Technical Note TN2013

The 'plst' Resource

CONTENTS

This Technote describes the `'plst'` resource for Carbon-based applications.

This note is directed at anyone who is creating a single-binary Carbon CFM application.

[Feb 07 2001]

What is a `'plst'` (property list) resource?

1. A `'plst'` resource provides the information that previously resided in the 'vers', `'open'`, `'FREF'`, `'BNDL'`, and `'kind'` resources of pre-Mac OS X systems. In fact, on Mac OS X, the information in a `'plst'` resource supersedes the information in the old-style resources.
2. A `'plst'` resource is only useful to a single binary CFM Carbon application running on Mac OS X. A `'plst'` resource is ignored on pre-Mac OS X systems. Therefore you need to maintain your `'vers'`, `'open'`, `'FREF'`, `'BNDL'`, and `'kind'` resources if you plan to also run on pre-Mac OS X systems.
3. A `'plst'` resource must be of ID 0.
4. The `'plst'` resource is a single executable's equivalent of a bundled application's Info.plist file.

Back to top

How do I make a `'plst'` resource?

You can easily create a `'plst'` resource by hand. A `'plst'` resource is a list of key/value pairs which you can paste into a `'plst'` resource in your application. For example, you can create a new resource of type `'plst'` with ResEdit, and paste in the complete XML plist description. We recommend that you use the XML format for your `'plst'` resource.

Although there is no 'plst' resource template, you could also add the contents of your 'plst' file to your project by reading it in with the following line in any .r file:

read 'plst' (0) "MyFile.plist"

If the .r and .plist files are not in the same folder make sure that you use a full or relative path.

You can also use the `PropertyListEditor` application to create and edit a property list. The `PropertyListEditor` application can be found on Mac OS X in Developer/Applications provided you have installed the Developer Tools. We recommend that you use the PropertyListEditor to create the text for your 'plst'.

The utility "pl" is also available on Mac OS X. You can use "pl" to check the construction of your property list. Usage of this tool in the Terminal application would be, for example:

```
pl  yourPropertyListFile
```

Back to top

What keys should I use?

Single-file (non-bundled) applications should define the following keys:

`CFBundleIdentifier`

`CFBundleName`
`CFBundlePackageType`
`CFBundleSignature`
`CFBundleVersion`
`CFBundleShortVersionString`
`CFBundleLongVersionString`
`CFBundleIconFile`
`NSHumanReadableCopyright`
`NSAppleScriptEnabled`

The `CFBundlePackageType` (file type) and `CFBundleSignature` (creator) must match the type and creator of your executable binary.

LaunchServices uses the `NSAppleScriptEnabled` key in the same way it would the presence of an 'aete' resource to determine whether a Classic or single-file CFM app with plst is scriptable.

Optional, but highly recommended keys:

`LSPrefersCarbon`
`LSPrefersClassic`
`LSRequiresCarbon`
`LSRequiresClassic`

> **Note:**
> Only one of the above four keys makes sense for a given app. In the absence of any specific declaration in the `'plst'` for one of these four keys, the `LSPrefersCarbon` key is "ON" by default if the app has a `'carb'` 0 resource but the "LSRequiresCarbon" key is "ON" by default if the app does not have a `'carb'` 0 resource.

The `LSPrefersCarbon` and `LSPrefersClassic` keys allow the user to override, via the Mac OS X Finder's Show Info "Open in the Classic environment" check-box, the developer-provided default, while the `LSRequiresCarbon` and `LSRequiresClassic` keys do not.

If your application supports documents, it can also define an entry per document type that includes the following:

`CFBundleTypeExtensions` - Extensions for this file type
`CFBundleTypeOSTypes` - OSTypes for this file type
`CFBundleTypeIconFile` - Resource ID for the icon for this file type
`CFBundleTypeName` - User visible name for this file type, i.e., the kind string used by the Finder and others.
`CFBundleTypeRole` - Role of the application for this file type (Editor/Viewer)
`LSTypeIsPackage` - This document type is a package, whether defined by a `CFBundleOSType` or `CFBundleFileExtension`. "Off" by default. Applications that claim package types for documents (such as .rtfd for TextEdit and .pbproj for Project Builder) should add this key (with Boolean value Yes).

