Preference Manifest Files for Managed Clients Overview

2008-10-15

Contents

Tables

About This Manual

This manual describes the preference manifest file, which is an XML document describing an application's preference keys that can be managed by Workgroup Manager, a component of Apple Computer's managed client solution for Mac OS X. Application developers create a preference manifest file to make their application's preference keys available for management by Workgroup Manager. Developers use the Property List Editor, provided on the Xcode Tools CD, to create and modify preference manifest files.

Conventions Used in This Manual

The `Letter Gothic` font is used to indicate text that you type or see displayed. This manual includes special text elements to highlight important or supplemental information:

> **Note:** Text set off in this manner presents sidelights or interesting points of information.

> **Important:**   Text set off in this manner—with the word Important—presents important information or instructions.

> ⚠ **Warning:** Text set off in this manner—with the word Warning—indicates potentially serious problems.

See Also

For information on Workgroup Manager, see *Mac OS X Server User Management* at http://www.apple.com/server/documentation.

A client management discussion list is available at http://lists.apple.com/mailman/listinfo/client-management.

Preferences Programming Topics for Core Foundation provides information on the best way to work with preferences.

See Also

Concepts

A preference manifest file is an XML plist file that describes an application's preferences and makes them available for management by Workgroup Manager, a component of Apple Computer's managed client solution for Mac OS X. Application developers use the Property List Editor, provided on the Xcode Tools CD, to create preference manifest files.

This chapter describes the structure of the preference manifest file, the keys that are found in a preference manifest file, and whether a key is required or optional. It also provides examples of key usage.

Structure of a Preference Manifest File

The structure of a preference manifest file consists of an outermost dictionary that provides a general description of the preference manifest file, followed by a series of nested dictionaries and arrays containing **manifest keys**. Manifest keys are used to specify the data type, default value, and other characteristics of a **preference key**. A preference key is a key within the user's preference. That is, manifest keys appear in a preference manifest file; together, manifest keys specify a preference key within a user's preferences.

Outermost Dictionary

The outermost dictionary contains manifest keys that describe the preference manifest file in general. Some manifest keys are only valid in the outermost dictionary.

The outermost dictionary typically contains the following manifest keys:

- a `pfm_description` key that provides a general description of the preference manifest file.

- a `pfm_title` key that provides a name that will be shown in Workgroup Manager's Preference Manifest Editor as the name for the application's preferences as a whole

- a `pfm_format_version` key that specifies the preference manifest format version the file uses; this key is only valid in the outermost dictionary

- a `pfm_version` key that specifies the version number for this preference manifest file so that it can be distinguished from other versions that might be released; this key is only valid in the outermost dictionary

- a `pfm_domain` key specifying the application domain for this preference manifest file, such as `com.apple.myapp`, where `myapp` is the typically name of the application's bundle ID, which is specified in the application's `info.plist` and is the name that the application uses to retrieve preferences through the `CPPreferences` or `NSUserDefaults` API calls.

- a `pfm_subkeys` key indicating that an array of dictionaries follows, with each dictionary using keys to describe a preference

Here is an example of an outermost dictionary for an application named `myapp`:

```
<dict>
    <key>pfm_description</key>
    <string>Configurable preferences for MyApp.</string>
    <key>pfm_title</key>
    <string>myapp</string>
    <key>pfm_format_version</key>
    <real>1.0</real>
    <key>pfm_version</key>
    <real>0.9</real>
    <key>pfm_domain</key>
    <string>com.apple.myapp</string>
    <key>pfm_subkeys</key>
```

Inner Dictionaries

The outermost dictionary is followed by a series of inner dictionaries that can contain other dictionaries.

