Internet Config Reference

Carbon > Networking

2003-05-01

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

.Mac is a registered service mark of Apple Inc.

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

Finder is a trademark of Apple Inc.

PowerPC and and the PowerPC logo are trademarks of International Business Machines Corporation, used under license therefrom.

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

Internet Config Reference

Framework:	Carbon/Carbon.h
Declared in	InternetConfig.h

Overview

Internet Config, a Mac OS 8 and 9 API, supports centralized entry and management of Internet preferences for all of a user's Internet applications. For example, email programs and Web browsers can obtain a user's name, email address, home page, incoming mail server, and similar preferences from one common place that is easily edited by the user via the Internet Config application.

Mac OS X applications should employ Launch Services and System Configuration for managing Internet preferences. In Mac OS X, Internet Config calls through to these newer APIs. Using them directly increases your application's efficiency.

If you use Internet Config in Mac OS X, perhaps to maintain backward compatibility for your application with Mac OS 8 and 9, here are some ways to optimize performance.

- Getting multiple preferences at once.

  Your program may access several Internet Config preferences within a single operation. For example, an FTP program on startup might always get the kICFTPHost, kICFTPProxyHost, kICFTPProxyAccount, kICFTPProxyPassword, and kICFTPProxyUser preferences. Bracket such a sequence of ICGetPref calls with an ICBegin and ICEnd function pair for significantly faster performance.

- Calling Internet Config less often.

  Your application can cache Internet Config preference data and then watch for changes using one of the preference coherency strategies described in the developer's online documentation:

  http://www.quinn.echidna.id.au/Quinn/Config/Prog_Docs.html#PreferenceCoherency

- Using preferences from underlying frameworks.

  Starting with Mac OS X, Internet Config is no longer the final authority on Internet preferences. For example, Launch Services stores URL helper application settings, and System Configuration stores the various proxy settings. When called by an application, Internet Config must call through to the appropriate underlying framework. Call Launch Services and System Configuration directly to improve performance.

The Mac OS 8 and 9 implementation of Internet Config is in the public domain, and Carbon supports all of its commonly-used functions. Functions that are not compatible with preemptive threads are not supported, and Internet Config no longer supports multiple configuration files. This document contains information on replacement functions.

For more information on Internet Config see its developer's web site:

http://www.quinn.echidna.id.au/Quinn/Config

Functions by Task

Starting and Stopping Internet Config

These functions let you create, configure, and destroy connections to Internet Config, denoted by the `ICInstance` type. Although it is usual to create one connection when your program starts and destroy it when it terminates, you can create an arbitrary number of connections at any time.

`ICStart` (page 51)
: Call this routine to start using Internet Config—typically at application initialization time—passing it your program's creator code.

`ICGetVersion` (page 44)
: Returns the version of Internet Config installed on the system.

`ICStop` (page 51)
: Call this when your application is done using Internet Config, passing it the instance you got from `ICStart`.

Getting Information About an Instance

`ICGetConfigName` (page 38)
: Returns a displayable string that represents the instance's current configuration.

`ICGetSeed` (page 43)
: Returns the seed for the current preferences set.

`ICGetPerm` (page 42)
: Returns the current permissions for this instance, i.e. the permission value you used when you called `ICBegin`, or returns `ioNoPerm` if you haven't call `ICBegin`.

Preparing to Read and Write Preferences

These routines are not always required because the commonly used reading and writing calls perform this operation automatically. However, even in that case, these routines are useful if you are making repeated calls because they allow those calls to work faster.

`ICBegin` (page 15)
: Prepares Internet Config to read (and, optionally, to write) preferences.

`ICEnd` (page 36)
: Tells Internet Config that you have finished accessing preference information.

Reading and Writing Preferences

`ICGetPref` (page 42)
: Given a preference's key, gets the preference's data and places it into a buffer that you supply.

ICSetPref (page 49)
    Sets a preference given its key, attributes, and a buffer containing the preference data.

ICFindPrefHandle (page 37)
    Gets a preference's data, like ICGetPref, but returns the data in a handle.

ICSetPrefHandle (page 50)
    Sets a preference, like ICSetPref, but takes its input as a handle.

ICGetDefaultPref (page 40)
    Gets the default value for the preference associated with the specified key.

ICDeletePref (page 35)
    Deletes the preference for the specified key.

Enumerating All Preferences

You must call ICBegin before calling any of these routines.

ICCountPref (page 29)
    Returns the total number of preferences available.

ICGetIndPref (page 40)
    Returns the key associated with the preference at the specified index.

Accessing the User Interface

We recommend that you do not provide a user interface for editing Internet Config preferences from within your application. In Mac OS X, simply provide a way for the user to open the Internet pane of System Preferences. The ICEditPreferences function provides support for this.

ICEditPreferences (page 36)
    Opens the Internet pane of System Preferences.

URL Functions

ICParseURL (page 47)
    Parses a URL out of the specified text and returns it to the calling program.

ICLaunchURL (page 44)
    Parses a URL out of the specified text and feeds it to the appropriate helper application.

ICCreateGURLEvent (page 31)
    Creates a GURL Apple event, targetted at the application with the specified creator code

ICSendGURLEvent (page 48)
    Sends a specified Apple Event to the target application.

Mapping-Database High-Level Functions

Internet Config's high-level functions are suitable for applications that want to easily look up a file type and creator based on an extension, or vice versa. These functions are significantly slower than their lower level counterparts, especially if you call them repeatedly.

ICMapFilename (page 46)
Given a filename, returns the most appropriate ICMapEntry based on the filename extension.

ICMapTypeCreator (page 46)
Given a file type and creator (and optionally a filename), returns the most appropriate ICMapEntry.

Mapping-Database Mid-Level Functions

These functions are useful if you are doing multiple searches because they avoid the overhead of accessing the mappings database each time.

ICMapEntriesFilename (page 45)
Given a filename, returns the most appropriate ICMapEntry based on the filename extension—without the overhead of a mappings database access.

ICMapEntriesTypeCreator (page 45)
Given a file type and creator (and optionally a filename), returns the most appropriate ICMapEntry—without the overhead of a mappings database access.

Mapping-Database Low-Level Functions

Internet Config's low-level functions give you access to the primitive operations used to implement the other mapping functions.

ICCountMapEntries (page 29)
Counts the number of entries in the mappings database preference provided.

ICGetIndMapEntry (page 40)
Returns the mappings database entry for a specified index .

ICGetMapEntry (page 41)
Returns a mappings database entry based on its position in the Mapping preference.

ICSetMapEntry (page 49)
Sets an entry in a specified Mapping preference.

ICDeleteMapEntry (page 35)
Deletes an entry in the specified Mapping preference.

ICAddMapEntry (page 14)
Adds an entry to the mappings database.

Profile Functions

ICGetCurrentProfile (page 39)
Returns the profile ID of the current profile.

ICSetCurrentProfile (page 48)
Sets the current profile to a specified profile ID.

ICCountProfiles (page 30)
Returns the number of available profiles.

ICGetIndProfile (page 41)
Returns the ID of the profile at the specified index.

`ICGetProfileName` (page 43)
Returns the name of the profile that has the specified ID.

`ICSetProfileName` (page 50)
Sets the name of the profile that has the specified ID.

`ICAddProfile` (page 15)
Creates a new profile and returns its ID.

`ICDeleteProfile` (page 36)
Deletes the profile that has the specified ID.

Deprecated functions

`ICGetPrefHandle` (page 43)
Deprecated. While this function works in Mac OS X, you should use ICFindPrefHandle instead.

Unsupported Functions

`ICCSetPref` (page 33)

`ICCAddMapEntry` (page 15)

`ICCAddProfile` (page 16)

`ICCBegin` (page 16)

`ICCChooseConfig` (page 16)

`ICCChooseNewConfig` (page 16)

`ICCCountMapEntries` (page 17)

`ICCCountPref` (page 17)

`ICCCountProfiles` (page 17)

`ICCCreateGURLEvent` (page 18)

`ICCDefaultFileName` (page 18)

`ICCDeleteMapEntry` (page 18)

`ICCDeletePref` (page 19)

ICCDeleteProfile (page 19)

ICCEditPreferences (page 19)

ICCEnd (page 19)

ICCFindConfigFile (page 20)

ICCFindPrefHandle (page 20)

ICCFindUserConfigFile (page 20)

ICCGeneralFindConfigFile (page 21)

ICCGetComponentInstance (page 21)

ICCGetConfigName (page 21)

ICCGetConfigReference (page 22)

ICCGetCurrentProfile (page 22)

ICCGetDefaultPref (page 22)

ICCGetIndMapEntry (page 23)

ICCGetIndPref (page 23)

ICCGetIndProfile (page 23)

ICCGetMapEntry (page 24)

ICCGetMappingInterruptSafe (page 24)

ICCGetPerm (page 24)

ICCGetPref (page 25)

ICCGetPrefHandle (page 25)

ICCGetProfileName (page 25)

ICCGetSeed (page 26)

ICCGetSeedInterruptSafe (page 26)

ICCGetVersion (page 26)

ICChooseConfig (page 27)

ICChooseNewConfig (page 27)

ICCLaunchURL (page 27)

ICCMapEntriesFilename (page 28)

ICCMapEntriesTypeCreator (page 28)

ICCMapFilename (page 28)

ICCMapTypeCreator (page 29)

ICCParseURL (page 30)

ICCRefreshCaches (page 31)

ICCRequiresInterruptSafe (page 31)

ICCSendGURLEvent (page 31)

ICCSetConfigReference (page 32)

ICCSetCurrentProfile (page 32)

ICCSetMapEntry (page 32)

ICCSetPrefHandle (page 33)

ICCSetProfileName (page 33)

ICCSpecifyConfigFile (page 34)

ICCStart (page 34)

ICCStop (page 34)

ICDefaultFileName (page 35)

Functions

ICAddMapEntry

Adds an entry to the mappings database.

```
OSStatus ICAddMapEntry (
   ICInstance inst,
   Handle entries,
   const ICMapEntry *entry
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICAddProfile

Creates a new profile and returns its ID.

