option](#) is cleared.

NOTE: The Palm OS may allocate more than the requested amount to allow for growth in OS-owned stack use. Most applications rarely use all of the default stack size, so increase your application's stack only if you are running out of space. Consider decreasing the stack size to make more room in the dynamic heap for other dynamic memory requests.

Generate XPRF Resource option

When selected, instructs Constructor to generate the `xprf` resource.

XPRF - Flag No Overlay option

When selected, instructs Constructor to generate the `xprf` resource with no overlays.

If the **Flag No Overlay** option is set, then Palm OS will no longer automatically open overlays for the application. You may want to use this option to prevent others from building a localized version of your application.

This option is dimmed if [Generate XPRF Resource option](#) is cleared.

Editor Windows

Each resource type has a resource editor. To open an individual resource for editing, use one of the following techniques:

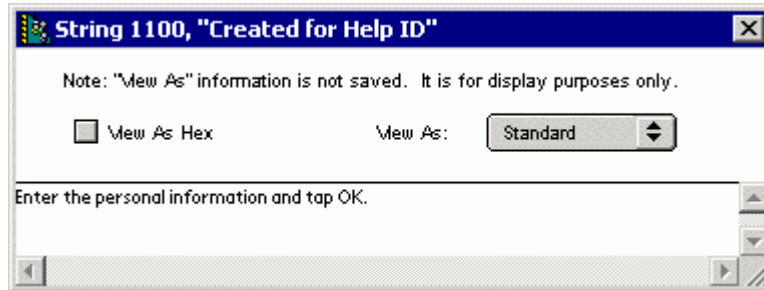
- Double-click the resource in the project window.
- Select the resource(s) and press the RETURN or ENTER key.
- Select the resource(s) and choose **Edit** > [Edit Resource](#).

Constructor opens an editor window that allows you to modify the resource. Each kind of resource has a different editor window. [Figure 1.4](#) shows the character string editor window.

Constructor Fundamentals

Property Inspector Window

Figure 1.4 Character String Editor Window



TIP: Macintosh: To save all windows in Constructor, hold down **OPTION** and choose **File > Save**.

Windows: To save all windows in Constructor, choose **Save All** from the **File** menu.

See these topics for information about various types of resource editors:

- “[Form Layout Window](#)” on page 32
- “[Menu Bar Editor Window](#)” on page 45
- “[String Editor Window](#)” on page 54
- “[Alert Layout Window](#)” on page 63
- “[Bitmap Editor Window](#)” on page 82

Property Inspector Window

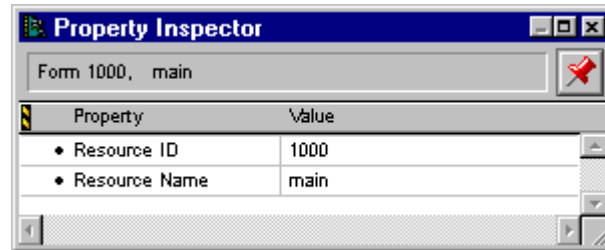
To open the Property Inspector Window, choose **Window > [Property Inspector](#)**.

The Property Inspector window displays attributes of a selected item, which may be:

- A resource in the [Project Window](#)
- A pane in a Layout Window or Hierarchy Window
- A menu or menu item in the Menu Bar Editor Window

When you select one of these items, its properties appear and can be edited in the Property Inspector Window. [Figure 1.5](#) shows an example of this window displaying properties for a resource.

Figure 1.5 Resource Properties



The top of the window identifies the item whose properties are displayed.

The Property Inspector Window remains synchronized with whatever other view is open. If you modify the data in the Property Inspector, all other views on the same data update simultaneously. If you modify the data in some other view, for example, the [Project Window](#), the Property Inspector updates its data at the same time.

To archive the content of the Property Inspector window for a specific view (to make sure its contents don't change when you switch from one view to another) you can "pin down" the content by clicking the push pin icon in the upper-right corner of the Property Inspector window. After clicking the push pin, the Property Inspector window shows only the information for its associated view. To see a Property Inspector window for a different view, choose the desired view; then choose **Window > Property Inspector**. A new Property Inspector window appears for the different view.

For a discussion of how the Property Inspector works with various items in Constructor, see the topics listed in [Table 1.3](#).

Table 1.3 Using the Property Inspector Window

To use the Property Inspector with	See this topic
Forms	“Setting Form Properties” on page 36
Menu bars and menus	“Setting Menu Bar Properties” on page 49
Character strings and help text	“Modifying a Character String” on page 61
Alerts	“Setting Alert Properties” on page 65

User Interface Features

Constructor has many of the standard user interface features. These include features that support:

- [Selection](#), for selecting items.
- [Moving, Copying, and Deleting](#), for relocating, duplicating, and removing items.
- [Lists](#), for working with an ordered group of items.
- [Text Editing](#), for modifying text.
- [Undo](#), for restoring items to a previous state.

The figures in this section use the Project window to illustrate the interface. The same techniques work in individual editor windows, when the technique is appropriate for the kind of data displayed in the window. Interface features are not available in all windows under all circumstances.

Selection

Constructor follows the standard behavior with respect to making selections. As a general rule, you must first select an item or items, and then perform an operation on the selection.

For selecting or canceling selection, use any of the following methods:

- To select an item, click the item.
- To extend the current selection, press SHIFT and click.
- To select contiguous items in one movement, drag the mouse. As you drag, all items that intersect are selected, and remain selected when you release the mouse. Note that for marquee selection to work properly, you must begin dragging in an empty area of the window. If you drag on an item, you move or copy the item instead of starting a marquee selection.
- To select all items in a window, choose **Edit** > [Select All](#).
- To cancel selection, click outside of the selection.

TIP: Macintosh: For a layout window, dragging in a top-level view (and not inside an individual pane) starts a marquee selection. If you drag inside an individual pane, you select the pane and then move or resize it. Hold down the CONTROL key when you begin a drag inside a pane to prevent moving or resizing the pane.

Windows: For a layout window, dragging in a top-level view (and not inside an individual pane) starts a marquee selection. If you drag inside an individual pane, you select the pane and then move or resize it.

Moving, Copying, and Deleting

To move or copy an item, drag the item. Dragging an item to a different location in the same window moves the item; dragging an item from one window to another window copies the item.

TIP: Macintosh: To copy an item within the same window, hold down **OPTION** as you drag or when you drop the item.

Windows: To copy an item within the same window, hold down **ALT** as you drag or when you drop the item.

Drag and drop also works in the bitmap editor, in both the Editing View and the Sample Views.

You can drop bitmaps onto either the Editing View or the Sample Views. If you drop on the Editing view, the original bitmap is scaled to fit. If you drop on the editing view, the bitmap does not scale.

In addition to drag and drop, the standard [Cut](#), [Copy](#), and [Paste](#) commands in the Edit menu work in the traditional manner at all appropriate times. You may also choose the **Edit > [Duplicate](#)** command to make a copy of an item in the same window.

When copying, duplicating, moving, or deleting a pane that contains other panes, your edits affect the container and its contents.

TIP: Macintosh: You can drag from the Sample Views to other windows in Constructor and applications that support drag and drop, including the Finder desktop and the Scrapbook.

TIP: Macintosh: To delete an item, select it. Then press the **DELETE** key. You can also use the **Edit > [Clear](#)** command from the menu.

Windows: To delete an item, select it; then press the **BACKSPACE** key. You can also use the **Edit > [Clear](#)** command.

Lists

Some Constructor windows, like the Project window, display lists of information. You can navigate through lists of items using the arrow keys. Each time you press a key, the next or previous item in a list is

selected. To extend a selection, hold down **SHIFT**, and then press an arrow key.

Constructor displays triangle icons to open and close lists. Click a triangle to expand or collapse an associated list.

You can extend a selection in a list by holding down **SHIFT** and then pressing an arrow key.

Text Editing

Many Constructor windows contain text entries, for example, the Project window displays a list of resource names.

To activate a text item for editing, click the item; then move the cursor away from the text, or wait a few seconds. The text becomes editable when a frame appears around the text or when the text insertion cursor appears within the item.

You navigate through text using the arrow keys. You select text by dragging through text; double-click a word to select the word.

To move from one item to the next in a series of text fields, use the **TAB** key.

Undo

Constructor provides single-level undo. Most operations can be undone (and subsequently redone) using the **Edit > Undo/Redo** command.

If the undo is unavailable after a given action, the Edit menu displays **Can't Undo**.

Constructor Fundamentals

User Interface Features

Working with Forms

This chapter provides information about modifying form resources using Constructor for Palm OS® software.

Working with Forms Overview

Working with forms comprises two concepts: the Constructor windows used to edit a form, and the tasks you perform on a form and its contents.

The following topics are included in this chapter:

- “[Windows for Forms](#)” - Windows you see while working on form resources.
- “[Editing Forms](#)” on page 34 - Tasks you perform on a form and its contents.

Windows for Forms

This section discusses the Constructor windows you use to create and edit forms and their contents. The following windows are used:

- “[Catalog Window](#)” - The window containing the user interface items you can add to a form.
- “[Form Layout Window](#)” on page 32 - The editor window that shows a view of a Palm OS form.
- “[Hierarchy Window](#)” on page 33 - The window that displays a list of the items within the form.

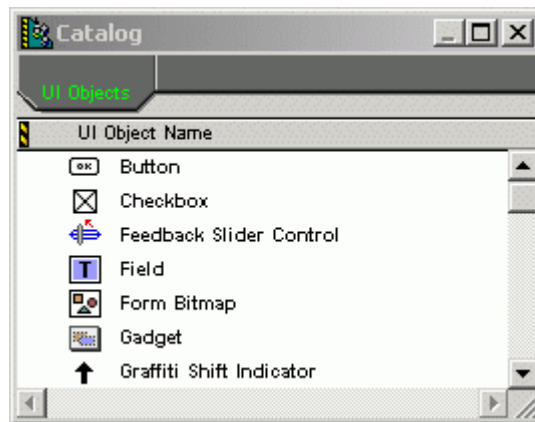
Catalog Window

The Catalog window lists the user interface items you can place in a form.

To open the Catalog window:

Choose [Window](#) > [Catalog](#).

Figure 2.1 The Catalog Window for Constructor for Palm OS



The Catalog window contains the items listed in [Table 2.1](#). For a description of all the user interface items available for Palm OS, see *Palm OS Programmer's Companion*.

Table 2.1 User Interface Items in the Catalog Window for Forms




















User Interface Item	Purpose or Behavior
 Button	Triggers an action after the item is depressed then released within the button's bounds.
 Checkbox	Toggles on and off when tapped. To specify that two or more check boxes are in the same group, select them and then choose Arrange > Make Object Group . For more information, see “ Grouping Items in a Form ” on page 42.
 Feedback Slider Control	Displays a data value within a range of values.
 Field	Accepts and displays user data entry in the form of text.
 Form Bitmap	Acts as a placeholder for a bitmap resource.

Table 2.1 User Interface Items in the Catalog Window for Forms (continued)

User Interface Item	Purpose or Behavior
 Gadget	Acts as a placeholder for an application-defined user interface item.
 Graffiti Shift Indicator	Displays the shift indicator for punctuation, symbol, uppercase shift, and uppercase lock.
 Graphic Button	Allows a graphic image to act as a button.
 Graphic Push Button	Allows a graphic image to act as a push button. To specify that two or more graphic push buttons are in the same group, select them and then choose Arrange > Make Object Group . For more information, see “ Grouping Items in a Form ” on page 42.
 Graphic Repeating Button	Allows a graphic image to act as a repeating button.
 Label	Displays non-editable text.
 List	Presents a box to the user containing a list of choices.
 Popup Trigger	Shows the selection from a popup list and lets the user display an associated popup to change the selection.
 Push Button	Allows the user to choose only one push button within a group of push buttons. To specify that two or more push buttons are in the same group, select them and then choose Arrange > Make Object Group . For more information, see “ Grouping Items in a Form ” on page 42.
 Repeating Button	Triggers a (typically) continuous action only while the item is depressed.
 Scrollbar	Allows the user to control which portion of a list or table is displayed. This item is not available when creating forms for Palm OS Version 1.

**Table 2.1 User Interface Items in the Catalog Window
for Forms (*continued*)**

User Interface Item	Purpose or Behavior
 Selector Trigger	Shows a setting and lets the user display an associated dialog box to change the setting. For more information on dialog boxes see “ Creating a Dialog Box ” on page 35.
 Slider Control	Allows the user to specify a data value within a range of values.
 Table	Displays a 2-dimensional array of information.

For information about adding items from the Catalog window to a form, see “[Adding Interface Items to a Form](#)” on page 38.

Form Layout Window

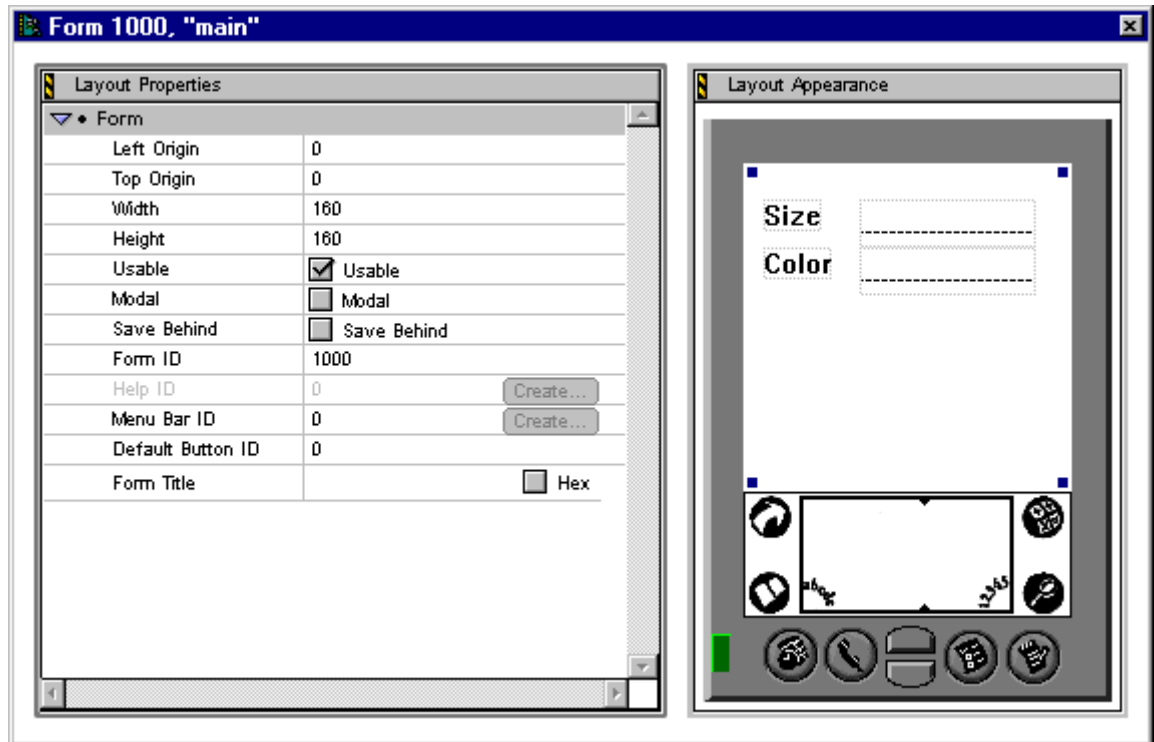
The Layout window is where you do most of your visual work in Constructor. This is the window in which you arrange user interface items in a form.

To open a form in a Layout window:

Do one of the following:

- Double-click a form resource in the Project window.
- Select a form resource in the Project window and press RETURN or ENTER.

Figure 2.2 The Form Layout Window



To learn how to create and edit forms, see "[Editing Forms](#)" on page 34.

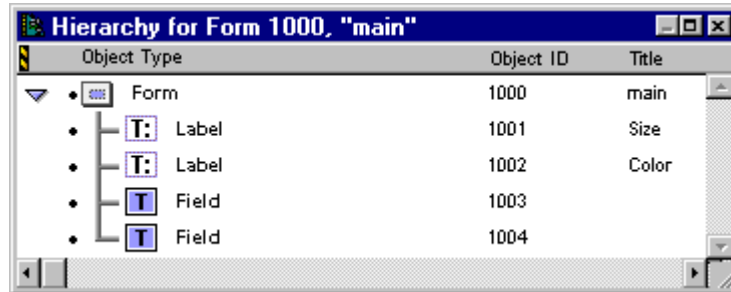
Hierarchy Window

The Hierarchy window provides another way to examine a form; it displays a hierarchical listing of a form and its contents.

To view a form in a hierarchy window:

1. Open the form from the Project window.
2. Choose [Layout](#) > [Show Object Hierarchy](#).

Figure 2.3 The Hierarchy window



Editing Forms

This section shows you how to work with form resources. Use the information in this section to learn about:

- [“Creating a Form”](#) on page 34
- [“Creating an About Box”](#) on page 35
- [“Creating a Dialog Box”](#) on page 35
- [“Setting Form Properties”](#) on page 36
- [“Adding Interface Items to a Form”](#) on page 38
- [“Adding Help to a Form”](#) on page 39
- [“Arranging Items in a Form”](#) on page 40
- [“Grouping Items in a Form”](#) on page 42
- [“Deleting Items in a Form”](#) on page 42

TIP: To prevent runtime problems with forms, make sure the [Keep IDs in sync option](#) is selected in the [Project Settings](#) section of the project window. This option ensures that resource and form IDs stay consistent.

Creating a Form

A Palm OS software application uses a form to hold a group of related interface items manipulated by the user. A Palm OS form is similar to a window on the Macintosh or in Windows.

To create a form:

1. Do one of the following:
 - Select the Forms icon in the Project window; then choose [Edit](#) > **New Form Resource**.
 - Choose [Edit](#) > **New Resource**, choose **Form** from the Resource Type pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to create the form.

An untitled form resource is displayed in the Project window. For more information on using the project window, see “[Resource Type and Name List](#)” on page 14.
2. Press RETURN or ENTER to open the form in a layout window.
See “[Form Layout Window](#)” on page 32, for more information.
3. Add and arrange items in the form.
To learn about adding and arranging items in the new form, see “[Editing Forms](#)” on page 34.

Creating an About Box

An About Box is a form that displays information about a Palm OS application and its author. The type of form used for an About Box in a Palm OS application is usually a dialog box. For more information, see “[Creating a Dialog Box](#)” on page 35.

Creating a Dialog Box

A dialog box is a form that displays information for the user to modify. Dialog boxes in Palm OS are modal, that is, a dialog box appears in front of other forms on the screen, forcing the user to interact before continuing.

To create a dialog box:

1. Do one of the following:
 - Click the Forms icon in the Project window to select it; then choose [Edit](#) > **New Form Resource**.

Working with Forms

Editing Forms

- Choose [Edit](#) > **New Resource**, choose **Form** from the Resource Type pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to create the form.

For more information on using the project window, see “[Resource Type and Name List](#)” on page 14.

2. Press RETURN or ENTER to open the form in a layout window.
See “[Form Layout Window](#)” on page 32, for more information about the layout window.

