CFStream Reference

Core Foundation > Networking

2007-05-03

Contents

CFStream Reference

Framework:	CoreFoundation/CoreFoundation.h
Declared in	CFFTPStream.h CFHTTPStream.h CFHost.h CFNetServices.h CFSocketStream.h CFStream.h
Companion guides	Getting Started with Networking CFNetwork Programming Guide

Overview

This document describes the generic `CFStream` functions, data types, and constants. See also `CFReadStreamRef` and `CFWriteStreamRef` for functions and constants specific to read and write streams respectively.

Functions

CFStreamCreateBoundPair

Creates a pair of read and write streams.

```
void CFStreamCreateBoundPair (
   CFAllocatorRef alloc,
   CFReadStreamRef *readStream,
   CFWriteStreamRef *writeStream,
   CFIndex transferBufferSize
);
```

Parameters

alloc

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

readStream

Upon return, a readable stream. Ownership follows the Create Rule.

writeStream

Upon return, a writable. Ownership follows the Create Rule.

transferBufferSize

The size of the buffer to use to transfer data from *readStream* to *writeStream*.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CFStream.h`

CFStreamCreatePairWithPeerSocketSignature

Creates readable and writable streams connected to a socket.

```
void CFStreamCreatePairWithPeerSocketSignature (
   CFAllocatorRef alloc,
   const CFSocketSignature *signature,
   CFReadStreamRef *readStream,
   CFWriteStreamRef *writeStream
);
```

Parameters

alloc

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

signature

A `CFSocketSignature` structure identifying the communication protocol and address to which the socket streams should connect.

readStream

On return, a readable stream connected to the socket address in *signature*. If you pass `NULL`, this function will not create a readable stream. Ownership follows the Create Rule.

writeStream

On return, a writable stream connected to the socket address in *signature*. If you pass `NULL`, this function will not create a writable stream. Ownership follows the Create Rule.

Discussion
The streams do not open a connection to the socket until one of the streams is opened.

Most properties are shared by both streams. Setting the property for one stream automatically sets the property for the other.

Availability
Available in Mac OS X v10.2 and later.

Declared In
`CFStream.h`

CFStreamCreatePairWithSocket

Creates readable and writable streams connected to a socket.

```
void CFStreamCreatePairWithSocket (
    CFAllocatorRef alloc,
    CFSocketNativeHandle sock,
    CFReadStreamRef *readStream,
    CFWriteStreamRef *writeStream
);
```

Parameters

alloc

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

sock

The pre-existing (and already connected) socket which the socket streams should use.

readStream

Upon return, a readable stream connected to the socket address in *signature*. If you pass `NULL`, this function will not create a readable stream. Ownership follows the Create Rule.

writeStream

Upon return, a writable stream connected to the socket address in *signature*. If you pass `NULL`, this function will not create a writable stream. Ownership follows the Create Rule.

Discussion

Most properties are shared by both streams. Setting the property for one stream automatically sets the property for the other.

Availability

Available in Mac OS X v10.1 and later.

Related Sample Code

CocoaEcho

CocoaHTTPServer

CocoaSOAP

Declared In

`CFStream.h`

CFStreamCreatePairWithSocketToHost

Creates readable and writable streams connected to a TCP/IP port of a particular host.

```
void CFStreamCreatePairWithSocketToHost (
   CFAllocatorRef alloc,
   CFStringRef host,
   UInt32 port,
   CFReadStreamRef *readStream,
   CFWriteStreamRef *writeStream
);
```

Parameters

alloc

The allocator to use to allocate memory for the `CFReadStream` and `CFWriteStream` objects. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

host
: The host name to which the socket streams should connect. The host can be specified using an IPv4 or IPv6 address or a fully qualified DNS host name.

port
: The TCP port number to which the socket streams should connect.

readStream
: Upon return, a readable stream connected to the socket address in *port*. If you pass `NULL`, this function will not create a readable stream. Ownership follows the Create Rule.

writeStream
: Upon return, a writable stream connected to the socket address in *port*. If you pass `NULL`, this function will not create a writable stream. Ownership follows the Create Rule.

Discussion

The streams do not open a connection to the specified host until one of the streams is opened.

Most properties are shared by both streams. Setting the property for one stream automatically sets the property for the other.

Availability

Available in Mac OS X v10.1 and later.

Declared In

`CFStream.h`

Data Types

CFStreamError

The structure returned by `CFReadStreamGetError` and `CFWriteStreamGetError`. (**Deprecated.** Use `CFReadStreamCopyError` and `CFWriteStreamCopyError` instead.)

```
typedef struct {
CFStreamErrorDomain domain;
SInt32 error
} CFStreamError;
```

Fields

`domain`
: The error domain that should be used to interpret the error. See `CFStream Error Domain Constants` (page 11) for possible values.