```
OSStatus ICAddProfile (
   ICInstance inst,
   ICProfileID prototypeID,
   ICProfileID *newID
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICBegin

Prepares Internet Config to read (and, optionally, to write) preferences.

```
OSStatus ICBegin (
   ICInstance inst,
   ICPerm perm
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICCAddMapEntry

Unsupported

```
OSStatus ICCAddMapEntry (
    ComponentInstance inst,
    Handle entries,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICAddMapEntry` (page 14) instead.

Declared In

`InternetConfig.h`

ICCAddProfile

Unsupported

```
OSStatus ICCAddProfile (
    ComponentInstance inst,
    ICProfileID prototypeID,
    ICProfileID *newID
);
```

Carbon Porting Notes

Use `ICAddProfile` (page 15) instead.

Declared In

`InternetConfig.h`

ICCBegin

Unsupported

```
OSStatus ICCBegin (
    ComponentInstance inst,
    ICPerm perm
);
```

Carbon Porting Notes

Use `ICBegin` (page 15) instead.

Declared In

`InternetConfig.h`

ICCChooseConfig

Unsupported

```
OSStatus ICCChooseConfig (
    ComponentInstance inst
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

`InternetConfig.h`

ICCChooseNewConfig

Unsupported

```
OSStatus ICCChooseNewConfig (
    ComponentInstance inst
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

`InternetConfig.h`

ICCCountMapEntries

Unsupported

```
OSStatus ICCCountMapEntries (
    ComponentInstance inst,
    Handle entries,
    SInt32 *count
);
```

Carbon Porting Notes

Use `ICCountMapEntries` (page 29) instead.

Declared In

`InternetConfig.h`

ICCCountPref

Unsupported

```
OSStatus ICCCountPref (
    ComponentInstance inst,
    SInt32 *count
);
```

Carbon Porting Notes

Use `ICCountPref` (page 29) instead.

Declared In

`InternetConfig.h`

ICCCountProfiles

Unsupported

```
OSStatus ICCCountProfiles (
    ComponentInstance inst,
    SInt32 *count
);
```

Carbon Porting Notes

Use `ICCountProfiles` (page 30) instead.

Declared In

`InternetConfig.h`

ICCCreateGURLEvent

Unsupported

```
OSStatus ICCCreateGURLEvent (
    ComponentInstance inst,
    OSType helperCreator,
    Handle urlH,
    AppleEvent *theEvent
);
```

Carbon Porting Notes

Use `ICCreateGURLEvent` (page 31) instead.

Declared In

`InternetConfig.h`

ICCDefaultFileName

Unsupported

```
OSStatus ICCDefaultFileName (
    ComponentInstance inst,
    Str63 name
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICCDeleteMapEntry

Unsupported

```
OSStatus ICCDeleteMapEntry (
    ComponentInstance inst,
    Handle entries,
    SInt32 pos
);
```

Carbon Porting Notes

Use `ICDeleteMapEntry` (page 35) instead.

Declared In

`InternetConfig.h`

ICCDeletePref

Unsupported

```
OSStatus ICCDeletePref (
    ComponentInstance inst,
    ConstStr255Param key
);
```

Carbon Porting Notes

Use `ICDeletePref` (page 35) instead.

Declared In

`InternetConfig.h`

ICCDeleteProfile

Unsupported

```
OSStatus ICCDeleteProfile (
    ComponentInstance inst,
    ICProfileID thisID
);
```

Carbon Porting Notes

Use `ICDeleteProfile` (page 36) instead.

Declared In

`InternetConfig.h`

ICCEditPreferences

Unsupported

```
OSStatus ICCEditPreferences (
    ComponentInstance inst,
    ConstStr255Param key
);
```

Carbon Porting Notes

Use `ICEditPreferences` (page 36) instead.

Declared In

`InternetConfig.h`

ICCEnd

Unsupported

```
OSStatus ICCEnd (
    ComponentInstance inst
);
```

Carbon Porting Notes

Use `ICEnd` (page 36) instead.

Declared In

`InternetConfig.h`

ICCFindConfigFile

Unsupported

```
OSStatus ICCFindConfigFile (
    ComponentInstance inst,
    SInt16 count,
    ICDirSpecArrayPtr folders
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICCFindPrefHandle

Unsupported

```
OSStatus ICCFindPrefHandle (
    ComponentInstance inst,
    ConstStr255Param key,
    ICAttr *attr,
    Handle prefh
);
```

Carbon Porting Notes

Use `ICFindPrefHandle` (page 37) instead.

Declared In

`InternetConfig.h`

ICCFindUserConfigFile

Unsupported

```
OSStatus ICCFindUserConfigFile (
    ComponentInstance inst,
    ICDirSpec *where
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICCGeneralFindConfigFile

Unsupported

```
OSStatus ICCGeneralFindConfigFile (
    ComponentInstance inst,
    Boolean searchPrefs,
    Boolean canCreate,
    SInt16 count,
    ICDirSpecArrayPtr folders
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICCGetComponentInstance

Unsupported

```
OSStatus ICCGetComponentInstance (
    ComponentInstance inst,
    ComponentInstance *componentInst
);
```

Carbon Porting Notes

Because Internet Config is not component-based under Mac OS X, use `ICGetVersion` (page 44) instead.

Declared In

`InternetConfig.h`

ICCGetConfigName

Unsupported

```
OSStatus ICCGetConfigName (
    ComponentInstance inst,
    Boolean longname,
    Str255 name
);
```

Carbon Porting Notes

Use `ICGetConfigName` (page 38) instead.

Declared In

`InternetConfig.h`

ICCGetConfigReference

Unsupported

```
OSStatus ICCGetConfigReference (
    ComponentInstance inst,
    ICConfigRefHandle ref
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

`InternetConfig.h`

ICCGetCurrentProfile

Unsupported

```
OSStatus ICCGetCurrentProfile (
    ComponentInstance inst,
    ICProfileID *currentID
);
```

Carbon Porting Notes

Use `ICGetCurrentProfile` (page 39) instead.

Declared In

`InternetConfig.h`

ICCGetDefaultPref

Unsupported

```
OSStatus ICCGetDefaultPref (
    ComponentInstance inst,
    ConstStr255Param key,
    Handle prefH
);
```

Carbon Porting Notes

Use `ICGetDefaultPref` (page 40) instead.

Declared In

`InternetConfig.h`

ICCGetIndMapEntry

Unsupported

```
OSStatus ICCGetIndMapEntry (
    ComponentInstance inst,
    Handle entries,
    SInt32 index,
    SInt32 *pos,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICGetIndMapEntry` (page 40) instead.

Declared In

`InternetConfig.h`

ICCGetIndPref

Unsupported

```
OSStatus ICCGetIndPref (
    ComponentInstance inst,
    SInt32 index,
    Str255 key
);
```

Carbon Porting Notes

Use `ICGetIndPref` (page 40) instead.

Declared In

`InternetConfig.h`

ICCGetIndProfile

Unsupported

```
OSStatus ICCGetIndProfile (
    ComponentInstance inst,
    SInt32 index,
    ICProfileID *thisID
);
```

Carbon Porting Notes

Use `ICGetIndProfile` (page 41) instead.