Here is an example of an inner dictionary that might follow the `pfm_subkeys` key in the sample outermost dictionary:

```
<array>
    <dict>
        <key>pfm_name</key>
        <string>autohide</string>
        <key>pfm_title</key>
        <key>Hiding</key>
        <key>pfm_type</key>
        <string>boolean</string>
        <key>pfm_default</key>
        <false/>
        <key>pfm_description</key>
        <string>True to turn hiding on.</string>
        <key>pfm_targets</key>
        <array>
            <string>user</string>
            <string>user-managed</string>
        </array>
    </dict>
    // Other dictionaries containing manifest keys that  describe
    // preference keys.
</array> // End of the array of dictionaries.
</dict> // End of the preference manifest file.
```

In the example of an inner dictionary,

■ the `pfm_name` key defines the name (`autohide`) of the preference key.

■ the `pfm_title` key specifies the user-visible name shown in the Workgroup Manager's Preference Manifest Editor.

■ the `pfm_type` key defines `boolean` as the data type of this preference key.

■ the `pfm_default` key defines the default value of this preference key as `false`.

■ the `pfm_description` key provides, in this case, a reminder that to enable this preference key, it's value should be set to `true`. You can use the `pfm_description` key to provide any instructions or information needed for the proper setting of this preference key.

■ the `pfm_targets` key specifies when and where this preference key should be used by Mac OS X client management software to properly manage a user or computer. In this example, `user` specifies that the key may be set in a user's preferences (at `~/Library/Preferences/com.apple.myapp.plist`) and `user-managed` specifies that the key may be set in a user's managed preferences (at `/Library/Managed Preferences/`*username*`/com.apple.myapp.plist`, where *username* is the logged in user's short name).

Preference Manifest Keys

For each valid manifest key, Table 1-1 (page 11) provides the key's data type, the default value that the Workgroup Manager determines for a the key when it is not specified, and a description. Manifest keys are required or optional as noted in the Description column below.

Table 1-1   Manifest keys

Manifest key	Data type	Default value	Description
`pfm_default`	Same as the data type of the `pfm_type` key.	`0`, `false`, `{}`, `""`, etc.	Specifies the preference key's default value.
`pfm_description`	String	`""`	Specifies a description of the preference key. This manifest key is optional. See the section `Localization` (page 21) for additional information about specifying the value of this key.
`pfm_domain`	String	If this key is not specified, the value of `pfm_domain` specified in the outermost dictionary is used.	Specifies the application domain for this preference key. This manifest key is only valid in the outermost dictionary and in direct `pfm_subkey` definitions. For additional information, see the section "Using the pfm_domain Key" (page 14).
`pfm_format_version`	Real	`1.0`	Specifies the preference manifest format version. This manifest key is only valid in the outermost dictionary.
`pfm_name`	String		Specifies the name of a preference key. This key is required for all keys except the subkeys of an array.

Manifest key	Data type	Default value	Description
`pfm_range_list`	Array of values having the same data type as specified by the `pfm_type` key.	`{}`	Specifies an array of legal values for this preference key, if applicable. This key is optional and if not specified, any values are allowed or any values within the range specified by `pfm_range_min` and `pfm_range_max`, if specified, are allowed.
`pfm_range_max`	Same as the data type specified by the `pfm_type` key.	+<infinity>	Specifies the maximum value for this preference key. This key is optional.
`pfm_range_min`	Same as the data type specified by the `pfm_type` key.	-<infinity>	Specifies the minimum value for this preference key. This key is optional.
`pfm_repetition_max`	Integer	1	Specifies the maximum number of times this preference key can be repeated. Values greater than one are only valid for preference keys that are arrays. Use –1 to specify unlimited repetition. This key is optional.
`pfm_repetition_min`	Integer	0	Specifies the minimum number of times this preference key must be repeated. Values greater than 1 are only valid for keys that are arrays. Use of this key is optional.
`pfm_targets`	Array	Value of the `pfm_targets` key specified in the next level up or `user` if the `pfm_targets` key is not specified in the next level up.	Specifies values that indicate when a preference is to be processed. Possible values are `user`, `user-managed`, and `system-managed`. For additional information, see the section "Values for the pfm_targets Key" (page 15).

