PackageMaker User Guide

Tools > Files & Software Installation

2009-01-06

Contents

Introduction **Introduction to PackageMaker User Guide 7**

Organization of This Document 7
See Also 8

Chapter 1 **Packaging Overview 9**

Chapter 2 **Packaging Workflow 13**

Create a PackageMaker Project 14
Identify the Product Components 14
Configure Component Packages 15
    Component Package Configuration Pane 15
    Component Package Contents Pane 16
    Component Package Components Pane 17
    Component Package Scripts Pane 18
Configure the Product Package 19
    Product Package Configuration Pane 19
    Product Package Requirements Pane 19
    Product Package Actions Pane 20
Configure the Product Package Install Choices 21
Build the Product Package 23

Glossary 25

Document Revision History 27

Figures and Tables

Chapter 1 **Packaging Overview 9**

Figure 1-1 The packaging process 10

Chapter 2 **Packaging Workflow 13**

Figure 2-1 PackageMaker project window 13
Figure 2-2 Install Properties dialog 14
Figure 2-3 Component Package Configuration pane 15
Figure 2-4 Component Package Contents pane 16
Figure 2-5 Component Package Components pane 17
Figure 2-6 Component Package Scripts pane 18
Figure 2-7 Product Package Configuration pane 19
Figure 2-8 Product Package Requirements pane 20
Figure 2-9 Product Package Actions pane 21
Figure 2-10 Choice Configuration pane 22
Figure 2-11 Choice requirement editor 23
Table 2-1 Product package installation requirement examples 20

Introduction to PackageMaker User Guide

Software packaging is the process used to put a software product into an *installation package* so that it can be installed by the users of the product on their computers. To package a product you use the PackageMaker application, which is part of the Xcode developer software suite.

This document shows how to use PackageMaker to create installation packages.

People who take finished product files and create installation packages for them are known as **packagers**. With PackageMaker, packagers accomplish these objectives:

- Enclose a software product in a transport-agnostic container for delivery to users
- Define the user install experience
- Specify how product files are placed on the user's file system

Software requirements: This document focuses on PackageMaker 3.0, introduced in Xcode 3.0. Most of the descriptions in this document do not apply to earlier releases of PackageMaker.

You should read this document if you have a software product that you want to deliver to its users in a way that allows you to define certain aspects of the user install experience and details about how the product files are to be placed on the user's file system.

To get the most out of this document, you should be familiar with *Software Delivery Guide*, which describes the software delivery model used in Mac OS X.

Organization of This Document

This document contains the following chapters:

- "Packaging Overview" (page 9). Describes briefly the packaging process in the context of the Mac OS X software delivery model.
- "Packaging Workflow" (page 13). Explains the workflow you should follow when creating installation packages.

This document also contains a glossary and a revision history.

See Also

- *Software Delivery Guide*. Explains how to deliver and install software using the PackageMaker and Installer applications. Provides a larger picture of the Mac OS X software-delivery model and more detailed information about the installation process.
- *File System Overview*. Conceptual information and guidelines describing the structure and usage of the Mac OS X file system. Contains information about the Mac OS X file system domains and the recommended locations for software components according to their type.

Packaging Overview

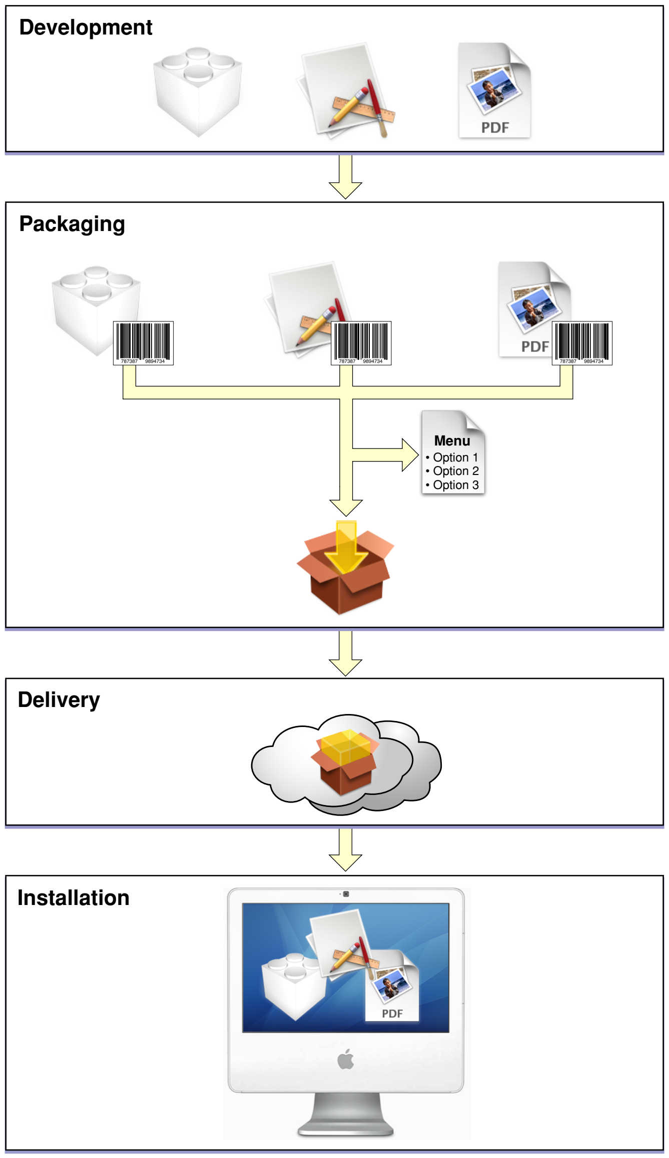
Packaging is one of the processes that make up the Mac OS X software-delivery model. An **installation package** is a file package that contains product files (the **payload**), instructions on how to add them to a Mac OS X–based system, and information used to create an appropriate install experience for the user. When users open your installation package, the Installer application guides them through the installation process, which ensures that their computer meets the installation requirements defined in the package before placing the payload on the user's file system, among other tasks.