Declared In

`InternetConfig.h`

ICCGetMapEntry

Unsupported

```
OSStatus ICCGetMapEntry (
    ComponentInstance inst,
    Handle entries,
    SInt32 pos,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICGetMapEntry` (page 41) instead.

Declared In

`InternetConfig.h`

ICCGetMappingInterruptSafe

Unsupported

```
OSStatus ICCGetMappingInterruptSafe (
    ComponentInstance inst,
    Ptr *mappingPref,
    SInt32 *mappingPrefSize
);
```

Carbon Porting Notes

`ICCGetMappingInterruptSafe` is not compatible with preemptive threads. If your application relies on this API, contact Apple Developer Technical Support.

Declared In

`InternetConfig.h`

ICCGetPerm

Unsupported

```
OSStatus ICCGetPerm (
    ComponentInstance inst,
    ICPerm *perm
);
```

Carbon Porting Notes

Use ICGetPerm (page 42) instead.

Declared In

InternetConfig.h

ICCGetPref

Unsupported

```
OSStatus ICCGetPref (
    ComponentInstance inst,
    ConstStr255Param key,
    ICAttr *attr,
    Ptr buf,
    SInt32 *size
);
```

Carbon Porting Notes

Use ICGetPref (page 42) instead.

Declared In

InternetConfig.h

ICCGetPrefHandle

Unsupported

```
OSStatus ICCGetPrefHandle (
    ComponentInstance inst,
    ConstStr255Param key,
    ICAttr *attr,
    Handle *prefh
);
```

Carbon Porting Notes

Use ICFindPrefHandle (page 37) instead.

Declared In

InternetConfig.h

ICCGetProfileName

Unsupported

```
OSStatus ICCGetProfileName (
    ComponentInstance inst,
    ICProfileID thisID,
    Str255 name
);
```

Carbon Porting Notes

Use `ICGetProfileName` (page 43) instead.

Declared In

`InternetConfig.h`

ICCGetSeed

Unsupported

```
OSStatus ICCGetSeed (
    ComponentInstance inst,
    SInt32 *seed
);
```

Carbon Porting Notes

Use `ICGetSeed` (page 43) instead.

Declared In

`InternetConfig.h`

ICCGetSeedInterruptSafe

Unsupported

```
OSStatus ICCGetSeedInterruptSafe (
    ComponentInstance inst,
    SInt32 *seed
);
```

Carbon Porting Notes

`ICCGetSeedInterruptSafe` is not compatible with preemptive threads. If your application relies on this API, contact Apple Developer Technical Support.

Declared In

`InternetConfig.h`

ICCGetVersion

Unsupported

```
OSStatus ICCGetVersion (
    ComponentInstance inst,
    SInt32 whichVersion,
    UInt32 *version
);
```

Carbon Porting Notes

Use `ICGetVersion` (page 44) instead.

Declared In

`InternetConfig.h`

ICChooseConfig

Unsupported

```
OSStatus ICChooseConfig (
    ICInstance inst
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

`InternetConfig.h`

ICChooseNewConfig

Unsupported

```
OSStatus ICChooseNewConfig (
    ICInstance inst
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

`InternetConfig.h`

ICCLaunchURL

Unsupported

```
OSStatus ICCLaunchURL (
    ComponentInstance inst,
    ConstStr255Param hint,
    Ptr data,
    SInt32 len,
    SInt32 *selStart,
    SInt32 *selEnd
);
```

Carbon Porting Notes

Use `ICLaunchURL` (page 44) instead.

Declared In

`InternetConfig.h`

ICCMapEntriesFilename

Unsupported

```
OSStatus ICCMapEntriesFilename (
    ComponentInstance inst,
    Handle entries,
    ConstStr255Param filename,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICMapEntriesFilename` (page 45) instead.

Declared In

`InternetConfig.h`

ICCMapEntriesTypeCreator

Unsupported

```
OSStatus ICCMapEntriesTypeCreator (
    ComponentInstance inst,
    Handle entries,
    OSType fType,
    OSType fCreator,
    ConstStr255Param filename,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICMapEntriesTypeCreator` (page 45) instead.

Declared In

`InternetConfig.h`

ICCMapFilename

Unsupported

```
OSStatus ICCMapFilename (
    ComponentInstance inst,
    ConstStr255Param filename,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICMapFilename` (page 46) instead.

Declared In

`InternetConfig.h`

ICCMapTypeCreator

Unsupported

```
OSStatus ICCMapTypeCreator (
    ComponentInstance inst,
    OSType fType,
    OSType fCreator,
    ConstStr255Param filename,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICMapTypeCreator` (page 46) instead.

Declared In

`InternetConfig.h`

ICCountMapEntries

Counts the number of entries in the mappings database preference provided.

```
OSStatus ICCountMapEntries (
   ICInstance inst,
   Handle entries,
   long *count
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICCountPref

Returns the total number of preferences available.

```
OSStatus ICCountPref (
   ICInstance inst,
   long *count
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICCountProfiles

Returns the number of available profiles.

```
OSStatus ICCountProfiles (
   ICInstance inst,
   long *count
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICCParseURL

Unsupported

```
OSStatus ICCParseURL (
    ComponentInstance inst,
    ConstStr255Param hint,
    Ptr data,
    SInt32 len,
    SInt32 *selStart,
    SInt32 *selEnd,
    Handle url
);
```

Carbon Porting Notes

Use `ICParseURL` (page 47) instead.

Declared In

`InternetConfig.h`

ICCreateGURLEvent

Creates a GURL Apple event, targetted at the application with the specified creator code

```
OSStatus ICCreateGURLEvent (
   ICInstance inst,
   OSType helperCreator,
   Handle urlH,
   AppleEvent *theEvent
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICCRefreshCaches

Unsupported

```
OSStatus ICCRefreshCaches (
    ComponentInstance inst
);
```

Carbon Porting Notes

This function is obsolete, because there is no supported way to modify the Internet Config database without going through the Internet Config API. There is no replacement function.

Declared In

`InternetConfig.h`

ICCRequiresInterruptSafe

Unsupported

```
OSStatus ICCRequiresInterruptSafe (
    ComponentInstance inst
);
```

Carbon Porting Notes

This function is not compatible with preemptive threads. There is no replacement.

Declared In

`InternetConfig.h`

ICCSendGURLEvent

Unsupported

```
OSStatus ICCSendGURLEvent (
    ComponentInstance inst,
    AppleEvent *theEvent
);
```

Carbon Porting Notes

Use `ICSendGURLEvent` (page 48) instead.

Declared In

`InternetConfig.h`

ICCSetConfigReference

Unsupported

```
OSStatus ICCSetConfigReference (
    ComponentInstance inst,
    ICConfigRefHandle ref,
    SInt32 flags
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

`InternetConfig.h`

ICCSetCurrentProfile

Unsupported

```
OSStatus ICCSetCurrentProfile (
    ComponentInstance inst,
    ICProfileID newID
);
```

Carbon Porting Notes

Use `ICSetCurrentProfile` (page 48) instead.

Declared In

`InternetConfig.h`

ICCSetMapEntry

Unsupported

```
OSStatus ICCSetMapEntry (
    ComponentInstance inst,
    Handle entries,
    SInt32 pos,
    ICMapEntry *entry
);
```

Carbon Porting Notes

Use `ICSetMapEntry` (page 49)`ICSetMapEntry` instead.

Declared In

`InternetConfig.h`

ICCSetPref

Unsupported

```
OSStatus ICCSetPref (
    ComponentInstance inst,
    ConstStr255Param key,
    ICAttr attr,
    Ptr buf,
    SInt32 size
);
```

Carbon Porting Notes

Use `ICSetPref` (page 49) instead.

Declared In

`InternetConfig.h`

ICCSetPrefHandle

Unsupported

```
OSStatus ICCSetPrefHandle (
    ComponentInstance inst,
    ConstStr255Param key,
    ICAttr attr,
    Handle prefh
);
```

Carbon Porting Notes

Use `ICSetPrefHandle` (page 50) instead.

Declared In

`InternetConfig.h`

ICCSetProfileName

Unsupported

```
OSStatus ICCSetProfileName (
    ComponentInstance inst,
    ICProfileID thisID,
    ConstStr255Param name
);
```

Carbon Porting Notes

Use `ICSetProfileName` (page 50) instead.

Declared In

`InternetConfig.h`

ICCSpecifyConfigFile

Unsupported

```
OSStatus ICCSpecifyConfigFile (
    ComponentInstance inst,
    FSSpec *config
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICCStart

Unsupported

```
OSStatus ICCStart (
    ComponentInstance *inst,
    OSType creator
);
```

Carbon Porting Notes

Use `ICStart` (page 51) instead.

Declared In

`InternetConfig.h`

ICCStop

Unsupported

```
OSStatus ICCStop (
    ComponentInstance inst
);
```

Carbon Porting Notes

Use `ICStop` (page 51) instead.

Declared In

`InternetConfig.h`

ICDefaultFileName

Unsupported

```
OSStatus ICDefaultFileName (
    ICInstance inst,
    Str63 name
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICDeleteMapEntry

Deletes an entry in the specified Mapping preference.

```
OSStatus ICDeleteMapEntry (
   ICInstance inst,
   Handle entries,
   long pos
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICDeletePref

Deletes the preference for the specified key.

```
OSStatus ICDeletePref (
   ICInstance inst,
   ConstStr255Param key
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICDeleteProfile

Deletes the profile that has the specified ID.

```
OSStatus ICDeleteProfile (
   ICInstance inst,
   ICProfileID thisID
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICEditPreferences

Opens the Internet pane of System Preferences.

```
OSStatus ICEditPreferences (
   ICInstance inst,
   ConstStr255Param key
);
```

Return Value
A result code. See "Result Codes" (page 80).

Discussion
This function launches the Internet preference control appropriate to the operating system version. In Mac OS X, it opens the Internet pane of System Preferences. In Mac OS 8 and 9 wiith Carbon, it launches the Internet control panel. On earlier systems, it launches Internet Config.

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICEnd

Tells Internet Config that you have finished accessing preference information.