3. In the Layout window, click the [Modal](#) option to make the form a modal dialog box.

4. Add and arrange items to the dialog box.

To learn about adding and arranging items in the new form, see “[Editing Forms](#)” on page 34.

TIP: For a modal frame to be correctly visible, the values for both Left Origin and Top Origin should be 2 or greater, and the values for both Width and Height should be 156 or less.

Setting Form Properties

Layout Properties determine how a form appears and how it will be used by a user.

For information about the form layout window, see “[Form Layout Window](#)” on page 32.

Left Origin

Specifies, in pixels, the left-most edge of the form.

Top Origin

Specifies, in pixels, the top-most edge of the form.

Width

Specifies, in pixels, the width of the form.

Height

Specifies, in pixels, the height of the form.

Usable

Specifies the visibility of the form, either visible or invisible. If the Usable option is selected, the form is displayed in the Palm OS application; if the option is not selected, the form is not displayed.

See *Palm OS Programmer's Companion* for more information about hiding and showing forms from a Palm OS application.

Modal

Specifies how events outside of the form boundary are interpreted, that is, if a user must interact with the form before continuing with other tasks.

If the Modal option is selected, the form does not respond to taps outside the form (this option is selected for dialog boxes). See [“Creating a Dialog Box”](#) on page 35, for more information.

Save Behind

Specifies how the area behind the form appears on the screen. If Save Behind is selected, the Palm OS saves the area of the screen the form obscures; then restores the area when the form is hidden.

Help ID

Specifies the help text resource ID that appears when the user taps the Help icon in a modal form. This field is only available if the [Modal](#) option is selected.

To select a Help ID option:

- If you enter a resource ID that does not match the ID of a character string resource in the project, a Create button appears next to the Help ID field. Click **Create** to create a new character string.
- If the resource ID you enter matches the IDs of a character string resource in the project, an Edit button appears next to the Help ID field. Click **Edit** to open the character string resource in a string editor window to edit the help text for the form.
- Entering an ID of 0 indicates that the form does not have help text.

See “[Adding Help to a Form](#)” on page 39, and “[Working with Character Strings Overview](#)” on page 53, for related information.

Menu Bar ID

Specifies the resource ID of the menu bar associated with the form.

To select a Menu Bar ID option:

- If you enter a resource ID that does not match the ID of a menu bar in the project, a Create button appears next to the Menu Bar ID field. Click **Create** to create a new menu bar resource.
- If the resource ID you enter matches the IDs of a menu bar resource in the project, an Edit button appears next to the Menu Bar ID field. Click **Edit** to open the menu bar resource in a menu bar editor window.
- Entering an ID of 0 indicates that the form does not have a menu bar.

See “[Adding a Menu Bar to a Form](#)” on page 40, and “[Working with Menu Bars and Menus Overview](#)” on page 43, for related information.

Default Button ID

Specifies the resource ID of the form’s default button. Palm OS software simulates the tapping of this button when the form is dismissed.

Form Title

Specifies the title of the form (displayed at the top of the form).

Adding Interface Items to a Form

An interface item is a visual object that appears in a form and presents information or allows the user to manipulate information. Examples of interface items are buttons, check boxes, and editable text fields.

To add an interface item to a form:

- Copy and paste the item or drag it from another window.
- Drag items from the [Catalog Window](#) onto the form.

For a list of items you may add to a form, see “[Catalog Window](#)” on page 29.

When you add a user interface item to a form, Constructor assigns the item a unique ID number. Note that Constructor does not allow you to change the ID of an item if the ID is already in use by another item in the current form.

NOTE: To keep resource ID numbers of interface items in a form from conflicting with the ID numbers of items in other forms, Constructor imposes a limit of 99 user interface items in a form.

Resource IDs should be less than 9999. Resource IDs of 10000 or above are reserved for use by Palm OS.

For related information on resource ID numbering, see “[Keep IDs in sync option](#)” on page 19.

After you have added items to a form, you can arrange them, group them, or delete them. See the following sections for more information:

- “[Arranging Items in a Form](#)” on page 40
- “[Grouping Items in a Form](#)” on page 42
- “[Deleting Items in a Form](#)” on page 42

Adding Help to a Form

Palm OS software displays a modal form containing help text when a user taps the Help icon in the top right area of a modal form.

Note that non-modal forms cannot use help text. For more information on creating modal forms, see “[Creating a Dialog Box](#)” on page 35.

To add help to a form:

1. Open the form to which you want to add help.
See “[Form Layout Window](#)” on page 32, for more information.
2. Make sure the [Modal](#) option is selected.
When the Modal option is selected, the [Help ID](#) field is displayed.
3. In the [Help ID](#) field, enter a resource ID.
If you’ve already entered a character string to use as help text, enter that character string’s resource ID. Otherwise, enter a unique resource ID.
4. Create or edit the help text.
Click the button that appears next to the [Help ID](#) field. See “[Help ID](#)” on page 37, for more information.

Adding a Menu Bar to a Form

If a form has a menu bar, the Palm OS software displays it when the user taps the Menu icon.

To add a menu bar to a form:

1. Open the form to which you want to add a menu bar.
See “[Form Layout Window](#)” on page 32, for information.
2. In the [Menu Bar ID](#), enter a resource ID.
If you’ve already created a menu bar and menus for this form, enter its resource ID. Otherwise, enter a unique resource ID.
3. Create or edit the menu bar and menus.
Click the button that appears next to the [Menu Bar ID](#) field.
See “[Menu Bar ID](#)” on page 38, for more information.

Arranging Items in a Form

This section provides information about the various ways you can arrange items in a form.

To select an obscured item:

If an item is obscured by another pane, do one of the following:

- **Macintosh:** To select an item hidden by another, hold down COMMAND and click the hidden item. Continue clicking until the item you want is selected. Release the COMMAND key; then hold down OPTION and drag, resize, or double-click the hidden selected item.
- **Windows:** To select an item obscured by another, hold down CTRL and click the hidden item. Continue clicking until the item you want is selected. Release the CTRL key; then hold down ALT and drag, resize, or double-click the hidden selected item.

To move an item:

To move an item, do one of the following:

- **Macintosh:** Begin dragging; then hold down SHIFT, COMMAND, or SHIFT+COMMAND to constrain the movement of the items being moved.
- **Windows:** Begin dragging; then hold down SHIFT, CTRL, or SHIFT+CTRL to constrain the movement of the items being moved.

NOTE: Holding down SHIFT key before you begin dragging changes the selection. Be sure to begin dragging first; then hold down SHIFT.

To nudge an item:

Nudging lets you move an item one pixel at a time. To nudge an item, first select and then use the arrow keys to move the item.

Using a grid

Constructor provides a grid to help you arrange panes.

To display or hide a grid:

1. Open the desired form.
2. Choose [Layout](#) > [Show Grid/Hide Grid](#) to display the grid.

To align items to a grid:

Do one of the following:

- To snap all items to the grid, choose [Layout](#) > [Snap to Grid/Don't Snap to Grid](#).
- To snap only a selected item to the grid, select the item; then hold down CTRL (Windows) or COMMAND (Macintosh) and continue dragging.

To set the grid size:

Choose [Layout](#) > [Edit Grid](#); then enter the desired grid spacing.

To resize an item:

Select the item; then drag a handle at the corner to resize the item.

Grouping Items in a Form

You can assemble push buttons, graphic push buttons, or check boxes into an *object group* so that only one control may be selected at a time within the group.

To create an object group:

- Select all of the items you want in the same group
- Choose [Arrange](#) > [Make Object Group](#)

Use `FrmSetControlGroupSelection` and `FrmGetControlGroupSelection` to work with object groups. See *Palm OS Programmer's API Reference* for more information.

Deleting Items in a Form

To remove an item from a form, you must first select it. Once you've selected the item, do one of the following:

- Press DELETE or BACKSPACE
- Choose [Edit](#) > [Clear](#)
- Choose [Edit](#) > [Cut](#)
- Drag the item onto the trash can on the desktop (Macintosh only)

Working with Menu Bars and Menus

This chapter introduces Menu Bar resources and Menu resources, used to create menu bars and menus for Palm OS® software applications. In addition, information about how to add menu items to Menu resources is provided.

Working with Menu Bars and Menus Overview

Menu bars and menus in Constructor are similar to those in displayed in software applications. In a software application, a menu bar resource defines the list of menu resources to include in the menu bar at runtime; in Constructor, the Menu Bar editor displays a list of menus and lets you modify them.

The following topics are included in this chapter:

- “[Menu Bars and Menus](#)” - How to create Menu Bar and Menu resources.
- “[Editing Menu Bar and Menu Resources](#)” on page 48 - How to edit Menu Bar and Menu resources.

Menu Bars and Menus

This section shows you how to create Menu Bar and Menu resources for your Palm OS software applications.

Creating a Menu Bar Resource

A Palm OS software application uses a menu bar to provide a user with a list of menus. To display a menu bar and its menus, the user taps the Menu icon on a Palm Powered handheld.

Working with Menu Bars and Menus

Menu Bars and Menus

In Constructor, there are two ways to create a Menu Bar resource. You can use either of two Menu Bar resource commands available from the Edit menu.

To create a menu bar resource:

In the Project window, click the Menu Bar resource icon; then use either of the following methods to create a menu bar resource.

Method 1

1. Choose the first [Edit](#) > **New Menu Bar Resource** command.
An untitled Menu Bar resource appears in the Menu Bar resources section in the Project window.
2. Click the untitled Menu Bar resource icon; then choose **Window > Property Inspector**.
3. In the Property Inspector, type a number for the Resource ID or accept the default number provided.
4. In the Resource Name field, type the name of the Menu Bar resource; then close the Property Inspector.

Method 2

1. Choose the second [Edit](#) > **New Menu Bar Resource** command. The Create New Resource dialog box appears.
In the New Resource dialog box, Menu Bar is the selected resource type.
2. In the Resource Name field, enter a name for the Menu Bar resource.
3. In the Resource ID field, enter a number for the Menu Bar resource.
4. Click **Create** to create the Menu Bar resource.

TIP: You can use the Create New Resource dialog box to create any type of resource. Select the desired resource type from the Resource Type pull-down menu; then enter a Resource Name and Resource ID. The resource is automatically added to the correct area of the project window.

Resource IDs should be less than 9999. Resource IDs of 10000 or above are reserved for use by Palm OS.

Menu Bar Editor Window

The Menu Bar editor window displays the collection of menu resources contained in the menu bar resource.

To open the Menu Bar editor window, do one of the following:

- Double-click a menu bar resource in the Project window.
- Select a menu bar resource and press RETURN or ENTER.

The Menu Bar editor window displays:

- Descriptions of menu content, for example, Menu Text and Shortcut.
- The menu bar, corresponding to the appearance of the menu bar in the application.
- The menu command text, corresponding to the appearance of a menu command in the application.
- The shortcut stroke associated with the command.

Figure 3.1 The Menu Bar Editor Window



Creating a Menu Resource

There are two ways to create a Menu resource: from a Menu Bar editor window or from the Project window. When you create a Menu resource from a Menu Bar editor window, it is automatically associated with the selected menu bar; when you create a menu from the Project window, it is not initially affiliated with any menu bar resource.

To create a menu from a menu bar editor window:

1. Open a menu bar resource window using one of the methods described in “[Menu Bar Editor Window](#)” on page 45.
2. Choose **Edit > New Menu**. An untitled menu name appears in the menu bar.
3. Type the name of the menu.
4. Add and arrange menu items in the menu.

For information about adding items to menus and arranging menus, see “[Editing Menu Bar and Menu Resources](#)” on page 48.

To create a menu from the Project window:

Method 1

1. Click the Menu resource icon in the Project window; then choose the first **Edit > New Menu Resource** command.
A new untitled menu appears in the Project window.
2. Using one of the methods described in “[Menu Editor Window](#)” on page 47, open the menu resource window.

Method 2

1. Click the Menu resource icon in the Project window; then choose the second **Edit > New Menu Resource** command.
The Create New Resource dialog box appears, with Menu as the selected resource type.
2. In the Resource Name field, enter a name for the Menu resource.
3. In the Resource ID field, enter a number for the Menu resource.
4. Click **Create** to create the Menu resource.

In addition to creating menus using the Menu Bar editor window and the project menu, you can add existing menus to the Menu Bar editor window.

To add an existing menu resource:

1. Open the Menu Bar resource using one of the methods described in “[Menu Bar Editor Window](#)” on page 45.
2. Drag the desired menu resource from the project window onto the menu bar editor window.

As you drag, a dotted gray line appears in the menu bar to indicate where the menu will be placed when you release the mouse button. You can rearrange the order of menus later. For more information, see “[Arranging Menus](#)” on page 49.

Menu Editor Window

The Menu editor window is nearly identical to the Menu Bar editor window with one exception—the Menu editor window displays a single menu, while the Menu Bar editor can display several menus.

Working with Menu Bars and Menus

Editing Menu Bar and Menu Resources

To open a Menu editor window:

Do one of the following:

- Double-click a Menu resource in the Project window.
- Select a Menu resource and press RETURN.
- Select a Menu resource and press ENTER.

Note that if a Menu Bar editor window is open, all single Menu resources affiliated with the Menu Bar resource are dimmed in the Project window. To display a single menu, close the Menu Bar resource window that contains the multiple menu resources; then open the desired single menu from the Menu resource area of the Project window.

Figure 3.2 The Menu Editor Window



Editing Menu Bar and Menu Resources

This section provides instructions for editing menu bar and menu resources, including:

- [“Setting Menu Bar Properties”](#)
- [“Arranging Menus”](#) on page 49
- [“Deleting Menus”](#) on page 50

Setting Menu Bar Properties

You use the Inspector window to change the properties of a Menu Bar or Menu resource. The Inspector window includes the resource name and the resource ID.

To change the properties of a menu bar resource:

1. Select the Menu Bar resource in the Project window.
2. Choose [Window](#) > [Property Inspector](#).

The Property Inspector window appears with the menu bar resource's ID and name displayed. See "[Property Inspector Window](#)" on page 22, for related information.

3. Enter a Resource ID, a Resource Name, and then close the Property Inspector to update the menu bar resource.

To change the properties of a menu resource:

1. Select the Menu in either the Menu Bar window or in the Project window.

The Property Inspector window appears with the menu bar resource's ID and name displayed. See "[Property Inspector Window](#)" on page 22, for related information.

2. Enter a Resource ID, a Resource Name, and then close the Property Inspector to update the menu resource.

Arranging Menus

You can change the order of menus in the Menu Bar editor window.

To arrange menus in the menu bar:

1. Open the Menu Bar editor window containing the Menu resources you want.
2. Drag the menu names within the menu bar. As you drag, a gray line appears to indicate where the menu will be positioned when you release the mouse button.

Deleting Menus

You can delete unwanted menus from the Project menu or from the menu bar editor window.

To delete a menu from a menu bar:

Do one of the following:

- In the Project window, select the menu you want to delete; then choose [Edit](#) > [Remove Menu](#).
- In the menu bar editor window, click the menu name to select it; then choose **Edit > Remove Menu**.

Either of these methods remove a menu from a menu bar without deleting the menu's resource from the project.

Creating Menu Commands

This section shows you how to create and manage menu commands. All the tasks presented in this section may be performed in either the Menu Bar editor window or the Menu editor window.

NOTE: As a general rule, the Menu Bar editor is more useful than the Menu editor because you can work with several menus. However, the Menu Bar editor does not display submenus, so you must use the Menu editor for submenus, pop-up menus, and so forth.

Read these topics to learn about:

- [“Setting Menu Properties”](#)
- [“Adding a Command to a Menu”](#) on page 51
- [“Modifying a Command in a Menu”](#) on page 52
- [“Arranging Commands in a Menu”](#) on page 52
- [“Deleting Commands from a Menu”](#) on page 52

Setting Menu Properties

Use the Inspector window to change the description and resource ID of a menu resource.

WARNING! Changing a menu's resource ID may corrupt menu bar resources that refer to that menu.

Resource IDs should be less than 9999. Resource IDs of 10000 or above are reserved for use by Palm OS.

To change a menu resource's properties:

1. Select the menu resource in the Project window.
2. Choose [Window](#) > [Property Inspector](#).
The Property Inspector window displays the menu resource's ID and name.
3. In the Property Inspector window, click the value you want to change; then enter the new value.

Adding a Command to a Menu

This section shows you how to add commands to Menu resources.

To add commands to a menu:

1. Open the menu in a menu bar editor window or menu editor window.
See "[Menu Bar Editor Window](#)" on page 45, or "[Menu Editor Window](#)" on page 47, for more information.
2. Select a location for the command by selecting an item in the current Menu bar. The new command will appear after this selection. If there are no items in the menu yet, the new item becomes the first command on the menu.
3. Choose **New Menu Item** or [Edit](#) > [New Separator Item](#) from the menu.
Choosing **New Menu Item** inserts a new untitled menu choice. Choosing [New Separator Item](#) inserts a menu separator, which displays a thin gray line.

4. Type the menu command's name.
5. In addition, you may specify a Graffiti or Graffiti® 2 command stroke for the Menu command. To specify a command stroke, press TAB and type the letter for the command stroke.

Modifying a Command in a Menu

To modify a menu command:

1. Open the menu in a Menu Bar editor window or menu editor window.

See “[Menu Bar Editor Window](#)” on page 45, or “[Menu Editor Window](#)” on page 47, for more information.

2. Click the command you want to change.

To change an item's name, click it; then edit the item. To change the command's Graffiti or Graffiti 2 command stroke, click the item and then edit it.

3. Press the TAB key to edit the next field.

Arranging Commands in a Menu

To move a menu command, drag it within the menu. A small arrow and a flashing horizontal line indicate where the menu item will be inserted when you release the mouse button.

You can also drag menu items from one menu to another in the same window or in different windows.

Deleting Commands from a Menu

To remove one or more menu items from a menu, first select the item or items; then do one of the following:

- Press DELETE
- Press BACKSPACE
- Choose [Edit](#) > **Cut Menu Item** or [Edit](#) > **Clear Menu Item**. Drag the command onto the trash can on the desktop (Macintosh only)

Working with Character Strings

This chapter introduces Character String resources, used for data like help text and default values for text fields.

Working with Character Strings Overview

Strings, stored in a 'tSTR' resource, are a common feature of Palm OS® software applications. Character string resources are used to hold data like help text, default values for text fields, and other information. Each 'tSTR' resource contains a text string.

This chapter provides instructions for how to manage and modify the contents of a character string resource, including:

- “[Windows for Character Strings](#)”
- “[Editing Character Strings](#)” on page 58

Windows for Character Strings

This section discusses the Constructor windows used to create and edit character strings:

- [String Editor Window](#), the window used to enter and edit text and set the display font for the character string.
- [String List Editor Window](#), the window used to enter and edit a string list resource and set the display font.
- [App Info String List Editor Window](#), the window to enter and edit an application information string list resource.
- [Character Map Window](#), the window to enter individual characters for a character string.