The preferred software delivery mechanism for a self-contained application is the **manual install**, where users drag the product from its container, a disk image, onto their file system. The installation package–based mechanism is the preferred method for delivering a multicomponent product that isn't self-contained in a bundle. A **managed install**, which is steered by the Installer application after the user opens an installation package, can take advantage of advanced features such as better package management through the Installer package database, downloadable packages, and certificate-based signing. Mac OS X leverages these features to provide users an improved install experience.

There are two types of installation packages: product packages and component packages. **Product packages** contain the payload for an entire product, either as a single component or distributed among several component packages. **Component packages** enclose a single component of a product and are generally contained within product packages. In addition, product packages can refer to external component packages through **package references**.

PackageMaker is the application you use to create installation packages. Figure 1-1 shows the packaging process within the software development-packaging-delivery-installation workflow. The rest of this document focuses on the packaging process.

Figure 1-1 The packaging process



To package a product you:

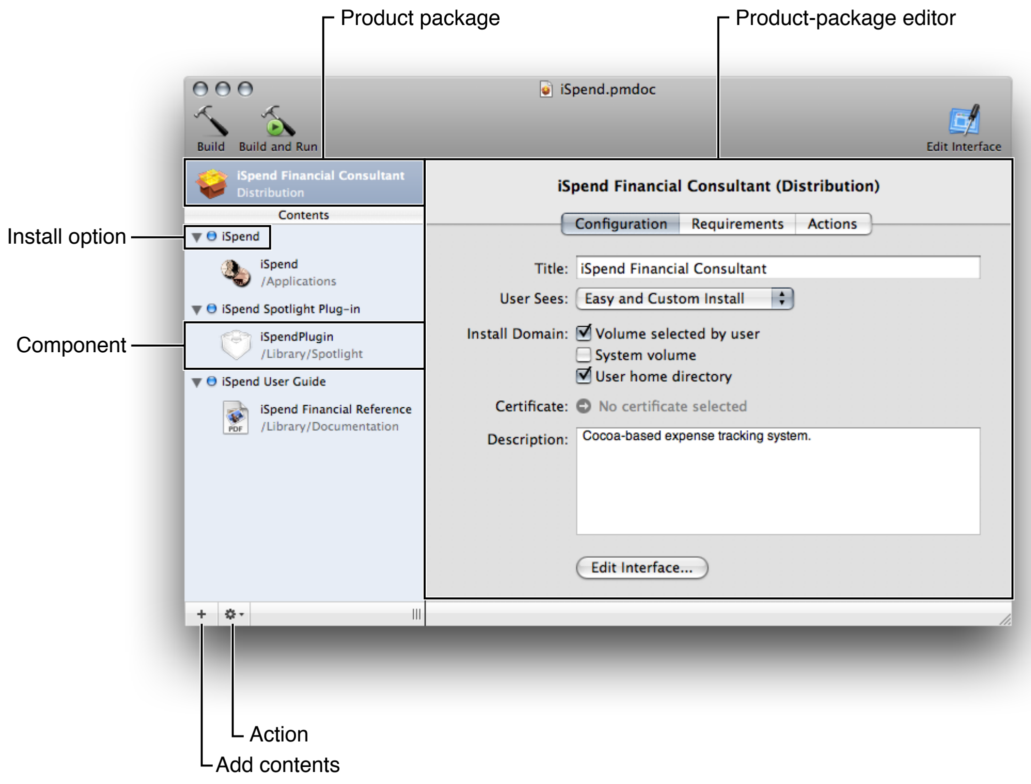
1. Identify and collect the product's components
2. Create a PackageMaker project
3. Add the product's components to the project
4. Configure component packages
5. Configure the product package
6. Define install options (in multicomponent products)
7. Build and test the product package

"Packaging Workflow" (page 13) describes the packaging workflow in detail.

Packaging Workflow

PackageMaker (`<Xcode>/Applications/Utilities`) is the application you use to create installation packages. Figure 2-1 shows the PackageMaker project window. The list following the figure describes the items identified in it.

Figure 2-1 PackageMaker project window



iSpend Financial Consultant (Distribution)

Title: iSpend Financial Consultant

User Sees: Easy and Custom Install

Install Domain: [✓] Volume selected by user; [ ] System volume; [✓] User home directory

Certificate: No certificate selected

Description: Cocoa-based expense tracking system.

- The left side of the window contains the **package list**. The first item in the list represents the *product package* the project generates. The items in the Contents pane represent **component packages** and, in product packages with more than one component package, install choices (the items with a blue dot next to them).
- The right side of the project window contains an editor for the item selected in the package list. Figure 2-1 shows the product-package editor.
- The plus sign (+) icon at the bottom-left corner of the window is the **Add Contents button**.
- The gear icon is the Action pop-up menu (also known as the **shortcut menu**). Its contents correspond to possible actions on the selected item.

The following sections depict the workflow you should follow when creating installation packages.

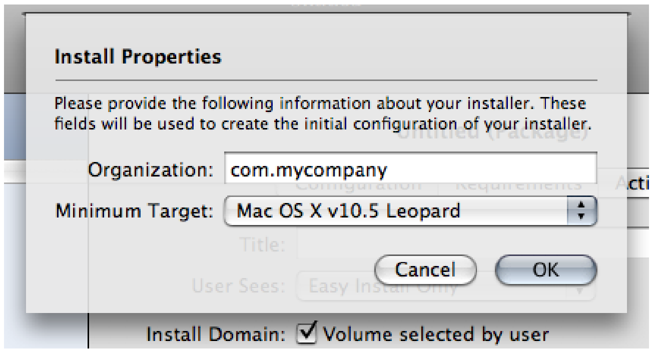
Create a PackageMaker Project

A PackageMaker project, the central piece of the your packaging experience, is where you identify a product to be packaged and define the install experience for the users of the product. To create a PackageMaker project, choose:

PackageMaker menu bar: File > New.