```
OSStatus ICEnd (
   ICInstance inst
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICFindConfigFile

Unsupported

```
OSStatus ICFindConfigFile (
    ICInstance inst,
    SInt16 count,
    ICDirSpecArrayPtr folders
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICFindPrefHandle

Gets a preference's data, like `ICGetPref`, but returns the data in a handle.

```
OSStatus ICFindPrefHandle (
   ICInstance inst,
   ConstStr255Param key,
   ICAttr *attr,
   Handle prefh
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICFindUserConfigFile

Unsupported

```
OSStatus ICFindUserConfigFile (
    ICInstance inst,
    ICDirSpec *where
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICGeneralFindConfigFile

Unsupported

```
OSStatus ICGeneralFindConfigFile (
    ICInstance inst,
    Boolean searchPrefs,
    Boolean canCreate,
    SInt16 count,
    ICDirSpecArrayPtr folders
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICGetComponentInstance

Unsupported

```
OSStatus ICGetComponentInstance (
    ICInstance inst,
    ComponentInstance *componentInst
);
```

Carbon Porting Notes

Because Internet Config is not component-based under X, use `ICGetVersion` instead of `ICGetComponentInstance`.

Declared In

`InternetConfig.h`

ICGetConfigName

Returns a displayable string that represents the instance’s current configuration.

```
OSStatus ICGetConfigName (
   ICInstance inst,
   Boolean longname,
   Str255 name
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetConfigReference

Unsupported

```
OSStatus ICGetConfigReference (
    ICInstance inst,
    ICConfigRefHandle ref
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

`InternetConfig.h`

ICGetCurrentProfile

Returns the profile ID of the current profile.

```
OSStatus ICGetCurrentProfile (
   ICInstance inst,
   ICProfileID *currentID
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetDefaultPref

Gets the default value for the preference associated with the specified key.

```
OSStatus ICGetDefaultPref (
   ICInstance inst,
   ConstStr255Param key,
   Handle prefH
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetIndMapEntry

Returns the mappings database entry for a specified index .

```
OSStatus ICGetIndMapEntry (
   ICInstance inst,
   Handle entries,
   long index,
   long *pos,
   ICMapEntry *entry
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetIndPref

Returns the key associated with the preference at the specified index.

```
OSStatus ICGetIndPref (
   ICInstance inst,
   long index,
   Str255 key
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICGetIndProfile

Returns the ID of the profile at the specified index.

```
OSStatus ICGetIndProfile (
   ICInstance inst,
   long index,
   ICProfileID *thisID
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICGetMapEntry

Returns a mappings database entry based on its position in the Mapping preference.

```
OSStatus ICGetMapEntry (
   ICInstance inst,
   Handle entries,
   long pos,
   ICMapEntry *entry
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICGetMappingInterruptSafe

Unsupported

```
OSStatus ICGetMappingInterruptSafe (
    ICInstance inst,
    Ptr *mappingPref,
    SInt32 *mappingPrefSize
);
```

Carbon Porting Notes

Because `ICGetMappingInterruptSafe` is not compatible with preemptive threads, it has been removed from Carbon.

Declared In

`InternetConfig.h`

ICGetPerm

Returns the current permissions for this instance, i.e. the permission value you used when you called `ICBegin` , or returns `ioNoPerm` if you haven't call `ICBegin`.

```
OSStatus ICGetPerm (
   ICInstance inst,
   ICPerm *perm
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetPref

Given a preference's key, gets the preference's data and places it into a buffer that you supply.

```
OSStatus ICGetPref (
   ICInstance inst,
   ConstStr255Param key,
   ICAttr *attr,
   void *buf,
   long *size
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetPrefHandle

Deprecated. While this function works in Mac OS X, you should use ICFindPrefHandle instead.

```
OSStatus ICGetPrefHandle (
   ICInstance inst,
   ConstStr255Param key,
   ICAttr *attr,
   Handle *prefh
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetProfileName

Returns the name of the profile that has the specified ID.

```
OSStatus ICGetProfileName (
   ICInstance inst,
   ICProfileID thisID,
   Str255 name
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICGetSeed

Returns the seed for the current preferences set.

```
OSStatus ICGetSeed (
   ICInstance inst,
   long *seed
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICGetSeedInterruptSafe

Unsupported

```
OSStatus ICGetSeedInterruptSafe (
    ICInstance inst,
    SInt32 *seed
);
```

Carbon Porting Notes

Because `ICGetSeedInterruptSafe` is not compatible with preemptive threads, it has been removed from Carbon.

Declared In
`InternetConfig.h`

ICGetVersion

Returns the version of Internet Config installed on the system.

```
OSStatus ICGetVersion (
   ICInstance inst,
   long whichVersion,
   UInt32 *version
);
```

Return Value
A result code. See "Result Codes" (page 80).

Discussion
For the range of versions, refer to "Version Constants" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICLaunchURL

Parses a URL out of the specified text and feeds it to the appropriate helper application.

```
OSStatus ICLaunchURL (
   ICInstance inst,
   ConstStr255Param hint,
   const void *data,
   long len,
   long *selStart,
   long *selEnd
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICMapEntriesFilename

Given a filename, returns the most appropriate `ICMapEntry` based on the filename extension—without the overhead of a mappings database access.

```
OSStatus ICMapEntriesFilename (
   ICInstance inst,
   Handle entries,
   ConstStr255Param filename,
   ICMapEntry *entry
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICMapEntriesTypeCreator

Given a file type and creator (and optionally a filename), returns the most appropriate `ICMapEntry`—without the overhead of a mappings database access.

```
OSStatus ICMapEntriesTypeCreator (
   ICInstance inst,
   Handle entries,
   OSType fType,
   OSType fCreator,
   ConstStr255Param filename,
   ICMapEntry *entry
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICMapFilename

Given a filename, returns the most appropriate `ICMapEntry` based on the filename extension.

```
OSStatus ICMapFilename (
   ICInstance inst,
   ConstStr255Param filename,
   ICMapEntry *entry
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICMapTypeCreator

Given a file type and creator (and optionally a filename), returns the most appropriate `ICMapEntry`.

```
OSStatus ICMapTypeCreator (
   ICInstance inst,
   OSType fType,
   OSType fCreator,
   ConstStr255Param filename,
   ICMapEntry *entry
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICParseURL

Parses a URL out of the specified text and returns it to the calling program.

```
OSStatus ICParseURL (
   ICInstance inst,
   ConstStr255Param hint,
   const void *data,
   long len,
   long *selStart,
   long *selEnd,
   Handle url
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICRefreshCaches

Unsupported

```
OSStatus ICRefreshCaches (
    ICInstance inst
);
```

Carbon Porting Notes

This function `ICRefreshCaches` is obsolete. There is no supported way to modify the Internet Config database without going through the Internet Config API. There is no replacement function.

Declared In
`InternetConfig.h`

ICRequiresInterruptSafe

Unsupported

```
OSStatus ICRequiresInterruptSafe (
    ICInstance inst
);
```

Carbon Porting Notes

Because ICRequiresInterruptSafe is not compatible with preemptive threads, it has been removed from Carbon.

Declared In

InternetConfig.h

ICSendGURLEvent

Sends a specified Apple Event to the target application.

```
OSStatus ICSendGURLEvent (
   ICInstance inst,
   AppleEvent *theEvent
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

InternetConfig.h

ICSetConfigReference

Unsupported

```
OSStatus ICSetConfigReference (
    ICInstance inst,
    ICConfigRefHandle ref,
    SInt32 flags
);
```

Carbon Porting Notes

Because Internet Config no longer supports multiple configuration files, this function is obsolete, and there is no replacement function.

Declared In

InternetConfig.h

ICSetCurrentProfile

Sets the current profile to a specified profile ID.

```
OSStatus ICSetCurrentProfile (
   ICInstance inst,
   ICProfileID newID
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICSetMapEntry

Sets an entry in a specified Mapping preference.

```
OSStatus ICSetMapEntry (
   ICInstance inst,
   Handle entries,
   long pos,
   const ICMapEntry *entry
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In
`InternetConfig.h`

ICSetPref

Sets a preference given its key, attributes, and a buffer containing the preference data.

```
OSStatus ICSetPref (
   ICInstance inst,
   ConstStr255Param key,
   ICAttr attr,
   const void *buf,
   long size
);
```

Return Value
A result code. See "Result Codes" (page 80).

Availability
Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICSetPrefHandle

Sets a preference, like `ICSetPref`, but takes its input as a handle.

```
OSStatus ICSetPrefHandle (
   ICInstance inst,
   ConstStr255Param key,
   ICAttr attr,
   Handle prefh
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICSetProfileName

Sets the name of the profile that has the specified ID.

