FontSync Reference

Carbon > Text & Fonts

2002-10-01

Contents

FontSync Reference

Framework: ApplicationServices/ApplicationServices.h

Declared in FontSync.h

Overview

FontSync is an API that provides a way for your application to identify fonts based upon the content of the font, rather than just the font name. Your application can use FontSync to compare fonts that are available on different computers. This document is relevant for anyone who is writing a text-intensive application that must minimize font mismatch errors when a file is moved from one computer to another. To use this document, you should understand the basics of fonts and be familiar with FontSync references and profiles.

Carbon supports FontSync. However, in Mac OS X, Apple recommends that you use FontSync only in OS X version 10.1 and later.

Functions by Task

Determining Availability, Version, and Feature Information

`FNSEnabled` (page 7)
: Indicates whether FontSync is enabled.

`FNSMatchDefaultsGet` (page 7)
: Determines the default match options used by FontSync functions performing font matching.

`FNSSysInfoGet` (page 35)
: Determines version and feature information for the version of FontSync installed on the user's system.

Providing User Interface Support

`FNSReferenceCountNames` (page 21)
: Determines the number of internal font names in a reference.

`FNSReferenceFindName` (page 24)
: Finds the first name that matches the given font name parameters, if any.

`FNSReferenceGetFamilyInfo` (page 27)
: Obtains information about a font family represented by a font reference.

`FNSReferenceGetIndName` (page 28)
: Finds the font name string and other font name parameters for an indexed font name.

Searching by Font Reference

`FNSProfileMatchReference` (page 15)
Obtains a list of the references in a profile that match a given reference.

`FNSReferenceMatchFamilies` (page 32)
Obtains a list of font families that match a reference.

`FNSReferenceMatchFonts` (page 33)
Obtains a list of font objects that match a reference.

Working With FontSync Profiles

`FNSProfileAddReference` (page 8)
Adds a font reference to a profile.

`FNSProfileClear` (page 9)
Removes all font references from a profile.

`FNSProfileClose` (page 10)
Closes the file associated with a font profile and disposes of run-time data.

`FNSProfileCompact` (page 10)
Compacts a font profile.

`FNSProfileCountReferences` (page 11)
Determines the number of font references in a font profile.

`FNSProfileCreate` (page 12)
Creates an empty FontSync profile using an `FSSpec`.

`FNSProfileCreateWithFSRef` (page 13)
Creates an empty FontSync profile using an `FSRef`.

`FNSProfileGetIndReference` (page 14)
Retrieves an indexed font reference from a profile.

`FNSProfileGetVersion` (page 15)
Retrieves the format version of an open font profile.

`FNSProfileOpen` (page 17)
Opens an existing font profile using an `FSSpec`.

`FNSProfileOpenWithFSRef` (page 18)
Opens an existing font profile using an `FSRef`.

`FNSProfileRemoveIndReference` (page 19)
Deletes an indexed font reference from a profile.

`FNSProfileRemoveReference` (page 20)
Deletes a font reference from a profile.

Working With FontSync References

`FNSReferenceCreate` (page 21)
Creates a font reference based on a font object.

Functions

FNSEnabled

Indicates whether FontSync is enabled.

```
Boolean FNSEnabled (
   void
);
```

Return Value
A `Boolean` value indicating whether FontSync is enabled. If `true`, your application can perform FontSync operations. See the Mac Types documentation for a description of the `Boolean` data type.

Discussion
You should check the flag returned by the `FNSEnabled` function before starting a sequence of FontSync calls, although it has no effect on the operation of the rest of the FontSync API.

Version Notes
Available beginning with FontSync 1.0. In FontSync 1.0, `FNSEnabled` always returns `true`.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSMatchDefaultsGet

Determines the default match options used by FontSync functions performing font matching.

```
FNSMatchOptions FNSMatchDefaultsGet (
   void
);
```

Return Value
A bit mask indicating the default match options. This is the value used when the mask constant `kFNSMatchDefaults` is passed to FontSync functions that perform matching. See the description of the `FNSMatchOptions` data type.

Discussion
The `FNSMatchDefaultsGet` function retrieves the bit mask used when the mask constant `kFNSMatchDefaults` is passed to the functions `FNSReferenceMatch` (page 31), `FNSProfileMatchReference` (page 15), `FNSReferenceMatchFonts` (page 33), and `FNSReferenceMatchFamilies` (page 32). The bit mask value is read from a preferences file which is created when the user sets match criteria via the control panel. The preference file is maintained by the FontSync library. If there is no preferences file, or it is unreadable, the implementation-defined fallback value `kFNSMatchAll` is returned.

There is no API for setting the default match criteria. Your application can specify options that are different from the user's preferences via the bitmask in the FontSync matching calls.

Version Notes
Available beginning with FontSync 1.0. In FontSync 1.0, the implementation-defined fallback value is `kFNSMatchAll`, that is, all defined options turned on. In other words, if the user does not set the match criteria via the control panel, FontSync uses the implementation-defined fallback value of all match criteria selected.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileAddReference

Adds a font reference to a profile.

```
OSStatus FNSProfileAddReference (
   FNSFontProfile iProfile,
   FNSFontReference iReference
);
```

Parameters

`iProfile`
A reference to the font profile to which you want to add a font reference. The profile must be writable.

`iReference`
A reference to the font reference you wish to add.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr`

indicates that a profile does not have a valid structure. The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSDuplicateReferenceErr` indicates that an identical reference already exists in the profile. In this case, the new one is not added. The File Manager error `permErr` indicates that the file is either locked and not editable or opened for read-only access. `FNSProfileAddReference` may return other File Manager errors. Memory Manager errors indicate that you did not have enough memory available in your heap.

Discussion

The `FNSProfileAddReference` function adds a font reference to a profile that has read/write access. If an identical reference already exists in the profile, the reference is not added and the result code `kFNSDuplicateReferenceErr` is returned. A matching reference is not necessarily identical, since not all the data in a font reference is examined when a matching operation is performed.

Version Notes

Available beginning with FontSync 1.0.

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSProfileClear

Removes all font references from a profile.

```
OSStatus FNSProfileClear (
   FNSFontProfile iProfile
);
```

Parameters

`iProfile`

A font profile reference. Pass a reference to the font profile whose references you wish to remove. The profile must be editable (that is, opened with read/write access).

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that the font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that the profile does not have a valid structure. The File Manager error `permErr` indicates that the file is either locked and not editable or opened for read-only access. `FNSProfileClear` may return other File Manager errors.

Discussion

The `FNSProfileClear` function clears all font references from a specified profile. Note that this is only true for editable profiles (that is, those opened with read/write access). The file of the font profile remains the same size after this operation.

Version Notes

Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileClose

Closes the file associated with a font profile and disposes of run-time data.

```
OSStatus FNSProfileClose (
   FNSFontProfile iProfile
);
```

Parameters

`iProfile`
  A pointer to a font profile reference. Pass a reference to the font profile you wish to close.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure. `FNSProfileClose` may return File Manager errors.

Discussion
The `FNSProfileClose` function closes the file associated with a font profile. Any memory associated with the reference is released. You should call the function `FNSProfileCompact` (page 10) before closing a profile that has been edited, since closing a profile does not automatically compact it.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileCompact

Compacts a font profile.

```
OSStatus FNSProfileCompact (
   FNSFontProfile iProfile
);
```

Parameters

iProfile

A font profile reference. Pass a reference to the font profile you wish to compact. The profile must be editable (that is, opened with read/write access).