Back to top

A sample `'plst'`

Here's a sample `'plst'` resource that SimpleText might have:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist SYSTEM "file://localhost/System/Library/DTDs/PropertyList.dtd">
<plist version="0.9">
<dict>
    <key>CFBundleInfoDictionaryVersion</key>
    <string>6.0</string>
    <key>CFBundleIdentifier</key>
    <string>com.apple.SimpleText</string>
    <key>CFBundleName</key>
    <string>Simple Text</string>
```

```
    <key>CFBundlePackageType</key>
    <string>APPL</string>
    <key>CFBundleSignature</key>
    <string>ttxt</string>
    <key>CFBundleDevelopmentRegion</key>
    <string>English</string>
    <key>LSPrefersCarbon</key>
    <true/>
    <key>CFBundleVersion</key>
    <string>1.4</string>
    <key>CFBundleShortVersionString</key>
    <string>1.4</string>
    <key>CFBundleLongVersionString</key>
    <string>1.4, Copyright 1985-2001 Apple Computer</string>
    <key>CFBundleIconFile</key>
    <string>128</string>
    <key>NSAppleScriptEnabled</key>
    <string>Yes</string>
    <key>NSHumanReadableCopyright</key>
    <string>Copyright (c) 1985-2001 Apple Computer</string>
    <key>CFBundleDocumentTypes</key>
    <array>
        <dict>
            <key>CFBundleTypeOSTypes</key>
            <array>
                <string>TEXT</string>
            </array>
            <key>CFBundleTypeIconFile</key>
            <string>129</string>
            <key>CFBundleTypeName</key>
            <string>Text document</string>
            <key>CFBundleTypeExtensions</key>
            <array>
                <string>txt</string>
                <string>text</string>
            </array>
            <key>CFBundleTypeRole</key>
            <string>Editor</string>
        </dict>
        <dict>
            <key>CFBundleTypeOSTypes</key>
            <array>
                <string>ttro</string>
            </array>
            <key>CFBundleTypeIconFile</key>
            <string>130</string>
            <key>CFBundleTypeName</key>
            <string>Read Only document</string>
            <key>CFBundleTypeRole</key>
            <string>Viewer</string>
        </dict>
        <dict>
            <key>CFBundleTypeOSTypes</key>
            <array>
                <string>PICT</string>
            </array>
            <key>CFBundleTypeIconFile</key>
            <string>131</string>
            <key>CFBundleTypeName</key>
            <string>PICT document</string>
            <key>CFBundleTypeExtensions</key>
            <array>
                <string>pict</string>
            </array>
            <key>CFBundleTypeRole</key>
            <string>Viewer</string>
        </dict>
    </array>
</dict>
</plist>
```

Back to top

Additional Notes & Comments

CFBundleIconFile and CFBundleTypeIconFile

The CFBundleIconFile and CFBundleTypeIconFile values refer to 'icns' resource ids. If your app were packaged as a bundle, rather than a single binary app, the values would refer to .icns files within your bundle. You may need to log out/log in to Mac OS X to see the result of modifications to these and other keys.

CFBundleTypeOSTypes of CFBundleDocumentTypes

As stated in TN 1085, "Using the Drag Manager to Interact with and Manipulate File System Entities", the fileCreator 'MACS' and the fileType 'fold' and 'disk' tell the Finder that your application will accept folders and volumes dropped onto your application's icon. These types are also honored as valid CFBundleTypeOSTypes, however there is no need to specify creator because only type is used to determine whether a file is accepted. This is similar to placing an extension on a document and specifying that extension as a CFBundleTypeExtension value.

CFBundleInfoDictionaryVersion

The CFBundleInfoDictionaryVersionkey is simply the version of the info dictionary format itself, so that in case Apple ever decides to change this format in the future we will be able to distinguish old from new. The value for this key should be 6.0

CSResourcesFileMapped

The CSResourcesFileMappedkey causes the CFBundle to open the resource fork as a mapped read-only file; attempts to write to this file would fail with a seg fault. There are two memory footprint advantages to adopting this key. The first is that the resource-map is not copied into memory (the in-file map is just used). The second one is that all handles are special handles that just point to the file mapped data (instead of an allocated copy). This key is a boolean, and by default its value is false.

The 'carb' resource

An empty (0-byte length) 'carb' resource of ID 0 was an early method of informing Mac OS X that the application possessing the 'carb' resource was a Carbon application. The 'plst' resource supersedes the 'carb' resource, except in one case: on Mac OS 9.1, the Process Manager will arrange for increased stack and heap memory for a Carbon application provided the application possesses a 'carb' 0 resource.

Back to top

References

You can find information related to this topic in the Mac OS X Release Notes for InfoPlists and CFBundles, and in the System Overview book, and in Core Foundation documentation concerning CFBundles.

A good place to start for information about XML is the www.XML.com website.

Back to top

Downloadables

Acrobat version of this Note (116K). Download

Back to top