Manifest key	Data type	Default value	Description
`pfm_title`	String	Value of the `pfm_name` key.	Specifies the title of the preference, which is displayed by the Workgroup Manager's Preference Manifest Editor. This key is optional, and if not specified, the Workgroup Manager's Preference Manifest Editor displays the value of the `pfm_name` key as the title. See the section Localization (page 21) for additional information about specifying the value of this key.
`pfm_type`	String		Specifies the type of the preference, which can be a standard plist type (`array`, `boolean`, `date`, `data`, `dictionary`, `integer`, `real`, `string`, or `url`), `union policy` for MCX union policy manifest key definitions, or `alias` for a standard alias. This key is required and has no default.
`pfm_upk_input_keys`	Array of strings	`{pfm_name}`	Specifies the input key names for a union policy manifest key. For additional information, see the section "Defining a Preference Key of Type Union Policy" (page 20).
`pfm_upk_output_name`	String		Specifies the output key name for a union policy manifest key.
`pfm_upk_output_-replace`	Boolean	`false`	Specifies whether to replace user preferences with the output of a union policy.
`pfm_upk_output_type`	String	`array`	For union policy manifest keys, specifies whether to output an array or a dictionary. Use `array` to output an array or `dictionary` to output a dictionary.
`pfm_upk_remove_-duplicates`	Boolean	`false`	Specifies whether to remove duplicate keys when a union policy results in a combination of keys that contains duplicates.
`pfm_version`	Real	`1.0`	Specifies the version of this preference manifest file. Increment this value to override previous version numbers. This manifest key is valid in the outermost dictionary only.
`pfm_subkeys`	Array of dictionary	`{}`	Specifies that nested manifest keys follow. This key is optional and is only used for manifest keys that are dictionaries or arrays.
`pfm_mcx_version`	Integer	`1`	Specifies the use of the `mcx_preference_-version_`*n* key, where *n* is a positive integer starting with 2. This key is optional.

Default Values

If a manifest key is not defined for a preference key, Workgroup Manager uses context and usage to determine a default value for the undefined key. If a default value cannot be determined, the preference key or the entire preference manifest file may be ignored.

Using the pfm_domain Key

The recommended approach is for each preference manifest file to specify just one domain. When a preference manifest file specifies one domain, the `pfm_domain` manifest key does not have to be specified for each preference key. When the `pfm_domain` is not specified for a preference key, the Workgroup Manager gets the domain for the preference key from the `pfm_domain` key in the outermost dictionary. Here is an example:

```
<key>pfm_format_version</key>
<real>1.0</real>
<key>pfm_version</key>
<real>1.0</real>
<key>pfm_domain</key>
<string>com.apple.myapp</string>
<key>pfm_subkeys</key>
<array>
    <dict>
        <key>pfm_name</key>
        <string>Preference One</string>
        // Additional information for "Preference One"
        <key>pfm_name</key>
        <string>Preference Two</key>
        // Additional information for "Preference Two"
    </dict>
</array>
```

In this example, the domain for `Preference One` and `Preference Two` is `com.apple.myapp` as specified by the `pfm_domain` key in the outmost dictionary.

Although doing so is not recommended, it is possible to specify more than one domain in a preference manifest file. Here is an example:

```
<key>pfm_format_version</key>
<real>1.0</real>
<key>pfm_version</key>
<real>1.0</real>
<key>pfm_subkeys</key>
<array>
    <dict>
        <key>pfm_name</key>
        <string>Preference One</string>
        // Additional information for "Preference One"
        <key>pfm_domain</key>
        <string>com.apple.myapp1</string>
        <key>pfm_name</key>
        <string>Preference Two</string>
        // Additional information for "Preference Two"
        <key>pfm_domain</key>
        <string>com,apple.myapp2</string>
    <dict>
```

```
</array>
```

Values for the pfm_targets Key

The value of the `pfm_targets` key is an array consisting of any combination of `user`, `user-managed`, and `system-managed`. These values tell the MCX client when (before or after login) to process the preference keys for a particular application domain.