`error`
: The error code.

Availability

Available in Mac OS X v10.1 and later.

Declared In

`CFStream.h`

CFStreamClientContext

A structure provided when an application registers itself to receive stream-related events.

```
struct CFStreamClientContext {
    CFIndex version;
    void *info;
    void *(*retain)(void *info);
    void (*release)(void *info);
    CFStringRef (*copyDescription)(void *info);
} CFStreamClientContext;
```

Fields

`version`
: An integer of type `CFIndex`. Currently the only valid value is zero.

`info`
: A pointer to allocated memory containing user-defined data that will be valid for as long as the client is registered with the stream. You may assign `NULL` if your callback function doesn't want to receive user-defined data.

`retain`
: A pointer to a function callback that retains the data pointed to by the `info` field.You may set this function pointer to `NULL`.

`release`
: A pointer to a function callback that releases the data pointed to by the `info` field.You may set this function pointer to `NULL` but doing so might result in memory leaks.

`copyDescription`
: A pointer to a function callback that provides a description of the data pointed to by the `info` field. In implementing this function, return a reference to a `CFString` object that describes your allocator, particularly some characteristics of your user-defined data. You may set this function pointer to `NULL`, in which case Core Foundation will provide a rudimentary description.

Declared In

`CoreFoundation/CFStream.h`

Constants

CFStream Status Constants

Constants that describe the status of a stream.

```
typedef enum {
    kCFStreamStatusNotOpen = 0,
    kCFStreamStatusOpening,
    kCFStreamStatusOpen,
    kCFStreamStatusReading,
    kCFStreamStatusWriting,
    kCFStreamStatusAtEnd,
    kCFStreamStatusClosed,
    kCFStreamStatusError
} CFStreamStatus;
```

Constants

`kCFStreamStatusNotOpen`

The stream is not open for reading or writing.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamStatusOpening`

The stream is being opened for reading or for writing.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamStatusOpen`

The stream is open.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamStatusReading`

The stream is being read from.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamStatusWriting`

The stream is being written to.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamStatusAtEnd`

There is no more data to read, or no more data can be written.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamStatusClosed`

The stream is closed.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamStatusError`

An error occurred on the stream.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

Discussion
The `CFStreamStatus` enumeration defines constants that describe the status of a stream. These values are returned by `CFReadStreamGetStatus` and `CFWriteStreamGetStatus`.

Declared In
`CoreFoundation/CFStream.h`

CFStream Error Domain Constants

Defines constants for values returned in the domain field of the `CFStreamError` structure. (**Deprecated.** These constants are returned by `CFReadStreamGetError` and `CFWriteStreamGetError`; use `CFReadStreamCopyError` and `CFWriteStreamCopyError` instead.)

```
typedef enum {
    kCFStreamErrorDomainCustom = -1,
    kCFStreamErrorDomainPOSIX = 1,
    kCFStreamErrorDomainMacOSStatus,
} CFStreamErrorDomain;
```

Constants

`kCFStreamErrorDomainCustom`

The error code is a custom error code.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamErrorDomainPOSIX`

The error code is an error code defined in `errno.h`.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamErrorDomainMacOSStatus`

The error is an OSStatus value defined in `MacErrors.h`.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

Discussion
These constants indicate how the error code in the `error` field in the `CFStreamError` (page 8) structure should be interpreted.

Declared In
`CoreFoundation/CFStream.h`

CFStream Error Domain Constants (CFHost)

Defines constants for values returned in the domain field of the `CFStreamError` structure.

```
const SInt32 kCFStreamErrorDomainNetDB;
const SInt32 kCFStreamErrorDomainNetServices;
const SInt32 kCFstreamErrorDomainMach;
const SInt32 kCFStreamErrorDomainFTP;
const SInt32 kCFStreamErrorDomainHTTP;
const int kCFStreamErrorDomainSOCKS;
const SInt32 kCFStreamErrorDomainSystemConfiguration;
const int kCFStreamErrorDomainSSL;
```

Constants

`kCFStreamErrorDomainNetDB`

The error code is an error code defined in `netdb.h`.

Available in Mac OS X v10.3 and later.

Declared in `CFHost.h`.

`kCFStreamErrorDomainNetServices`

The error code is a `CFNetService` error code. For details, see the `CFNetService Error Constants` enumeration.

Available in Mac OS X v10.2 and later.

Declared in `CFNetServices.h`.

`kCFStreamErrorDomainMach`

The error code is a Mach error code defined in `mach/error.h`.

Available in Mac OS X v10.2 and later.

Declared in `CFNetServices.h`.

`kCFStreamErrorDomainFTP`

The error code is an FTP error code.

Available in Mac OS X v10.3 and later.