PackageMaker displays the Install Properties dialog, shown in Figure 2-2.

Figure 2-2 Install Properties dialog



Install Properties

Please provide the following information about your installer. These fields will be used to create the initial configuration of your installer.

Organization: com.mycompany

Minimum Target: Mac OS X v10.5 Leopard

In this dialog you specify the following package properties:

- **Provider Identifier (Organization):** Identifies the entity responsible for the package's contents. PackageMaker uses the provider identifier to generate default package identifiers for the contained component packages. See "Component Package Configuration Pane" (page 15).
- **Target OS (Minimum Target):** The earliest Mac OS X release on which you intend the package to be installed.

Identify the Product Components

To define the package's payload, locate the product components to be included in the package and add them to the Contents pane in the project window. You can add components by dragging them from a Finder window or by choosing:

PackageMaker menu bar: Project > Add Contents.

After adding a second component to a project, PackageMaker creates an install choice for each component you add to the project, unless you add it directly to an existing install choice.

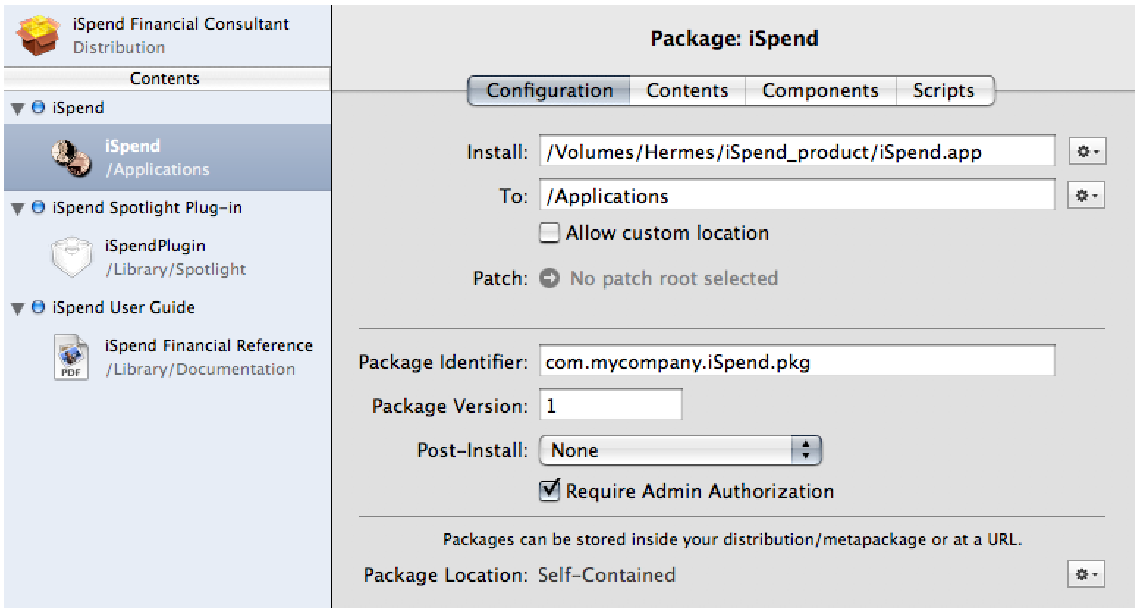
Configure Component Packages

You configure a component package in the **component package editor**, which contains four panes: Configuration, Contents, Components, and Scripts. They are described in the following sections.

Component Package Configuration Pane

The Component Package Configuration pane (shown in Figure 2-3) is where you specify essential information about the component package and its install experience. The list following the figure describes the items it specifies.

Figure 2-3 Component Package Configuration pane



Install: /Volumes/Hermes/iSpend_product/iSpend.app
To: /Applications
Allow custom location
Patch: No patch root selected
Package Identifier: com.mycompany.iSpend.pkg
Package Version: 1
Post-Install: None
[✓] Require Admin Authorization
Packages can be stored inside your distribution/metapackage or at a URL.
Package Location: Self-Contained

- **Component source (Install):** Pathname to the component's root directory. This is the location of the component's files on your file system. These files make up the component package's payload.
- **Destination (To):** Location in the target computer's file system where the component is to be placed.
- **Custom destination consent (Allow custom location):** Specifies whether the user installing the component package can specify a different destination.
- **Package identifier:** String, in the form of a universal type identifier (UTI), that identifies the component package. For example, `com.apple.iSpend.pkg`.

Install effect: The Installer application uses the package identifier to identify the component package in the Installer package database.

- **Package version number:** Positive integer that specifies the iteration of the component package. (This version number is not related to any versioning information specified in the payload.)

Install effect: The Installer application uses the package version number, together with the package identifier, to determine whether to install the contained component as a new item in the target computer or to upgrade an existing copy of the component.

- **Finalization action (Post-install):** Action to require the user perform after the installation process is complete. The available actions are log-out, restart, and shutdown.

Install effect: After the installation process is finished, the Installer application displays a dialog indicating the action to be performed. When the user clicks the dialog's default button, Installer carries out the action.