```
OSStatus ICSetProfileName (
   ICInstance inst,
   ICProfileID thisID,
   ConstStr255Param name
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICSpecifyConfigFile

Unsupported

```
OSStatus ICSpecifyConfigFile (
    ICInstance inst,
    FSSpec *config
);
```

Carbon Porting Notes

Functions related to finding different IC database files are obsolete. You can simply remove these calls from your code—there are no replacement functions.

Declared In

`InternetConfig.h`

ICStart

Call this routine to start using Internet Config—typically at application initialization time—passing it your program's creator code.

```
OSStatus ICStart (
   ICInstance *inst,
   OSType signature
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

ICStop

Call this when your application is done using Internet Config, passing it the instance you got from `ICStart`
.

```
OSStatus ICStop (
   ICInstance inst
);
```

Return Value

A result code. See "Result Codes" (page 80).

Availability

Available in CarbonLib 1.0.2 and later when Internet Config 2.0 or later is present.

Available in Mac OS X 10.0 and later.

Declared In

`InternetConfig.h`

Data Types

ICAppSpec

A fixed length record used to specify an application.

```
struct ICAppSpec {
    OSType fCreator;
    Str63 name;
};
typedef struct ICAppSpec ICAppSpec;
typedef ICAppSpec * ICAppSpecPtr;
typedef ICAppSpecPtr * ICAppSpecHandle;
```

Discussion

The program using this specification is expected to look up the location of the application in the desktop database. The name is provided to display to the user, and should not affect the search.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`InternetConfig.h`

ICAppSpecList

Represents a list of application specifications.

```
struct ICAppSpecList {
    short numberOfItems;
    ICAppSpec appSpecs[1];
};
typedef struct ICAppSpecList ICAppSpecList;
typedef ICAppSpecList * ICAppSpecListPtr;
typedef ICAppSpecListPtr * ICAppSpecListHandle;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`InternetConfig.h`

ICCharTable

Specifies a mapping from MacRoman encoding (kTextEncodingMacRoman) to Mac Net ASCII (kTextEncodingMacRomanLatin1) and vice versa.

```
struct ICCharTable {
    unsigned char netToMac[256];
    unsigned char macToNet[256];
};
typedef struct ICCharTable ICCharTable;
typedef ICCharTable * ICCharTablePtr;
typedef ICCharTablePtr * ICCharTableHandle;
```

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

Declared In
`InternetConfig.h`

ICConfigRef

Stores a permanent reference to an Internet Config configuration.

```
struct ICConfigRef {
    OSType manufacturer;
};
typedef struct ICConfigRef ICConfigRef;
typedef ICConfigRef * ICConfigRefPtr;
typedef ICConfigRefPtr * ICConfigRefHandle;
```

Discussion
The `ICConfigRef` type varies in length and only the first four bytes, representing the manufacturer field, has a public meaning. For more information see the IC Internals Documentation.

Availability
Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In
`InternetConfig.h`

ICDirSpec

Holds the `vRefNum` and `dirID` of a directory.

```
struct ICDirSpec {
    short vRefNum;
    long dirID;
};
typedef struct ICDirSpec ICDirSpec;
typedef ICDirSpec ICDirSpecArray[4];
typedef ICDirSpecArray * ICDirSpecArrayPtr;
```

Discussion
An array of `ICDirSpec` records is supplied to the pre-Carbon-only function `ICFindConfigFile` to specify the search path. This array is defined to contain just 4 elements, but is in fact arbitrarily extensible.

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

Not available to 64-bit applications.

Declared In
`InternetConfig.h`

ICError

Used for all error results from Internet Config.

```
typedef long ICError;
```

Discussion
ICError uses a long integer because Internet Config makes calls to Component Manager, which uses long integers for error codes.

Special Considerations

This data type is available only if you define `OLDROUTINENAMES`.

Availability
Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In
`InternetConfig.h`

ICFileInfo

Defines the format of a key.

```
struct ICFileInfo {
    OSType fType;
    OSType fCreator;
    Str63 name;
};
typedef struct ICFileInfo ICFileInfo;
typedef ICFileInfo * ICFileInfoPtr;
typedef ICFileInfoPtr * ICFileInfoHandle;
```

Special Considerations

This data type defines the format of a key. That key data type has previouisly been removed from the header file. `ICFileInfo` is deprecated and will also be removed, but for the moment, it is available if you define `OLDROUTINENAMES`.

Availability
Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In
`InternetConfig.h`

ICFileSpec

A variable length data structure used to specify a file or folder.

```
struct ICFileSpec {
    Str31 volName;
    long volCreationDate;
    FSSpec fss;
    AliasRecord alias;
};
typedef struct ICFileSpec ICFileSpec;
typedef ICFileSpec * ICFileSpecPtr;
typedef ICFileSpecPtr * ICFileSpecHandle;
```

Discussion
`ICFileSpec` contains both an alias and a 'poor man's alias.' All modern applications can use the real alias and ignore the poor man's alias. The latter was included so that programs running under System 6 could specify file positions using `vol_name` and `vol_creation_date` for the volume, `fss.parID` for the directory on the volume, and `fss.name` for the file in the directory.

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

Declared In
`InternetConfig.h`

ICFontRecord

A fixed length record used to specify a font, size and face.

```
struct ICFontRecord {
    short size;
    Style face;
    char pad;
    Str255 font;
};
typedef struct ICFontRecord ICFontRecord;
typedef ICFontRecord * ICFontRecordPtr;
typedef ICFontRecordPtr * ICFontRecordHandle;
```

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

Declared In
`InternetConfig.h`

ICInstance

An opaque type used to hold a reference to a session with Internet Config.

```
typedef struct OpaqueICInstance * ICInstance;
```

Discussion
Applications can create IC instances by calling `ICStart`, use them with any of the API routines, and destroy them by calling `ICStop`.

An ICInstance is a pointer, so you can use the system nil value to denote an invalid instance. Do not pass an ICInstance between processes. Also, do not pass an ICInstance between instruction set architectures, i.e. from PowerPC to 68K or vice versa.

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

Declared In
InternetConfig.h

ICMapEntry

Specifies an Internet Config map entry.

```
struct ICMapEntry {
    short totalLength;
    ICFixedLength fixedLength;
    short version;
    OSType fileType;
    OSType fileCreator;
    OSType postCreator;
    ICMapEntryFlags flags;
    Str255 extension;
    Str255 creatorAppName;
    Str255 postAppName;
    Str255 MIMEType;
    Str255 entryName;
};
typedef struct ICMapEntry ICMapEntry;
typedef ICMapEntry * ICMapEntryPtr;
typedef ICMapEntryPtr * ICMapEntryHandle;
```

Fields

totalLength

The length of the map entry, in bytes, from beginning of the record. This includes the fixed length, the length of the packed Pascal strings at the end of the entry, and the length of the user data that follow those strings.

fixedLength

The length of the fixed part of the map entry, in bytes, from beginning of the record.

version

The version number of the map entry. The only version currently defined is 0.

fileType

The four-character file type for the map entry.

fileCreator

The four-character creator code for the map entry.

postCreator

The creator code for the post-processing application for the map entry. Applications should consult this field only if the ICmap_post_bit is set in the flags field, but even if that bit is not set, applications can determine the post-processing application by looking at this field. If no post processing application has even been set, this field's value will be 0.

`flags`
: Flags associated with this map entry. For flag definitions, see "Map Constants" (page 74), "Map Entry Flags" (page 75), and "Map Entry Masks" (page 76).

`extension`
: The filename extension for this map entry.

`creatorAppName`
: The name of the creator application identified by the `fileCreator` field. You can use this field to display the application name even if the application is not installed.

`postAppName`
: The name of the post-processing application identified by the `postCreator` field. You can use this field to display the post-processing application's name even if the that application is not installed. See the `post_creator` field description for details as to when this field is valid.

`MIMEType`
: The MIME type for the map entry, for example, "application/zip".

`entryName`
: The user-visible name for the map entry.

Discussion

The value of the Mappings preference is not an array, so you cannot index it directly. It is instead a packed array of `ICMapEntry` types, with each entry packed to remove the empty space at the end of the strings. Entries can start on an odd address and entries can contain user data . We strongly recommend that you access this data structure using the routines described in this section. These routines return an unpacked `ICMapEntry`, which is a lot easier to deal with.

Each `ICMapEntry` data type provides a filename extension, a MIME type, and a file type and creator. The database of `ICMapEntry` types is not normalised; that is, there can be multiple entries with the same creator, file type, MIME type, filename extension, and so on. In general, each application determines how these entries are used in a particular circumstance, although IC does define some policies by way of its high level mapping routines.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`InternetConfig.h`

ICServiceEntry

Specifies an Internet Config TCP service entry.

```
struct ICServiceEntry {
    Str255 name;
    short port;
    ICServiceEntryFlags flags;
};
typedef struct ICServiceEntry ICServiceEntry;
typedef ICServiceEntry * ICServiceEntryPtr;
typedef ICServiceEntryPtr * ICServiceEntryHandle;
```

Fields

`name`
: The name for the TCP service entry.

`port`

The port for the TCP service entry.

`flags`

For descriptions of the flags, see "Services Constants" (page 77), "Services Bits" (page 79), and "Services Masks" (page 79).