The value of the `pfm_targets` key works in conjunction with settings made by the administrator that indicate when an application preference should be processed:

■ *Always* — This is an application preference that is set and cannot be changed.

■ *Once* — This is an application preference that is set but can be permanently changed.

■ *Often*— This is an application preference that is set and can be temporarily changed. When the user logs back in the preference is set to its original value.

It is up to the application to conform to these settings and display the correct user interface.

When `user` is a value of the `pfm_targets` key, any application preferences set by the administrator to Once and Often are processed just after authentication and before any user applications run. The key may be set in a user's preferences at `~/Library/Preferences/com.apple.myapp.plist`.

When `user-managed` is a value of the `pfm_targets` key, any application preferences set by the administrator to Always are processed just after authentication and before any user applications run. The key may be set in a user's managed preferences at `~/Library/Managed Preferences/`*username*`/com.apple.myapp.plist.` where *username* is the short name of the logged-in user.

When `system-managed` is a value of the `pfm_targets` key, any application preferences set by the system administrator to Always are processed at startup time and after every user logout. For example, the preference that controls whether to show `loginwindow` with a list of users or with a field in which the user's name is entered has to processed before any user logs in. The key will be set in a property list file (`.plist`) in `/Library/Managed Preferences/`.

In actual practice, an application preference can be set anytime before the application runs. Log in is a convenient time because no user applications or processes have run yet. In this release, an application preference can activate at any time, such as waking from sleep, the computer is plugged into the network, or after a certain amount of time has elapsed. The preferences take effect immediately, but an application may not notice newly activated preferences until the applications are relaunched.

File Naming Conventions

An application should use one preference manifest file per domain and use the following file-naming convention:

domain.manifest

where *domain* is the value specified by the `pfm_domain` key in the preference manifest file. If the domain is `com.apple.myapp`, the name of the preference manifest file would be `com.apple.myapp.manifest`.

The preference manifest file should be placed in the application's bundle in the `./Contents/Resources/` subdirectory. If the domain is `com.apple.myapp`, the path for the preference manifest file would be:

```
./Contents/Resources/com.apple.myapp.manifest
```

Sample Preference Key Definitions

This section uses a series of examples to describe the definition of a preference key for each `pfm_type`.

Defining a Preference Key of Type Alias

A preference key whose `pfm_type` is `alias` causes the Workgroup Manager's Preference Manifest Editor to allow you to specify an alias to a file or folder. The data corresponding to the alias record is stored in the preference data as a `data` plist type.

Here is a sample definition for a preference key whose `pfm_type` is `alias`:

```
<key>pfm_name</key>
<string>backupVolume</string>
<key>pfm_type</key>
<string>alias</string>
<key>pfm_title</key>
<string>Backup Volume</string>
<key>pfm_description</key>
<string>Alias to the backup volume</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
```

Defining a Preference Key of Type Array

Here is a sample definition for a preference key whose `pfm_type` is `array`:

```
<key>pfm_name</key>
<string>OptionCheckboxStringsTemp</string>
<key>pfm_type</key>
<string>array</string>
<key>pfm_default</key>
<array>
    <string>antialias</string>
    <string>fullcolor</string>
</array>
<key>pfm_title</key>
<string>Checkbox Strings</string>
<key>pfm_description</key>
<string>One array element for each visible checkbox containing  the internal
name of  that checkbox.</string>
<key>pfm_targets</key>
```

```
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
<key>pfm_subkeys</key>
<array>
    <dict>

        <key>pfm_description</key>
        <string>Checkbox string values.</string>
        <key>pfm_name</key>
        <string>checkboxnames</string>
        <key>pfm_rangelist</string>
        <array>
            <string>antialias</string>
            <string>fullcolor</string>
            <string>invertcolor</string>
        </array>
        <key>pfm_repetition_max</key>
        <integer>3</integer>
        <key>pfm_repetition_min</key>
        <integer>0</integer>
        <key>pfm_title</key>
        <string>Checkbox name</string>
        <key>pfm_type</key>
        <string>string</string>
    </dict>
</array>
```