String Editor Window

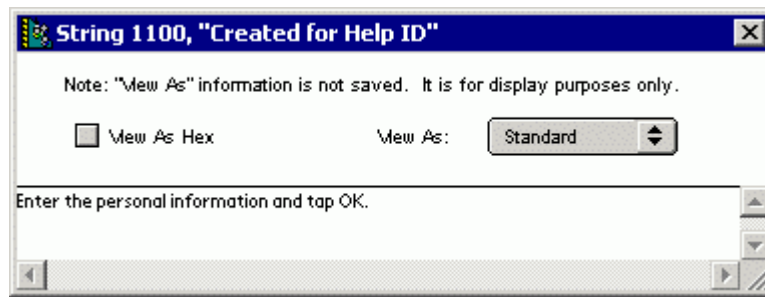
Use the String Editor window to enter and edit a character string resource (' tSTR ') and specify in what font the string will be displayed.

To open a string in a string editor window:

Do one of the following:

- Double-click a string resource in the Project window.
- Select a string resource in the Project window and press ENTER or RETURN.

Figure 4.1 The String Editor Window



To learn how to create and edit character strings, see “[Editing Character Strings](#)” on page 58.

String List Editor Window

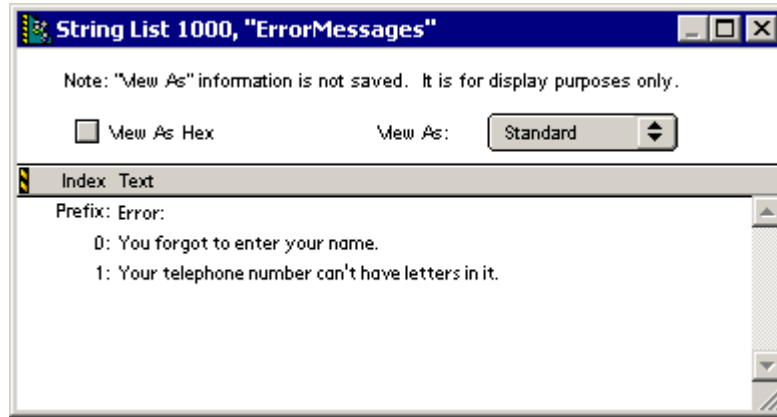
The String List Editor window is used to enter and edit a string list resource (' tSTL '). A string list resource is a related group of strings.

To open a string in a String List editor window:

Do one of the following:

- Double-click a String List resource in the Project window.
- Select a String List resource in the Project window and press RETURN or ENTER.

Figure 4.2 The String List Editor Window



Use the `SysStringByIndex()` routine to access a string within a string list resource. When you access a string in a string list resource, the **Prefix** string is added to the indexed string. For the example where these values are set in the String List Resource editor:

```
Prefix:"Error: "  
0: "You forgot to enter your name."  
1: "Your telephone number can't have letters in it."
```

You could use the following code to access the strings:

```
Char myString[ 60 ];  
SysStringByIndex( MyStringList, 0 /* index */, myString, sizeof(myString) );  
// result: myString = "Error: You forgot to enter your name."  
  
SysStringByIndex( MyStringList, 1 /* index */, myString, sizeof(myString) );  
// result: myString = "Error: Your telephone number can't have letters in it."
```

App Info String List Editor Window

Use the App Info String List Editor window to enter and edit an application information string list resource ('tAIS'). An app info string list resource is often used to hold an application's initial categories in an application's database. Use the `CategoryInitialize()` routine to access an app info string list.

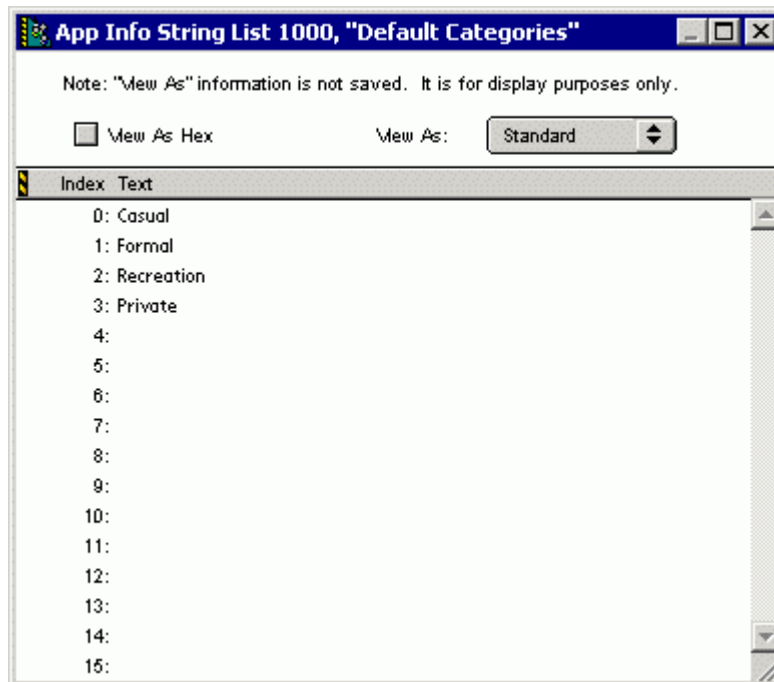
NOTE: On Palm OS 3.5 and earlier, an error may occur if the default category names list does not contain exactly 16 strings. The strings may be blank, but there should be exactly 16 strings.

To open a string in a App Info String List editor window:

Do one of the following:

- Double-click an App Info String List resource in the Project window.
- Select an App Info String List resource in the Project window and press RETURN or ENTER.

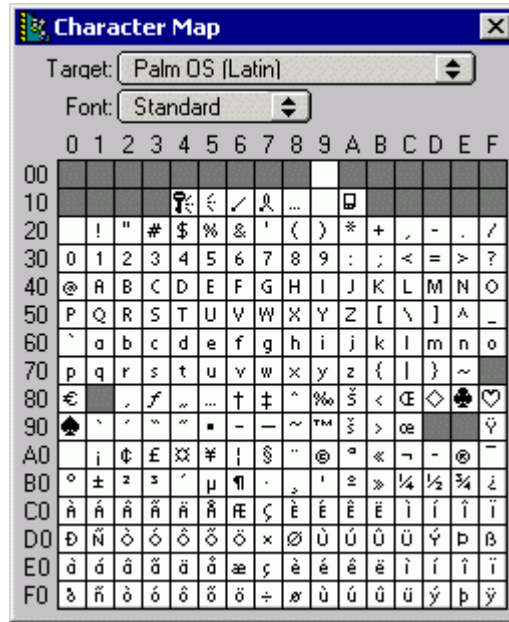
Figure 4.3 The App Info String List Editor Window



Character Map Window

Use the Character Map window to enter individual characters for a character string. The Character Map window displays all of the characters that can be used in a Palm OS interface.

Figure 4.4 Character Map Window



To enter characters from the Character Map window into a character string:

- Open one of the character string edit windows ([String Editor Window](#), [String List Editor Window](#), or [App Info String List Editor Window](#)) to edit a character string.
- Open the Character Map window by choosing **Window > Character Map**.
- Double-click the character you want to enter into the character string.
 - On Windows, the character is copied to the Clipboard. Make the string edit window the active window, and paste the character into the string edit window using **Edit > Paste**.
 - On Macintosh, the character is written directly to the string that you are editing in the string edit window.

Editing Character Strings

This section shows you how to work with text in Strings, String Lists, and App Info String Lists resources, including:

- “[Creating a Character String](#)”
- “[Creating a Character String List](#)” on page 59
- “[Creating a Category String List](#)” on page 59
- “[Creating Help Text](#)” on page 60
- “[Modifying a Character String](#)” on page 61
- “[Deleting Character Strings](#)” on page 62

Creating a Character String

To create a character string resource:

1. Do one of the following:
 - Select the Strings icon in the Project window; then choose **Edit** > **New String Resource**. A new, untitled Strings resource appears in the Project window.
 - From anywhere in the Project window, choose **Edit** > **New Resource**. Select **String** from the **Resource Type** pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to add the resource to the Strings section of the Project window.

For more information about using the Project window, see “[Resource Type and Name List](#)” on page 14.

2. Select the string; then press RETURN or ENTER to open the string in a String editor window.

For more information, see “[String Editor Window](#)” on page 54.

3. Enter the character string and set its font.

For information about setting the string font, see “[Setting the String Font](#)” on page 61.

Creating a Character String List

To create a character string list resource:

1. Do one of the following:
 - Select the String Lists icon in the Project window; then choose **Edit** > **New String List Resource**. A new, untitled string list resource appears in the Project window.
For more information on using the project window, see “[Resource Type and Name List](#)” on page 14.
 - From anywhere in the Project window, choose **Edit** > **New Resource**. Select **String Lists** from the Resource Type pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to add the resource to the String Lists section of the Project window.
2. Select the string list in the Project window; then press RETURN or ENTER to open the string list resource in the String List Editor window.
See “[String List Editor Window](#)” on page 54, for more information.
3. Enter the character string list and set its font. For information about setting the string font, see “[Setting the String Font](#)” on page 61.

Creating a Category String List

Category names may be stored in an App Info String List resource.

To create list of category names for a database:

1. Do one of the following:
 - Select the App Info String Lists icon in the Project window; then choose **Edit** > **New App Info String List Resource**. An untitled resource appears in the Project window.
For more information on using the project window, see “[Resource Type and Name List](#)” on page 14.
 - From anywhere in the Project window, choose **Edit** > **New Resource**. Select **App Info String Lists** from the Resource Type pop-up menu, enter a Resource Name and Resource ID,

Working with Character Strings

Editing Character Strings

and then click **Create** to add the resource to the App Info String Lists section of the Project window.

2. Select the string list; then press RETURN or ENTER to open the string list in a App Info String List editor window. See “[App Info String List Editor Window](#)” on page 55, for more information.
3. Enter the categories list and set the category list font. For information about setting the string font, see “[Setting the String Font](#)” on page 61.

Creating Help Text

A character string resource is used to create help text or an online tip for a form or alert.

To learn about adding an online tip to a form, see “[Adding Help to a Form](#)” on page 39. To learn about adding an online tip to an alert, see “[Adding Help to an Alert](#)” on page 67.

Setting String Properties

Use the Inspector window to change the description and resource ID of a string resource.

NOTE: Changing a string’s resource ID may affect other resources that refer to that string. Be sure to update the properties of any forms, alerts, and other resources that refer to string resources you’ve modified.

Resource IDs should be less than 9999. Resource IDs of 10000 or above are reserved for use by Palm OS.

To change a string resource’s properties:

1. Select the string resource in the Project window.
2. Choose [Window](#) > [Property Inspector](#) to open the resource in the Property Inspector window.
3. Enter the new value in the Value area to change the string resource’s properties.

Modifying a Character String

To modify a character string's contents:

1. From the Project window, open the string in either of the following ways:
 - Double-click the string icon.
 - Select the string; then press RETURN or ENTER.

See “[String Editor Window](#)” on page 54, “[String List Editor Window](#)” on page 54, and “[App Info String List Editor Window](#)” on page 55, for more information.

2. Edit the string's text.

Note that some Palm OS characters may not appear correctly on your desktop computer. Select the **View as Hex** option to display and edit the text as hexadecimal values instead of ASCII characters. For more information, see “[Text Editing](#)” on page 27.

Setting the String Font

To set a string's font:

1. Open the string in an editor window.

See “[String Editor Window](#)” on page 54, “[String List Editor Window](#)” on page 54, and “[App Info String List Editor Window](#)” on page 55, for more information.

2. From the View as: pop-up menu, choose the string's font.

Constructor for Palm OS displays a list of Palm OS fonts available depending on the **Palm OS Target** setting selected in your project settings. For more information about the **Palm OS Target** setting, see “[Palm OS Target option](#)” on page 19.

NOTE: The string font selection is merely a convenience for you to preview the string as it would appear if it were displayed in the selected font at run time. The font selection is not saved, and the selection does not affect the string resource generated for your Palm application.

Deleting Character Strings

To remove a String, String List, or App Info String List resource:

1. Select the resource in the Project window.
2. Do one of the following:
 - Press DELETE or BACKSPACE.
 - Choose [Edit](#) > [Clear](#).
 - Choose [Edit](#) > [Cut](#).
 - Drag the string resource onto the trash can (Macintosh only).

To remove a string within a string list or app info string list:

1. Open the string list of app info string list in an editor window.
2. Click the number to the left of the string you want to remove. (When you click a number, the number is not highlighted.)
3. Do one of the following:
 - Press DELETE or BACKSPACE.
 - Choose [Edit](#) > [Clear](#).
 - Choose [Edit](#) > [Cut](#).
 - Drag the string resource onto the trash can (Macintosh only).

Working with Alerts

This chapter introduces Alert resources, used to communicate important messages to users.

Working With Alerts Overview

This chapter discusses how to manage and modify the contents of Alert resources.

Working with alerts comprises two concepts: the Constructor windows used to create and edit an alert resource, and the tasks you perform when working with alert resources.

- “[Windows for Alerts](#)” - Windows you use to work on alert resources.
- “[Editing Alerts](#)” on page 64 - Tasks you perform on alert resources.

Windows for Alerts

This section discusses the Constructor windows you use to create and edit alert resources and their contents.

- “[Alert Layout Window](#)” - The editor window that displays a view of a Palm OS® software alert.

Alert Layout Window

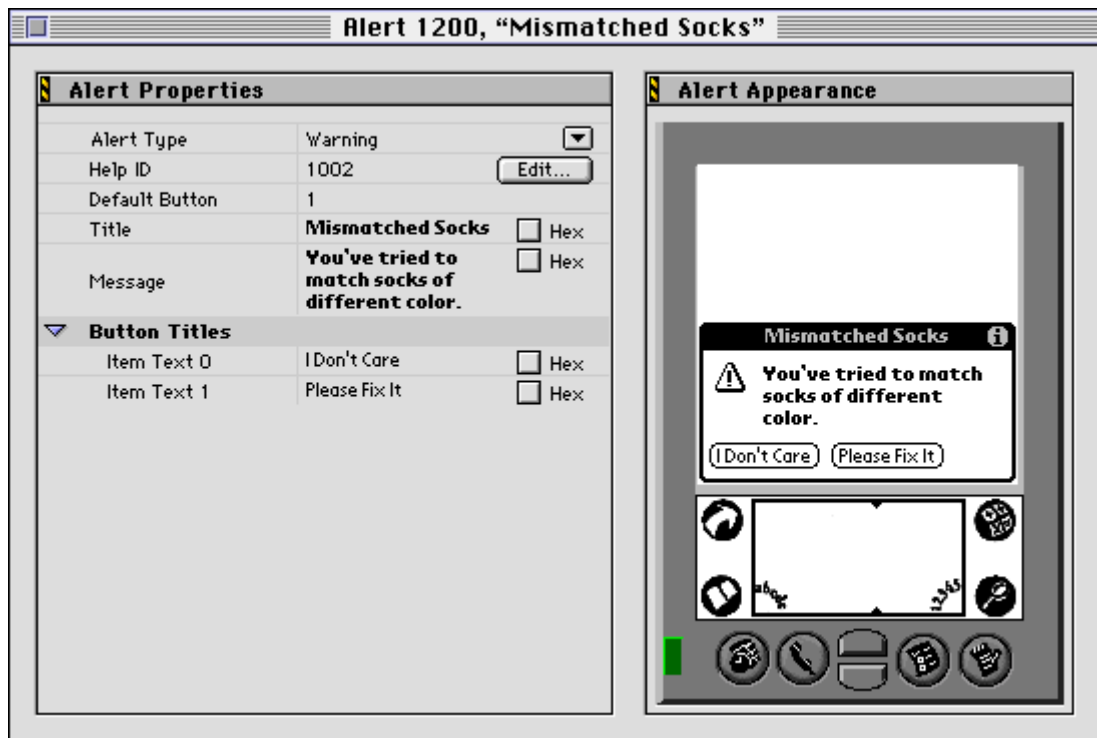
The Layout window is where most visual work is completed. For example, you use the Layout window to specify the text and buttons that appear in an alert resource.

To open an Alert resource:

Do one of the following:

- Double-click an Alert resource in the Project window.
- Select an alert resource in the Project window and press RETURN or ENTER.

Figure 5.1 The Alert Layout Window



Editing Alerts

This section shows you how to work with alert resources. Use this section to learn about:

- [“Creating an Alert”](#) on page 65
- [“Setting Alert Properties”](#) on page 65
- [“Adding Help to an Alert”](#) on page 67
- [“Setting the Alert’s Message”](#) on page 68

- “[Adding Buttons to an Alert](#)” on page 68
- “[Deleting Buttons from an Alert](#)” on page 68

Creating an Alert

An alert resource is used to communicate an important message to a user. The user must respond to the alert before they continue using the application.

To create an alert resource:

1. Do one of the following:
 - Select the Alert icon in the Project window; then choose [Edit](#) > **New String Resource**. A new, untitled Strings resource appears in the Project window.
 - From anywhere in the Project window, choose [Edit](#) > **New Resource**. Select **Alert** from the Resource Type pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to add the resource to the Alert section of the Project window.

For information about using the Project window, see “[Resource Type and Name List](#)” on page 14.

2. Select the Alert resource; then press RETURN or ENTER to open the Alert resource in an Alert layout window.

For more information, see “[Alert Layout Window](#)” on page 63.

3. Add and arrange items in the alert.

To learn how to add and arrange items in the alert, see “[Editing Alerts](#)” on page 64.

Setting Alert Properties

The Alert Properties section of the Alert layout window lets you control how an alert appears and how it will be used by the user.

For information about the Alert layout window, see “[Alert Layout Window](#)” on page 63.





Alert Type

Specifies the nature and importance of the alert message.

From the **Alert Type** pop-up menu, choose one of the items listed in [Table 5.1](#).

For information about designing alerts, see *Palm OS Programmer's Companion*.

Table 5.1 Items in the Alert Type Pop-up Menu

Choosing this item	Displays	To indicate that
Information		The user should be aware of something, but it will not cause problems at a later time.
Confirmation		The user must acknowledge the alert before continuing.
Warning		The application will initiate an irreversible action.
Error		An error has developed that cannot be reversed.

Help ID

Specifies the resource ID of the help text that appears when the user taps the Help icon in an alert.

If you enter a resource ID that does not match an existing character string resource in the project, a Create button appears next to the Help ID field. Click **Create** to enter a new character string in the String editor window.

If the resource ID that you enter matches the ID of a character string resource already in the project, an Edit button appears next to the String Help ID field. Click **Edit** to open the character string resource in a String editor window; then edit the help text for the alert.

Entering an ID of 0 means that the alert does not have help text.

See “[Adding Help to an Alert](#)” on page 67, and “[Working with Character Strings Overview](#)” on page 53, for related information.

Default Button

Specifies the resource ID of the form's default button. Palm OS software simulates the tapping of this button when the form is dismissed.

Title

Specifies the title of the alert, which appears at the top of the alert dialog box.

Message

Specifies the text that appears in the body of the alert. Enter text that clearly describes the problem or issue.

Button Titles

Displays a list of the button or buttons displayed in the Alert dialog box.

For information about adding buttons to or deleting buttons from an alert, see “[Adding Buttons to an Alert](#)” on page 68, and “[Deleting Buttons from an Alert](#)” on page 68.