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

Declared In
`InternetConfig.h`

ICServices

Specifies the mapping between TCP service names and their ports.

```
struct ICServices {
    short count;
    ICServiceEntry services[1];
};
typedef struct ICServices ICServices;
typedef ICServices * ICServicesPtr;
typedef ICServicesPtr * ICServicesHandle;
```

Fields

`count`

`services`

An unbounded array of `ICServiceEntry` (page 57)records.

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

Declared In
`InternetConfig.h`

internetConfigurationComponent

```
typedef ComponentInstance internetConfigurationComponent;
```

Special Considerations

This data type is obsolete. Use the data type `ICInstance` (page 55) instead.

Availability
Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In
`InternetConfig.h`

Constants

Apple Event Constants

```
enum {
    kInternetEventClass = 'GURL',
    kAEGetURL = 'GURL',
    kAEFetchURL = 'FURL',
    keyAEAttaching = 'Atch'
};
```

Attribute Constants

These are old names, mapped to newer names for backward compatibility, available only if you define OLDROUTINENAMES.

```
enum {
    ICattr_no_change = (unsigned long) (kICAttrNoChange),
    ICattr_locked_bit = kICAttrLockedBit,
    ICattr_locked_mask = kICAttrLockedMask,
    ICattr_volatile_bit = kICAttrVolatileBit,
    ICattr_volatile_mask = kICAttrVolatileMask,
    icNoUserInteraction_bit = kICNoUserInteractionBit,
    icNoUserInteraction_mask = kICNoUserInteractionMask,
    ICfiletype = kICFileType,
    ICcreator = kICCreator
};
```

Component Identifiers

Define the component type, subtype, and manufacturer of the Internet Config component.

```
enum {
    kICComponentType = 'PREF',
    kICComponentSubType = 'ICAp',
    kICComponentManufacturer = 'JPQE'
};
```

Component Identifiers (Deprecated)

```
enum {
    internetConfigurationComponentType = 'PREF',
    internetConfigurationComponentSubType = 'ICAp',
    internetConfigurationComponentInterfaceVersion0 = 0x00000000,
    internetConfigurationComponentInterfaceVersion1 = 0x00010000,
    internetConfigurationComponentInterfaceVersion2 = 0x00020000,
    internetConfigurationComponentInterfaceVersion3 = 0x00030000,
    internetConfigurationComponentInterfaceVersion =
internetConfigurationComponentInterfaceVersion3
};
```

Discussion

These constants are deprecated. Please use `Component Interface Version` (page 60) and `Component Identifiers` (page 59) instead.

Component Interface Version

Define the possible version numbers returned by the Internet Config component.

```
enum {
    kICComponentInterfaceVersion0 = 0x00000000,
    kICComponentInterfaceVersion1 = 0x00010000,
    kICComponentInterfaceVersion2 = 0x00020000,
    kICComponentInterfaceVersion3 = 0x00030000,
    kICComponentInterfaceVersion4 = 0x00040000,
    kICComponentInterfaceVersion = kICComponentInterfaceVersion4
};
```

Constants

`kICComponentInterfaceVersion0`

Internet Config versions greater than or equal to 1.0.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICComponentInterfaceVersion1`

Internet Config versions greater than or equal to 1.1.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICComponentInterfaceVersion2`

Internet Config versions greater than or equal to 1.2.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICComponentInterfaceVersion3`

Internet Config versions greater than or equal to 2.0.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICComponentInterfaceVersion4`

Internet Config versions greater than or equal to 2.5.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICComponentInterfaceVersion`

The current version number is 4.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Discussion

These constants define the possible version numbers returned by the Internet Config component in response to a `ICGetVersion` (page 44) call. The constants define only the high word of the version number, which is the version of the programming interface. The low word of the version number is the implementation version, which changes with each bug fix release of Internet Config.

Component Selectors

Represent component selectors for each Internet Config function.

```
enum {
    kICCStart = 0,
    kICCStop = 1,
    kICCGetVersion = 50,
    kICCFindConfigFile = 2,
    kICCFindUserConfigFile = 14,
    kICCGeneralFindConfigFile = 30,
    kICCChooseConfig = 33,
    kICCChooseNewConfig = 34,
    kICCGetConfigName = 35,
    kICCGetConfigReference = 31,
    kICCSetConfigReference = 32,
    kICCSpecifyConfigFile = 3,
    kICCRefreshCaches = 47,
    kICCGetSeed = 4,
    kICCGetPerm = 13,
    kICCDefaultFileName = 11,
    kICCBegin = 5,
    kICCGetPref = 6,
    kICCSetPref = 7,
    kICCFindPrefHandle = 36,
    kICCGetPrefHandle = 26,
    kICCSetPrefHandle = 27,
    kICCCountPref = 8,
    kICCGetIndPref = 9,
    kICCDeletePref = 12,
    kICCEnd = 10,
    kICCGetDefaultPref = 49,
    kICCEditPreferences = 15,
    kICCLaunchURL = 17,
    kICCParseURL = 16,
    kICCCreateGURLEvent = 51,
    kICCSendGURLEvent = 52,
    kICCMapFilename = 24,
    kICCMapTypeCreator = 25,
    kICCMapEntriesFilename = 28,
    kICCMapEntriesTypeCreator = 29,
    kICCCountMapEntries = 18,
    kICCGetIndMapEntry = 19,
    kICCGetMapEntry = 20,
    kICCSetMapEntry = 21,
    kICCDeleteMapEntry = 22,
    kICCAddMapEntry = 23,
    kICCGetCurrentProfile = 37,
    kICCSetCurrentProfile = 38,
    kICCCountProfiles = 39,
    kICCGetIndProfile = 40,
    kICCGetProfileName = 41,
    kICCSetProfileName = 42,
    kICCAddProfile = 43,
    kICCDeleteProfile = 44,
    kICCRequiresInterruptSafe = 45,
    kICCGetMappingInterruptSafe = 46,
    kICCGetSeedInterruptSafe = 48,
    kICCFirstSelector = kICCStart,
    kICCLastSelector = 52
};
```

Discussion

These constants are component selectors for each Internet Config fuction. You can use these values as parameters to the Component Manager's `ComponentFunctionImplemented` function. These are also useful for authors of override components.

Component Selector Proc Information

```
enum {
    kICCStartProcInfo = 1008,
    kICCStopProcInfo = 240,
    kICCGetVersionProcInfo = 4080,
    kICCFindConfigFileProcInfo = 3824,
    kICCFindUserConfigFileProcInfo = 1008,
    kICCGeneralFindConfigFileProcInfo = 58864,
    kICCChooseConfigProcInfo = 240,
    kICCChooseNewConfigProcInfo = 240,
    kICCGetConfigNameProcInfo = 3568,
    kICCGetConfigReferenceProcInfo = 1008,
    kICCSetConfigReferenceProcInfo = 4080,
    kICCSpecifyConfigFileProcInfo = 1008,
    kICCRefreshCachesProcInfo = 240,
    kICCGetSeedProcInfo = 1008,
    kICCGetPermProcInfo = 1008,
    kICCDefaultFileNameProcInfo = 1008,
    kICCGetComponentInstanceProcInfo = 1008,
    kICCBeginProcInfo = 496,
    kICCGetPrefProcInfo = 65520,
    kICCSetPrefProcInfo = 65520,
    kICCFindPrefHandleProcInfo = 16368,
    kICCGetPrefHandleProcInfo = 16368,
    kICCSetPrefHandleProcInfo = 16368,
    kICCCountPrefProcInfo = 1008,
    kICCGetIndPrefProcInfo = 4080,
    kICCDeletePrefProcInfo = 1008,
    kICCEndProcInfo = 240,
    kICCGetDefaultPrefProcInfo = 4080,
    kICCEditPreferencesProcInfo = 1008,
    kICCLaunchURLProcInfo = 262128,
    kICCParseURLProcInfo = 1048560,
    kICCCreateGURLEventProcInfo = 16368,
    kICCSendGURLEventProcInfo = 1008,
    kICCMapFilenameProcInfo = 4080,
    kICCMapTypeCreatorProcInfo = 65520,
    kICCMapEntriesFilenameProcInfo = 16368,
    kICCMapEntriesTypeCreatorProcInfo = 262128,
    kICCCountMapEntriesProcInfo = 4080,
    kICCGetIndMapEntryProcInfo = 65520,
    kICCGetMapEntryProcInfo = 16368,
    kICCSetMapEntryProcInfo = 16368,
    kICCDeleteMapEntryProcInfo = 4080,
    kICCAddMapEntryProcInfo = 4080,
    kICCGetCurrentProfileProcInfo = 1008,
    kICCSetCurrentProfileProcInfo = 1008,
    kICCCountProfilesProcInfo = 1008,
    kICCGetIndProfileProcInfo = 4080,
    kICCGetProfileNameProcInfo = 4080,
    kICCSetProfileNameProcInfo = 4080,
    kICCAddProfileProcInfo = 4080,
    kICCDeleteProfileProcInfo = 1008,
    kICCRequiresInterruptSafeProcInfo = 240,
    kICCGetMappingInterruptSafeProcInfo = 4080,
    kICCGetSeedInterruptSafeProcInfo = 1008
};
```

Edit Preference Constants

Specify AppleEvent details for Internet Config preference requests.

```
enum {
    kICEditPreferenceEventClass = 'ICAp',
    kICEditPreferenceEvent = 'ICAp',
    keyICEditPreferenceDestination = 'dest'
};
```

Discussion

These constants specify details of the AppleEvent that the ICEditPreferences (page 36) function sends to the Internet Config application to request it to open the edit window for a specific preference. Consult the Internet Config terminology resource ('aete') for the meaning of these constants.