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure. `FNSProfileCompact` may return File Manager errors.

Discussion

The `FNSProfileCompact` function eliminates excess space created when creating a font profile (that is, the space you designate for not-yet-existent font references). This space is necessary to minimize growing the file and shuffling data. If a profile has not been opened for read/write access, `FNSProfileCompact` simply returns without doing anything.

You should call `FNSProfileCompact` before closing a profile that has been edited.

Version Notes

Available beginning with FontSync 1.0.

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSProfileCountReferences

Determines the number of font references in a font profile.

```
OSStatus FNSProfileCountReferences (
   FNSFontProfile iProfile,
   ItemCount *oCount
);
```

Parameters

iProfile

A reference to the font profile whose font references you wish to count.

oCount

On return, a pointer to the number of font references in the profile.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that a profile is valid, but created by a later version of FontSync, or that a profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileCreate

Creates an empty FontSync profile using an `FSSpec`.

```
OSStatus FNSProfileCreate (
   const FSSpec *iFile,
   FourCharCode iCreator,
   ItemCount iEstNumRefs,
   FNSObjectVersion iDesiredVersion,
   FNSFontProfile *oProfile
);
```

Parameters

`iFile`
: A pointer to the file that you want to initialize as an empty font profile.

`iCreator`
: The creator code to set for the file. To specify the file creator code assigned by FontSync, pass the `kFNSCreatorDefault` constant, described in "Font Profile Constants" (page 41).

`iEstNumRefs`
: The estimated number of font references that the font profile will contain. Estimating this value minimizes the number of times the file needs to be grown, since the new profile will usually immediately have font references added to it. Pass 0 if you don't know how many font references your profile will contain.

`iDesiredVersion`
: The desired format version of the font profile. Pass a value in the range returned by the function `FNSSysInfoGet` (page 35) in the `oCurProfileVersion` and `oMinProfileVersion` fields of the system information structure. To specify the most recent version supported by the FontSync library regardless of format version, pass the constant `kFNSVersionDontCare`, described in "Version Constants" (page 42).

`oProfile`
: On return, a pointer to a reference to the newly-created font profile.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that you requested an unsupported profile format version. Memory Manager errors indicate that the font profile could not be created because you did not have enough memory available in your heap. `FNSProfileCreate` may return File Manager errors.

Discussion

The `FNSProfileCreate` function creates an empty file containing a FontSync font profile. The newly-created font profile is ready for use. You can add font references to the profile by calling the function `FNSProfileAddReference` (page 8).

`FNSProfileCreate` requires that you specify the desired profile version format because there will likely be changes to the profile file format in future versions. This allows earlier versions of FontSync to use the font profiles you create.

Version Notes

Available beginning with FontSync 1.0. In FontSync 1.0, if you specify the constant `kFNSCreatorDefault` in the `iCreator` parameter of the function `FNSProfileCreate`, FontSync assigns the creator code `'fns'`

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSProfileCreateWithFSRef

Creates an empty FontSync profile using an `FSRef`.

```
OSStatus FNSProfileCreateWithFSRef (
   const FSRef *iParentDirectory,
   UniCharCount iNameLength,
   const UniChar *iName,
   FourCharCode iCreator,
   ItemCount iEstNumRefs,
   FNSObjectVersion iDesiredVersion,
   FNSFontProfile *oProfile
);
```

Parameters

`iParentDirectory`
: A pointer to the parent directory of the file that you want to initialize as an empty font profile.

`iNameLength`
: The number of `UniChar` characters in the `iName` parameter.

`iName`
: The name of the file in which you are storing the profile.

`iCreator`
: The creator code to set for the file. To specify the file creator code assigned by FontSync, pass the `kFNSCreatorDefault` constant, described in "Font Profile Constants" (page 41).

`iEstNumRefs`
: The estimated number of font references that the font profile will contain. Estimating this value minimizes the number of times the file needs to be grown, since the new profile will usually immediately have font references added to it. Pass 0 if you don't know how many font references your profile will contain.

iDesiredVersion

The desired format version of the font profile. Pass a value in the range returned by the function `FNSSysInfoGet` (page 35) in the `oCurProfileVersion` and `oMinProfileVersion` fields of the system information structure. To specify the most recent version supported by the FontSync library regardless of format version, pass the constant `kFNSVersionDontCare`, described in "Version Constants" (page 42).

oProfile

On return, a pointer to the newly-created font profile.

Return Value

A result code. See "FontSync Result Codes" (page 42).

Discussion

The function `FNSProfileCreateWithFSRef` works similarly to the function `FNSProfileCreate`, except that `FNSProfileCreateWithFSRef` uses an `FSRef` instead of an `FSSpec`. An `FSSpec` cannot handle Unicode names that are too long, as long names are truncated. In addition, an `FSSpec` cannot be shared between processes since an `FSSpec` references volume IDs which are different between different processes.

Availability

Not available in CarbonLib 1.0.

Available in Mac OS X 10.1 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSProfileGetIndReference

Retrieves an indexed font reference from a profile.

```
OSStatus FNSProfileGetIndReference (
   FNSFontProfile iProfile,
   UInt32 iWhichReference,
   FNSFontReference *oReference
);
```

Parameters

iProfile

A reference to the font profile whose indexed font reference you want to determine.

iWhichReference

An index into the list of font references in the profile. Pass a value between 0 and one less than the number of references in the profile returned by the function `FNSProfileCountReferences` (page 11).

oReference

On return, a pointer to a reference to the indexed font reference.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that the font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `inputOutOfBounds` indicates that the specified index was out of range. The result code `kFNSInvalidProfileErr` indicates that the profile does not have a valid structure. `FNSProfileGetIndReference` may return File Manager errors. Memory Manager errors indicate that you did not have enough memory available in your heap.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileGetVersion

Retrieves the format version of an open font profile.

```
OSStatus FNSProfileGetVersion (
   FNSFontProfile iProfile,
   FNSObjectVersion *oVersion
);
```

Parameters

`iProfile`

A reference to the font profile whose format version number you wish to obtain.

`oVersion`

On return, a pointer to the format version number of the font profile.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSInvalidProfileErr` indicates that the profile does not have a valid structure.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileMatchReference

Obtains a list of the references in a profile that match a given reference.

```
OSStatus FNSProfileMatchReference (
   FNSFontProfile iProfile,
   FNSFontReference iReference,
   FNSMatchOptions iMatchOptions,
   ItemCount iOutputSize,
   UInt32 oIndices[],
   ItemCount *oNumMatches
);
```

Parameters

iProfile
: A reference to the font profile containing the font references you wish to compare.

iReference
: A reference to a font reference against which you are performing the comparison.

iMatchOptions
: A bit mask you can use to set the matching option bits to be used in the comparison. To specify the global default match criteria, pass the bit mask returned by the function `FNSMatchDefaultsGet` (page 7). Your application can specify options that are different from the user's preferences via this mask.

iOutputSize
: The number of font references you want passed back in the `oIndices` array. This may be less than the actual number of matches passed back in the `oNumMatches` parameter. To determine this value, see the discussion below.

oIndices
: On return, a pointer to an array of indices identifying the font references that matched. The number of indices returned is limited by the value you specify in the `iOutputSize` parameter. The total number of matching references is passed back in the `oNumMatches` parameter.

oNumMatches
: On return, a pointer to the total number of matching font references. This value may be greater than the number of indices passed back in the `oIndices` array.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported version number. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure. The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSMismatchErr` indicates that no matches were found. The File Manager error `permErr` indicates that the file is either locked and not editable or opened for read-only access. `FNSProfileMatchReference` may return other File Manager errors. Memory Manager errors indicate that you did not have enough memory available in your heap.