Defining a Preference Key of Type Boolean

Here is a sample definition for a preference key whose `pfm_type` is `boolean`.

```
<key>pfm_name</key>
<string>reverseHilite</string>
<key>pfm_type</key>
<string>boolean</string>
<key>pfm_default</key>
<false/>
<key>pfm_title</key>
<string>Reverse Highlighting</string>
<key>pfm_description</key>
<string>Set to true for black on white highlighting.</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
```

Defining a Preference Key of Type Date

Here is a sample definition for a preference key whose `pfm_type` is `date`.

```
<key>pf_domain</key>
<string>com.apple.myapp</string>
<key>pfm_name</key>
<string>database creation time</string>
<key>pfm_type</key>
<string>date</string>
<key>pfm_title</key>
<string>Database Creation Date</string>
<key>pfm_description</key>
<string>Date when the main database was created.</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
```

Defining a Preference Key of Type Data

Here is a sample definition for a preference key whose `pfm_type` is `data`.

```
<key>pfm_name</key>
<string>preference data</string>
<key>pfm_type</key>
<string>data</string>
<key>pfm_title</key>
<string>Database Bit Flags</string>
<key>pfm_description</key>
<string>Database Bit Flags. 10 bytes long.</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
```

Defining a Preference Key of Type Dictionary

Here is a sample definition for a preference key whose `pfm_type` is `dictionary`.

```
<key>pfm_name</key>
<string>document objects</string>
<key>pfm_type</key>
<string>dictionary</string>
<key>pfm_title</key>
<string>Document Object Dictionary</string>
<key>pfm_description</key>
<string>Dictionary of document objects.</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
<key>pfm_subkeys</key>
```

```
<array>
    <dict>
        <key>pfm_name</key>
        <string>Preference One</string>
        // Other keys describing this preference.
    </dict>
    <dict>
        <key>pfm_name</key>
        <string>Preference Two</string>
        // Other keys describing this preference.
    </dict>
</array>
```

Defining a Preference Key of Type Integer

Here is a sample definition for a preference key whose `pfm_type` is `integer`.

```
<key>pfm_name</key>
<string>zoom factor</string>
<key>pfm_type</key>
<string>integer</string>
<key>pfm_default</key>
<integer>100</integer>
<key>pfm_range_max</key>
<integer>1000</integer>
<key>pfm_range_min</key>
<integer>0</integer>
<key>pfm_title</key>
<string>Zoom Factor</string>
<key>pfm_description</key>
<string>Document zoom factor. Only multiples of 10 are valid.</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
```

Defining a Preference Key of Type Real

Here is a sample definition for a preference key whose `pfm_type` is `real`. In this example, `pfm_range_max` is not specified, so the Workgroup Manager would use the default, +<infinity>. The `pfm_description` key is not specified, so the preference key has no description.

```
<key>pfm_name</key>
<string>scaling</string>
<key>pfm_type</key>
<string>real</string>
<key>pfm_default</key>
<real>1.0</real>
<key>pfm_range_min</key>
<real>0.01</real>
<key>pfm_title</key>
<string>Scaling Factor</string>
<key>pfm_targets</key>
```

```
<array>
    <string>user</string>
    <string>user-managed</strings>
</array>
```

Defining a Preference Key of Type String

Here is a sample definition for a preference key whose `pfm_type` is `string`.

```
<key>pfm_name</key>
<string>input source</string>
<key>pfm_type</key>
<string>string</string>
<key>pfm_default</key>
<string>file</string>
<key>pfm_range_list</key>
<array>
    <string>file</string>
    <string>network</string>
</array>
<key>pfm_title</key>
<string>Input Source</string>
<key>pfm_description</key>
<string>Input can be taken from a file or the network.</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
```

Defining a Preference Key of Type Union Policy

Union policy keys provide a way to specify behavior when two or more preference keys come into play as the result of a single event. Consider the case of what application tiles should appear in the application dock. For example, the following could be specified:

■ that all computers show "Calculator" in the dock

■ that Workgroup "MyWorkGroup" shows "DVD Player" in the dock

■ that user "Jimmy" shows "TextEdit" in the dock

When "Jimmy" (who belongs to workgroup "MyWorkGroup") logs in, what should appear in his dock? Just "TextEdit" or should "Calculator," "DVD Player," and "TextEdit" be in his dock?