Adding Help to an Alert

When a user taps the Help icon at the top right of the alert, Palm OS software displays a modal form containing help text. Optionally, you may add help to the alert.

To add help to an alert:

1. Open the Alert resource from the Project window.
See “[Alert Layout Window](#)” on page 63, for information about how to open Alert resources.
2. Enter a resource ID in the [Help ID](#) field.
If you've already entered a character string to use as help text, enter that character string's resource ID. Otherwise, enter a unique resource ID.
3. Click the button that appears next to the **Help ID** field—(**Create** or **Edit**, respective to the Help ID you enter.)
4. Enter or edit the desired help text.

5. Click the button that appears next to the [Help ID](#) field. See “[Help ID](#)” on page 66, for more information.

Setting the Alert’s Message

The Alert resource’s message text describes the situation or issue the user must acknowledge.

To define the Alert message:

1. Open the Alert resource from the Project window.
See “[Alert Layout Window](#)” on page 63, for more information.
2. In the **Message** field, enter the message text you want.
3. Either close the Alert window or move to another field.

Adding Buttons to an Alert

Buttons in an Alert resource let the user dismiss or act on a variety of choices.

To add buttons to an alert:

1. From the Project window, open the Alert to which you want to add a button.
See “[Alert Layout Window](#)” on page 63, for more information.
2. Click [Button Titles](#) in the Alert Properties section of the Alert window; then choose [Edit](#) > **New Button Title**. A new entry appears in the [Button Titles](#) list.
3. Click in the field to the right of the button title you want; then enter the text for the button.

Deleting Buttons from an Alert

To remove a button from an alert:

1. From the Project window, open the Alert from which you want to delete a button.
See “[Alert Layout Window](#)” on page 63, for more information.

2. Click a button ID to select the button you want to delete. (If the list of buttons in the alert layout window is hidden, expand the [Button Titles](#) list.
3. Choose [Edit](#) > **Delete Button Title**.

Working with Icons and Bitmap Images

This chapter introduces the Constructor's bitmap editor, used to create and modify bitmaps, icons, bitmap families, and icon families within a variety of resources.

Working with Icons and Bitmaps Overview

Constructor has a simple, powerful, and intuitive bitmap picture editor. Constructor uses a single bitmap editor to modify bitmaps in a variety of resource types:

- Icons
- Pictures

This chapter shows you how to manage and modify the contents of bitmap resources, including:

- “[Bitmap Families](#)” on page 72 - With Constructor 1.7 and later, you can create and store up to ten bitmaps rendered in different bit depths and densities.
- “[Application Icon Families](#)” on page 75 - With Constructor 1.7 and later, you can create and store up to ten application icons rendered in different bit depths and densities.
- “[Icons in Palm OS 3.5 and Later](#)” on page 77 - How to upgrade icons to multibit (grayscale) icons and make sure they are recognized by any version of Palm OS® software.
- “[Bitmap Images in Palm OS 3.5 and Later](#)” on page 80 - How to name bitmap images to take advantage of Palm OS 3.0 features.
- “[Bitmap Editor Window](#)” on page 82 - Identifying various parts of the editor window.

Working with Icons and Bitmap Images

Bitmap Families

- “[Using the Bitmap Editor Tools](#)” on page 85 - A description of each bitmap editor tool and how to use them.
- “[Editing Icons and Bitmaps](#)” on page 93 - Tips and ideas for using the bitmap editor.

NOTE: Use of color and grayscale icons and bitmaps is dependent on the Palm OS version. The 2-bit (4-level grayscale) icons and bitmaps are only supported in Palm OS 3.0 and later. The 4-bit (16-level grayscale) and 8-bit color icons and bitmaps are only supported in Palm OS 3.5 and later. The 16-bit color icons and bitmaps are only supported in Palm OS 4.0 and later.

Make sure that your application verifies the system software and displays the appropriate image type for the version of the system software you are using. See “[Icons in Palm OS 3.5 and Later](#)” on page 77, for more information.

Grayscale and compressed bitmap images must follow a special naming convention to be recognized by Palm OS 3.0 software. See “[Bitmap Images in Palm OS 3.5 and Later](#)” on page 80, for more information.

Bitmap Families

The bitmap family resource element (`Tbmp`) lets you store up to five bitmap images rendered in different bit depths in a single resource.

The following rules apply to bitmap families:

- A maximum of ten bitmaps can be included in one bitmap family.
- Bitmap images must be stored in increasing order of density first and then bit depth. The bit depths must be unique by density.

For example, bitmap images of the following order would be correct:

- Density = Normal; Depth = 1-bit
- Density = Normal; Depth = 8-bit
- Density = One and One Half; Depth = 1-bit
- Density = One and One Half; Depth = 8-bit
- Density = Double; Depth = 1-bit
- Density = Double; Depth = 8-bit

However, the following order would not be correct because the densities are not in increasing order:

- Density = Normal; Depth = 1-bit
 - Density = Double; Depth = 1-bit
 - Density = Normal; Depth = 8-bit
 - Density = Double; Depth = 8-bit
- The height and width of all bitmaps in a Bitmap Family Resource must be equivalent in normal density coordinates.

For example, if the Bitmap Family height and width is 16, then the height and width of normal density bitmaps will be 16. But the height and width of one-and-one-half density bitmaps will be 24, while the height and width of double density bitmaps will be 32.

For one-and-one-half density versions of a bitmap, the dimensions are calculated by multiplying 1.5 times the normal size and then rounding down. For example, if the bitmap family's Normal size is 1-by-1, then the One and One Half size is also 1-by-1. However, if the Normal size is 1-by-2, then the One and One Half size is 1-by-3.

To create a bitmap family resource:

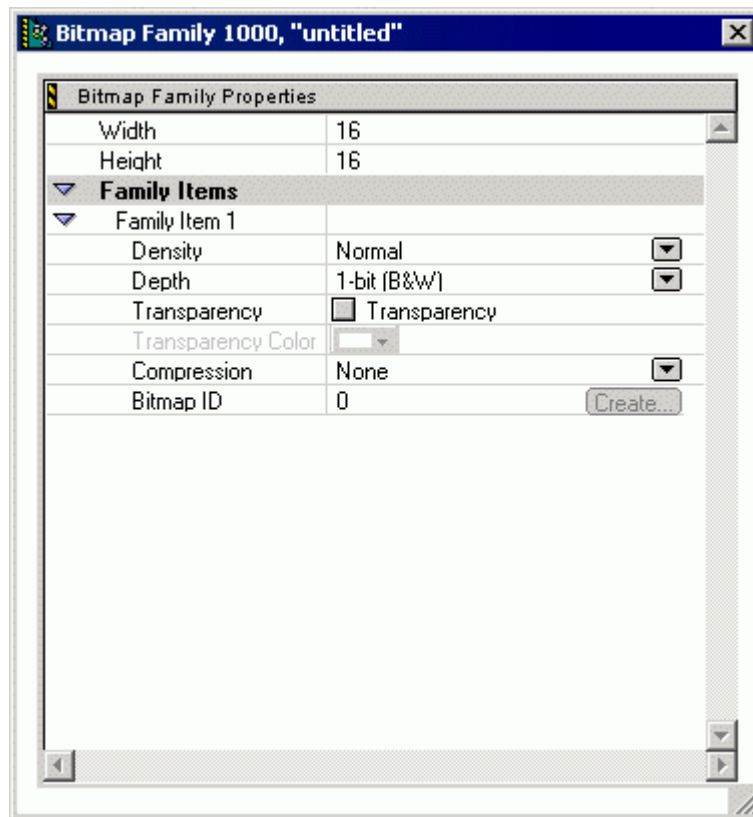
1. Select the Bitmap Families Resource icon in the Project window.

Working with Icons and Bitmap Images

Bitmap Families



2. Choose **Edit** > **New Bitmap Family Resource**; then double-click the untitled icon under the resource to open the Bitmap Family dialog box.



Follow the instructions later in this chapter to use the tools to create the bitmap.

Application Icon Families

The application icon family resource element (tAIB) lets you store up to five icons rendered in different bit depths in a single resource.

The following rules apply to application icon families:

- **Large Application Icons (Resource ID 1000):** The maximum pixel height and width is 22-by-22 at normal density, 33-by-33 at one-and-one-half density, and 44-by-44 at double density.
- **Small Application Icons (Resource ID 1001):** The maximum pixel height is 9 and maximum pixel width is 15 at normal density (13-by-22 at one-and-one-half density and 18-by-30 at double density).

Working with Icons and Bitmap Images

Application Icon Families

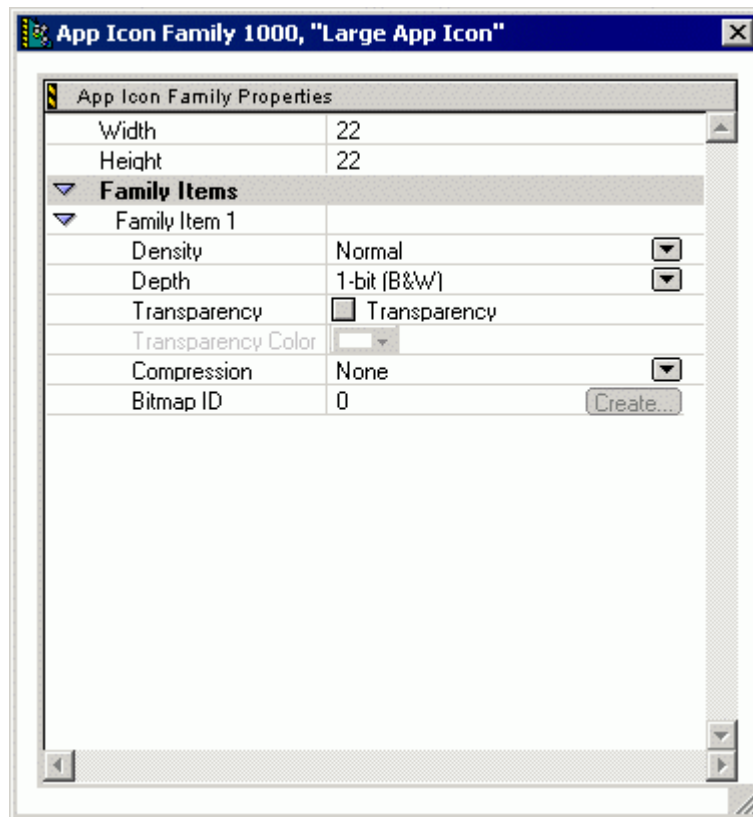
- A maximum of ten icons can be included in one application icon family.
- Bitmap images must be stored in increasing order of density first and then bit depth. The bit depths must be unique by density.

To create an application icon family resource:

1. Select the Application Icon Family resource in the Project window.



2. Choose **Edit > New Application Icon Family**; then double-click the untitled icon under the resource to open the App Icon Family editor.



Follow the instructions later in this chapter to use the tools to create the application icon.

Icons in Palm OS 3.5 and Later

Constructor lets you create these types of icons:

- Black and white icons, for use with versions of the Palm OS earlier than version 3.0
- *Multibit* (grayscale and color) icons, for use with Palm OS software version 3.0 and later

It is recommended that you use the App Icon Family editor to create icons for new projects. The Multibit Icon editor is included with Constructor for compatibility with older Constructor files. The Multibit Icon editor is limited to 1-bit black and white icons and 2-bit grayscale icons.

The topics in this section are:

- “[Updating Older Icons](#)”
- “[Important Icon Numbering and Size Conventions](#)” on page 80

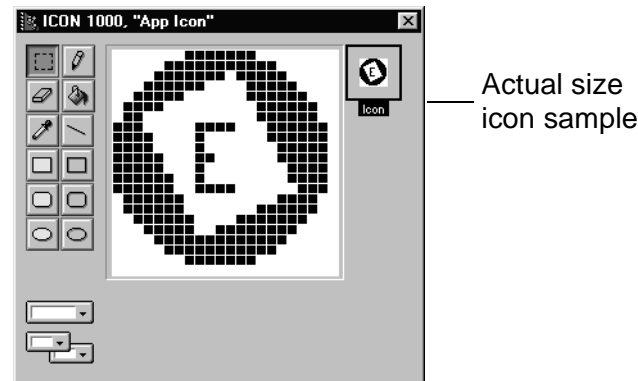
Updating Older Icons

You use the multibit editor to create both a black and white and a grayscale icon that share the same ID. Having both icons available ensures that the correct icon is displayed, regardless of the version of the Palm OS software.

To update a black and white icon to a multibit icon:

1. Open the Constructor project containing the black and white icon you want to update.
2. From the Icon area of the Project window, double-click the desired icon to open it in the Icon editor window.

Figure 6.1 Icon Editor Window



In the top right area of the window, a black and white sample icon appears.

3. Choose **Edit > Select All**; then choose **Edit > Copy** to copy the black and white sample icon to the Clipboard.

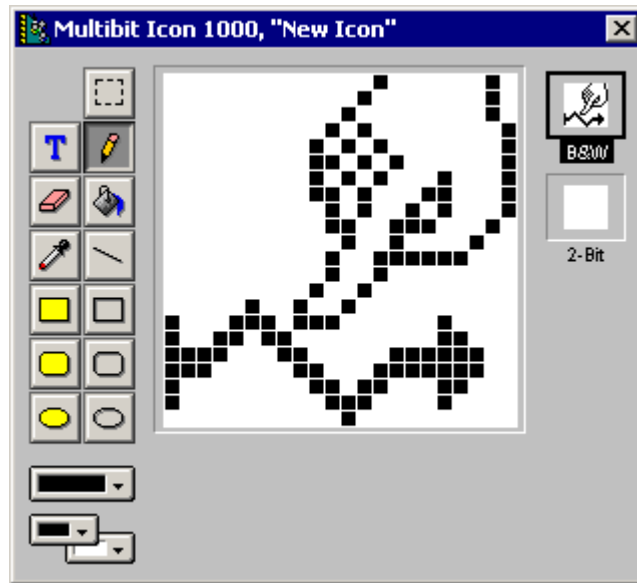
Next, you'll create a multibit icon.

4. In the Project window, click the Multibit Icons icon; then choose **Edit > New Multibit Icon Resource**.

A new, untitled multibit icon resource appears.

5. Click the 'untitled' name text to select it; then rename the icon with a new name. Be sure to name the multibit icon something different than the original black and white icon.
6. Double-click the renamed icon to open it in the Multibit Icon editor window.

Figure 6.2 Multibit Icon Editor Window



7. Click the **B&W** sample view in the top-right corner of the Multibit Icon editor window; then choose **Edit > Paste** to paste the old icon from the Clipboard.
8. Next, click the **2-Bit** sample view and choose **Edit > Paste** again to paste the older icon onto the canvas area. This **2-Bit** sample view shows the grayscale version of the icon (which is still black and white, because it has not been edited).
9. With the 2-Bit sample view selected, you can now edit the black and white icon with gray values to make it a grayscale icon.

For information about drawing and editing icons, see "[Editing Icons and Bitmaps](#)" on page 93. For information about working with two versions of the same icon, see "[Viewing Sample Views](#)" on page 84.

Important Icon Numbering and Size Conventions

After updating black and white icons to multibit icons, make sure that all versions of Palm OS software recognize and display your application's icons accurately by following the conventions listed in [Table 6.1](#).

For information about changing a resource's ID, see the "[Property Inspector Window](#)" on page 22.

Table 6.1 Icon IDs and Dimensions

The multibit icon with this ID:	Should be the following size:
1000	22 pixels high by 22 pixels wide at normal density (33-by-33 at one-and-one-half density, 44-by-44 at double density), which is the entire size of the canvas presented by Constructor in a multibit icon editor window.
1001	The top left corner of the multibit icon editor's canvas, 9 pixels high by 15 pixels wide at normal density (13-by-22 at one-and-one-half density, 18-by-30 pixels at double density). Pixels outside this rectangle are ignored by Palm OS software.

Follow these icon conventions for all application icons ('`TAIB`' resources). You should use the App Icon Family editor to create the application icons. The Multibit Icon editor and Icon editor are for compatibility purposes only, and are included with Constructor so that you can edit old Constructor resource formats.

Bitmap Images in Palm OS 3.5 and Later

It is recommended that you use the Bitmap Family editor to create all bitmaps. The Bitmap Family editor allows you to specify bit-depth, density, compression, and transparency of bitmaps. See

“[Bitmap Families](#)” on page 72, for more information about the Bitmap Family editor.

The Bitmap editor is still supported, but with the Bitmap editor, you need to use resource name encoding to specify bit-depth and compression.

To take advantage of the grayscale bitmap display features in Palm OS Version 3.5, follow the naming conventions in [Table 6.2](#). By default, if the name of a bitmap does not start with one of the prefixes provided in [Table 6.2](#), any version of the Palm OS software displays the bitmap in black and white.

For information on changing a resource’s name, use “[Property Inspector Window](#)” on page 22.

NOTE: The naming conventions in [Table 6.2](#) are applicable only for Palm OS version 3.0 and higher. Earlier versions of Palm OS software can only display black and white (1-bit per pixel) images, so be sure your application verifies the system version at runtime before displaying bitmap images.

Table 6.2 Bitmap Naming Conventions for the Bitmap Editor

If the bitmap resource’s name begins with:	It is displayed by Palm OS 3.0 software as:
/-2/	A grayscale image with 2 bits per pixel
/-1/	A black and white image with 1 bit per pixel
/-c/	Compresses the bitmap resource to save memory
/-c-2/	Compresses the resource and displays it as a 2-bit grayscale image
/-c-1/	Compresses the resource and displays it as a black and white image

Bitmap Editor Window

Constructor uses the same bitmap editor, with a few small differences, for all resources containing bitmaps.

The Constructor bitmap editor has the following components:

- [Tool Palette](#) - Tools used to control drawing operations.
- [Drawing on the Canvas](#) - The area where you draw, usually displayed in a magnified view called *fat bits*.
- [Viewing Sample Views](#) - Actual-size representations of icons.

Tool Palette

The tool palette provides a set of tools for working with bitmap icons. To select a tool in the tool palette; click it.

Table 6.3 The Bitmap Editor Tools

















Tool	Icons	Purpose
Lasso Tool		Selects an irregularly shaped area (available on Macintosh only).
Marquee Tool		Selects a rectangular area.
Text Tool		Adds text to a bitmap (available on Macintosh only).
Pencil Tool		Draws pixel by pixel, usually with the foreground color.
Eraser Tool		Erases pixels with the background color
Paint Bucket Tool		Fills an area of contiguous pixels of one color with current pattern.
Dropper Tool		Sets the foreground or background color (also affects pattern color).

Table 6.3 The Bitmap Editor Tools (*continued*)

Tool	Icons	Purpose
Line Tool		Draws a line with the foreground color.
Rectangle Tools		Draws a filled rectangle using the current pattern.
Rectangle Tools		Draws a rectangle outlined with the foreground color.
Rectangle Tools		Draws a filled rounded rectangle using the current pattern.
Rectangle Tools		Draws a rounded rectangle outlined with the foreground color.
Oval Tools		Draws a filled oval using the current pattern.
Oval Tools		Draws an outlined oval, using the foreground color.
Pattern Tool		Sets the fill pattern, based on the foreground and background colors.
Color Tool		Sets the foreground or background color.

