CFPlugIn Reference

Core Foundation

2006-02-07

Apple Inc.
© 2003, 2006 Apple Computer, Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

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

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

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

iPhone is a trademark of Apple Inc.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

CFPlugIn Reference

Derived From:	CFType
Framework:	CoreFoundation/CoreFoundation.h
Companion guide	Plug-ins
Declared in	CFBundle.h CFPlugIn.h

Overview

`CFPlugIn` provides a standard architecture for application extensions. With `CFPlugIn`, you can design your application as a host framework that uses a set of executable code modules called plug-ins to provide certain well-defined areas of functionality. This approach allows third-party developers to add features to your application without requiring access to your source code. You can also bundle together plug-ins for multiple platforms and let `CFPlugIn` transparently load the appropriate plug-in at runtime. You can use `CFPlugIn` to add plug-in capability to, or write a plug-in for, your application.

Functions by Task

Creating Plug-Ins

`CFPlugInCreate` (page 7)
Creates a CFPlugIn given its URL.

`CFPlugInInstanceCreate` (page 9)
Creates a `CFPlugIn` instance of a given type using a given factory.

Registration

`CFPlugInRegisterFactoryFunction` (page 10)
Registers a factory function and its UUID with a `CFPlugIn` object.

`CFPlugInRegisterFactoryFunctionByName` (page 11)
Registers a factory function with a `CFPlugIn` object using the function's name instead of its UUID.

`CFPlugInRegisterPlugInType` (page 11)
Registers a type and its corresponding factory function with a `CFPlugIn` object.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

`CFPlugInUnregisterFactory` (page 13)
Removes the given function from a plug-in's list of registered factory functions.

`CFPlugInUnregisterPlugInType` (page 13)
Removes the given type from a plug-in's list of registered types.

CFPlugIn Miscellaneous Functions

`CFPlugInAddInstanceForFactory` (page 6)
Registers a new instance of a type with `CFPlugIn`. .

`CFPlugInFindFactoriesForPlugInType` (page 7)
Searches all registered plug-ins for factory functions capable of creating an instance of the given type.

`CFPlugInFindFactoriesForPlugInTypeInPlugIn` (page 8)
Searches the given plug-in for factory functions capable of creating an instance of the given type.

`CFPlugInGetBundle` (page 8)
Returns a plug-in's bundle.

`CFPlugInGetTypeID` (page 9)
Returns the type identifier for the `CFPlugIn` opaque type.

`CFPlugInIsLoadOnDemand` (page 10)
Determines where or not a plug-in is loaded on demand.

`CFPlugInRemoveInstanceForFactory` (page 12)
Unregisters an instance of a type with `CFPlugIn`.

`CFPlugInSetLoadOnDemand` (page 12)
Enables or disables load on demand for plug-ins that do dynamic registration (only when a client requests an instance of a supported type).

Functions

CFPlugInAddInstanceForFactory

Registers a new instance of a type with `CFPlugIn`. .

```
void CFPlugInAddInstanceForFactory (
   CFUUIDRef factoryID
);
```

Parameters

factoryID
The `CFUUID` object representing the plug-in factory.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
CoreRecipes
Departments and Employees

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

SampleCMPlugIn
Spotlight
SpotlightFortunes

Declared In
`CFPlugIn.h`

CFPlugInCreate

Creates a CFPlugIn given its URL.

```
CFPlugInRef CFPlugInCreate (
   CFAllocatorRef allocator,
   CFURLRef plugInURL
);
```

Parameters

allocator
: The allocator to use to allocate memory for the new plug-in. Pass `NULL` or `kCFAllocatorDefault` to use the default allocator.

plugInURL
: The location of the plug-in.

Return Value
A new plug-in. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
BasicPlugIn

Declared In
`CFPlugIn.h`

CFPlugInFindFactoriesForPlugInType

Searches all registered plug-ins for factory functions capable of creating an instance of the given type.

```
CFArrayRef CFPlugInFindFactoriesForPlugInType (
   CFUUIDRef typeUUID
);
```

Parameters

typeUUID
: A UUID type.

Return Value
An array of UUIDs for factory functions capable of creating an instance of the given type.

Availability
Available in CarbonLib v1.1 and later.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInFindFactoriesForPlugInTypeInPlugIn

Searches the given plug-in for factory functions capable of creating an instance of the given type.

```
CFArrayRef CFPlugInFindFactoriesForPlugInTypeInPlugIn (
   CFUUIDRef typeUUID,
   CFPlugInRef plugIn
);
```

Parameters

typeUUID

A UUID type.

plugIn

The plug-in to search.

Return Value
An array of UUIDs for factory functions capable of creating an instance of the given type.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
BasicPlugIn

Declared In
`CFPlugIn.h`

CFPlugInGetBundle

Returns a plug-in's bundle.

```
CFBundleRef CFPlugInGetBundle (
   CFPlugInRef plugIn
);
```

Parameters

plugIn

The plug-in whose bundle to obtain.