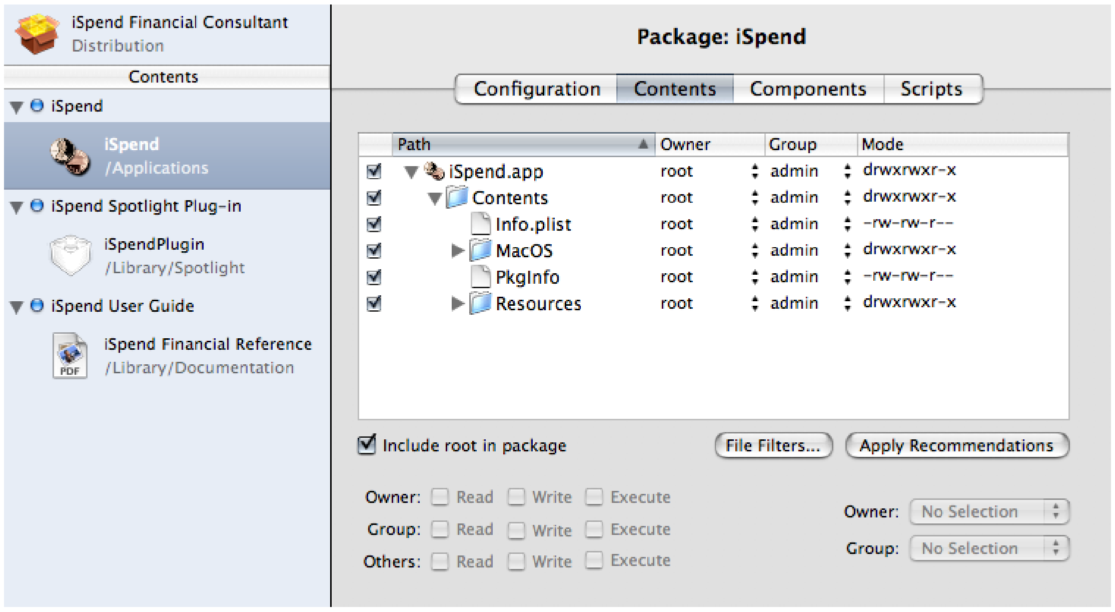
- **Administrator-authentication requirement (Require admin authorization):** Specifies whether the user must authenticate as an administrator of the computer before performing the install. This is needed when the user can install a product in one of the privileged file-system domains, such as the local domain (for example, `/Applications`). You don't need to select this option when the user can install your package only on their home directory (see "Product Package Configuration Pane" (page 19) for more information).

Install effect: The Installer application displays the Mac OS X Authentication dialog. If the user who authenticates is not an administrator of the computer, the installation process does not proceed.

Component Package Contents Pane

The Component Package Contents pane (shown in Figure 2-4) is where you specify the ownership and access permissions of each of the files that make up the component.

Figure 2-4 Component Package Contents pane



	Path	Owner	Group	Mode
✓	iSpend.app	root	admin	drwxrwxr-x
✓	Contents	root	admin	drwxrwxr-x
✓	Info.plist	root	admin	-rw-rw-r--
✓	MacOS	root	admin	drwxrwxr-x
✓	PkgInfo	root	admin	-rw-rw-r--
✓	Resources	root	admin	drwxrwxr-x

[✓] Include root in package

Owner: ☐ Read ☐ Write ☐ Execute
Group: ☐ Read ☐ Write ☐ Execute
Others: ☐ Read ☐ Write ☐ Execute

Owner: No Selection
Group: No Selection

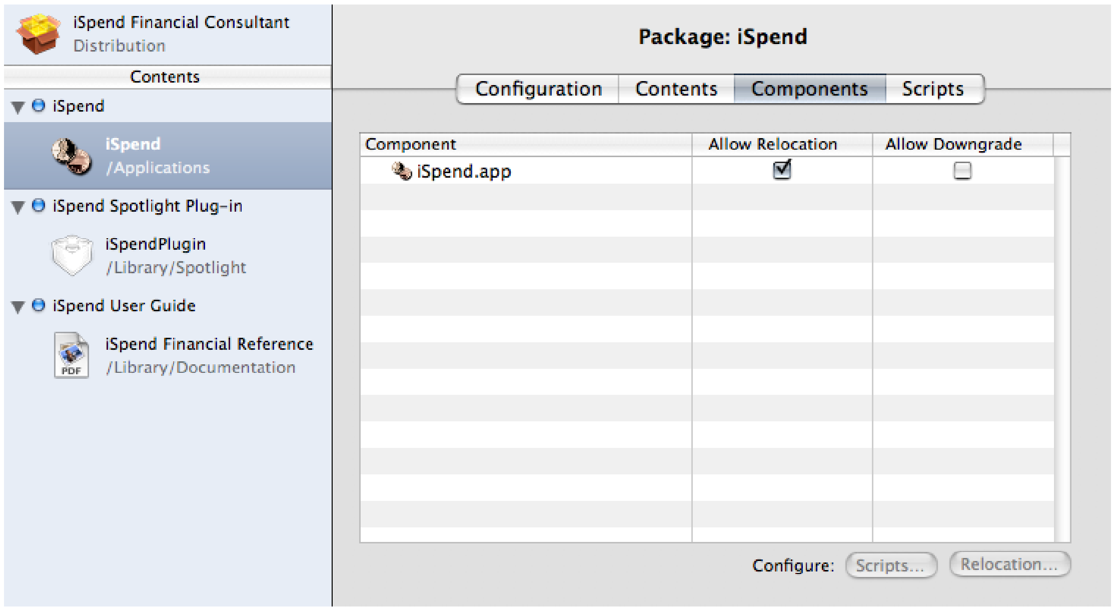
Ownership and access permissions: The files that the Installer application places on the target computer have the same ownership and access permissions of the payload's files. Therefore, you must set up the owner and access permissions of component files appropriately before building the installation package; otherwise, users may have difficulty manipulating those files after they are installed or Installer may be unable to copy payloads to their destinations.

In most cases, the owner should be `root` and the group `admin`. Also, PackageMaker can set the owner, group, and access permissions of the component files to those that work best in Mac OS X, according to the component type.