File Specification Header Size

```
enum {
    kICFileSpecHeaderSize = sizeof(ICFileSpec) - sizeof(AliasRecord)
};
```

File Specification Header Size (Deprecated)

```
enum {
    ICfile_spec_header_size = kICFileSpecHeaderSize
};
```

Discussion

This constant is available only if you define OLDROUTINENAMES.

File Type Constants

Define the file type and creator code for the Internet preferences file.

```
enum {
    kICFileType = 'ICAp',
    kICCreator = 'ICAp'
};
```

Constants

kICFileType

The Finder file type for the Internet preferences file.

Available in Mac OS X v10.0 and later.

Declared in InternetConfig.h.

kICCreator

The Finder creator code for the Internet preferences file.

Available in Mac OS X v10.0 and later.

Declared in InternetConfig.h.

Keys

```
#define kICReservedKey                  "\pkICReservedKey"
#define kICArchieAll                    "\pArchieAll"
#define kICArchiePreferred              "\pArchiePreferred"
#define kICCharacterSet                 "\pCharacterSet"
#define kICDocumentFont                 "\pDocumentFont"
#define kICDownloadFolder               "\pDownloadFolder"
#define kICEmail                        "\pEmail"
#define kICFTPHost                      "\pFTPHost"
#define kICFTPProxyAccount              "\pFTPProxyAccount"
#define kICFTPProxyHost                 "\pFTPProxyHost"
#define kICFTPProxyPassword             "\pFTPProxyPassword"
#define kICFTPProxyUser                 "\pFTPProxyUser"
#define kICFingerHost                   "\pFingerHost"
#define kICGopherHost                   "\pGopherHost"
#define kICGopherProxy                  "\pGopherProxy"
#define kICHTTPProxyHost                "\pHTTPProxyHost"
#define kICHelper                       "\pHelper•"
#define kICHelperDesc                   "\pHelperDesc•"
#define kICHelperList                   "\pHelperList•"
#define kICIRCHost                      "\pIRCHost"
#define kICInfoMacAll                   "\pInfoMacAll"
#define kICInfoMacPreferred             "\pInfoMacPreferred"
#define kICLDAPSearchbase               "\pLDAPSearchbase"
#define kICLDAPServer                   "\pLDAPServer"
#define kICListFont                     "\pListFont"
#define kICMacSearchHost                "\pMacSearchHost"
#define kICMailAccount                  "\pMailAccount"
#define kICMailHeaders                  "\pMailHeaders"
#define kICMailPassword                 "\pMailPassword"
#define kICMapping                      "\pMapping"
#define kICNNTPHost                     "\pNNTPHost"
#define kICNTPHost                      "\pNTPHost"
#define kICNewMailDialog                "\pNewMailDialog"
#define kICNewMailFlashIcon             "\pNewMailFlashIcon"
#define kICNewMailPlaySound             "\pNewMailPlaySound"
#define kICNewMailSoundName             "\pNewMailSoundName"
#define kICNewsAuthPassword             "\pNewsAuthPassword"
#define kICNewsAuthUsername             "\pNewsAuthUsername"
#define kICNewsHeaders                  "\pNewsHeaders"
#define kICNoProxyDomains               "\pNoProxyDomains"
#define kICOrganization                 "\pOrganization"
#define kICPhHost                       "\pPhHost"
#define kICPlan                         "\pPlan"
#define kICPrinterFont                  "\pPrinterFont"
#define kICQuotingString                "\pQuotingString"
#define kICRealName                     "\pRealName"
#define kICRTSPProxyHost                "\pRTSPProxyHost"
#define kICSMTPHost                     "\pSMTPHost"
#define kICScreenFont                   "\pScreenFont"
#define kICServices                     "\pServices"
#define kICSignature                    "\pSignature"
#define kICSnailMailAddress             "\pSnailMailAddress"
#define kICSocksHost                    "\pSocksHost"
#define kICTelnetHost                   "\pTelnetHost"
#define kICUMichAll                     "\pUMichAll"
#define kICUMichPreferred               "\pUMichPreferred"
```

```
#define kICUseFTPProxy                  "\pUseFTPProxy"
#define kICUseGopherProxy               "\pUseGopherProxy"
#define kICUseHTTPProxy                 "\pUseHTTPProxy"
#define kICUsePassiveFTP                "\pUsePassiveFTP"
#define kICUseRTSPProxy                 "\pUseRTSPProxy"
#define kICUseSocks                     "\pUseSocks"
#define kICWAISGateway                  "\pWAISGateway"
#define kICWWWHomePage                  "\pWWWHomePage"
#define kICWebBackgroundColour          "\pWebBackgroundColour"
#define kICWebReadColor                 "\p646F6777•WebReadColor"
#define kICWebSearchPagePrefs           "\pWebSearchPagePrefs"
#define kICWebTextColor                 "\pWebTextColor"
#define kICWebUnderlineLinks            "\p646F6777•WebUnderlineLinks"
#define kICWebUnreadColor               "\p646F6777•WebUnreadColor"
#define kICWhoisHost                    "\pWhoisHost"
```

Constants

`kICReservedKey`

Reserved for use by Internet Config.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICArchieAll`

Formatted string list of Archie servers.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICArchiePreferred`

Formatted string naming the preferred Archie server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICCharacterSet`

For conversion between kTextEncodingMacRoman (“Mac”) and kTextEncodingMacRomanLatin1 (“Net”). Deprecated. Use the Text Encoding Conversion manager and Unicode instead.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICDocumentFont`

Font to use for proportional text.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICDownloadFolder`

Location in the file system for newly downloaded files.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICEmail`

Return address for the user.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICFTPHost`

Default FTP server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICFTPProxyAccount`

Second-level FTP proxy authorization. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICFTPProxyHost`

Default FTP proxy server. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICFTPProxyPassword`

Password (scrambled) for FTP proxy user. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICFTPProxyUser`

First-level FTP proxy authorization. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICFingerHost`

Default finger-protocol server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICGopherHost`

Default gopher-protocol server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICGopherProxy`

Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICHTTPProxyHost`

Default HTTP proxy server. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICHelper`

Helpers for URL schemes.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICHelperDesc`
: Descriptions for URL schemes.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICHelperList`
: Common helpers for URL schemes.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICIRCHost`
: Internet Relay Chat server.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICInfoMacAll`
: Formatted list of Info-Mac servers.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICInfoMacPreferred`
: Formatted string containing preferred Info-Mac server.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICLDAPSearchbase`
: Default LDAP search base.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICLDAPServer`
: Default LDAP server.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICListFont`
: Font used for lists, such as news article lists.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICMacSearchHost`
: Host for MacSearch queries.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICMailAccount`
: Account from which to fetch email.

  Available in Mac OS X v10.0 and later.

  Declared in `InternetConfig.h`.

`kICMailHeaders`
: Additional headers for email messages.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMailPassword`

Password (scrambled) for the default mail account.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapping`

File-type mapping.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNNTPHost`

Default NNTP server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNTPHost`

Default NTP (Network Time Protocol) server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNewMailDialog`

Boolean value indicating whether to display a new email dialog.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNewMailFlashIcon`

Boolean value indicating whether to flash a new email icon.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNewMailPlaySound`

Boolean value indicating whether to play a new email sound.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNewMailSoundName`

The name of the preferred new email sound.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNewsAuthPassword`

Password (scrambled) for authorized news account.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNewsAuthUsername`

User name for authorized news account.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNewsHeaders`

Additional headers for news messages.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICNoProxyDomains`

List of domains not to be proxied.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICOrganization`

String for the X-Organization MIME header.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICPhHost`

Default Ph protocol server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICPlan`

Default response for finger servers.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICPrinterFont`

Font used to print the monospaced text displayed on screen (see kICScreenFont).

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICQuotingString`

A short string used to indicate quoting in email and news replies.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICRealName`

Real name of user.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICRTSPProxyHost`

Default RTSP proxy server. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.1 and later.

Declared in `InternetConfig.h`.