Drawing on the Canvas

The canvas is the main area of the Constructor bitmap editor window. You draw and edit the contents of the canvas area to modify the bitmap icon it represents, regardless of the kind of resource in which that bitmap appears.

The canvas area displays a magnified view of an icon. Each magnified bit, sometimes called a *fat bit*, represents a single pixel in the actual size bitmap.

Working with Icons and Bitmap Images

Bitmap Editor Window

To edit a bitmap, select a tool from the [Tool Palette](#), then move the cursor onto the canvas area. When the cursor is positioned in the canvas area, the cursor changes to reflect the currently selected tool.

As you edit the bitmap on the canvas, the [Viewing Sample Views](#) are updated.

The canvas supports the usual mechanisms for transferring information between windows and between applications: Cut, Copy, and Paste. You can also drag and drop a bitmap from the Sample View of one bitmap onto the canvas area of another bitmap.

When you drop an item on the canvas, the original image is not resized to fit the new canvas dimensions. If the bitmap is larger than the canvas, it is cropped to fit onto the current canvas.

Macintosh OS: You may drag from another Constructor window, or from an application that supports drag and drop, such as the Finder or the Scrapbook.

Viewing Sample Views

Sample views are displayed at the right side of the Icon editor or Multibit Icon editor window. The sample views display icons at actual size.

If more than one sample view is available, each sample view represents either a different related resource or a different portion of the same resource. The canvas area displays the currently selected sample view, which is outlined in black. To switch from one sample view to another, click the sample view icon you want.

You can drag and drop sample views in the same window or between two discrete windows.

Macintosh OS: You may also drag bitmap images between Constructor and other applications that support drag and drop. If you drop a bitmap from a different source onto a sample view in a Constructor editor window, the bitmap is scaled to fit Constructor's canvas size. In addition, if the sample view you drag and drop contains more colors than the bitmap you're dragging to, excess colors are automatically converted to the sample views color palette.

Using the Bitmap Editor Tools

This section provides information about the individual tools on the [Tool Palette](#).

- “[Lasso Tool](#)” on page 85
- “[Marquee Tool](#)” on page 86
- “[Text Tool](#)” on page 86
- “[Pencil Tool](#)” on page 87
- “[Eraser Tool](#)” on page 88
- “[Paint Bucket Tool](#)” on page 88
- “[Dropper Tool](#)” on page 89
- “[Rectangle Tools](#)” on page 90
- “[Oval Tools](#)” on page 90
- “[Line Tool](#)” on page 89
- “[Pattern Tool](#)” on page 91
- “[Color Tool](#)” on page 92

Lasso Tool

The lasso tool creates irregularly shaped selections by letting you drag the mouse in a freehand motion. (The lasso tool is available only for the Macintosh.)

To create a selection using the lasso tool:

1. Click the lasso tool in the tool palette.
2. Drag the lasso tool around the shape you want to select. As you drag, a thin line follows to create a selection path.
3. Close the path in either of the following ways:
 - Cross the starting point of the path to end the path.
 - Release the mouse button anywhere to draw a selection that ends by drawing a straight line between the starting point and the point at which you release the mouse button.

The lasso tool ignores pixels in the current background color. If an object is surrounded by a background color, drawing the lasso

Working with Icons and Bitmap Images

Using the Bitmap Editor Tools

loosely around the object and then releasing the mouse button automatically tightens around the object to select it.

TIP: Double-click the lasso tool to select the outermost outline of non-background-color pixels in the canvas.

See also “[Tool Palette](#)” on page 82, “[Marquee Tool](#)” on page 86, and “[Selecting Pixels](#)” on page 94.

Marquee Tool

The marquee tool creates rectangular or square selections.

To create a selection using the marquee tool:

1. Click the marquee tool in the tool palette.
2. Do one of the following:
 - Drag to create a rectangular selection.
 - Hold down SHIFT and drag to constrain the selection to a square.

TIP: Double-click the marquee tool in the tool palette to select the entire canvas area.

See also “[Tool Palette](#)” on page 82, “[Lasso Tool](#)” on page 85, and “[Selecting Pixels](#)” on page 94.

Text Tool

The text tool is used to enter text in a bitmap.

To create text:

1. Click the text tool in the tool palette.
2. Click in the canvas area to set an insertion point; then type the text you want.

The initial position of text defines the margin, either left or right, depending on the justification settings. By default, text

wraps at the edge of a bitmap; for a forced return, use RETURN or ENTER.

3. Use the Font and Style menus to choose text characteristics. By default, text appears in the foreground color, and may have only one font, style, and color per entry.

Once you complete a text entry, it becomes a bitmap image and is not longer editable as text. You can however, edit the pixels that define the text on a pixel-by-pixel basis.

Note that the text editor does not support the Cut, Copy, or Paste commands.

TIP: To increase or decrease font size by one point as you enter text, type COMMAND-up arrow or COMMAND-down arrow, respectively.

See also “[Tool Palette](#)” on page 82.

Pencil Tool

The pencil tool draws single pixels or lines of pixels using the foreground color. The size of the pencil is one pixel, and cannot be changed.

To use the tool, select it in the tool palette. Clicking an existing pixel changes it to the current foreground color; clicking a blank area of the canvas adds a pixel using the current foreground color.

If you click and drag the pencil, each pixel in your path changes to the foreground color. If the first pixel clicked is already in the foreground color, then the pencil draws in the background color.

TIP: Change any non-foreground color pixel to the background color by double-clicking the desired pixel with the pencil.

To constrain the pencil to a straight line, press **SHIFT** before you begin dragging. The pencil is constrained to move in a straight line, either horizontally or vertically.

See also “[Tool Palette](#)” on page 82, “[Paint Bucket Tool](#)” on page 88, and “[Line Tool](#)” on page 89.

Eraser Tool

The eraser tool erases pixels. As you drag to erase pixels, the current background color is revealed. Be sure to set the background color to white if you want the eraser tool to reveal a white background.

To use the eraser tool:

1. Click the eraser tool in the tool palette.
2. Drag in the canvas area over the pixels you want to erase.
3. If desired, constrain the eraser to a straight line by pressing **SHIFT** before you begin dragging.

TIP: Double-click the eraser tool to erase the entire canvas.

See also “[Tool Palette](#)” on page 82.

Paint Bucket Tool

The paint bucket tool fills an area with the current pattern. Pattern colors are based on the current foreground and background colors.

To use the paint bucket tool:

1. Click the paint bucket tool in the tool palette to select it.
2. Select a pattern from the pattern pop-up menu.
3. Click the paint bucket tool in the canvas area to add the current pattern to the canvas.

TIP: Note that clicking in an enclosed area (an area of contiguous pixels that start and end at the same point) may produce varied results. The paint bucket tool fills an area with the current pattern, *not* the current foreground color (although the pattern is based on the current foreground and background colors). To fill an area with either the foreground or background color, select the foreground or background solid color swatch in the pattern menu before clicking the paint bucket tool in the canvas area.

See also “[Tool Palette](#)” on page 82, “[Pattern Tool](#)” on page 91, and “[Color Tool](#)” on page 92.

Dropper Tool

The dropper tool lets you set foreground and background colors.

To use the dropper tool:

1. Click the dropper in the tool palette.
2. Do either of the following:
 - To set the foreground color, position the dropper on the canvas and begin dragging. As you drag, the foreground swatch changes color. Release the mouse button when the color you want is displayed in the foreground swatch.
 - To set the background color, position the dropper on the canvas, hold down SHIFT, and then begin dragging. As you drag, the background swatch changes color. Release the mouse button when the color you want is displayed in the background swatch.

TIP: For Macintosh: You can set foreground and background color at any time without selecting the dropper tool. OPTION-click a pixel to set the foreground color. SHIFT+OPTION-click a pixel to set the background color.

TIP: For Windows: You can set foreground and background color at any time without selecting the dropper tool. ALT-click a pixel to set the foreground color. SHIFT+ALT-click a pixel to set the background color.

Line Tool

The line tool draws lines using the foreground color. The line tool has a default width of 1 pixel, which may not be changed.

To use the line tool:

1. Click the line in the tool palette.

2. Position the pointer in the canvas area; then do one of the following:
 - Drag to draw a line.
 - Hold down **SHIFT**, and then drag to draw a line constrained to 45° angle.

See also “[Tool Palette](#)” on page 82 and “[Pencil Tool](#)” on page 87.

Rectangle Tools

You use the filled and empty rectangle tools to draw filled and outline rectangles. The filled rectangle tool fills a rectangle with the current pattern, and the empty rectangle tool creates an outlined rectangle using the current foreground color.

To use the rectangle tools:

1. Select either the filled rectangle tool or the empty rectangle tool and position it in the canvas area.
2. Do one of the following:
 - For a filled rectangle, drag from one edge to another to create a rectangle filled with the current pattern.
 - For an empty rectangle, drag from one edge to another to create an outlined rectangle. The current foreground color creates a 1-pixel outline. The width of the outline may not be changed.
3. To constrain either rectangle tool to a square, hold down **SHIFT** before you begin dragging.

See also “[Tool Palette](#)” on page 82, “[Rectangle Tools](#)” on page 90, and “[Drawing Shapes](#)” on page 96.

Oval Tools

You use the oval tools to draw filled and outlined ovals. The filled oval tool fills an oval with the current pattern, and the empty oval tool creates an outline using the current foreground color.

To use the oval tools:

1. Select either the filled oval tool or the empty oval tool and position it in the canvas area.

2. Do one of the following:
 - For a filled oval, drag from one edge to another to create an oval filled with the current pattern.
 - For an empty oval, drag from one edge to another to create an outline oval. The current foreground color creates a 1-pixel outline. The width of the outline may not be changed.
3. To constrain either oval tool to a circle, hold down **SHIFT** before you begin dragging.

See also “[Tool Palette](#)” on page 82, “[Oval Tools](#)” on page 90, and “[Drawing Shapes](#)” on page 96.

Pattern Tool

The pattern tool is used to select patterns.

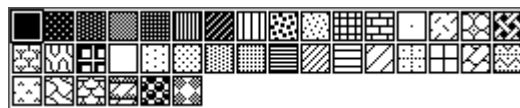
To select a pattern:

1. Click and hold down the mouse button on the pattern tool to display the available patterns.
2. Drag to the desired pattern; then release the mouse button.
3. Position the mouse pointer in the canvas area; then use one of the available pattern tools to add the pattern.

Patterns are based on the current foreground and background colors.

[Figure 6.3](#) illustrates the pattern pop-up menu with the current pattern outlined. In this case, it is the solid fill pattern in the top left corner of the pop-up menu.

Figure 6.3 The pattern pop-up menu



You cannot edit the default patterns available in Constructor.

See also “[Tool Palette](#)” on page 82, “[Color Tool](#)” on page 92, and “[Setting Colors and Patterns](#)” on page 95.

Color Tool

The color tool lets you choose foreground and background colors. The topmost swatch is the foreground swatch, and the backmost swatch is the background color.

Palm OS resources support:

- 1-bit black and white bitmaps
- 2-bit (4-level grayscale) bitmaps.
- 4-bit (16-level grayscale) bitmaps
- 8-bit (256-color) bitmaps
- 16-bit (thousands of colors) bitmaps

To select a foreground or background color:

1. Hold down the mouse button on either the foreground or background swatch to open a pop-up menu.
2. Drag to the desired color and release the mouse button.

The foreground and background color swatches available in the pop-up menu depend on the type of resource you are editing and the setting in the Colors menu.

For example, if you are working with a black-and-white resource, only the Black & White (1-bit depth) command is available from the Colors menu; if you are working with a full-color resource, the Colors menu lets you select from Black & White (1-bit depth) through Thousands of Colors (16-bit depth).

TIP: For Macintosh: You can also set the foreground color to match an existing pixel by clicking a pixel with the dropper tool, or OPTION-clicking a pixel at any time. Set the background color by SHIFT-clicking a pixel with the dropper tool, or SHIFT+OPTION-clicking a pixel at any time.

TIP: For Windows: You can also set the foreground color to match an existing pixel by clicking a pixel with the dropper tool, or ALT-clicking a pixel at any time. Set the background color by SHIFT-clicking a pixel with the dropper tool, or SHIFT+ALT-clicking a pixel at any time.

See also “[Tool Palette](#)” on page 82, “[Dropper Tool](#)” on page 89, “[Setting Colors and Patterns](#)” on page 95, and “[Setting Colors and Patterns](#)” on page 95.

Editing Icons and Bitmaps

Editing a bitmap using Constructor is a simple task. In general, you select a tool from the tool palette, move it onto the canvas area, and begin modifying the image. Operations may be performed in any order, on any pixel, at any time, and may be undone or redone using the **Edit > Undo** and **Edit > Redo** commands.

However, some functions require additional explanation, which are included here for your convenience. These topics include:

- “[Selecting Pixels](#)” - Choosing a part of the bitmap image.
- “[Nudging Pixels](#)” on page 94 - Moving selected pixels a pixel at a time.
- “[Copying Pixels](#)” on page 94 - Copying a selection.
- “[Rotating and Inverting Pixels](#)” on page 95 - Manipulating selections.
- “[Setting Colors and Patterns](#)” on page 95 - Painting parts of a bitmap.
- “[Drag and Drop](#)” on page 96 - Using drag and drop effectively.
- “[Drawing Shapes](#)” on page 96 - Tips on applying basic shapes to an image.
- “[Setting Colors and Patterns](#)” on page 95 - Tips for specifying color and patterns for drawing.

Selecting Pixels

You can select pixels in a number of ways; use the method that best suits the operation you want to perform.

To select pixels:

Do any of the following:

- Use the [Lasso Tool](#) to select irregularly shaped areas.
- Double-click the [Lasso Tool](#) to select the outermost outline of non-background-color pixels on the canvas.
- Use the [Marquee Tool](#) to select rectangular shapes. Use the SHIFT key to constrain the selection marquee to a square. Double-click the [Lasso Tool](#) to select the entire bitmap. (Macintosh only).
- Choose [Edit](#) > [Select All](#) everything in the canvas area.

Nudging Pixels

You can *nudge* a selection to move it one pixel at a time, which may provide more precise placement than dragging the selection with the mouse.

To nudge a selection:

1. Select the pixels you want to move.
2. Use the arrow keys to move the selection in the desired direction. Each time you press an arrow key, the selection moves one pixel in the direction of the arrow key you press.

Copying Pixels

You can copy pixels in two different ways: using the [Edit](#) > [Copy](#) command, or using a keyboard shortcut.

To copy using a keyboard shortcut:

- On the Macintosh, hold down OPTION and drag the selection. Be sure to press OPTION before you begin dragging.
- In Windows, hold down ALT and drag the selection. Be sure to press ALT before you begin dragging.

Rotating and Inverting Pixels

When a bitmap editor window is active, the [Options](#) menu provides commands that let you flip, rotate and invert a selection:

- The [Flip Vertical](#) command flips a selection vertically (up and down).
- The [Flip Horizontal](#) command flips a selection horizontally (left to right).
- The [Rotate Right](#) command rotates a selection clockwise.
- The [Rotate Left](#) command rotates a selection counter-clockwise.
- The [Invert Pixels](#) command inverts the pixels in a selection.

Setting Colors and Patterns

There are three ways to set foreground and background colors.

1. For the Macintosh:

To set the foreground color, do one of the following:

- Choose a color from the foreground [Color Tool](#) swatch in the editor window.
- Click a pixel with the [Dropper Tool](#).
- When any tool other than the dropper tool is active, OPTION-click a pixel (OPTION selects the dropper tool from the keyboard when another tool is active.)

To set the background color, do one of the following:

- Choose a color from the background [Color Tool](#) swatch in the editor window.
- SHIFT-click a pixel with the [Dropper Tool](#).
- When any tool other than the dropper tool is active, SHIFT+OPTION-click a pixel. (OPTION selects the dropper tool from the keyboard when another tool is active.)

2. For Windows:

To set the foreground color, do one of the following:

- Choose a color from the foreground [Color Tool](#) swatch in the editor window.

Working with Icons and Bitmap Images

Editing Icons and Bitmaps

- Click a pixel with the [Dropper Tool](#).
To set the background color, do one of the following:
 - Choose a color from the background [Color Tool](#) swatch in the editor window.
 - Select the [Dropper Tool](#), then SHIFT-click a pixel
3. In addition, you can use the [Colors](#) > [Font](#) command to set the foreground and background colors to black and white, respectively.

The [Colors](#) > [Swap Fore & Back Colors](#) command reverses the current foreground and background colors.

Any change in either foreground or background color affects the colors in the current pattern. By selectively choosing colors, patterns, and transparency, you can quickly create complex effects such as grid lines, slashes, and interesting color blends.

Drawing Shapes

To draw filled and outlined shapes, use the filled and outlined shape tools. For information about how to draw shapes, see [Oval Tools](#) and [Rectangle Tools](#).

You can constrain the shape with the SHIFT key. You may press or release the SHIFT key at any time while drawing the shape. While the SHIFT key is pressed, the rectangle tools draw squares and the oval tools draw circles.

TIP: To create a shape that is both filled and outlined, draw an outlined shape; then use the [Paint Bucket Tool](#) to fill the shape with the pattern.

Drag and Drop

The bitmap editor has powerful drag and drop features, especially for [Viewing Sample Views](#). Although you cannot edit the [Viewing Sample Views](#) directly, they are fully available for drag and drop.

Tips for using drag and drop:

For Macintosh: When editing a bitmap, you can make an instant backup to save work in progress. Simply drag the sample to the desktop, or to any folder. If you later decide you want to revert to a previous stage of development, drag your saved work from the desktop back into the sample view.

If you have a favorite bitmap that you want to use as a basis for further work, use Constructor to open the source file that contains the original bitmap resource. Then you can simply drag the original bitmap into your own Constructor project, and you're on your way.

Working with Icons and Bitmap Images

Editing Icons and Bitmaps

Working with Fonts

This chapter describes how to use Constructor for Palm OS® to add font family resources to your application.

Working with Font Families Overview

Constructor allows you to define font families that you can use in your application.

The following topics are included in this chapter:

- “[Fonts and Font Families](#)” on page 99
- “[Creating a Font Family](#)” on page 100
- “[Windows for Font Families](#)” on page 102

Fonts and Font Families

Palm OS uses the font resource type ('NFNT') to store normal (that is, single density) bitmap fonts. To support high density screens, Palm OS 5 defines the extended font resource type ('nfnt'), which may contain single density font data, one-and-one-half density font data, double density font data, or all three density versions of the font data. (Normally, the extended font resource contains both single and double density versions of the font data.)

When a font specified by the extended font resource type is used on a handheld, Palm OS will automatically draw text using the version of the font data that matches the screen density.

The Constructor font family resource is used to build a Palm OS extended font resource. You must first create a 'NFNT' resource for each of the densities used in the font family. Constructor does not provide an editor for font resources, but there are other font tools available (for example, ResEdit and Resorcerer on the Macintosh).

For one-and-one-half density versions of a font, the font data size is calculated by multiplying 1.5 times the normal size and then rounding down.