Component Package Components Pane

The Component Package Components pane (shown in Figure 2-5) specifies whether the component is relocatable or downgradable.

Figure 2-5 Component Package Components pane



Component	Allow Relocation	Allow Downgrade
iSpend.app	✓	

A **relocatable component** is one that may be moved by the user after it's installed. For example, after installing an application into `/Applications`, a user with administrator privileges may move it to `/Volumes/Family/Applications`. When the user installs a relocatable component a second time on the same computer, the Installer application searches for the component's existing files in additional locations in the file system, not just the location at which the component was installed, according to the Installer package database).

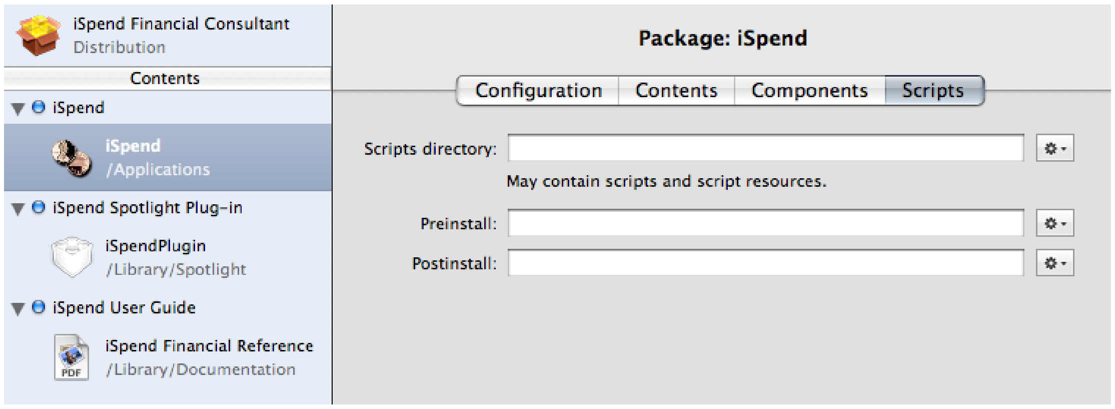
In this pane you also specify whether a component can be downgraded. A **downgradable component** is one that can be replaced with an earlier version during an install. When the user reinstalls a earlier version of an existing downgradable component, Installer replaces component files that exist in both the payload and the target computer with the ones in the payload and deletes files that are not present in the payload.

Important: A component's downgradability is specified by the installation package that may downgrade the component, not by the installation package from which the component was installed. And none of this information is kept in the component itself. That is, if an early release of an installation package specifies one of its components as downgradable but a later release deems the same component as not downgradable, after the user installs the later package, they can still downgrade the component using the earlier package.

Component Package Scripts Pane

The Component Package Scripts pane (shown in Figure 2-6) specifies install operations—which are implemented as executable files—to perform before (preinstall) or after (postinstall) the component is installed.

Figure 2-6 Component Package Scripts pane



Scripts directory:
May contain scripts and script resources.
Preinstall:
Postinstall:

Install effect: The Installer application executes the specified install operation either at the beginning of the install or at the end of the install. But Installer also warns users that it's about to execute unsecured code when they open the installation package.

Note: Consider defining preinstall and postinstall actions on the product package (see "Product Package Actions Pane" (page 20)) instead of preinstall and postinstall operation in component packages. The latter are inherently less secure and, therefore, causes the warning described earlier.

Important: In Mac OS X v10.5 clients, the only install operations available are preinstall and postinstall.

For detailed information about install operations, see *Software Delivery Guide*.

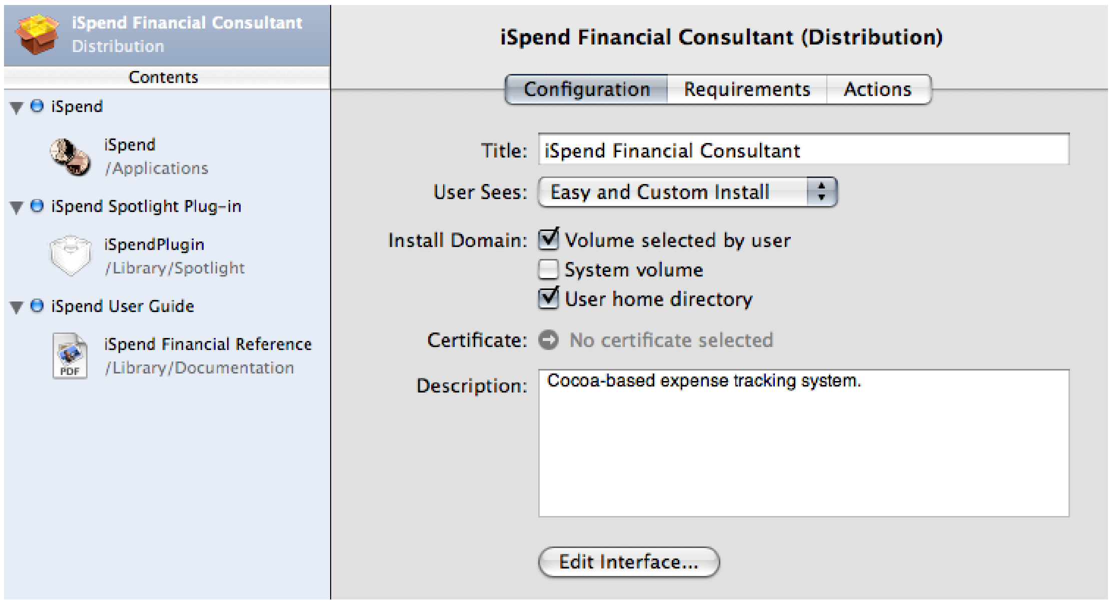
Configure the Product Package

When you select the product package in the package list, the **product package editor** appears on the right side of the project window. In this editor you define the product's packaging details and some installation information. The following sections describe the panes in the product-package editor.