Union policy keys set up rules so that array or dictionary keys found in user, group, or computer list records can be merged into a single array or dictionary in the resulting preference file.

The following example specifies a union policy for application tiles in:

```
<key>pfm_description</key>
<string>Application tiles union policy keys for user domain</string>
<key>pfm_name</key>
<string>appTilesUPKUser</string>
```

```
<key>pfm_remove_duplicates</key>
<true/>
<key>pfm_targets</key>
<array>
    <string>user</string>
</array>
<key>pfm_title</key>
<string>Application Tiles Union Policy - User</string>
<key>pfm_type</key>
<string>union policy</string>
<key>pfm_upk_input_keys</key>
<array>
    <string>AppItems-Raw</string>
</array>
<key>pfm_upk_output_name</key>
<string>persistent-apps</string>
<key>pfm_upk_output_type</key>
<string>array</string>
```

Defining a Preference Key of Type URL

A preference key whose `pfm_type` is `url` causes the Workgroup Manager's Preference Manifest Editor to allow you to choose a file or folder. If you choose a `.webloc` file, the URL contained within the `.webloc` file is extracted and stored as a string type in the preference data. If you choose a file that is not a `.webloc` file, a `file://`-based URL is created to the selected file and this URL is stored as a string in the preference data.

Here is a sample definition for a preference key whose `pfm_type` is `url`.

```
<key>pfm_name</key>
<string>help source</string>
<key>pfm_type</key>
<string>url</string>
<key>pfm_default</key>
<string>http://help.apple.com</string>
<key>pfm_title</key>
<string>Web Help URL</string>
<key>pfm_description</key>
<string>Web-based help url.</string>
<key>pfm_targets</key>
<array>
    <string>user</string>
    <string>user-managed</string>
</array>
```

Localization

The Workgroup Manager's Preference Manifest Editor interprets the values specified by the `pfm_title` key and the `pfm_description` key as three comma-separated strings.

> **Note:** A localized preference manifest must itself be a bundle and cannot refer to resources of a larger application bundle that may contain the preference manifest.

Using the three comma-separated strings, the bundle containing the preference manifest file, and `NSBundle::-localizedStringForKey()`, the Workgroup Manager's Preference Manifest Editor finds and displays localized strings.

`NSBundle::-localizedStringForKey()` takes three strings as parameters: the name of a key, the value of a key, and the name of a string table to search for a localized string.

If the value of a `pfm_title` or `pfm_description` key contains no commas, the value is interpreted as the second parameter to `NSBundle::-localizedStringForKey()`, that is, the default string to be used if the key is not found. Here is an example:

```
<key>pfm_title</key>
<string>hiding</string>
```

If the value of a `pfm_title` or `pfm_description` key contains only one comma, the two resulting strings are interpreted as the first two parameters of `NSBundle::-localizedStringForKey()` and the empty string is used as the third parameter. The following examples are equivalent to each other:

```
<string>hiding, HIDING</string>
<string>hiding, HIDING,</string>
<string>hiding, HIDING, Localized.strings</string>
```

Any leading and trailing spaces that the strings may contain are ignored. The following examples are equivalent to each other:

```
<string>hiding, , myStringTable</string>
<string>hiding, HIDING, myStringTable</string>
```

To embed a comma within a string, precede the comma with a backslash (\ ) character. Here is an example that embeds two commas in the second string:

```
<string>hiding, Hiding\, Seeking\, &amp; Viewing ,</string>
```

Tasks

This chapter provides an example of a preference manifest file.