For double density versions of a font, the font data used must be exactly proportional to the single density font data in all metrics.

When you have completed your font resources ('NFNT'), you can define the font family resource in Constructor, listing each font density to be used and the ID of the 'NFNT' resource to be used for that density. The font densities must be listed in increasing order: first Normal density, then One and One Half density, then Double density.

When you compile your application, PalmRez uses the data from the font family resource ('tfnt') and the referenced 'NFNT' font resources and combines them to produce a Palm OS extended font resource ('nfnt'). The output extended font resource will have the same resource ID as the font family resource. Note that the resource IDs of the 'NFNT' resources themselves do not have any effect on the output resource ID or resource data.

Creating a Font Family

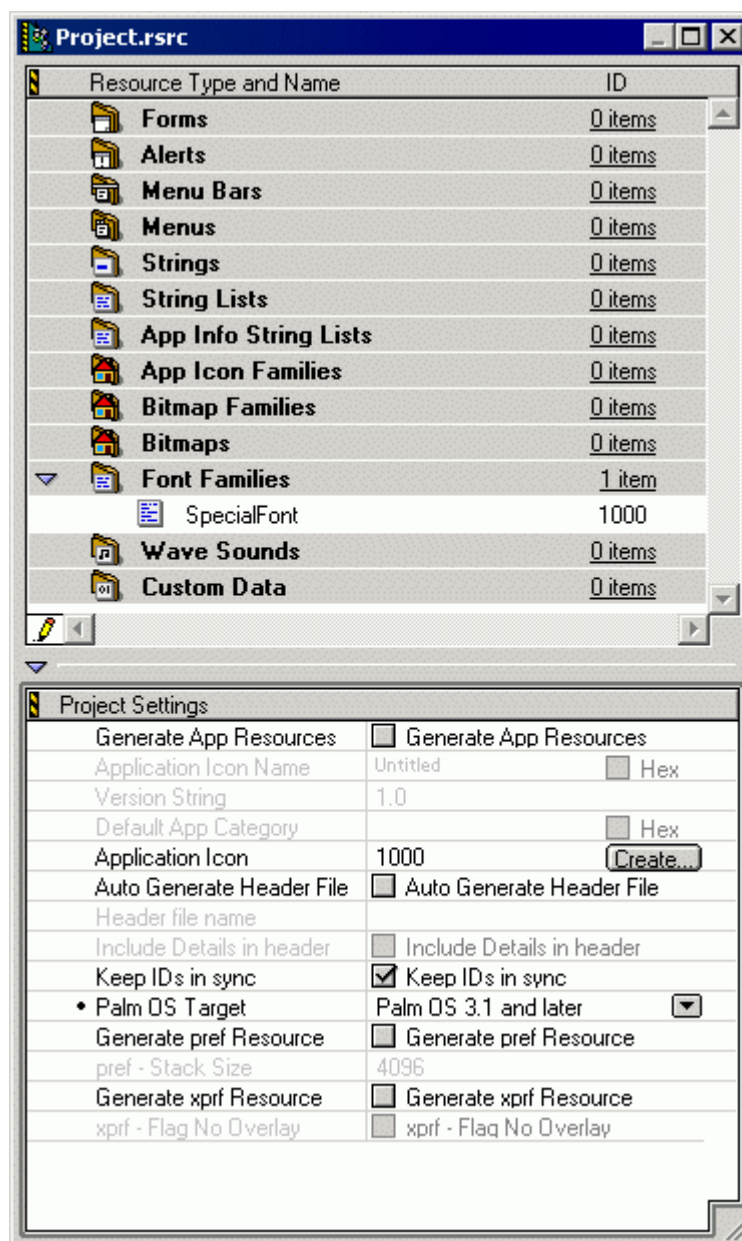
To create a font family resource:

From anywhere in the Project window, choose **Edit > New Resource**. Select **Font Family** from the **Resource Type** pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to add the resource to the Font Families section of the Project window (shown in [Figure 7.1](#) on page 101).

Once you have created a font family and have the Font Families icon in the Project window, you can create a font family resource by doing the following:

- Select the Font Families icon in the Project window
- Choose **Edit > New Font Family Resource**. A new, untitled font family resource appears in the Project window.

Figure 7.1 Project with Font Families



For more information about using the Project window, see “[Resource Type and Name List](#)” on page 14.

Windows for Font Families

This section discusses the Constructor windows used to create and edit font families.

Font Family Editor Window

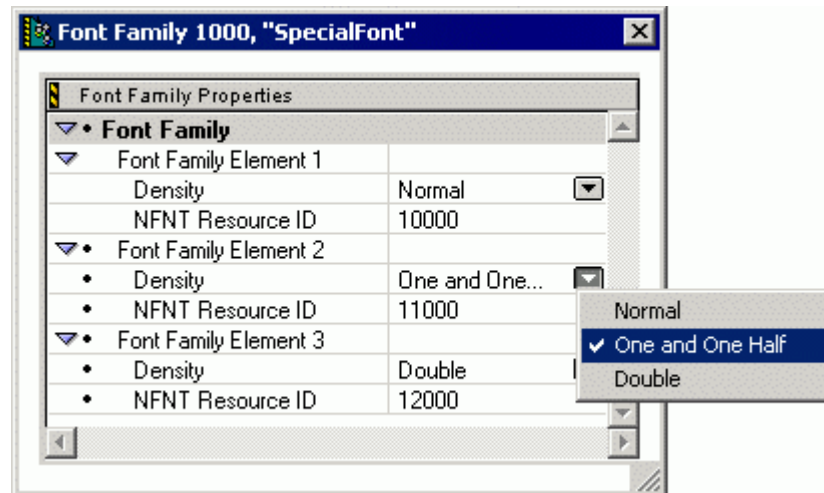
Use the Font Family Editor window to specify the information about NFNT resources you want included in a font family.

To open a font family in a font family editor window:

Do one of the following:

- Double-click a font family resource in the Project window.
- Select a font family resource in the Project window and press ENTER or RETURN.

Figure 7.2 Font Family Editor Window



Working with Sound Resources

This chapter describes how to use Constructor for Palm OS® to add sound resources to your application.

Working with Sound Resources Overview

With Constructor 1.8 and later, you can define sound resources that you can use in your application. Sound resources are stored in a Palm OS 'wave' resource.

When you create the sound resource, you specify an existing wave file (a file with a WAV extension) that will be included in the resource file for your application. Constructor does not support recording sounds or changing the format of existing sounds; Constructor simply uses the data from the wave file unchanged.

Palm OS supports uncompressed wave files (PCM) and wave files with IMA ADPCM compression. However, Constructor does not check whether the wave file is valid for Palm OS. If you use a wave file that has some other compression format, Constructor will accept it, but it will not play on the Palm OS at runtime.

This chapter provides instructions for how to manage and modify the defined sound resources, including:

- “[Creating a Sound Resource](#)” on page 104
- “[Windows for Sound Resources](#)” on page 104

Creating a Sound Resource

To create a wave sound resource:

1. Do one of the following:
 - Select the Wave Sounds icon in the Project window; then choose [Edit](#) > **New Wave Sound Resource**. A new, untitled wave sound resource appears in the Project window.
 - From anywhere in the Project window, choose [Edit](#) > **New Resource**. Select **Wave Sound** from the **Resource Type** pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to add the resource to the Wave Sounds section of the Project window.

For more information about using the Project window, see “[Resource Type and Name List](#)” on page 14.

2. Select the wave sound resource; then press RETURN or ENTER to open the wave sound in a Wave Sound Editor window.

For more information, see “[Wave Sound Editor Window](#)” on page 104.

3. Click **Browse** to select a wave file for the wave sound resource.

Windows for Sound Resources

This section discusses the Constructor windows used to add wave sound resources to your interface.

Wave Sound Editor Window

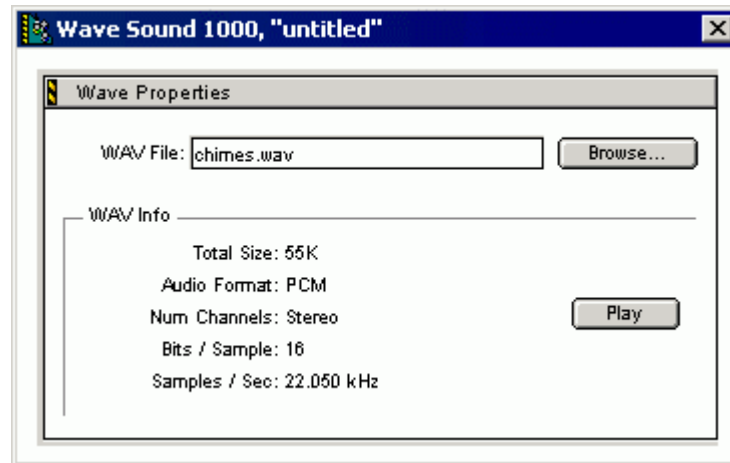
Use the Wave Sound Editor window to select the wave file you want to add to your application or to play the specified wave file.

To open a sound resource in a wave sound editor window:

Do one of the following:

- Double-click a wave sound resource in the Project window.
- Select a wave sound resource in the Project window and press ENTER or RETURN.

Figure 8.1 Wave Sound Editor Window



To specify sound resource properties in a wave sound editor window:

Do one of the following:

- Click **Browse** to select a wave file for the wave sound resource. Constructor opens the standard file open dialog for you to select a file with a WAV file extension.
- Enter a wave filename into the WAV File field.

NOTE: It is best to enter only the filename of the wave file, not a full path name or relative path name. PalmRez uses the CodeWarrior project access paths to locate the file when it compiles the wave file. If the wave file is in the same folder as the resource file (RSRC), or if the wave file is in another directory specified by the access paths, then the wave file will be found automatically if you enter the filename only. You can move your project more easily, both on your own computer and to another platform, by not using full path names or relative path names.

The properties of the wave sound resource consists only of the name of the wave file. The other properties displayed in the Wave Sound Editor window are implicitly specified by the wave file data.

Working with Sound Resources

Windows for Sound Resources

When you specify a wave filename, Constructor attempts to locate the file in the same directory as the Constructor project file.

- If Constructor locates the wave file, then Constructor reads the wave file and displays the properties of the wave file in the “WAV Info” section of the Wave Sound Editor Window. Constructor also enables the **Play** button, which you can use to preview the sound file.
- If Constructor is unable to locate the wave file or if Constructor cannot read the wave file, the “WAV Info” section of the Wave Sound Editor Window is left blank and the **Play** button is disabled.

The fact that Constructor cannot locate or read a wave file does not necessarily indicate a problem with the wave file. As long as the file can be located by CodeWarrior and by PalmRez when you compile your application, and as long as the wave file is in the file format supported by Palm OS, then the sound should play correctly in your application.

To play a sound resource in a wave sound editor window:

Click the **Play** button to play the selected wave file.

Working with Custom Data Resources

This chapter describes how to use Constructor for Palm OS® to add custom resource data to your application.

Working with Custom Data Resources Overview

With Constructor 1.9 and later, you can define custom data resources that you can use in your application. Custom data resources are stored in a resource type that you name yourself using the Custom Data Editor window.

You specify the custom data for your resource either by placing the data in a file or by manually entering hex-encoded binary data into the Custom Data Editor window.

A custom data resource can be used whenever you need to use a resource that isn't already provided by Constructor. For example, all QVGA-aware applications written for HandEra handhelds need to include a special 'sKst' resource. You can create an 'sKst' resource by using the Custom Data Editor window.

This chapter provides instructions for how to manage and modify custom data resources, including:

- “[Creating a Custom Data Resource](#)” on page 108
- “[Windows for Custom Data Resources](#)” on page 110

Creating a Custom Data Resource

To create a custom data resource:

1. Do one of the following:
 - Select the Custom Data icon in the Project window; then choose [Edit](#) > **New Custom Data Resource**. A new, untitled custom data resource appears in the Project window (shown in [Figure 9.1](#) on page 109).
 - From anywhere in the Project window, choose [Edit](#) > **New Resource**. Select **Custom Data** from the **Resource Type** pop-up menu, enter a Resource Name and Resource ID, and then click **Create** to add the resource to the Custom Data section of the Project window (shown in [Figure 9.1](#) on page 109).
2. Select the custom data resource; then press RETURN or ENTER to open the custom data in a Custom Data Editor window.
For more information, see “[Custom Data Editor Window](#)” on page 110.

Figure 9.1 Project with Custom Data Resources



For more information about using the Project window, see [“Resource Type and Name List”](#) on page 14.

Windows for Custom Data Resources

This section discusses the Constructor windows used to add custom data resources to your interface.

Custom Data Editor Window

Use the Custom Data Editor window to enter the information about custom data resources you want included in your application.

To open a custom data resource in a custom data editor window:

Do one of the following:

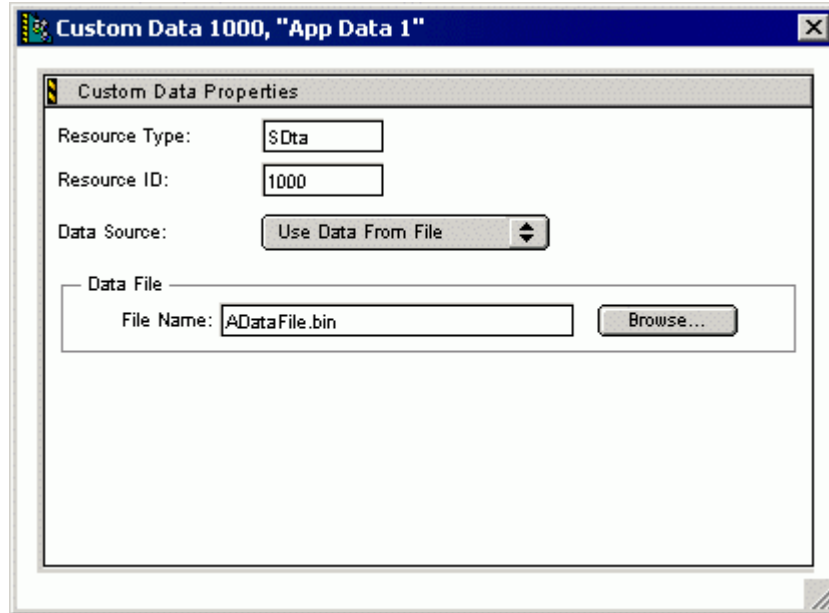
- Double-click a custom data resource in the Project window.
- Select a custom data resource in the Project window and press ENTER or RETURN.

To specify custom data resource properties:

There are two ways you can set the properties for your custom data resource:

- Specify the custom data by using a data file. This method uses the editor shown in [Figure 9.2](#) on page 111.
- Specify the custom data by using inline data. This method uses the editor shown in [Figure 9.3](#) on page 112.

Figure 9.2 Custom Data Editor Window with Data from a File

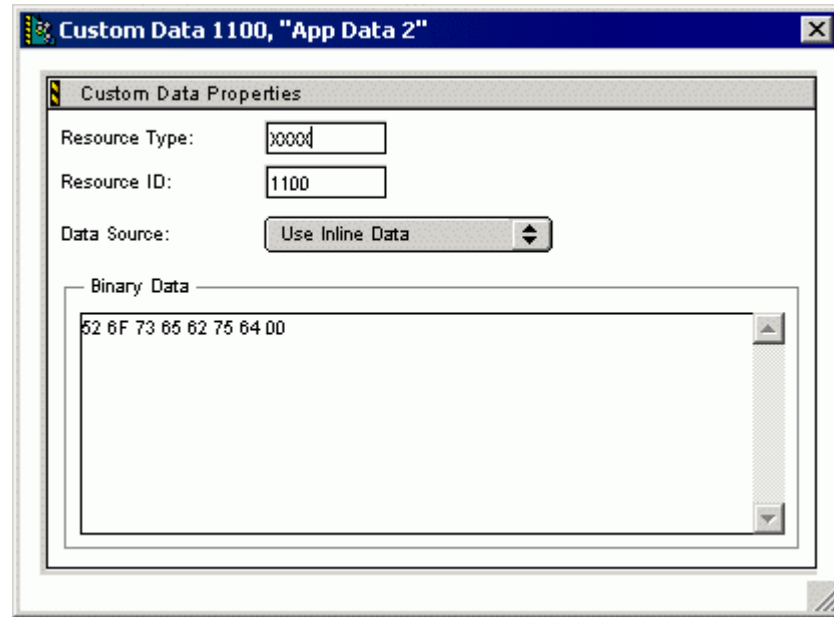


To specify custom data by using a data file:

1. Enter the **Resource Type** and **Resource ID** information.
2. Select **Use Data From File** from the Data Source menu.
3. Do one of the following:
 - Click **Browse** select the data file for the custom data resource. Constructor opens the standard file open dialog for you to select a file.
 - Enter a data file filename into the File Name field.

NOTE: It is best to enter only the filename of the data file, not a full path name or relative path name. PalmRez uses the CodeWarrior project access paths to locate the file when it compiles the data file. If the data file is in the same folder as the resource file (RSRC), or if the data file is in another directory specified by the access paths, then the data file will be found automatically if you enter the filename only. You can move your project more easily, both on your own computer and to another platform, by not using full path names or relative path names.

Figure 9.3 Custom Data Editor Window with Inline Data



To specify custom data by using inline data:

- Enter the **Resource Type** and **Resource ID** information.
- Select **Use Inline Data** from the Data Source menu.
- Enter the binary data into the **Binary Data** entry box.

Menu Reference

This chapter provides a description of each Constructor for Palm OS® menu item.

Constructor Menu Reference Overview

The Constructor menus include:

- [File](#)
- [Edit](#)
- [Window](#)
- [Arrange](#)
- [Layout](#)
- [Options](#)
- [Colors](#)
- [Font](#)
- [Style](#)

The menus available depend on the active resource editor and context in Constructor.

File

Use the **File** menu commands for managing Constructor project files.

New Project File

Creates a new Constructor project file.

See “[Constructor Project Files](#)” on page 8.

Open Project File...

Opens an existing Constructor project file.

See “[Constructor Project Files](#)” on page 8.

Close

Closes the active window. If you close the project window and there are unsaved changes, you are prompted to save before closing.

This item is enabled when there is a window open.

See “[Constructor Project Files](#)” on page 8.

Save

Saves the active project file and, if the header file generation option is selected, generates and saves an interface header file for the project. If the project hasn’t been saved yet, this command displays the standard file dialog box so you can specify a location for the file.

This item is enabled when there is a project open that has been changed without being saved.

See also “[Generate Header File](#)” on page 115, “[Constructor Project Files](#)” on page 8, and “[Header File Generation](#)” on page 12.

Save As...

Saves a project file using a new name, which may be saved in the same or another location.

This item is enabled when there is a project open.

See “[Constructor Project Files](#)” on page 8.

Generate Header File

Generates and saves an interface header file for the project. Use this command to generate and save the interface file without having to save the project file.

See also “[Save](#)” and “[Header File Generation](#)” on page 12.

Page Setup...

Displays the standard dialog box for choosing page setup options such as paper size and page orientation.

Print...

Prints the active Constructor window.

This item is enabled when there is a document open.