Discussion

The `FNSProfileMatchReference` function obtains a list of the font references that match a specified reference. Since there may be more than one matching reference, a list is returned.

The number of font references passed back in the `oIndices` array is limited by the value you specify in the `iOutputSize` parameter. The actual number of matches is passed back in the `oNumMatches` parameter. You can check this value to determine whether the `oIndices` array was large enough to contain the matches.

If you want to determine whether the profile has a matching font, but don't care which one, pass 0 for the `iOutputSize` parameter and `NULL` for the `oNumMatches` parameter. The result code `noErr` indicates that matches were found, while the result code `kFNSMismatchErr` indicates that no matches were found.

To determine the number of matches, call `FNSProfileMatchReference` and pass 0 for the `iOutputSize` parameter. The pointer passed back in the `oNumMatches` parameter will point to the actual number of matches. You can then call `FNSProfileMatchReference` again, passing the returned number of matches in the `iOutputSize` parameter.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileOpen

Opens an existing font profile using an `FSSpec`.

```
OSStatus FNSProfileOpen (
   const FSSpec *iFile,
   Boolean iOpenForWrite,
   FNSFontProfile *oProfile
);
```

Parameters

`iFile`
: A pointer to the font profile file that you wish to open.

`iOpenForWrite`
: A flag indicating whether the profile file is read/write or read-only. Pass `true` to allow read/write access. This is necessary if the profile is going to be editable.

`oProfile`
: On return, a pointer to a reference to the open font profile.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure. Memory Manager errors indicate that you did not have enough memory available in your heap. `FNSProfileOpen` may return File Manager errors.

Discussion
The `FNSProfileOpen` function opens an already-existing font profile (that is, one that contains font references). If you want to make the font profile editable, pass true in the `iOpenForWrite` parameter. `FNSProfileOpen` will not open an empty profile created by the function `FNSProfileCreate` (page 12).

Font profiles are housed in a file. FontSync attempts to moderate access to this file. Ideally, it tries to either allow many readers or exactly one writer but not both. The Mac OS File Manager does not allow this kind of exclusion on local volumes, so it may still be possible for someone to get write access to a profile when there are active readers. Rather than complicating the implementation to work around this limitation, FontSync profile files are treated like most document files. That is, the caller is responsible for making sure this does not occur. If the user wishes to modify a profile, your application should make a copy of the file, modify the copy, and swap file names when done. This has the added benefit of preserving the original profile if an error leaves the new profile invalid.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileOpenWithFSRef

Opens an existing font profile using an `FSRef`.

```
OSStatus FNSProfileOpenWithFSRef (
   const FSRef *iFile,
   Boolean iOpenForWrite,
   FNSFontProfile *oProfile
);
```

Parameters

`iFile`

A pointer to the `FSRef` that specifies the file that you wish to open.

`iOpenForWrite`

A flag indicating whether the profile file is read/write or read-only. Pass `true` to allow read/write access. This is necessary if the profile is going to be editable.

`oProfile`

On return, a pointer to a reference to the open font profile.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure. Memory Manager errors indicate that you did not have enough memory available in your heap. `FNSProfileOpen` may return File Manager errors.

Discussion
The function `FNSProfileOpenWithFSRef` works similarly to the function `FNSProfileOpen`, `FNSProfileOpenWithFSRef` uses an `FSRef` instead of an `FSSpec`. An `FSSpec` cannot handle Unicode names that are too long, as long names are truncated. In addition, an `FSSpec` cannot be shared between processes since an `FSSpec` references volume IDs which are different between different processes.

Availability
Not available in CarbonLib 1.0.

Available in Mac OS X 10.1 and later

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileRemoveIndReference

Deletes an indexed font reference from a profile.

```
OSStatus FNSProfileRemoveIndReference (
   FNSFontProfile iProfile,
   UInt32 iIndex
);
```

Parameters

`iProfile`

A reference to the font profile whose indexed font reference you want to delete. The profile must be writable.

`iIndex`

An index into the list of font references in the profile. Pass a value between 0 and one less than the number of references in the profile, returned by the function `FNSProfileCountReferences` (page 11). Note that this will change the indices of all succeeding references.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure. The result code `permErr` indicates that a specified file is locked and not writable. The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `inputOutOfBounds` indicates that the specified index was out of range. The File Manager error `permErr` indicates that the file is either locked and not editable or opened for read-only access. `FNSProfileRemoveIndReference` may return other File Manager errors. Memory Manager errors indicate that you did not have enough memory available in your heap.

Discussion
The `FNSProfileRemoveIndReference` function deletes an indexed font reference from an editable profile. The reference must be identical to the reference specified in the `iReference` parameter. A matching reference is not enough, since not all the data in a font reference is examined when a matching operation is performed.

You can use either `FNSProfileRemoveIndReference` or the function `FNSProfileRemoveReference` (page 20) to remove a font reference, depending upon what you know about the reference. If you know its value, call `FNSProfileRemoveReference` (page 20). If you know its index in the list of font references, call `FNSProfileRemoveIndReference`.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSProfileRemoveReference

Deletes a font reference from a profile.

```
OSStatus FNSProfileRemoveReference (
   FNSFontProfile iProfile,
   FNSFontReference iReference
);
```

Parameters

`iProfile`

A reference to the font profile whose font reference you want to delete. The profile must be writable.

`iReference`

A reference to the font reference you wish to remove.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadProfileVersionErr` indicates that a font profile has an unsupported format version. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. The result code `kFNSInvalidProfileErr` indicates that a profile does not have a valid structure. The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSMismatchErr` indicates that the reference you wish to remove is not in the profile. The File Manager error `permErr` indicates that the file is either locked and not editable or opened for read-only access. `FNSProfileRemoveReference` may return other File Manager errors. Memory Manager errors indicate that you did not have enough memory available in your heap.

Discussion
The `FNSProfileRemoveReference` function deletes a font reference from an editable profile. The reference must be identical to the reference specified in the `iReference` parameter. A matching reference is not necessarily identical, since not all the data in a font reference is examined when a matching operation is performed.

You can use either `FNSProfileRemoveReference` or the function `FNSProfileRemoveIndReference` (page 19) to remove a font reference, depending upon what you know about the reference. If you know the value of the reference, call `FNSProfileRemoveReference`. If you know its index in the list of font references, call `FNSProfileRemoveIndReference` (page 19).