Preference Manifest Example

This example is a preference manifest file for the Dock keys that control hiding, position, tile size, andfolders added by the managed client software. The name of the file is `com.apple.dock.manifest`.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>pfm_description</key>
    <string>Dock preferences. Preferences are divided between  simple "appears"
 keys  and the more complicated Application and Document tile dictionaries,
</string>
    <key>pfm_domain</key>
    <string>com.apple.dock</string>
    <key>pfm_name</key>
    <string>Dock</string>
    <key>pfm_subkeys</key>
    <array>
        <dict>
            <key>pfm_default</key>
            <false/>
            <key>pfm_description</key>
            <string>True to turn hiding on.</string>
            <key>pfm_name</key>
            <string>autohide</string>
            <key>pfm_targets</key>
            <array>
                <string>user</string>
                <string>user-managed</string>
            </array>
            <key>pfm_title</key>
            <string>Hiding</string>
            <key>pfm_type</key>
            <string>boolean</string>
        </dict>
        <dict>
            <key>pfm_default</key>
            <string>bottom</string>
            <key>pfm_description</key>
            <string>Left, right, or bottom Dock position</string>
            <key>pfm_name</key>
            <string>orientation</string>
```

```
            <key>pfm_range_list</key>
            <array>
                <string>left</string>
                <string>bottom</string>
                <string>right</string>
            </array>
            <key>pfm_targets</key>
            <array>
                <string>user</string>
                <string>user-managed</string>
            </array>
            <key>pfm_title</key>
            <string>Position</string>
            <key>pfm_type</key>
            <string>string</string>
        </dict>
        <dict>
            <key>pfm_default</key>
            <real>64.0</real>
            <key>pfm_description</key>
            <string>Tile size.</string>
            <key>pfm_name</key>
            <string>tilesize</string>
            <key>pfm_range_max</key>
            <real>128.0</real>
            <key>pfm_range_min</key>
            <real>16.0</real>
            <key>pfm_targets</key>
            <array>
                <string>user</string>
                <string>user-managed</string>
            </array>
            <key>pfm_title</key>
            <string>Tile Size</string>
            <key>pfm_type</key>
            <string>real</string>
        </dict>
        <dict>
            <key>pfm_description</key>
            <string>Folders added by Managed Client.</string>
            <key>pfm_name</key>
            <string>MCXDockSpecialFolders-Raw</string>
            <key>pfm_repetition_max</key>
            <integer>0</integer>
            <key>pfm_subkeys</key>
            <array>
                <dict>
                    <key>pfm_description</key>
                  <string>Flag for one folder added by Managed  Client.</string>
                    <key>pfm_name</key>
                    <string>mcxfolderflag</string>
                    <key>pfm_range_list</key>
                    <array>
                        <string>AddDockMCXMyApplicationsFolder</string>
                        <string>AddDockMCXDocumentsFolder</string>
                        <string>AddDockMCXOriginalNetworkHomeFolder</string>
                    </array>
                    <key>pfm_repetition_max</key>
```

```
                        <integer>3</integer>
                        <key>pfm_repetition_min</key>
                        <integer>0</integer>
                        <key>pfm_title</key>
                        <string>Folder Flag</string>
                        <key>pfm_type</key>
                        <string>string</string>
                    </dict>
                </array>
                <key>pfm_targets</key>
                <array>
                    <string>user</string>
                    <string>user-managed</string>
                </array>
                <key>pfm_title</key>
                <string>Managed Client Special Folders</string>
                <key>pfm_type</key>
                <string>array</string>
            </dict>
        </array>
        <key>pfm_title</key>
        <string>Dock</string>
        <key>pfm_version</key>
        <real>0.5</real>
</dict>
</plist>
```

Preference Manifest Example

Document Revision History

This table describes the changes to *Preference Manifest Files for Managed Clients Overview*.

Date	Notes
2008-10-15	Corrected typos in example code
2005-04-29	New document that describes the format of preference manifest files, which are XML documents for storing and managing application preferences.