Product Package Configuration Pane

The Product Package Configuration pane (shown in Figure 2-7) is where you enter information about the product package, such as its title and description. You can also add other product description files, such as the Welcome, Read Me, License, and Conclusion files (see *Software Delivery Guide* for details).

Figure 2-7 Product Package Configuration pane



iSpend Financial Consultant (Distribution)

Title: iSpend Financial Consultant

User Sees: Easy and Custom Install

Install Domain: [✓ Volume selected by user] System volume [✓ User home directory]

Certificate: No certificate selected

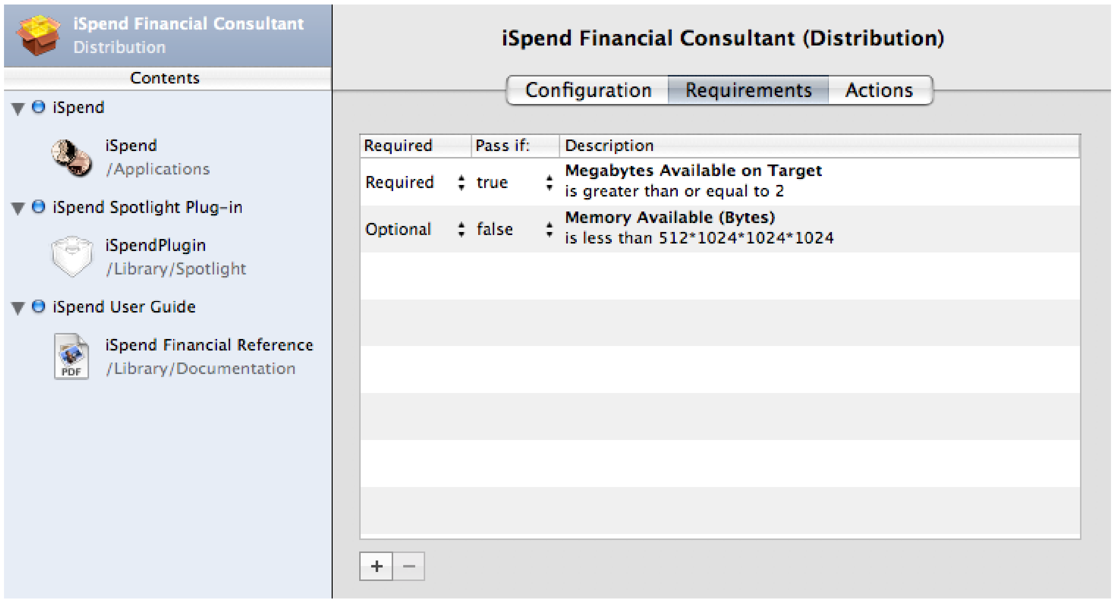
Description: Cocoa-based expense tracking system.

This pane also lets you specify which type of install the user can perform on the product package: easy, custom, or both. In addition, you can specify the locations into which the user can install the product: any volume, the system volume, or the user's home directory.

Product Package Requirements Pane

A **package requirement** is a test the Installer application performs at the beginning of the installation process. These requirements may be optional. Any unmet, non-optional package requirement prevents the installation process from continuing. Figure 2-8 shows the Product Package Requirements pane, where you specify these requirements. In this case, the package has two requirements, one required and one optional. Their definition and install effects are described in Table 2-1.

Figure 2-8 Product Package Requirements pane



Required	Pass if:	Description
Required	true	**Megabytes Available on Target** is greater than or equal to 2
Optional	false	**Memory Available (Bytes)** is less than 512*1024*1024*1024

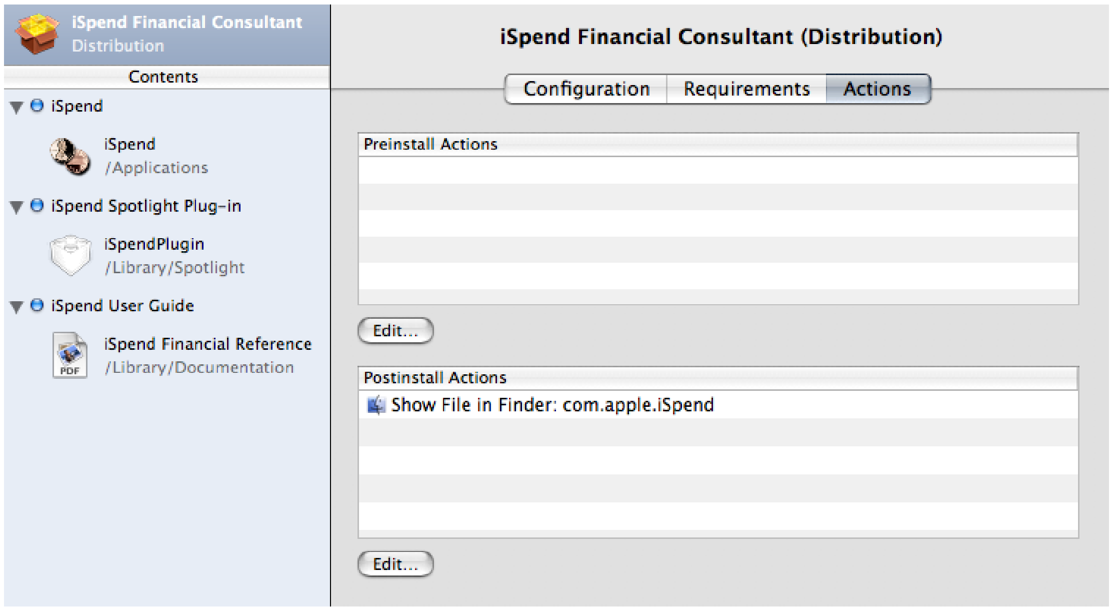
Table 2-1 Product package installation requirement examples

Description	Required	Pass if	Install effect
Disk space on target volume is at least 2MB	Yes	`true`	Prevents install unless destination volume has at least 2MB of free space.
Computer RAM is less than 512MB	No	`false`	Displays a warning when the computer has less than 512MB of RAM.