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceCountNames

Determines the number of internal font names in a reference.

```
OSStatus FNSReferenceCountNames (
   FNSFontReference iReference,
   ItemCount *oNameCount
);
```

Parameters

iReference
: A reference to the font reference whose font names you wish to count.

oNameCount
: On return, a pointer to the number of internal font names, other than the font family name passed to the `GetFNum` function, recorded in the reference. The font family passed to the function `GetFNum` is available by calling the function `GetFamilyInfo`. This includes the PostScript and unique names, if available.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSInsufficientDataErr` indicates that the mask constant `kFNSMissingDataNoMatch` was set and both references being compared are missing the same data. The result code `kFNSMismatchErr` indicates that no font names were recorded in the reference.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceCreate

Creates a font reference based on a font object.

```
OSStatus FNSReferenceCreate (
   FMFont iFont,
   FNSObjectVersion iDesiredVersion,
   FNSFontReference *oReference
);
```

Parameters

iFont

The font object ID representing the font whose reference you wish to create.

iDesiredVersion

The desired font reference format version number. Pass a value between the oldest and current format version numbers supported by the FontSync library. You can determine this range by examining the `oCurRefVersion` and `oMinRefVersion` fields of the `FNSSysInfo` (page 38) structure. This structure is passed back in the `ioInfo` parameter of the function `FNSSysInfoGet` (page 35). To specify the most recent version supported by the FontSync library regardless of format version, pass the constant `kFNSVersionDontCare`, described in "Version Constants" (page 42).

oReference

On return, a pointer to a reference to the newly-created FontSync reference.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that you requested an unsupported reference format version. The Font Manager result code `kFMInvalidFontErr` indicates that a font is invalid. Memory Manager errors indicate that a font reference could not be created because you did not have enough memory available in your heap.

Discussion

You should call the `FNSReferenceCreate` function to create a font reference if your application uses ATSUI to render text. If the specified font object is associated with a font family, the newly-created font reference will contain the QuickDraw Text-specific information from that associated family.

The `FNSReferenceCreate` function requires that you specify the desired font reference format version because there will likely be changes to the nature of the "fingerprints" in a font reference in future versions. This allows earlier versions of FontSync to use the font references you create.

Version Notes

Available beginning with FontSync 1.0. In FontSync 1.0, a font object can only belong to one family. FontSync uses the family returned by the Font Manager function `FMGetFontFamilyInstanceFromFont`.

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSReferenceCreateFromFamily

Creates a font reference based on a font family and style.

```
OSStatus FNSReferenceCreateFromFamily (
   FMFontFamily iFamily,
   FMFontStyle iStyle,
   FNSObjectVersion iDesiredVersion,
   FNSFontReference *oReference,
   FMFontStyle *oActualStyle
);
```

Parameters

`iFamily`
: The font family of the font whose reference you wish to create.

`iStyle`
: The style of the font family. This value is often not the actual style of the font reference being created, since there are often left-over style bits. The actual style of the newly-created font reference is passed back in the `oActualStyle` parameter. For more information, see the discussion.

`iDesiredVersion`
: The desired format version of the font reference. Pass a value in the range returned by the function `FNSSysInfoGet` (page 35) in the `oCurRefVersion` and `oMinRefVersion` fields of the system information structure. To specify the most recent version supported by the FontSync library regardless of format version, pass the constant `kFNSVersionDontCare`, described in "Version Constants" (page 42).

`oReference`
: On return, a pointer to a reference to the newly-created FontSync reference.

`oActualStyle`
: On return, a pointer to the actual style of the newly-created font reference. This value may differ from the value you passed in the `iStyle` parameter. For more information, see the discussion. This value may be `NULL`.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that you requested an unsupported reference format version. The Font Manager result code `kFMInvalidFontFamilyErr` indicates that a font family is invalid. Memory Manager errors indicate that a font reference could not be created because you did not have enough memory available in your heap.

Discussion
You should call the `FNSReferenceCreateFromFamily` function to create a font reference if your application uses QuickDraw Text to render text.

The style you specify in the `iStyle` parameter is often not the actual style of the font reference being created, since there may not be a real face corresponding to that style. For example, a family may not have a real italic face, so any italicization is handled by skewing the glyphs. The actual style of the newly-created font reference is passed back in the `oActualStyle` parameter.

The `FNSReferenceCreateFromFamily` function requires that you specify the desired font reference format version because there will likely be changes to the nature of the "fingerprints" in a font reference in future versions. This allows earlier versions of FontSync to use the font references you create.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceDispose

Disposes of the storage associated with a font reference.

```
OSStatus FNSReferenceDispose (
   FNSFontReference iReference
);
```

Parameters

`iReference`
A pointer to the font reference whose associated memory you wish to dispose of.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version. This may indicate that the font reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. Memory Manager errors indicate that you did not have enough memory available in your heap.

Special Considerations

You should not use a font reference after calling the `FNSReferenceDispose` function to dispose of its storage.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceFindName

Finds the first name that matches the given font name parameters, if any.

```
OSStatus FNSReferenceFindName (
   FNSFontReference iReference,
   FontNameCode iFontNameCode,
   FontPlatformCode iFontNamePlatform,
   FontScriptCode iFontNameScript,
   FontLanguageCode iFontNameLanguage,
   ByteCount iMaximumNameLength,
   Ptr oName,
   ByteCount *oActualNameLength,
   ItemCount *oFontNameIndex
);
```

Parameters

iReference

A reference to the font reference whose font name you are searching for.

iFontNameCode

The type of the font name string you are searching for.

iFontNamePlatform

The encoding of the font name string you are searching for. You can pass the `kFontNoPlatform` constant if you do not care about the encoding of a font name. In this case, `FNSReferenceFindName` will pass back the first name matching the other font name parameters.

iFontNameScript

The script code of the font name string you are searching for. You can pass the `kFontNoScript` constant if you do not care about the script ID. In this case, `FNSReferenceFindName` will pass back the first name matching the other font name parameters.

iFontNameLanguage

The language code of the font name string you are searching for. You can pass the `kFontNoLanguage` constant if you do not care about the language of the font name. In this case, `FNSReferenceFindName` will pass back the first name matching the other font name parameters.

iMaximumNameLength

The maximum length of the font name. Typically, this is equivalent to the size of the buffer allocated to contain the font name pointed to by the `oName` parameter. To determine this length, see the discussion below.

oName

A pointer to a buffer. Before calling `FNSReferenceFindName`, pass a pointer to memory that you have allocated for this buffer. On return, the buffer contains the font name string. If the buffer you allocate is not large enough, `FNSReferenceFindName` passes back a partial string.

oActualNameLength

On return, a pointer to the actual length of the font name string. This may be greater than the value passed in the `iMaximumNameLength` parameter. You should check this value to make sure that you allocated enough memory for the buffer.

oFontNameIndex

On return, a pointer to a 0-based index of the font name in the font name table. This can be used with the function `FNSReferenceGetIndName` (page 28) to determine the actual values of unknown font name parameters.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSInsufficientDataErr` indicates that the mask constant `kFNSMissingDataNoMatch` was set and both references being compared are missing the same data. The result code `kFNSNameNotFoundErr` indicates that there was no name in the font reference that matched the given parameters.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceFlatten

Flattens a font reference.

```
OSStatus FNSReferenceFlatten (
   FNSFontReference iReference,
   void *oFlatReference,
   ByteCount *oFlattenedSize
);
```

Parameters

`iReference`
: The font reference that you want to flatten.

`oFlatReference`
: A pointer to the storage for the font reference to be flattened. Pass a `NULL` pointer if you wish to determine the size of the flattened reference without actually creating it.

`oFlattenedSize`
: On return, a pointer to the flattened size (in bytes) of the font reference.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid.

Discussion
The `FNSReferenceFlatten` function flattens a font reference into a form which can be stored externally (for example, in a document or embedded in an Apple event), and returns the size of the flattened reference in the `oFlattenedSize` parameter. `FNSReferenceFlatten` assumes that the storage pointed to by `iFlatReference` is large enough to hold the data and will always contain a full flattened reference.