Declared in `CFFTPStream.h`.

`kCFStreamErrorDomainHTTP`

The error code is an HTTP error code.

Available in Mac OS X v10.1 and later.

Declared in `CFHTTPStream.h`.

`kCFStreamErrorDomainSOCKS`

The error code is a SOCKS proxy error.

Available in Mac OS X v10.2 and later.

Declared in `CFSocketStream.h`.

`kCFStreamErrorDomainSystemConfiguration`

The error code is a system configuration error code as defined in `System/ConfigurationSystemConfiguration.h`.

Available in Mac OS X v10.3 and later.

Declared in `CFHost.h`.

`kCFStreamErrorDomainSSL`

The error code is an SSL error code as defined in `Security/SecureTransport.h`.

Available in Mac OS X v10.1 and later.

Declared in `CFSocketStream.h`.

Discussion

These constants indicate how the error code in the `error` field in the `CFStreamError` (page 8) structure should be interpreted.

Availability

Available in Mac OS X version 10.0 and later.

Declared In

`CFNetwork.framework/CFHost.h`

CFStream Event Type Constants

Defines constants for stream-related events.

```
typedef enum {
kCFStreamEventNone = 0,
kCFStreamEventOpenCompleted = 1,
kCFStreamEventHasBytesAvailable = 2,
kCFStreamEventCanAcceptBytes = 4,
kCFStreamEventErrorOccurred = 8,
kCFStreamEventEndEncountered = 16
} CFStreamEventType;
```

Constants

`kCFStreamEventNone`

No event has occurred.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamEventOpenCompleted`

The open has completed successfully.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamEventHasBytesAvailable`

The stream has bytes to be read.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamEventCanAcceptBytes`

The stream can accept bytes for writing.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamEventErrorOccurred`

An error has occurred on the stream.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamEventEndEncountered`

The end of the stream has been reached.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

Discussion
This enumeration defines constants for stream-related events.

Availability
Available in Mac OS X version 10.0 and later.

Declared In
`CoreFoundation/CFStream.h`

Stream Properties

Stream property names that can be set or copied.

```
const CFStringRef kCFStreamPropertyAppendToFile;
const CFStringRef kCFStreamPropertyFileCurrentOffset;
const CFStringRef kCFStreamPropertyDataWritten;
const CFStringRef kCFStreamPropertySocketNativeHandle;
const CFStringRef kCFStreamPropertySocketRemoteHostName;
const CFStringRef kCFStreamPropertySocketRemotePortNumber;
```

Constants

`kCFStreamPropertyDataWritten`

Value is a `CFData` object that contains all the bytes written to a writable memory stream. You cannot modify this value.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamPropertyAppendToFile`

Value is a `CFBoolean` value that indicates whether to append the written data to a file, if it already exists, rather than to replace its contents.

You must set this value before opening the writable file stream. The default value is `kCFBooleanFalse`, indicating that the stream should replace any pre-existing file. You cannot read this value.

Declared in `CFStream.h`.

Available in Mac OS X version 10.2 and later.

`kCFStreamPropertyFileCurrentOffset`

Value is a `CFNumber` object containing the current file offset.

Available in Mac OS X v10.3 and later.

Declared in `CFStream.h`.

`kCFStreamPropertySocketNativeHandle`

Value is a `CFData` object that contains the native handle for a socket stream—of type `CFSocketNativeHandle`—to which the socket stream is connected.

This property is only available for socket streams. You cannot modify this value. You can read this value at any time.

Available in Mac OS X v10.1 and later.

Declared in `CFStream.h`.

`kCFStreamPropertySocketRemoteHostName`
: Value is a `CFString` object containing the name of the host to which the socket stream is connected or `NULL` if unknown.

  You cannot modify this value. You can read this value at any time.]

  Available in Mac OS X v10.1 and later.

  Declared in `CFStream.h`.

`kCFStreamPropertySocketRemotePortNumber`
: Value is a `CFNumber` object containing the remote port number to which the socket stream is connected or `NULL` if unknown.

  You cannot modify this value. You can read this value at any time.]

  Available in Mac OS X v10.1 and later.

  Declared in `CFStream.h`.

Discussion
Use `CFReadStreamCopyProperty` or `CFWriteStreamCopyProperty` to read the property values. Use `CFReadStreamSetProperty` or `CFWriteStreamSetProperty` to set the property values.

Availability
Available in Mac OS X v10.2 and later.

Declared In
`CoreFoundation/CFStream.h`

Document Revision History

This table describes the changes to *CFStream Reference*.

Date	Notes
2007-05-03	Updated to include API introduced in Mac OS X v10.5.
2006-01-10	First version of this document. Incorporates information from CFStreamRef and CFNetwork Services Programming Guide.

Index