Product Package Actions Pane

You may want to tell the Installer application to perform a particular action before or after the product package is installed. **Installation actions** let you specify such tasks. For example, you can instruct Installer to show the installed payload in a Finder window, as shown in Figure 2-9.

Figure 2-9 Product Package Actions pane



Show File in Finder: com.apple.iSpend

This postinstall action opens a Finder window displaying one of the installed product's components.

Configure the Product Package Install Choices

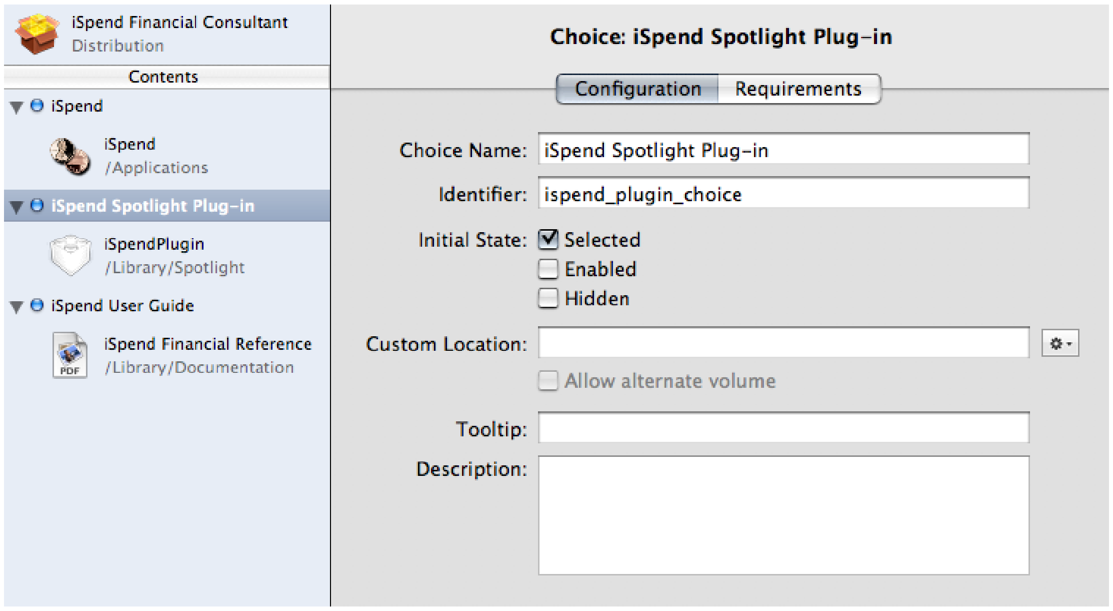
When your product contains multiple components, it may be appropriate to let the user decide which components to install. For example, a particular user of your product may not want to install a documentation or example component. In this case, the user should be able to remove unwanted components from the install process. PackageMaker and the Installer application allow you to define install choices to accomplish such outcome.

Install choices allow users of your product package to customize the install by selecting the components to be installed (unselected components are not installed). For example, if your product includes an application and a user guide as separate components, you may allow the user not to install the user guide by having the application and the user guide under separate choices. In the **install customization pane** (the Custom Install pane in the Installer application), the user selects the components to be installed.

Note: Install choices are not a mechanism by which installed components can be removed from a computer.

You configure an install choice by selecting it in the Contents pane in the project window and setting values for its properties in the **choice editor**, which contains two panes: Configuration and Requirements. Figure 2-10 shows the Choice Configuration pane.

Figure 2-10 Choice Configuration pane



Choice: iSpend Spotlight Plug-in

Choice Name: iSpend Spotlight Plug-in

Identifier: ispend_plugin_choice

Initial State: [✓ Selected] Enabled Hidden

Custom Location:

Allow alternate volume

Tooltip:

Description:

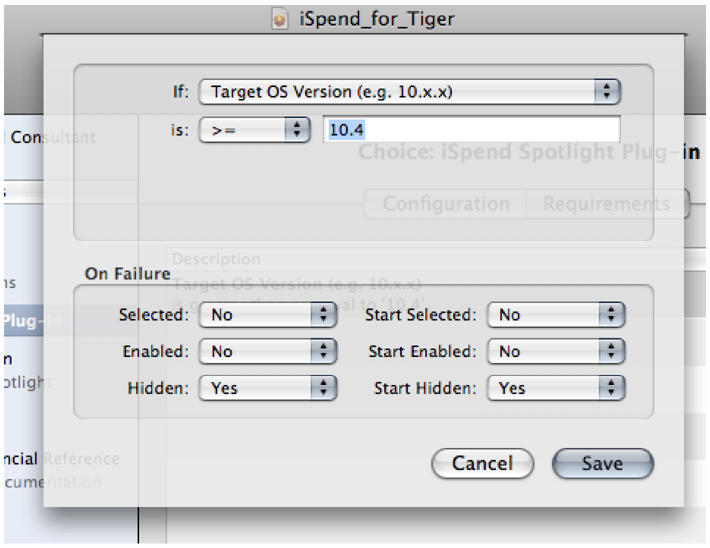
This pane lets you specify the following choice properties:

- **Choice Name:** The text the user sees in the install customization pane.
- **Identifier:** Identifies the choice within the package.
- **Initial State:** Specifies the value of three user-interface properties (selected, actionable, and visible) of the choice the first time the user arrives at the install customization pane:
  - **Selected:** Specifies whether the choice is selected.
  - **Enabled:** Specifies whether the user can change the selected state of the choice directly.
  - **Hidden:** Specifies whether the user can see the choice.
- **Destination (Custom Location):** Specifies the destination of the choice's components. Defining a choice destination overrides the destination specified for the choice's components in the component package editor (see "Component Package Configuration Pane" (page 15)). You may also allow the user to choose a different destination by selecting "Allow alternate volume." You should always specify a destination for every choice with product components.
- **Tooltip:** Short message (15 words or fewer) that appears when the user hovers the pointer over the choice in the install customization pane.
- **Description:** Information that appears in the install customization pane when the user selects the choice.