If you simply want to calculate the size of a flattened reference, you can pass a `NULL` pointer in the `iFlatReference` parameter or call the function `FNSReferenceFlattenedSize` (page 27).

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceFlattenedSize

Calculates the space required for the flattened form of a font reference.

```
OSStatus FNSReferenceFlattenedSize (
   FNSFontReference iReference,
   ByteCount *oFlattenedSize
);
```

Parameters

`iReference`
: The font reference whose flattened form you wish to compute.

`oFlattenedSize`
: On return, a pointer to the flattened size (in bytes) of the font reference.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that the font reference has an unsupported format version. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that the font reference is invalid.

Discussion
You can call the `FNSReferenceFlattenedSize` function to calculate the size of a flattened reference. You can also accomplish this by passing a `NULL` pointer in the `iFlatReference` parameter of the function `FNSReferenceFlatten` (page 26).

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceGetFamilyInfo

Obtains information about a font family represented by a font reference.

```
OSStatus FNSReferenceGetFamilyInfo (
   FNSFontReference iReference,
   Str255 oFamilyName,
   ScriptCode *oFamilyNameScript,
   FMFontStyle *oActualStyle
);
```

Parameters

iReference

A reference to the font reference representing a font family.

oFamilyName

On return, the name by which the font is known to the classic Font Manager (that is, the string you pass to the Font Manager function `GetFNum`). If you do not want to obtain this information, pass `NULL`.

oFamilyNameScript

On return, a pointer to the script code of the family name string. If you do not want to obtain this information, pass `NULL`.

oActualStyle

On return, a pointer to the actual QuickDraw style associated with the font reference. This is the value passed back in the `oActualStyle` parameter of the function `FNSReferenceCreateFromFamily` (page 22). If you do not want to obtain this information, pass `NULL`. For more information, see the discussion of `FNSReferenceCreateFromFamily` (page 22).

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSInsufficientDataErr` indicates that the mask constant `kFNSMissingDataNoMatch` was set and both references being compared are missing the same data. The result code `kFNSMismatchErr` indicates that no font names were recorded in the reference.

Version Notes

Available beginning with FontSync 1.0.

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSReferenceGetIndName

Finds the font name string and other font name parameters for an indexed font name.

```
OSStatus FNSReferenceGetIndName (
   FNSFontReference iReference,
   ItemCount iFontNameIndex,
   ByteCount iMaximumNameLength,
   Ptr oName,
   ByteCount *oActualNameLength,
   FontNameCode *oFontNameCode,
   FontPlatformCode *oFontNamePlatform,
   FontScriptCode *oFontNameScript,
   FontLanguageCode *oFontNameLanguage
);
```

Parameters

iReference

A reference to the font reference whose indexed font name you want information about.

iFontNameIndex

An index of the font name you want information about. Pass a value between 0 and one less than the count passed back by the function `FNSProfileCountReferences` (page 11).

iMaximumNameLength

The maximum length of the font name. Typically, this is equivalent to the size of the buffer allocated to contain the font name pointed to by the `oName` parameter. To determine this length, see the discussion below.

oName

A pointer to a buffer. Before calling `FNSReferenceGetIndName`, pass a pointer to memory that you have allocated for this buffer. If you are uncertain of how much memory to allocate, see the discussion below. On return, the buffer contains the font name string. If the buffer you allocate is not large enough, `FNSReferenceGetIndName` passes back a partial string.

oActualNameLength

On return, a pointer to the actual length of the font name string. This may be greater than the value passed in the `iMaximumNameLength` parameter. You should check this value to make sure that you allocated enough memory for the buffer.

oFontNameCode

On return, a pointer to the type of the font name string.

oFontNamePlatform

On return, a pointer to the encoding of the font name string.

oFontNameScript

On return, a pointer to the script ID of the font name string.

oFontNameLanguage

On return, a pointer to the language of the font name string.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSInsufficientDataErr` indicates that the mask constant `kFNSMissingDataNoMatch` was set and both references being compared are missing the same data. The result code `inputOutOfBounds` indicates that the specified index was out of range.

Discussion

You should call the `FNSReferenceGetIndName` function to iterate through the entries of a font name table to find the font name string, name code, language code, script code, and platform code of an indexed font name.

The best way to use `FNSReferenceGetIndName` is to call it twice:

- Pass the reference of the font whose name table you want to iterate in the `iReference` parameter, `NULL` for the `oName` parameter, and 0 for the other parameters. `FNSReferenceGetIndName` returns the length of the font name string in the `oActualNameLength` parameter.
- Allocate enough space for a font name buffer of the returned size, then call the function again, passing a pointer in the `oName` parameter; on return, the pointer references the font name string.

To find the index and font name of the first font in a name table matching given font name parameters, call the function `FNSReferenceFindName` (page 24).

Version Notes

Available beginning with FontSync 1.0.

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSReferenceGetVersion

Indicates the format version number of a font reference.

```
OSStatus FNSReferenceGetVersion (
   FNSFontReference iReference,
   FNSObjectVersion *oVersion
);
```

Parameters

`iReference`

The font reference whose format version number you wish to determine.

`oVersion`

On return, a pointer to the format version number of the specified font reference.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSInvalidReferenceErr` indicates that the font reference is invalid.

Version Notes

Available beginning with FontSync 1.0.

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceMatch

Compares font references using specified matching options.

```
OSStatus FNSReferenceMatch (
   FNSFontReference iReference1,
   FNSFontReference iReference2,
   FNSMatchOptions iOptions,
   FNSMatchOptions *oFailedMatchOptions
);
```

Parameters

`iReference1`
A font reference whose contents you wish to compare to the font reference in the `iReference2` parameter.

`iReference2`
A font reference whose contents you wish to compare to the font reference in the `iReference1`parameter.

`iOptions`
A bit mask you can use to set the match option bits to be used in the font comparison. To specify the global default match criteria, pass the bit mask returned by the function `FNSMatchDefaultsGet` (page 7). Your application can specify options that are different from the user's preferences via this mask.

`oFailedMatchOptions`
Before calling `FNSReferenceMatch`, pass `NULL` if you do not desire to know which match options failed. On return, a pointer to a bit mask that you can test to determine the match options that failed to match in the event of a mismatch.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSMismatchErr` indicates that a font reference did not match. The result code `kFNSInsufficientDataErr` indicates that the mask constant `kFNSMissingDataNoMatch` was set and both references being compared are missing the same data.

Discussion
The `FNSReferenceMatch` function returns a bit mask indicating the matching options that did not match when comparing two font references. You should specify which match options you wish to compare in the `iOptions` parameter. To specify the default match criteria, pass the bit mask returned by the function `FNSMatchDefaultsGet` (page 7). If the match fails, on return, the `oFailedMatchOptions` parameter contains a bit mask of the elements that failed to match. You can use the bit mask to determine the criteria under which the fonts failed to match.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceMatchFamilies

Obtains a list of font families that match a reference.

```
OSStatus FNSReferenceMatchFamilies (
   FNSFontReference iReference,
   FNSMatchOptions iMatchOptions,
   ItemCount iOutputSize,
   FMFontFamilyInstance oFonts[],
   ItemCount *oNumMatches
);
```

Parameters

`iReference`
A reference to the font reference whose matching font(s) you wish to determine.