`kICSMTPHost`

Default SMTP host.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICScreenFont`

Font for monospaced text.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICServices`

TCP and IP port-to-name mapping.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICSignature`

Block of text to append to outgoing news and email messages.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICSnailMailAddress`

Preferred postal mail address.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICSocksHost`

Default SOCKS host. (The host.domain format allows ":port" and " port")

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICTelnetHost`

Default Telnet host.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICUMichAll`

Formatted list of UMich servers.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICUMichPreferred`

Formatted string containing preferred UMich server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICUseFTPProxy`

Boolean value indicating whether to use an FTP proxy. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICUseGopherProxy`

Boolean value indicating whether to use a Gopher proxy. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICUseHTTPProxy`

Boolean value indicating whether to use an HTTP proxy. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICUsePassiveFTP`

Boolean value indicating whether to use the PASV command for FTP transfers.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICUseRTSPProxy`

Boolean value indicating whether to use an RTSP proxy. Read-only in Mac OS X; use System Configuration to set this value.

Available in Mac OS X v10.1 and later.

Declared in `InternetConfig.h`.

`kICUseSocks`

Boolean value indicating whether to use SOCKS.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWAISGateway`

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWWWHomePage`

User’s default Web home page.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWebBackgroundColour`

Background color for Web pages.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWebReadColor`

Color for Web links that have been visited.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWebSearchPagePrefs`

URL for the default search engine page.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWebTextColor`

Color for normal text on Web pages.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWebUnderlineLinks`

Boolean value indicating whether to underline Web links.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWebUnreadColor`

Color for Web links that have not been visited.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICWhoisHost`

Default whois server.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Map Constants

```
enum {
    ICmap_binary_bit = kICMapBinaryBit,
    ICmap_binary_mask = kICMapBinaryMask,
    ICmap_resource_fork_bit = kICMapResourceForkBit,
    ICmap_resource_fork_mask = kICMapResourceForkMask,
    ICmap_data_fork_bit = kICMapDataForkBit,
    ICmap_data_fork_mask = kICMapDataForkMask,
    ICmap_post_bit = kICMapPostBit,
    ICmap_post_mask = kICMapPostMask,
    ICmap_not_incoming_bit = kICMapNotIncomingBit,
    ICmap_not_incoming_mask = kICMapNotIncomingMask,
    ICmap_not_outgoing_bit = kICMapNotOutgoingBit,
    ICmap_not_outgoing_mask = kICMapNotOutgoingMask,
    ICmap_fixed_length = kICMapFixedLength
};
```

Discussion

For descriptions of these constants, see "Map Entry Flags" (page 75) and "Map Entry Masks" (page 76). These constants are available only if you define `OLDROUTINENAMES`.

Map Entry Flags

```
typedef long ICMapEntryFlags;
enum {
    kICMapBinaryBit = 0,
    kICMapResourceForkBit = 1,
    kICMapDataForkBit = 2,
    kICMapPostBit = 3,
    kICMapNotIncomingBit = 4,
    kICMapNotOutgoingBit = 5
};
```

Constants

`kICMapBinaryBit`

When this bit is set, indicates the file should be transferred in binary as opposed to text mode.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapResourceForkBit`

When this bit is set, indicates the resource fork of the file is significant.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapDataForkBit`

When this bit is set, indicates the data fork of the file is significant.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapPostBit`

When this bit is set, indicates post process using post fields.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapNotIncomingBit`

When this bit is set, indicates to ignore this mapping for incoming files.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapNotOutgoingBit`

When this bit is set, indicates to ignore this mapping for outgoing files.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Map Entry Masks

```
enum {
    kICMapBinaryMask = 0x00000001,
    kICMapResourceForkMask = 0x00000002,
    kICMapDataForkMask = 0x00000004,
    kICMapPostMask = 0x00000008,
    kICMapNotIncomingMask = 0x00000010,
    kICMapNotOutgoingMask = 0x00000020
};
```

Constants

`kICMapBinaryMask`

Indicates the file should be transferred in binary as opposed to text mode.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapResourceForkMask`

Indicates the resource fork of the file is significant.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapDataForkMask`

Indicates the data fork of the file is significant.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapPostMask`

Indicates to post process using post fields.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapNotIncomingMask`

Indicates to ignore this mapping for incoming files.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICMapNotOutgoingMask`

Indicates to ignore this mapping for outgoing files.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Map Fixed Length Constants

```
typedef short     ICFixedLength;
enum {
    kICMapFixedLength = 22
};
```

Constants

`kICMapFixedLength`

Use in the `fixedLength` field of a structure `ICMapEntry` (page 56).

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Services Constants

```
enum {
    ICservices_tcp_bit = kICServicesTCPBit,
    ICservices_tcp_mask = kICServicesTCPMask,
    ICservices_udp_bit = kICServicesUDPBit,
    ICservices_udp_mask = kICServicesUDPMask
};
```

Special Considerations

These constants are available only if you define `OLDROUTINENAMES`.

Permissions

The ICPerm type is used to specify whether you wish to access the preferences as read-only or read/write.

```
typedef UInt8 ICPerm;
enum {
    icNoPerm = 0,
    icReadOnlyPerm = 1,
    icReadWritePerm = 2
};
```

Preference Attribute Bits and Masks

```
typedef UInt32 ICAttr;
enum {
    kICAttrLockedBit = 0,
    kICAttrVolatileBit = 1
};
```

```
enum {
    kICAttrNoChange = 0xFFFFFFFF,
    kICAttrLockedMask = 0x00000001,
    kICAttrVolatileMask = 0x00000002
};
```

Constants

`kICAttrLockedBit`

If the locked bit is set, any attempt to set the preference will result in an error.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICAttrVolatileBit`

If the volatile bit is set, you should not cache the value of this preference because it is subject to non-seed changing changes.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICAttrNoChange`

Used when you call ICSetPref and do not want to mess around with attributes. You can supply this value and IC will not change the attribute of the preference.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICAttrLockedMask`

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICAttrVolatileMask`

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Discussion

The ICAttr type is simply a longint containing flags that describe the attributes of a key and its data.

Preference Attribute Masks

```
enum {
    kICAttrNoChange = 0xFFFFFFFF,
    kICAttrLockedMask = 0x00000001,
    kICAttrVolatileMask = 0x00000002
};
```

Profile IDs

The `ICProfileID` type is an opaque reference to a particular profile.

```
typedef long ICProfileID;
typedef ICProfileID * ICProfileIDPtr;
enum {
    kICNilProfileID = 0
};
```

Constants

`kICNilProfileID`

Use to denote no profile.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Services Bits

```
typedef short ICServiceEntryFlags;
enum {
    kICServicesTCPBit = 0,
    kICServicesUDPBit = 1
};
```

Constants

`kICServicesTCPBit`

When this bit is set, indicates the service is TCP.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICServicesUDPBit`

When this bit is set, indicates the service is UDP.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Discussion

Both these bits may be set to indicate the service is both TCP and UDP.

Services Masks

```
enum {
    kICServicesTCPMask = 0x00000001,
    kICServicesUDPMask = 0x00000002
};
```

Constants

`kICServicesTCPMask`

Indicates the service is TCP.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

`kICServicesUDPMask`

Indicates the service is UDP.

Available in Mac OS X v10.0 and later.

Declared in `InternetConfig.h`.

Discussion

Both these bits may be set to indicate the service is both TCP and UDP.

User Interaction Constants

```
enum {
    kICNoUserInteractionBit = 0
};
```

User Interaction Masks

```
enum {
    kICNoUserInteractionMask = 0x00000001
};
```

Version Constants

```
enum {
    kICComponentVersion = 0,
    kICNumVersion = 1
};
```

Result Codes

The most common result codes returned by Internet Config are listed in the table below.

Result Code	Value	Description
icPrefNotFoundErr	-666	The preference was not found. Available in Mac OS X v10.0 and later.
icPermErr	-667	The preference cannot be set because of premissions restrictions. Available in Mac OS X v10.0 and later.
icPrefDataErr	-668	There was a problem with the preference data. Available in Mac OS X v10.0 and later.
icInternalErr	-669	An internal error occurred. Available in Mac OS X v10.0 and later.
icTruncatedErr	-670	More data was present than was returned. Available in Mac OS X v10.0 and later.

Result Code	Value	Description
`icNoMoreWritersErr`	-671	You can't begin a write session because there is already one in progress. Available in Mac OS X v10.0 and later.
`icNothingToOverrideErr`	-672	There is no component for the override component to capture. Available in Mac OS X v10.0 and later.
`icNoURLErr`	-673	No URL was found. Available in Mac OS X v10.0 and later.
`icConfigNotFoundErr`	-674	No configuration was found. Available in Mac OS X v10.0 and later.
`icConfigInappropriateErr`	-675	The manufacturer code is incorrect. Available in Mac OS X v10.0 and later.
`icProfileNotFoundErr`	-676	The profile was not found. Available in Mac OS X v10.0 and later.
`icTooManyProfilesErr`	-677	There are too many profiles in the database. Available in Mac OS X v10.0 and later.

Document Revision History

This table describes the changes to *Internet Config Reference*.

Date	Notes
2003-05-01	Added documentation for functions and for most data types and constants. Added a cross-referenced appendix listing unsupported functions.
2002-12-01	Updated the EDD.
	Created a section for deprecated functions.
	Gave english names to constants and relocated related data types.
2001-07-01	Last version of this document.

Index