Return Value
The bundle for *plugIn*. Ownership follows the Get Rule.

Discussion
You should *always* use this function to get a plug-in's bundle. Never attempt to access the plug-in directly as a bundle.

Availability
Available in CarbonLib v1.0 and later.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInGetTypeID

Returns the type identifier for the `CFPlugIn` opaque type.

```
CFTypeID CFPlugInGetTypeID (
   void
);
```

Return Value
The type identifier for the `CFPlugIn` opaque type.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInInstanceCreate

Creates a `CFPlugIn` instance of a given type using a given factory.

```
void * CFPlugInInstanceCreate (
   CFAllocatorRef allocator,
   CFUUIDRef factoryUUID,
   CFUUIDRef typeUUID
);
```

Parameters

`allocator`
The allocator to use to allocate memory for the new object. Pass `NULL` or `kCFAllocatorDefault` to use the default allocator.

`factoryUUID`
The UUID representing the factory function to use to create a plug-in of the given type.

`typeUUID`
The UUID type.

Return Value
Returns the IUnknown interface for the new plug-in.

Discussion
The plug-in host uses this function to create an instance of the given type. Unless the plug-in is using dynamic registration, this function causes the plug-in's code to be loaded into memory.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Related Sample Code
BasicPlugIn

Declared In
`CFPlugIn.h`

CFPlugInIsLoadOnDemand

Determines where or not a plug-in is loaded on demand.

```
Boolean CFPlugInIsLoadOnDemand (
   CFPlugInRef plugIn
);
```

Parameters

plugIn

The plug-in to query.

Return Value
`true` if the plug-in is loaded only when a client requests an instance of a supported type, otherwise `false`.

Discussion
Plug-ins that do static registration are load on demand by default. Plug-ins that do dynamic registration are not load on demand by default.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInRegisterFactoryFunction

Registers a factory function and its UUID with a `CFPlugIn` object.

```
Boolean CFPlugInRegisterFactoryFunction (
   CFUUIDRef factoryUUID,
   CFPlugInFactoryFunction func
);
```

Parameters

factoryUUID

The `CFUUID` object representing the factory function to register.

func

The factory function pointer to register.

Return Value
`true` if the factory function was successfully registered, otherwise `false`.

Discussion
This function is used by a plug-in or host when performing dynamic registration.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInRegisterFactoryFunctionByName

Registers a factory function with a `CFPlugIn` object using the function's name instead of its UUID.

```
Boolean CFPlugInRegisterFactoryFunctionByName (
   CFUUIDRef factoryUUID,
   CFPlugInRef plugIn,
   CFStringRef functionName
);
```

Parameters

`factoryUUID`
The `CFUUID` object representing the factory function to register.

`plugIn`
The plug-in containing *`functionName`*.

`functionName`
The name of the factory function to register.

Return Value
`true` if the factory function was successfully registered, otherwise `false`.

Discussion
This function is used by a plug-in or host when performing dynamic registration.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInRegisterPlugInType

Registers a type and its corresponding factory function with a `CFPlugIn` object.

```
Boolean CFPlugInRegisterPlugInType (
   CFUUIDRef factoryUUID,
   CFUUIDRef typeUUID
);
```

Parameters

`factoryUUID`
The `CFUUID` object representing the factory function that can create the type being registered.

`typeUUID`
The UUID type to register.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Return Value
`true` if the factory function was successfully registered, otherwise `false`.

Discussion
This function is used by a plug-in or host when performing dynamic registration.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInRemoveInstanceForFactory

Unregisters an instance of a type with `CFPlugIn`.

```
void CFPlugInRemoveInstanceForFactory (
   CFUUIDRef factoryID
);
```

Parameters
`factoryID`
The `CFUUID` object representing the plug-in factory.

Discussion
If the instance counts of every factory in a plug-in are zero, the plug-in can be unloaded.]

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
CoreRecipes
Departments and Employees
SampleCMPlugIn
Spotlight
SpotlightFortunes

Declared In
`CFPlugIn.h`

CFPlugInSetLoadOnDemand

Enables or disables load on demand for plug-ins that do dynamic registration (only when a client requests an instance of a supported type).

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

```
void CFPlugInSetLoadOnDemand (
   CFPlugInRef plugIn,
   Boolean flag
);
```

Parameters

plugIn

The plug-in to be loaded on demand.

flag

`true` to enable load on demand, `false` otherwise.

Discussion

Plug-ins that do static registration are load on demand by default. Plug-ins that do dynamic registration are not load on demand by default.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In

`CFPlugIn.h`

CFPlugInUnregisterFactory

Removes the given function from a plug-in's list of registered factory functions.

```
Boolean CFPlugInUnregisterFactory (
   CFUUIDRef factoryUUID
);
```

Parameters

factoryUUID

The `CFUUID` object representing the factory to unregister.

Return Value

`true` if the factory function was successfully unregistered, otherwise `false`.

Discussion

Used by a plug-in or host when performing dynamic registration.