`iMatchOptions`
A bit mask you can use to set the matching option bits to be used in the comparison. To specify the global default match criteria, pass the bit mask returned by the function `FNSMatchDefaultsGet` (page 7). The total number of matching references is passed back in the `oNumMatches` parameter. Your application can specify options that are different from the user's preferences via this mask.

`iOutputSize`
The capacity of the `oFonts` array. This may be less than the actual number of matches passed back in the `oNumMatches` parameter.

`oFonts`
On return, a pointer to an array of indices identifying the fonts matching the specified reference. The number of indices returned is limited by the value you specify in the `iOutputSize` parameter.

`oNumMatches`
On return, a pointer to the total number of font families that match the specified reference. This value may be greater than the number of fonts passed back in the `oFonts` array.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSMismatchErr` indicates that no matches were found. Memory Manager errors indicate that you did not have enough memory available in your heap.

Discussion
The `FNSReferenceMatchFamilies` function maps a font reference to an active font that can be used with QuickDraw Text. Since there may be more than one such font, a list is returned.

The number of fonts passed back in the `oFonts` array is limited by the value you specify in the `iOutputSize` parameter. The actual number of matches is passed back in the `oNumMatches` parameter. You can check this value to determine whether the `oFonts` array was large enough to contain the matches.

If `FNSReferenceMatchFamilies` cannot find a font family that matches a font reference and someone has registered interest in this process, FontSync sends an Apple Event with the details of the request to the third party font-management utility in question. For more information, see the discussion for the function `FNSReferenceMatchFonts` (page 33).

If you want to determine whether the profile has a matching font, but don't care which one, pass 0 for the `iOutputSize` parameter and `NULL` for the `oNumMatches` parameter. The result code `noErr` indicates that matches were found, while the result code `kFNSMismatchErr` indicates that no matches were found.

To determine the number of matches, call `FNSReferenceMatchFamilies` and pass 0 for the `iOutputSize` parameter. The pointer passed back in the `oNumMatches` parameter will point to the actual number of matches. You can then call `FNSReferenceMatchFamilies` again, passing the returned number of matches in the `iOutputSize` parameter.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceMatchFonts

Obtains a list of font objects that match a reference.

```
OSStatus FNSReferenceMatchFonts (
   FNSFontReference iReference,
   FNSMatchOptions iMatchOptions,
   ItemCount iOutputSize,
   FMFont oFonts[],
   ItemCount *oNumMatches
);
```

Parameters

`iReference`
A reference to the font reference whose matching font(s) you wish to determine.

`iMatchOptions`
A bit mask you can use to set the matching option bits to be used in the comparison. To specify the global default match criteria, pass the bit mask returned by the function `FNSMatchDefaultsGet` (page 7). The total number of matching references is passed back in the `oNumMatches` parameter. Your application can specify options that are different from the user's preferences via this mask.

`iOutputSize`
The capacity of the `oFonts` array. This may be less than the actual number of matches passed back in the `oNumMatches` parameter.

`oFonts`
On return, a pointer to an array of indices identifying the fonts matching the specified reference. The number of indices returned is limited by the value you specify in the `iOutputSize` parameter.

`oNumMatches`
: On return, a pointer to the total number of font objects that match the specified reference. This value may be greater than the number of fonts passed back in the `oFonts` array.

Return Value
A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSInvalidReferenceErr` indicates that a font reference is invalid. The result code `kFNSMismatchErr` indicates that no matches were found. Memory Manager errors indicate that you did not have enough memory available in your heap.

Discussion
The `FNSReferenceMatchFonts` function passes back a list of active fonts which match the specified reference. `FNSReferenceMatchFonts` maps a font reference to an actual font that can be used with ATSUI. Since there may be more than one such font, a list is returned.

The number of fonts passed back in the `oFonts` array is limited by the value you specify in the `iOutputSize` parameter. The actual number of matches is passed back in the `oNumMatches` parameter. You can check this value to determine whether the `oFonts` array was large enough to contain the matches.

If you want to determine whether the profile has a matching font, but don't care which one, pass 0 for the `iOutputSize` parameter and `NULL` for the `oNumMatches` parameter. The result code `noErr` indicates that matches were found, while the result code `kFNSMismatchErr` indicates that no matches were found.

To determine the number of matches, call `FNSReferenceMatchFonts` and pass 0 for the `iOutputSize` parameter. The pointer passed back in the `oNumMatches` parameter will point to the actual number of matches. You can then call `FNSReferenceMatchFonts` again, passing the returned number of matches in the `iOutputSize` parameter.

If `FNSReferenceMatchFonts` cannot find an active font that matches a font reference and your application has registered interest in this process, FontSync sends an Apple Event with the details of the request to the third party font-management utility in question. The receiver should respond with a list of matching fonts, taking whatever steps are necessary to identify and activate them before replying to the event. Registration is handled by the simple expedient of installing a handler for the appropriate Apple Event. This handler will typically be installed in the system table, though FontSync will check for handlers both in the system and in the context's local handler table. The Apple Event will be a send-to-self.

Version Notes
Available beginning with FontSync 1.0.

Availability
Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`FontSync.h`

FNSReferenceUnflatten

Reconstitutes a flattened font reference.

```
OSStatus FNSReferenceUnflatten (
   const void *iFlatReference,
   ByteCount iFlattenedSize,
   FNSFontReference *oReference
);
```

Parameters

iFlatReference

A pointer to the flattened font reference.

iFlattenedSize

The size (in bytes) of the flattened font reference.

oReference

On return, a pointer to a reference to the reconstituted font reference.

Return Value

A result code. See "FontSync Result Codes" (page 42). The result code `kFNSBadReferenceVersionErr` indicates that a font reference has an unsupported format version. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid. The result code `kFNSBadFlattenedSizeErr` indicates that either the specified size doesn't match the size recorded in the flattened reference or the size was not large enough to hold a flattened reference. The result code `kFNSInvalidReferenceErr` indicates that a reconstructed reference is bad. Memory Manager errors indicate that you did not have enough memory available in your heap.

Discussion

The `FNSReferenceUnflatten` function reconstitutes a flattened font reference from its external form. For example, you could use `FNSReferenceUnflatten` to read a font reference out of a document. The `iFlattenedSize` parameter is not really necessary since a flattened reference contains its own size. However, you can use this value to check that you have passed the right amount of data for the flattened reference.

Version Notes

Available beginning with FontSync 1.0.

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`FontSync.h`

FNSSysInfoGet

Determines version and feature information for the version of FontSync installed on the user's system.

```
void FNSSysInfoGet (
   FNSSysInfo *ioInfo
);
```

Parameters

ioInfo

Before calling the FNSSysInfoGet function, pass a pointer to a FNSSysInfo (page 38) structure. Fill in the iSysInfoVersion field of the structure with the version of this structure. Pass the constant kFNSCurSysInfoVersion, described in "Version Constants" (page 42), to represent the current version. On return, FNSSysInfoGet fills in the remaining fields and passes back a pointer to the structure.

Discussion

Before calling the FNSSysInfoGet function, you should fill in the iSysInfoVersion field of the FNSSysInfo (page 38) structure with the version of this structure. Pass the constant kFNSCurSysInfoVersion, described in "Version Constants" (page 42) , to represent the current version. FNSSysInfoGet fills in the remaining fields and passes back the structure in the ioInfo parameter. The information it provides includes the version of FontSync running in the current context and available features, as well as the current and oldest font reference and profile format versions supported by the FontSync library.