As described earlier, a choice has user-interface properties that define the level of interaction the user has with the choice in the install customization pane. While you can specify a value for each UI property in the package, there may be choices that need information about the computer to determine the value of such properties. For example, if an optional plug-in component can work only in certain releases of Mac OS X, you may want to display the choice that contains the component only when the computer is running an appropriate operating system version. Here's where choice requirements can help.

A **choice requirement** is a test that compares a system property against a value and sets the initial and dynamic values of the choice's UI properties based on the test result. (The Installer application sets the dynamic values of the UI properties as the user selects or deselects choices in the install customization pane; see *Software Delivery Guide* for details.) Figure 2-11 shows the **choice requirement editor**, which lets you define the test and the resulting values of the choice UI properties if the test fails.

Figure 2-11 Choice requirement editor



If: Target OS Version (e.g. 10.x.x)
is: >= 10.4

On Failure

Selected:	No	Start Selected:	No
Enabled:	No	Start Enabled:	No
Hidden:	Yes	Start Hidden:	Yes

Build the Product Package

To build the package, choose:

PackageMaker menu bar: Project > Build.

A dialog appears asking for the name and location of the package.

Open the generated installation package in the Installer application and ensure that the resulting install experience is what you expect and that your product is installed correctly. You should perform comprehensive tests involving as many of the system configurations your product supports as possible. See *Software Delivery Guide* for more information.

Glossary

choice requirement A test that compares the value of a system property (such as the amount of random-access memory available) with a value. Choice requirements determine the value of a choice's user-interface properties: selected, actionable, and visible.

choice requirement editor Area of a PackageMaker project window that allows packagers to specify a choice requirement (and how it affects the value of the choice's user-interface properties). See also choice requirement.

component package Installer package that contains a single software component as its payload. See also, product package.

component package editor Area of a PackageMaker project window that specifies packaging and installation information about a product component. This editor is displayed when a component is selected in the Contents pane in the project window.

downgradable component A product component, such as an application binary or a plug-in, that can be replaced with an earlier version in an install process.

finalization action An action required after a completed installation process. The possible finalization actions are log-out, restart, and shutdown.

install choice An option users can select or deselect during the installation process to specify whether a product component is to be installed

install customization pane A pane users see while interacting with the Installer application if the package being installed allows the user to customize the install by choosing the product components to be installed. See also, product package, product component.

install operation An install-time operation performed by an executable file that is invoked at the beginning or at the end of the install. The two install operations supported in Mac OS X v10.5 are preinstall and postinstall.

installation action A task to be performed before or after an install. PackageMaker defines several installation actions, including Quit Application and Show File in Finder.

installation package A file package with the `pkg` or `mpkg` extension. Installation packages contain a payload and installation information used by the Installer or Remote Desktop applications to identify the payload's parts and generate an install experience for the user.

Installer package database System-level database of all the installation packages installed by the Installer application.

managed install An Installer application–driven installation process. Users open an installer package in Installer, which then guides them through the installation process.

manual install An user-driven installation process. In this software-installation method, users drag a product's files to a location of their choosing in their computer's file system. See also managed install.

package identifier Identifies the package within the Installer package database. See also Installer package database.

package list A pane in a PackageMaker project window that lists the packages the project defines. This list is divided in two parts: the installation-package file (which contains all the product's files) and the subpackages or package references that contain components of the product.

package requirement A test that determines whether a package can be installed on the computer. A package requirement can be optional; such requirements display a warning to the user but allows the install to proceed. Non-optional requirements prevent an install from taking place.

package version number Positive integer that identifies an iteration of a single-component product package, or an iteration of a component package within a product package. This version number should be incremented when the contents or installation details of the package are changed. See also product package component package.

payload The product or product components contained in an installation package. See also installation package.

product component Self-contained part of a product. A product can have one or more components. The Mac OS X file system contains special locations for several types of components. For example, application binaries are placed in `Application` directories, plug-ins are housed in `Plugin` directories, fonts live in `Fonts` directories, and so on.

product package Installation package that contains all the components of a product. Product packages with multicomponent products contain or reference component packages. See also installation package.

product package editor A pane in a PackageMaker project window that specifies packaging and installation information about a product. This pane is displayed when the product package is selected in the package list.

provider identifier Identifier for the entity responsible for the contents of an installation package; for example, `com.apple`. PackageMaker uses this identifier to generate default package identifiers for a product package's components. See also package identifier.

relocatable component A product component, such as an application binary or a plug-in, that the user may move after it has been installed.

target OS version The earliest release of Mac OS X in which the installation package is to be installed. The package is installable on the specified release and later. For example, a package whose target OS is Mac OS X v10.4 can be installed on computers running Mac OS X v10.4 and later releases.

volume requirement A test that compares the value of a volume property (such as free space) with a value. Volume requirements determine whether the user can choose a particular volume as the destination volume of a product package.

Document Revision History

This table describes the changes to *PackageMaker User Guide*.

Date	Notes
2009-01-06	Made minor technical corrections.
2007-07-23	New document that describes how to use PackageMaker to create installation packages.