Availability

Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Declared In

`CFPlugIn.h`

CFPlugInUnregisterPlugInType

Removes the given type from a plug-in's list of registered types.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

```
Boolean CFPlugInUnregisterPlugInType (
   CFUUIDRef factoryUUID,
   CFUUIDRef typeUUID
);
```

Parameters

factoryUUID

The `CFUUID` object representing the factory function for the type to unregister.

typeUUID

The UUID type to unregister.

Return Value

`true` if the factory function was successfully unregistered, otherwise `false`.

Discussion

Used by a plug-in or host when performing dynamic registration.

Availability

Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Declared In

`CFPlugIn.h`

Callbacks

CFPlugInDynamicRegisterFunction

A callback which provides a plug-in the opportunity to dynamically register its types with a host.

```
typedef void (*CFPlugInDynamicRegisterFunction) (
    CFPlugInRef plugIn
);
```

If you name your function `MyCallBack`, you would declare it like this:

```
void MyCallBack (
    CFPlugInRef plugIn
);
```

Parameters

plugIn

The `CFPlugIn` object that is engaged in dynamic registration. When using in C++, this parameter functions as a `this` pointer for the plug-in.

Discussion

This callback is called as a plug-in is being loaded. This provides the plugin the means to dynamically register its types and factories with a plug-in's host. The call is triggered by the presence of `kCFPlugInDynamicRegistrationKey` (page 16) in the plug-in's information property list.

Availability

Available in Mac OS X v10.0 and later.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Declared In
`CFPlugIn.h`

CFPlugInFactoryFunction

Callback function that a plug-in author must implement to create a plug-in instance.

```
typedef void *(*CFPlugInFactoryFunction) (
    CFAllocatorRef allocator,
    CFUUIDRef typeUUID
);
```

If you name your function `MyCallBack`, you would declare it like this:

```
void *MyCallBack (
    CFAllocatorRef allocator,
    CFUUIDRef typeUUID
);
```

Parameters

allocator
The allocator to use to allocate memory for the new object. Pass `NULL` or `kCFAllocatorDefault` to use the default allocator.

typeUUID
The UUID type to instantiate.

Discussion
The plug-in author's implementation of this function is registered with `CFPlugIn` either statically in the plug-in's information property list, or dynamically. This function is executed as a result of a call to `CFPlugInInstanceCreate` (page 9) by the plug-in host.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

CFPlugInUnloadFunction

Callback function that is called, if present, just before a plug-in's code is unloaded.

```
typedef void (*CFPlugInUnloadFunction) (
    CFPlugInRef plugIn
);
```

If you name your function `MyCallBack`, you would declare it like this:

```
void MyCallBack (
    CFPlugInRef plugIn
);
```


2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Parameters

plugIn

The `CFPlugIn` object that is about to be unloaded from memory. When writing in C++, this parameter functions as a `this` pointer for the plug-in.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`CFPlugIn.h`

Data Types

CFPlugInRef

A reference to a `CFPlugin` object.

```
typedef struct __CFBundle *CFPlugInRef;
```

Availability
Available in Mac OS X version 10.0 and later.

Declared In
`CFBundle.h`

Constants

Information Property List Keys

A plug-in's information property list can contain these keys used for registering types, factories, and interfaces.

```
const CFStringRef kCFPlugInDynamicRegistrationKey;
const CFStringRef kCFPlugInDynamicRegisterFunctionKey;
const CFStringRef kCFPlugInUnloadFunctionKey;
const CFStringRef kCFPlugInFactoriesKey;
const CFStringRef kCFPlugInTypesKey;
```

Constants

`kCFPlugInDynamicRegistrationKey`

Indicates whether a plug-in requires dynamic registration.

Available in Mac OS X v10.0 and later.

Declared in `CFPlugIn.h`.

`kCFPlugInDynamicRegisterFunctionKey`

Used to specify a plug-in's registration function.

Available in Mac OS X v10.0 and later.

Declared in `CFPlugIn.h`.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

`kCFPlugInUnloadFunctionKey`
: Used to specify a plug-in's unload function.

  Available in Mac OS X v10.0 and later.

  Declared in `CFPlugIn.h`.

`kCFPlugInFactoriesKey`
: Used to statically register factory functions.

  Available in Mac OS X v10.0 and later.

  Declared in `CFPlugIn.h`.

`kCFPlugInTypesKey`
: Used to statically register the factories that can create each supported type.

  Available in Mac OS X v10.0 and later.

  Declared in `CFPlugIn.h`.

Availability
Mac OS X version 10.0 and later

Declared In
`CFPlugIn.h`

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Document Revision History

This table describes the changes to *CFPlugIn Reference*.

Date	Notes
2006-02-07	Made formatting changes.
2005-08-11	Cosmetic changes to conform to documentation guidelines.
2003-01-01	First version of this document.

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Document Revision History

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.

Index

C

I

K

2006-02-07 | © 2003, 2006 Apple Computer, Inc. All Rights Reserved.