New fields may be added to the end of the structure in future versions of FontSync. FontSync uses the iSysInfoVersion field to determine which version of the structure you are using. The value of the current version constant kFNSCurSysInfoVersion will change accordingly.

Version Notes

Available beginning with FontSync 1.0. In FontSync 1.0, the current structure version is defined by the constant kFNSCurSysInfoVersion, described in "Version Constants" (page 42).

Availability

Available in CarbonLib 1.0 and later when Font Sync 1.0 or later is present.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

FontSync.h

Data Types

FNSFeatureFlags

Represents a mask that you can use to determine available FontSync features.

```
typedef UInt32 FNSFeatureFlags;
```

Discussion

The FNSFeatureFlags type defines a bit mask your application can use to determine available FontSync features. The function FNSSysInfoGet (page 35) passes back a mask of this type in the oFeatures field of the FNSSysInfo (page 38) structure in the ioInfo parameter. You can use this mask to determine available FontSync features.

Version Notes

Available beginning with FontSync 1.0. In FontSync 1.0, the value is 0, since no feature flags are defined.

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

Declared In
`FontSync.h`

FNSFontProfile

Represents a reference to a font profile.

```
typedef struct OpaqueFNSFontProfile * FNSFontProfile;
```

Discussion
The `FNSFontProfile` type is a reference to an opaque structure containing a collection of font references. It defines a set of fonts on the user's system. Although you do not need to use font profiles to iterate, identify, and match fonts on the user's system, they are necessary in building font menus and other font selection human interface elements.

You pass a font profile to FontSync functions that manipulate font profiles. A font reference is passed back by functions that create font profiles. For a description of these functions, see "Working With FontSync Profiles" (page 6).

Version Notes
Available beginning with FontSync 1.0.

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

Declared In
`FontSync.h`

FNSFontReference

Represents a reference to a font reference.

```
typedef struct OpaqueFNSFontReference * FNSFontReference;
```

Discussion
The `FNSFontReference` type is a reference to an opaque structure containing information about a font. Some of the data contained in a font reference includes the QuickDraw font family name, ATSUI-visible font name, type of font, font version, checksums of the data, and information from the font name table.

You pass a font reference to FontSync functions that manipulate font references. A font reference is passed back by functions that create font references. For a description of these functions, see "Working With FontSync References" (page 6).

Version Notes
Available beginning with FontSync 1.0.

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

Declared In
`FontSync.h`

FNSSysInfo

Contains FontSync version and feature information.

```
struct FNSSysInfo {
    FNSObjectVersion iSysInfoVersion;
    FNSFeatureFlags oFeatures;
    FNSObjectVersion oCurRefVersion;
    FNSObjectVersion oMinRefVersion;
    FNSObjectVersion oCurProfileVersion;
    FNSObjectVersion oMinProfileVersion;
    UInt16 oFontSyncVersion;
};
typedef struct FNSSysInfo FNSSysInfo;
```

Fields

`iSysInfoVersion`
: On input, the version of this parameter block structure. In FontSync 1.0, the version number of this structure is 1. Pass the constant `kFNSCurSysInfoVersion`, described in "Version Constants" (page 42). For more information, see the discussion.

`oFeatures`
: On output, the FontSync features that are available. In FontSync 1.0, no feature flags are defined.

`oCurRefVersion`
: On output, the current font reference format version supported by the FontSync library.

`oMinRefVersion`
: On output, the oldest font reference format version supported by the FontSync library.

`oCurProfileVersion`
: On output, the current font profile format version supported by the FontSync library.

`oMinProfileVersion`
: On output, the oldest font profile format version supported by the FontSync library.

`oFontSyncVersion`
: On output, a binary-coded decimal value indicating the version of FontSync currently running. The high-order 8 bits give the major version, the next four give the minor version, and the last four give the revision. For example, version 1.0 would be encoded as 0x0100.

Discussion

Before calling the function `FNSSysInfoGet` (page 35), you should fill in the `iSysInfoVersion` field of this structure with the version of this structure. Pass the constant `kFNSCurSysInfoVersion`, described in "Version Constants" (page 42), to represent the current version. `FNSSysInfoGet` (page 35) fills in the remaining fields and passes back the structure in the `ioInfo` parameter. The information it provides includes the version of FontSync running in the current context and available features, as well as the current and oldest font reference and profile format versions supported by the FontSync library.

New fields may be added to the end of the structure in future versions of FontSync. FontSync uses the `iSysInfoVersion` field to determine which version of the structure you are using. The value of the current version constant `kFNSCurSysInfoVersion` will change accordingly.

Version Notes

Available beginning with FontSync 1.0. In FontSync 1.0, the value of the `iSysInfoVersion` field is 1. The value of the `oFeatures` field is 0, since no feature flags are defined.

Availability

Available in Mac OS X v10.0 and later.

Declared In
`FontSync.h`

Constants

Matching Options

Represent a mask that you can use to set and determine default match options.

```
typedef UInt32 FNSMatchOptions;
enum {
    kFNSMatchNames = 0x00000001,
    kFNSMatchTechnology = 0x00000002,
    kFNSMatchGlyphs = 0x00000004,
    kFNSMatchEncodings = 0x00000008,
    kFNSMatchQDMetrics = 0x00000010,
    kFNSMatchATSUMetrics = 0x00000020,
    kFNSMatchKerning = 0x00000040,
    kFNSMatchWSLayout = 0x00000080,
    kFNSMatchAATLayout = 0x00000100,
    kFNSMatchPrintEncoding = 0x00000200,
    kFNSMissingDataNoMatch = 0x80000000,
    kFNSMatchAll = 0xFFFFFFFF,
    kFNSMatchDefaults = 0
};
```

Constants

`kFNSMatchNames`

If the bit specified by this mask is set, all significant font names must match. This includes the QuickDraw Text family, ATSUI, unique, full, manufacturer, and version names. Note that the PostScript names are also examined as part of the `kFNSMatchPrintEncoding` option.

Available in Mac OS X v10.0 and later.

Declared in `FontSync.h`.

`kFNSMatchTechnology`

If the bit specified by this mask is set, scaler technologies must match. It is possible to match other parts of the font across different technologies, but this is not supported by FontSync 1.0. As a result, even if this bit is not set, fonts of different technologies will probably not match under any other criteria.

Available in Mac OS X v10.0 and later.

Declared in `FontSync.h`.

`kFNSMatchGlyphs`

If the bit specified by this mask is set, glyph repertoires and outline/bitmap data must match.

Available in Mac OS X v10.0 and later.

Declared in `FontSync.h`.