Quit (Macintosh)

Quits Constructor. If any open files have not been saved, you are prompted to save before quitting the application.

Exit (Windows)

Quits Constructor. If any open files have not been saved, you are prompted to save before quitting the application.

Edit

In addition to standard **Edit** menu commands, such as **Copy** and **Paste**, the Constructor **Edit** menu contains commands that let you create and modify resources, panes, menus, and so forth.

In addition, many **Edit** commands append a word to the basic command that reflects what you’re currently working with, for example *resource*, *pane*, or *menu*. In some cases, there may be two or more types of items listed. For example, when editing a menu resource, the **Edit** menu displays three commands to create new items related to menus, including **New Menu**, **New Menu Item**, and **New Separator Item**.

Undo/Redo

The **Undo/Redo** description varies based on your last action. You can toggle between undoing and redoing an action by selecting **Undo** or **Redo**. Constructor provides a single-level undo.

Cut

Removes a selection to the Clipboard.

Copy

Copies a selection to the Clipboard.

Paste

Pastes a copy of the Clipboard's contents into the active window.

Clear

Clears a selection from the active window.

Select All

Selects all items in the active window.

Duplicate

Makes a duplicate of the selected items in the active window.

New Item

Creates a new item. The menu text reflects the nature of the item. It may be a resource, a menu, a menu item, and so forth.

New Separator Item

Inserts a menu separator; available only when editing menus.

Remove Menu

Removes a menu from the menu bar without deleting the menu resource from the project. This item is only available when editing a menu bar.

Edit Resource

The **Edit Resource** command opens a resource editor window, for example, the view editor, the menu editor, the text traits editor, and so forth.

When an editor is already open and an item is selected in the editor (such as a menu item), selecting this command opens the Property Inspector window for the selected item.

Preferences

Displays a notebook for setting project preferences.

General Tab - Macintosh Only

Select **Automatically Set Keyboard Script in Edit Fields** to automatically set the keyboard script to match the target text encoding.

For example, when you are working with a Japanese resource file on an English system, if you click in a field with Japanese text, Constructor will set the keyboard script to the Japanese input system automatically. If you click in a field with Latin text, Constructor will set the keyboard script to the Roman input system automatically.

PRC Editing Tab

Sets the target operating release that will be used when editing a PRC file:

- Auto-detect. In most cases, Constructor for Palm OS will detect the target operating system. However, in some cases, you may get better results by manually setting the operating system target.
- Palm OS 3.0.x and earlier. Select this option for 3.0.x or earlier systems.
- Palm OS 3.1 and later. Select this option for Latin-encoded Palm OS 3.1 and later systems.
- Palm OS for Japan: Select this option if Japanese resources are used in the PRC that you are editing.
- Palm OS Chinese (Simp.): Select this option if Simplified Chinese resources are used in the PRC that you are editing.

The default is **Auto-detect**.

Window

The **Window** menu contains commands that display or activate various Constructor windows. Menu items are displayed as appropriate.

Property Inspector

Displays or activates the Property Inspector window for a related resource. Click the close box to close the Property Inspector window. For more information about the Property Inspector window, see “[Property Inspector Window](#)” on page 22.

Catalog

Displays or activates the Catalog window. Click the close box to close the Catalog window. For more information about the Catalog window, see “[Catalog Window](#)” on page 29.

Alignment Palette

Displays or activates the Alignment Palette window. Click the close box to close the Alignment Palette window.

Character Map

Displays or activates the Character Map window. Click the close box to close the Character Map window. For more information about the Character Map window, see “[Character Map Window](#)” on page 56.

Zoom Window

Zooms the active window, has the same effect as clicking the window’s zoom box.

Arrange

The **Arrange** menu is available when a form layout window is active. This menu provides tools for aligning and distributing items in the form.

Arrange Objects

Opens the Arrange Objects dialog box.

Align Left Edges

Aligns selected items along the left of the left most selected item.

Align Centers Horizontally

Aligns selected panes halfway between the left of the left most selected item and the right of the right most selected item.

Align Right Edges

Aligns selected items along the right of the right most selected item.

Spread Horizontally

Distributes selected items evenly from the left of the left most selected item to the right of the right most selected item.

Spread Horizontally in Container

Distributes selected items from the left edge to the right edge of the container that holds all the items.

Align Top Edges

Aligns selected items along the top of the top most selected item.

Align Centers Vertically

Aligns selected panes halfway between the top of the top most selected item and the bottom of the bottom most selected item.

Align Bottom Edges

Aligns selected items along the bottom of the bottom most selected item.

Spread Vertically

Distributes selected items evenly from the top of the topmost selected item to the bottom of the bottommost selected item.

Spread Vertically in Container

Distributes selected items from the top edge to the bottom edge of the container that holds all the items.

Make Object Group

Assigns the same value to the group ID field of the selected user interface items in a form.

Use this command to assemble push buttons, graphic push buttons, or check boxes into a group so that only one control may be selected at a time within the group. For more information on user interface items, see [Table 2.1](#).

You may also assign a group ID to a user interface item by changing the group ID field in the item's "[Property Inspector Window](#)" on page 22.

A group ID value of 0 means that the user interface item is not in a group.

Auto-Size Object

This menu is enabled for controls that support auto-sizing, such as buttons, checkboxes, fields, and popup triggers.

Auto-sizing does not reposition the control, but it may change the width and the height. For example, when you auto-size a button, the width will be changed to fit the button label text, and the height will be changed to fit the current text font.

If auto-sizing a control would extend it beyond the form's bounds, then the size of the control is limited to remain within the form's bounds.

Layout

The **Layout** menu is available when a form layout window is active. Items in this menu control the layout grid, and item visibility. Most items in the menu are enabled at all times.

Show Grid/Hide Grid

Displays or hides a dotted rectangular grid, respectively.

Snap to Grid/Don't Snap to Grid

Snaps an object to the nearest grid line when the mouse button is released. The **Don't Snap to Grid** command lets you move objects without snapping them to the grid.

Edit Grid

Lets you edit the size of the grid, in pixels.

Show Object IDs/Hide Object IDs

Displays or hides user interface item resource IDs in the Layout Appearance section of the form, respectively

Show Object Edges/Hide Object Edges

Displays or hides the edges of panes that are otherwise invisible. Displaying edges shows the size of unboxed items, making it easier to adjust and avoid overlaps.

Show Invisible Objects/Hide Invisible Objects

Displays or hides items that have the *Usable* check box cleared, respectively.

Show Object Hierarchy

Opens the Hierarchy for Form window if it is not already displayed. Displays the hierarchy of objects and object types. To close the Hierarchy window, click the window's close box. This command is disabled if the Hierarchy window is open and active.

Options

The **Options** menu is available when a bitmap editor is active. Most items in the menu are enabled when pixels are selected in the editor.

Flip Vertical

Turns the selected pixels upside down (rotates the selection 180°).

Flip Horizontal

Swaps selected pixels left and right. This command creates a mirror image of the pixels.

Rotate Right

Rotates selected pixels clockwise.

Rotate Left

Rotates selected pixels counter-clockwise.

Make Transparent

Sets the background color to be transparent.

Invert Pixels

Makes white pixels black and black pixels white; available only when a selection is active.

Delete Image

This command is available to Palm OS[®] software icons and bitmap images.

Set Image Size

Displays a dialog that lets you set the image size in pixels. To have the image resized to fit the new image size, select **Stretch** before clicking **Resize**.

Colors

The **Colors** menu is available when the bitmap editor is active.

Black & White (1-bit depth)

Set the foreground color to black and the background color to white.

4 Grays (2-bit depth)

Sets the color swatch pop-up to 4 shades of gray.

16 Grays (4-bit depth)

Sets the color swatch pop-up to 16 shades of gray.

256 Colors (8-bit depth)

Sets the color swatch pop-up to 256-color palette.

Thousands of Colors (16-bit depth)

Sets the color swatch pop-up to use the color dialog box.

Swap Fore & Back Colors

Set the foreground color to the current background color and vice versa.

Font

The Font menu is available when the bitmap editor is active. The font menu lists all available fonts on your computer.

Style

The Style menu is available when the bitmap editor is active and contains commands to control text style, justification, and font size.

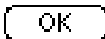
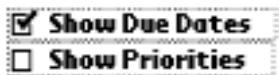


A

Catalog Resources

This appendix provides detailed descriptions of the attributes you set for the Constructor for Palm OS® catalog resources.


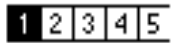





Catalog resources are available in Constructor's Catalog window and can be dragged directly on a form. [Table A.1](#) lists the available catalog resources. The Macintosh ResEdit resource name is included for reference only; it's not needed by developers who use Constructor exclusively and not relevant for Windows developers.

Table A.1 Catalog Resources

Name	Resource	Resource
tBTN	Button Resource	
tCBX	Check Box Resource	
tSLF	Feedback Slider Resource	
tFLD	Field Resource	Look Up: Text.....
tFBM	Form Bitmap Resource	(container for Bitmap resource)
tGDT	Gadget Resource	(application defined)
tGSI	Graffiti Shift Indicator Resource	
tGBN	Graphic Button Resource	
tGPB	Graphic Push Button Resource	
tGRB	Graphic Repeating Button Resource	

Catalog Resources

Table A.1 Catalog Resources (*continued*)

Name	Resource	Resource
tLBL	Label Resource	Set Date:
tLST	List Resource	
tPUT	Popup Trigger Resource	▼ Unfiled
tPBN	Push Button Resource	: 
tREP	Repeating Button Resource	
tSCL	Scroll Bar Resource	
tSLT	Selector Trigger Resource	
tsld	Slider Resource	
tTBL	Table Resource	

Button Resource

The button resource creates a command button. Most forms have a single row of command buttons at the bottom of the form.

Table A.2 Button resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the button in application code.
Button ID	ID assigned by Constructor
Left Origin	1 for left-most button on modeless form. 5 for left-most button on modal form. Determine left origin for button n by: $button(n-1) \text{ left} + button(n-1) \text{ width} + 6$
Top Origin	147 or 160 – $height$ – 1 for modeless forms $formHeight - buttonHeight - 5$ for modal forms
Width	36 or $label \text{ width} + 10$
Height	12 or $font \text{ height} + 3$ if not using standard font
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Anchor Left	Controls how the object resizes itself when its text label is changed. If checked, the left bound of the object is fixed; if unchecked, the right bound is fixed.
Frame	Checked
Non-bold Frame	Checked
Font	Use Standard font for most buttons.
Label	Text displayed inside the button.

Check Box Resource

The check box resource creates a check box. Check boxes often appear in a group. If so, the group should be non-exclusive (so that the user can enable multiple check boxes) and should be aligned vertically or horizontally. If you want to make sure that only one of a series of options is selected, use push buttons instead of check boxes.

Table A.3 Check box resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the button in application code.
Checkbox ID	ID assigned by Constructor
Left Origin	Form-relative position of left side of object.
Top Origin	Form-relative position of top of object.
Width	Determined at runtime.
Height	12 pixels minimum Height determines tappable area Check box is always centered in chosen height
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Selected	Initial selection state of the check box. If this box is checked (the default), the check box is initially checked.
Group ID	Do not use.
Font	Use Bold font for most check boxes.
Label	Text displayed to the right of the check box.

Feedback Slider Resource

The feedback slider resource creates a feedback slider. The difference between a feedback slider and a regular slider is that feedback sliders send `ctlRepeatEvents` while the user drags the slider thumb. The regular slider sends a single `ctlSelectEvent` when the user is done dragging the thumb.

Table A.4 Feedback slider resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the slider in application code.
SliderControl ID	ID assigned by Constructor
Left Origin	Form-relative position of the left side of the object.
Top Origin	Form-relative position of top of object.
Width	The default background bitmap looks best at these widths: 72, 93, 114, 135, or 156
Height	15 or height of thumb bitmap
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Initial Value	Initial setting of the slider thumb.
Minimum Value	Minimum value the slider can represent.
Maximum Value	Maximum value the slider can represent.
Page Jump Amount	Amount by which the thumb moves if the user taps to its left or right.

Table A.4 Feedback slider resource attributes (*continued*)

Attribute	Description
Thumb Bitmap	ID of the bitmap resource or bitmap family resource used to create the thumb. Use 0 to use the default thumb bitmap.
Background Bitmap	ID of the bitmap resource or bitmap family resource used to create the background. Use 0 to use the default background bitmap.

Field Resource

Use the field resource to create an editable text field that is either a single-line long or multiple lines long. You can also use the field resource to create noneditable text that is displayed on the form. It is easier to use noneditable text fields instead of labels if the text of the label changes dynamically.

Table A.5 Field resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the field in application code.
Field ID	ID assigned by Constructor
Left Origin	Form-relative position of left side of object.
Top Origin	Form-relative position of top of object.
Width	Ideally wide enough to show maximum characters
Height	11 or <i>font height</i> + 2 for single-line fields For multi-line fields, a multiple of the line height.
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Editable	If this box is checked, the field is editable. Noneditable fields don't accept user input but can be changed programmatically.

Table A.5 Field resource attributes (continued)

Attribute	Description
Underline	Check this box for editable fields. Uncheck for noneditable fields.
Single Line	Check this box for single-line fields so the field doesn't scroll horizontally and doesn't accept Return or Tab characters.
Dynamic Size	Uncheck this box if the Single Line box is checked. Check this box if you want the field to grow as the user enters more characters.
Left Justified	Left-justify editable single-line fields. Right-justify single-line numeric fields or noneditable fields used as labels. Not applicable to multi-line fields.
Max Characters	Maximum number of bytes that the user can enter into an editable field. Note that the number of bytes is not the same as the number of characters when multi-byte characters (for example, Japanese) are being used. This attribute has no effect on noneditable fields.
Font	Font used to draw the text in the field.
Auto Shift	If checked, Graffiti or Graffiti® 2 auto-shift rules are applied. This attribute should be enabled for most editable fields.
Has Scroll Bar	Check this box if you want to associate a scroll bar with a multi-line field.
Numeric	If checked, only the characters 0 through 9 and associated separators are allowed to be entered in the field. The associated separators are the thousands separator and the decimal character. The values of these two characters depend on the settings in the Formats preferences panel. Note that numeric fields do not allow plus signs.

Form Bitmap Resource

Use the form bitmap resource to display a bitmap at a fixed location on a form. For example, an about dialog often displays the application icon along with the application name, version number, and other pertinent information. You can use a form bitmap resource to display the icon on this dialog.

Table A.6 Form bitmap resource attributes

Attribute	Description
Object ID	ID assigned by Constructor
Object Identifier	Name you assign to refer to the object in application code.
Left Origin	Form-relative position of left side of object.
Top Origin	Form-relative position of top of object.
Bitmap Resource ID	ID of a bitmap resource or bitmap family resource to be drawn in the specified location.
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.

Gadget Resource

A gadget resource creates an application-defined resource. You must write code to draw the user interface element and respond to events associated with that element.

Table A.7 Gadget resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the object in application code.
Gadget ID	ID assigned by Constructor.

Table A.7 Gadget resource attributes (*continued*)

Attribute	Description
Left Origin	Form-relative position of left side of object.
Top Origin	Form-relative position of top of object.
Width	Width in pixels.
Height	Height in pixels
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.

Graffiti Shift Indicator Resource

The Graffiti Shift Indicator (GSI) resource should be displayed on all forms that contain an editable text field. After you place the GSI on a form, the system automatically displays the current shift status at that location.

Table A.8 Graffiti Shift Indicator resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the object in application code.
Left Origin	135 (modeless form) 142 (modal form)
Top Origin	150 (modeless form) <i>form height</i> – 13 (modal form)
Object ID	ID assigned by Constructor

Graphic Button Resource

The graphic button resource creates a command button whose label is a bitmap rather than a text label.

Table A.9 Graphic button resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the button in application code.
Graphic Button ID	ID assigned by Constructor
Left Origin	1 for left-most button on modeless form. 5 for left-most button on modal form. Determine left origin for button n by: $button(n-1) \text{ left} + button(n-1) \text{ width} + 6$
Top Origin	147 or 160 – <i>height</i> – 1 for modeless forms $formHeight - buttonHeight - 5$ for modal forms
Width	36 or $label \text{ width} + 10$
Height	$bitmap \text{ height} + 3$
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Anchor Left	Controls how the object resizes itself when its bitmap is changed. If checked, the left bound of the object is fixed; if unchecked, the right bound is fixed.
Frame	Checked
Non-bold Frame	Checked

Table A.9 Graphic button resource attributes (continued)

Attribute	Description
Bitmap Resource ID	Resource ID of the bitmap family that draws the graphic on this button. Assign a resource ID and click the Create button to create the graphic.
Selected Bitmap Resource ID	Resource ID of the bitmap family that draws the graphic when the button is selected. It's important to provide a different graphic for the selected bitmap on a color screen. The default is to invert the pixels, which draws funny on a color screen.

Graphic Push Button Resource

Use the graphic push button resource to create a push button whose label is a bitmap rather than text. You usually create more than one push button at a time. Push buttons should be aligned horizontally or vertically and have no space between.

Table A.10 Graphic push button resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the button in application code.
PushButton ID	ID assigned by Constructor
Left Origin	For a horizontal row of push buttons, the position of button(<i>n</i>) is: <i>button(n-1) left + button(n-1) width + 1</i>
Top Origin	All push buttons in same group have same top (if horizontal row).
Width	<i>label width + 2</i> minimum All push buttons in same group have same width.
Height	<i>bitmap height + 3</i> All push buttons in same group have same height.

Table A.10 Graphic push button resource attributes

Attribute	Description
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Group ID	Nonzero value between 1 and 65535 to identify the group. If 0, the push button is not assigned to a group.
Bitmap ID	Resource ID of the bitmap family that draws the graphic on this button. Assign a resource ID and click the Create button to create the graphic.
Selected Bitmap ID	Resource ID of the bitmap family that draws the graphic when the button is selected. It's important to provide a different graphic for the selected bitmap on a color screen. The default is to invert the pixels, which draws funny on a color screen.

Graphic Repeating Button Resource

Use the graphic repeating button resource to create a repeating button whose label is a bitmap rather than text. A repeating button can look like a command button or not. The difference between the two is that the repeating button sends events repeatedly while the user holds the pen down on the button. Command buttons wait until the user releases the pen and then send a single event.

Table A.11 Graphic repeating button resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the button in application code.
Graphic Repeating Button ID	ID assigned by Constructor

Table A.11 Graphic repeating button resource attributes (*continued*)

Attribute	Description
Left Origin	Form-relative position of left side of object.
Top Origin	Form-relative position of top of object.
Width	Width of object in pixels
Height	Height of object in pixels
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Anchor Left	Controls how the object resizes itself when its bitmap is changed. If checked, the left bound of the object is fixed; if unchecked, the right bound is fixed.
Frame	Uncheck if you don't want a frame to appear around the button. Repeating buttons typically don't have frames.
Non-bold Frame	Check if you have Frame checked and you want the frame to be a 1-pixel border.
Bitmap Resource ID	Resource ID of the bitmap family that draws the graphic on this button. Assign a resource ID and click the Create button to create the graphic.
Selected Bitmap Resource ID	Resource ID of the bitmap family that draws the graphic when the button is selected. It's important to provide a different graphic for the selected bitmap on a color screen. The default is to invert the pixels, which draws funny on a color screen.