`kFNSMatchEncodings`
: If the bit specified by this mask is set, the `'cmap'` tables must match. If the order of the `'cmap'` tables is different, although the tables are the same, this may be considered a mismatch, since it can cause QuickDraw Text to use a different `'cmap'table`.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchQDMetrics`
: If the bit specified by this mask is set, metrics used by QuickDraw Text must match. This includes the effect of `fractEnable` and any metric information in the `'FOND'` resource.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchATSUMetrics`
: If the bit specified by this mask is set, metrics used by ATSUI must match. This includes both horizontal and vertical metrics.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchKerning`
: If the bit specified by this mask is set, kerning data must match.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchWSLayout`
: If the bit specified by this mask is set, layout information given by an `'itl5'` table, whether attached directly to the font or the one provided in the script bundle, must match.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchAATLayout`
: If the bit specified by this mask is set, advanced layout information such as that used by ATSUI, must match. This includes such things as ligature and morphing tables. OpenType-style layout information is included in this option.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchPrintEncoding`
: If the bit specified by this mask is set, PostScript names and `'FOND'` re-encoding vectors must match. Note that it is an error for a font's internal PostScript name to be different from the one in the `'FOND'`, but FontSync will record both and consider them separately.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMissingDataNoMatch`
: If the bit specified by this mask is set, FontSync will report font reference mismatches when both fonts are missing data needed by a selected option. This is useful, since some older fonts may not have all the data needed for matching newer fonts. This makes the mask constant `kFNSMatchAll` specify the most stringent possible match criteria.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchAll`
: If the bit specified by this mask is set, all of the match options must match. In this case, the bit specified by the mask constant `kFNSMissingDataNoMatch` is also set, asserting the most stringent possible match criteria.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSMatchDefaults`
: If this constant is specified, the global default match criteria established by the API are used (that is, use all of the options described above in the match). If the user changes the FontSync Control Panel settings, that becomes the new default. This constant basically says to use whatever the user has set.

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

Discussion

The `FNSMatchOptions` enumeration defines masks your application can use to set or test match option bits. You can use this mask with the functions `FNSReferenceMatch` (page 31), `FNSProfileMatchReference` (page 15), `FNSReferenceMatchFonts` (page 33), and `FNSReferenceMatchFamilies` (page 32) to set the match options used during font comparison. If you wish, your application can specify options that are different from the user's preferences via this mask. You can use this mask to test the match option bits produced by the function `FNSMatchDefaultsGet` (page 7), thereby obtaining the default match options to use in a font comparison. You can also use this mask to test the match option bits produced by the function `FNSReferenceMatch`, thereby determining the match options under consideration that did not match.

At least one of the match options must be set. Having all of these bits clear is equivalent to saying "don't look at anything," which would allow any font to match. Since having all flags clear is nonsensical, the value of the mask constant `kFNSMatchDefaults` is 0. Setting undefined bits does not generate an error and provides backward compatibility.

Version Notes

Available beginning with FontSync 1.0.

Font Profile Constants

Represent the file type and default creator code of a font profile.

```
enum {
    kFNSCreatorDefault = 0,
    kFNSProfileFileType = 'fnsp'
};
```

Constants

`kFNSCreatorDefault`
: Assigns a file creator code instead of using one of your own. Pass this constant in the `iCreator` parameter of the function `FNSProfileCreate` (page 12).

  Available in Mac OS X v10.0 and later.

  Declared in `FontSync.h`.

`kFNSProfileFileType`
The file type of a profile. It is not used in the API, but is provided for convenience. For example, you can use it to put up a Navigation Services dialog box to select a profile.

Available in Mac OS X v10.0 and later.

Declared in `FontSync.h`.

Version Notes
Available beginning with FontSync 1.0. In FontSync 1.0, if you specify the constant `kFNSCreatorDefault` in the `iCreator` parameter of the function `FNSProfileCreate` (page 12), FontSync assigns the creator code `'fns'`.

Version Constants

Represents version information.

```
typedef UInt32 FNSObjectVersion;
enum {
    kFNSVersionDontCare = 0,
    kFNSCurSysInfoVersion = 1
};
```

Constants
`kFNSVersionDontCare`
Specifies the most recent font reference or font profile version supported by the FontSync library, regardless of version number. In FontSync 1.0, the most recent version for both font references and profiles is version 1. You pass this constant in the `iDesiredVersion` parameter of the functions `FNSSysInfo` (page 38), `FNSSysInfoGet` (page 35), and `FNSProfileCreate` (page 12).

Available in Mac OS X v10.0 and later.

Declared in `FontSync.h`.

`kFNSCurSysInfoVersion`
Identifies the current version of the parameter block structure `FNSSysInfo` (page 38) returned by the function `FNSSysInfoGet` (page 35). You pass this constant in the `iSysInfoVersion` field of the system information structure. The version of the structure used by FontSync 1.0 is version 1.

Available in Mac OS X v10.0 and later.

Declared in `FontSync.h`.

Discussion
You can pass the `kFNSVersionDontCare` constant in the `iDesiredVersion` parameter of the functions `FNSReferenceCreate` (page 21), `FNSReferenceCreateFromFamily` (page 22), and `FNSProfileCreate` (page 12), to specify the most recent font reference or font profile version supported by the FontSync library. You can use the `kFNSCurSysInfoVersion` constant in the `iSysInfoVersion` field of the structure `FNSSysInfo` (page 38) to indicate the current version of the structure.

Result Codes

The most common result codes returned by FontSync are listed below.

Result Code	Value	Description
`kFNSInvalidReferenceErr`	-29580	Returned by FontSync functions that operate on font references to indicate that the specified font reference is invalid. This can also be returned by the function `FNSReferenceUnflatten` if the reconstructed reference is bad. Available in Mac OS X v10.0 and later.
`kFNSBadReferenceVersionErr`	-29581	Returned by FontSync functions that operate on font references to indicate that the reference has an unsupported version number. This may indicate that the reference is valid, but created by a later version of FontSync, or that the reference is truly invalid This is also be returned by font reference creation functions if an unsupported reference version was requested. Available in Mac OS X v10.0 and later.
`kFNSInvalidProfileErr`	-29582	Returned by FontSync functions that operate on font profiles to indicate that the profile does not have a valid structure. Available in Mac OS X v10.0 and later.
`kFNSBadProfileVersionErr`	-29583	Returned by FontSync functions that operate on font profiles to indicate that the profile has an unsupported version number. This may indicate that the profile is valid, but created by a later version of FontSync, or that the profile is truly invalid. This is also be returned by font profile creation functions if an unsupported profile version was requested. Available in Mac OS X v10.0 and later.
`kFNSDuplicateReferenceErr`	-29584	Returned by the function `FNSProfileAddReference` to indicate that the font reference being added already exists in the profile. Available in Mac OS X v10.0 and later.
`kFNSMismatchErr`	-29585	Indicates that either a reference did not match, the reference you wish to remove is not in the profile, or that no font names were recorded in the reference. Available in Mac OS X v10.0 and later.
`kFNSInsufficientDataErr`	-29586	Returned by the functions `FNSReferenceMatch`, `FNSReferenceGetFamilyInfo`, `FNSReferenceFindName`, and `FNSReferenceGetIndName` to indicate that the mask constant `kFNSMissingDataNoMatch` has been set and both references being compared are missing the same data. Available in Mac OS X v10.0 and later.

Result Code	Value	Description
`kFNSBadFlattenedSizeErr`	-29587	Returned by the function `FNSReferenceUnflatten` to indicate that the specified size doesn't match the size recorded in the flattened reference or that the size was not large enough for a flattened reference. Available in Mac OS X v10.0 and later.
`kFNSNameNotFoundErr`	-29589	Returned by the function `FNSReferenceFindName` to indicate that there was no name in the font reference that matched the given parameters. Available in Mac OS X v10.0 and later.

Document Revision History

This table describes the changes to *FontSync Reference*.

Date	Notes
2003-06-16	Deleted a space from the title meta tag.
2002-10-23	Added clarification on the parameters in the function `FNSReferenceGetFamilyInfo`.
	Added documentation for the functions `FNSProfileOpenWithFSRef` and `FNSProfileCreateWithFSRef`.
	Fixed numerous typographical and formatting errors.

Index