Label Resource

Use a label to display noneditable text or labels on a form.

Table A.12 Label resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the object in application code.
Label ID	ID assigned by Constructor
Left Origin	Form-relative position of left side of object. Leave 3 pixels of white space between the label and its control.
Top Origin	Form-relative position of top of object. Align labels with the top of the object they label.
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Font	Use Bold font for most labels.
Text	Text of the label. Typing a carriage return places a carriage return in the label text.

List Resource

Use a list resource to create a stand-alone list or the list portion of a pop-up list. (The pop-up trigger resource defines the trigger for the pop-up list.)

Table A.13 List resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the list in application code.
List ID	ID assigned by Constructor.

Table A.13 List resource attributes (*continued*)

Attribute	Description
Left Origin	Form-relative position of left side of object. If creating pop-up list, use <i>trigger left</i> if trigger is left-aligned or <i>trigger left – list width</i> if trigger is right-aligned.
Top Origin	Form-relative position of top of object. If creating pop-up list, top is often set to <i>trigger top</i> . Palm OS repositions the list if necessary.
Width	5 + width of longest item label 10 + width of longest item label to allow scrolling If creating a Categories pop-up list, the width is set at runtime. Prior to Palm OS 3.5, if the list is narrower than one of its items, the list items would draw outside of the list. Take care to ensure that this does not occur with your interface.
Usable	If creating a pop-up list, uncheck this box.
Font	Use Standard font for most lists.
Visible Items	Number of items the list displays. 13 items for a full screen pop-up list 11 items for a full screen stand-alone list (modeless form) 10 items for a full screen stand-alone list (modal form) 0 for the Categories pop-up list (its size is determined at runtime) Set to the total number of items in your list if less than these numbers.
List Items	Text for the items to appear in the list. Type CTL+K or CMD+K to add items. Leave blank if you add items programmatically.

Popup Trigger Resource

A pop-up trigger resource creates the pop-up trigger portion of a pop-up list.

Table A.14 Pop-up trigger resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the pop-up trigger in application code.
Popup ID	ID assigned by Constructor.
Left Origin	Form-relative position of left side of button. Use 160 to align trigger with right edge of screen if it resizes to the left (Use this for Categories pop-up trigger.)
Top Origin	Form-relative position of top of trigger.
Width	Determined at runtime. For editing purposes, you can set to several pixels wider than the list to be sure you can select the trigger separately from the list.
Height	12 or <i>font height</i> + 3
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Anchor Left	Check this box if the trigger should resize to the right when the label changes. If the trigger is on the right edge of the screen and should resize to the left, uncheck this box. Uncheck this box for the Categories pop-up trigger.
Font	Use Standard font for most triggers.

Table A.14 Pop-up trigger resource attributes (continued)

Attribute	Description
Label	Usually leave this field blank and set the label programmatically.
List ID	ID of the list resource to be popped up when the user taps the trigger.

Push Button Resource

A push button resource creates a push button. A group of push buttons represents a set of options where only one option can be selected at a time. You typically create several push buttons aligned horizontally or vertically.

Table A.15 Push button resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the button in application code.
PushButton ID	ID assigned by Constructor
Left Origin	For a horizontal row of push buttons, the position of button(<i>n</i>) is: <i>button(n-1) left + button(n-1) width + 1</i>
Top Origin	All push buttons in same group have same top (if horizontal row)
Width	<i>label width + 2</i> minimum All push buttons in same group have same width.
Height	12 All push button in same group have same height.
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.

Table A.15 Push button resource attributes (*continued*)

Attribute	Description
Group ID	Nonzero value between 1 and 65535 to identify the group. If 0, the push button is not assigned to a group.
Font	Use Standard font for most push buttons.
Label	Text displayed inside the push button.

Repeating Button Resource

The repeating button resource creates a repeating button. Repeating buttons can look identical to command buttons or not. The difference between the two is that the repeating button sends events repeatedly while the user holds the pen down on the button. Command buttons wait until the user releases the pen and then send a single event.

The most common use of repeating buttons is to draw the scroll buttons in the lower-right portion of a form. They are also often used as increment decrement arrows.

Table A.16 Repeating button resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the button in application code.
Button ID	ID assigned by Constructor
Left Origin	Form-relative position of left side of object. 147 for scroll buttons
Top Origin	Form-relative position of top of object. 144 for scroll up button 152 for scroll down button
Width	Width of object in pixels 13 for scroll buttons

Table A.16 Repeating button resource attributes *(continued)*

Attribute	Description
Height	Height of object in pixels 8 for scroll buttons
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Anchor Left	Controls how the object resizes itself when its label is changed. If checked, the left bound of the object is fixed; if unchecked, the right bound is fixed.
Frame	Uncheck if you don't want a frame to appear around the button. Repeating buttons typically don't have frames.
Non-bold Frame	Check if you have Frame checked and you want the frame to be a 1-pixel border.
Font	Font used to draw the repeating button label. Use Symbol 7 font for scroll buttons.
Label	Text that appears inside the repeating button. Use 0x01 for the scroll up button. Use 0x02 for the scroll down button.

Scroll Bar Resource

Use the Scroll Bar Resource to create a scroll bar.

Table A.17 Scroll bar resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the scroll bar in application code.
Scrollbar ID	ID assigned by Constructor

Table A.17 Scroll bar resource attributes (*continued*)

Attribute	Description
Left Origin	153
Top Origin	Align with top of element to be scrolled
Width	7
Height	Height of element that the scroll bar scrolls
Usable	Check this box. The system ensures that the scroll bar only appears when the element that it is associated with is scrollable.
Value	Initial value of the scroll car. Typically 0 or the same as the minimum.
Minimum Value	Typically 0.
Maximum Value	Typically not set until run time. The number of lines it should take to scroll to the end of the element.
Page Size	Number of lines to scroll at one time. This value is often not set until run time. It should be at least one less than the number of lines that can be displayed at one time to provide context. That is, if the field can display ten lines of text, the page jump value should be nine.

Selector Trigger Resource

Use the selector trigger resource to create a selector trigger.

Table A.18 Selector trigger resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the trigger in application code.
Selector Trigger ID	ID assigned by Constructor

Table A.18 Selector trigger resource attributes (*continued*)

Attribute	Description
Left Origin	Form-relative position of the left side of the object.
Top Origin	Form-relative position of top of object.
Width	Determined at run time.
Height	13 pixels or odd number so that the corner pixels are black
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Anchor Left	Check if the trigger should resize to the right when the label changes. If the trigger is on the right edge of the screen and should resize to the left, uncheck this box.
Font	Use Standard font for most selector triggers.
Label	The initial value for the pop-up trigger. Usually leave blank and set at run time.

Slider Resource

The slider resource creates a slider.

Table A.19 Slider resource attributes

Attribute	Description
Object Identifier	Name you assign to refer to the slider in application code.
SliderControl ID	ID assigned by Constructor
Left Origin	Form-relative position of the left side of the object.

Catalog Resources

Slider Resource

Table A.19 Slider resource attributes (*continued*)

Attribute	Description
Top Origin	Form-relative position of top of object.
Width	The default background bitmap looks best at these widths: 72, 93, 114, 135, or 156
Height	15 or height of thumb bitmap
Usable	Uncheck this box if you don't want the object to appear on the screen initially. Nonusable objects can programmatically be set to usable.
Initial Value	Initial setting of the slider thumb.
Minimum Value	Minimum value the slider can represent.
Maximum Value	Maximum value the slider can represent.
Page Jump Amount	Amount by which the thumb moves if the user taps to its left or right.
Thumb Bitmap	ID of the bitmap resource or bitmap family resource used to create the thumb. Use 0 to use the default thumb bitmap.
Background Bitmap	ID of the bitmap resource or bitmap family resource used to create the background. Use 0 to use the default background bitmap.

Table Resource

Use the table resource to create a table.

Table A.20 Table resource attributes

Attribute	Value
Object Identifier	Name you assign to refer to the table in application code.
Table ID	ID assigned by Constructor
Left Origin	0 for modeless form 4 for modal form
Top Origin	18
Width	160 (modeless form) 148 (modal form)
Height	120 pixels (modeless form) 108 pixels (modal form)
Editable	Check this box if the user can modify the table. Uncheck this box if the user can only select items in the table.
Rows	Number of rows in the table. This value does not change at run time. Instead, you write code that changes the data that each row displays.
Column Widths	The number of columns in the table and each column's width. The widths must add up to the table width. Use CTL+K or CMD+K to add a new column.

Palm OS Resource Types

This appendix lists the Palm OS® resource types. This information is included for developers who may be using other tools to edit Constructor resources.

Palm OS Resource Types

Table B.1 Resources Used by Palm OS1

Palm Resources	Palm OS Resource Type	Constructor for Palm OS Resource Types
Form	tFRM	tFRM + (tBTN tCBX tFBM tFLD tgbn tGDT tgpB tgrb tGSI tLBL tLST tPBN tPUL tPUT tREP tSCL tsld tslf tSLT tTBL)
Alert	Talt	Talt
Menu bar	MBAR	MBAR + MENU
String	tSTR	tSTR
String list	tSTL	tSTL
Bitmap	Tbmp	(tbmf + PICT) PICT
Version string	tver	tver
Application icon name	tAIN	tAIN
Default application category	taic	taic
App Info string list (categories)	tAIS	tAIS
Application icon	tAIB	(taif + PICT) ICON cicn
Application preference	pref	pref
Application extended preference	xprf	xprf

Table B.2 Palm OS Form Objects

Form Object	Constructor for Palm OS Resource Types
Button	tBTN
Check box	tCBX
Feedback slider control	tslf
Field	tFLD
Form bitmap	tFBM
Gadget	tGDT
Graffiti shift indicator	tGSI
Graphic button	tgbn
Graphic push button	tgpb
Graphic repeating button	tgrb
Label	tLBL
List	tLST
Popup trigger	tPUT, tPUL
Push button	tPBN
Repeating button	tREP
Scroll bar	tSCL
Selector trigger	tSLT
Slider control	tsld
Table	tTBL

Index

Numerics

- 1.5X density screens x
- 16-bit color bitmaps 6, 92
- 1-bit bitmaps 92
- 2-bit grayscale bitmaps 6, 92
- 4-bit grayscale bitmaps 6, 92
- 8-bit color bitmaps 6, 92

A

- alert
 - creating and modifying 63
 - definition 5
- alignment palette window 118
- app icon family editor 76
- app info string list 5
- app info string list editor window 55
- application icon 18, 150
- application icon families 75
- application icon name option 17
- application icon option 18
- Arrange menu, Constructor 118
- ASCII 61
- auto generate header file option 18
- auto-detect 117
- Auto-Shift (field) 131

B

- bitmap
 - compressed 81
 - creating and modifying 71
 - definition 5
 - grayscale 81
- bitmap editor 71–97
- bitmap families 72
- bitmaps
 - rotating 95
- black and white bitmaps 92
- bootscreen bitmap family resource 7
- BPRC file 9
- bug fixes ix

C

C

- header file, generating 12, 18
- catalog resources 125
- catalog window 29, 118
- CategoryInitialize() 5, 55
- character map window 56, 118
- character set
 - ASCII 61
 - Palm OS 61
- character string
 - creating and modifying 53
 - definition 4
- character string list 4
- check box, group 30
- Chinese text encoding x
- clockwise rotation 95
- CodeWarrior IDE
 - using Constructor with xii
- color bitmaps 6, 92
- color tool 92
- color, setting in bitmap 95
- Colors menu, Constructor 122
- compression 81
- Constructor
 - catalog resources 125
 - version 1.6 xi
 - version 1.7 x
 - version 1.8 x
- controls
 - user interface 30
 - button 30
 - check box 30
 - feedback slider 30
 - field 30
 - form bitmap 30
 - gadget 31
 - graffiti shift indicator 31
 - graphic button 31
 - graphic push button 31
 - graphic repeating button 31
 - label 31
 - list 31
 - popup trigger 31
 - push button 31
 - repeating button 31
 - scroll bar 31

- selector trigger 32
- slider 32
- table 32
- copying pixels, bitmaps 94
- copying selections 25
- counter-clockwise rotation 95
- custom data resource 107

D

- default application category option 17
- deleting items 26
- double density bitmap 73
- double density font 100
- drag and drop in bitmaps 96
- dropper tool 89
- duplicating items 26

E

- Edit menu, Constructor 115
- edit>preferences 117
- editing bitmaps 93–97
 - copying pixels 94
 - drag and drop 96
 - nudging pixels 94
 - rotating pixels 95
 - selecting pixels 94
 - setting color 95
 - setting pattern 95
- editing PRC files 9, 117
- editing text items 27
- editor window
 - opening 21
- extended font resource 99

F

- families
 - application icon 75
 - bitmap 72
 - font 99
- feature resource 7
- field resource
 - Auto-Shift 131
 - Has Scrollbar 131
- File menu, Constructor 114

- file privileges 15
- filled oval tool 90
- font
 - Palm OS 61
- font editor 99
- font families 99
- font family editor window 102
- font family resource 99
- font resource 99
- font resource ID 100
- form
 - definition 3
 - ID consistency 19

G

- generate application resources option 17
- generate pref resource option 20
- generate xprf resource option 21
- generating header file 18
- generating source code 12, 18
- getting started ix
- graphic push button, group 31
- grayscale bitmaps 6, 92
- group 120
- group ID 120

H

- Has Scrollbar (field) 131
- header file name option 19
- header file, generating 12, 18
- help text
 - creating and modifying 53
 - definition 4
- hierarchy window 33

I

- icon
 - definition 5
 - for application 18, 80, 150
- include details in header option 19

K

- keep IDs in sync option 19

L

- large application icons 75
- lasso tool 85
- Layout menu, Constructor 120
- Layout window
 - for alerts 63
 - for forms 32
- line tool 89
- lists
 - in Constructor 26
 - opening and closing 27
 - selecting items in 27
- locale setting resource 7

M

- Mac OS xii
- marquee tool 86
- 'MBAR' resource 4, 150
- menu
 - definition 4
 - deleting an item 52
 - editor window 47
 - removing an item 52
 - removing from menu bar 50
- menu bar
 - definition 4
 - deleting a menu 50
 - editing 48–52
 - editor window 45
 - removing a menu 50
- Microsoft Windows xi
- moving selections 25
- multibit icons 80

N

- new features x
- 'NFNT' resource 99
- 'nfnt' resource 99
- normal density bitmaps 73
- normal density font 100
- nudging pixels 94

O

- object group 120
- one and one half density bitmap 73
- one and one half density font 100
- one and one half density resources x
- operating system options
 - auto-detect 117
 - Palm OS
 - 3.0x target 19, 117
 - 3.1 target 19, 117
 - Chinese (Simp.) target 20, 117
 - for Japan target 19, 117
- OPRC file 9
- options
 - application icon name 17
 - application icon option 18
 - auto generate header file 18
 - default application category 17
 - generate application resources 17
 - generate pref resource 20
 - generate xprf resource 21
 - header file name 19
 - include details in header 19
 - keep IDs in sync 19
 - Palm OS target 19
 - pref stack size 20
 - version string option 17
 - xprf flag no overlay 21
- Options menu, Constructor 121
- overlay resource 7

P

- paint bucket tool 88
- Palm OS
 - 3.0x target 19, 117
 - 3.1 target 19, 117
 - character set 61
 - Chinese (Simp.) target 20, 117
 - fonts 61
 - for Japan target 19, 117
 - resource 2
 - using Constructor for xii
- Palm OS target option 19
- pattern tool 91
- pattern, setting in bitmap 95

- PCM wave files 103
- pencil tool 87
- picture
 - creating and modifying 71
- PRC file 9
- 'pref' resource 20, 150
- pref stack size option 20
- preferences
 - autodetect 117
- preferences for PRC editing 117
- project file
 - definition 8
 - managing 8
- Project Settings 16
 - Palm OS Target option 19
- project window
 - definition 13
- property inspector 22–23
- property inspector window 118
- push button, group 31, 120

Q

- QVGA support x

R

- radio button, group 31, 120
- read/write permission 15
- Release Notes folder ix
- requirements xi
- resource
 - alert 5
 - app info string list 5
 - application extended preferences 21, 150
 - application icon 18, 150
 - application icon name 3, 17
 - application preferences 20, 150
 - application version 3, 17
 - bitmap 5
 - character string 4
 - character string list 4
 - defined 2
 - form 3
 - form ID 19
 - help text 4
 - icon 5

- ID, consistency 19
 - menu 4
 - menu bar 4
 - version 3
- resource ID for fonts 100
- resource types
 - 'MBAR' 4, 150
 - 'NFNT' 99
 - 'nfnt' 99
 - 'pref' 20, 150
 - 'sKst' 107
 - 'tAIB' 75
 - 'taic' 17, 150
 - 'tAIN' 17, 150
 - 'tAIS' 5, 55, 150
 - 'Talt' 5, 150
 - 'Tbmp' 72, 150
 - 'tFRM' 3, 150
 - 'tSTL' 4, 54, 150
 - 'tSTR' 4, 53, 54, 150
 - 'tver' 17, 150
 - 'wave' 103
 - 'xprf' 21, 150
- bootscreen bitmap family 7
- custom data 107
- defined. See also individual resource types
- feature 7
- in project window 14
- locale setting 7
- overlay 7
- short integer list 7
- soft constant 7

- Resource.frk 10
- rotating bitmaps 95
- rotating pixels, bitmaps 95

S

- selecting pixels, bitmaps 94
- selections
 - copying 25
 - making 24
 - moving 25
- short integer list resource 7
- Simplified Chinese support x
- single density font 100
- 'sKst' resource 107

- small application icons 75
- soft constant resource 7
- source code, generating 12, 18
- string
 - See character string.
- string editor window 54
- string list editor window 54
- SysStringByIndex() 5, 55
- system requirements xi

T

- 'tAIB' resource 75
- 'taic' resource 150, 17
- 'tAIN' resource 17, 150
- 'tAIS' resource 5, 55, 150
- 'Talt' resource 5, 150
- target options 19
- 'Tbmp' resource 72, 150
- text editing 27
- text tool 86
- 'tFRM' resource 3, 150
- 'tSTL' resource 4, 54, 150
- 'tSTR' resource 4, 53, 54, 150
- 'tver' resource 17, 150

U

- UI controls 30
- UI resources 125–147
- undo in Constructor 27
- user interface controls 30
 - button 30
 - check box 30
 - feedback slider 30
 - field 30

- form bitmap 30
- gadget 31
- graffiti shift indicator 31
- graphic button 31
- graphic push button 31
- graphic repeating button 31
- label 31
- list 31
- popup trigger 31
- push button 31
- repeating button 31
- scroll bar 31
- selector trigger 32
- slider 32
- table 32
- user interface features 24

V

- version control system
 - using Constructor with 8
- version string option 17
- View as Hex 61

W

- 'wave' resource 103
- Window menu, Constructor 118
- Windows xi
- windows
 - catalog 29
 - project 13

X

- xprf flag no overlay option 21
- 'xprf' resource 21, 150

