FileStream Class

An extension of the **StreamInterface** class that enables asynchronous reading and writing from/to a file on disk. This class disables the copy constructor. You must call **Open** to start reading or writing the file. The methods shown are not thread-safe, and can be called on any thread.

Syntax

```
class FileStream : public StreamInterface
```

Methods

Name	Description
void **Close**()	Closes the file. This will not send an SE_CLOSE error, because it is a deliberate action. You can call **Open** to open another file.
bool **DisableBuffering**()	By default, reading and writing are buffered. Calling this method blocks all write attempts to block until the disk bytes are updated.
FileStream()	Constructor.
~FileStream	Destructor. Calls **Close** to close the file, if it has not been explicitly closed already.
bool **GetPosition**(size_t **position*) const	Retrieves the current read/write position in the file. Returns True if the call succeeded, False otherwise.
bool **GetSize**(size_t **size*) const	Retrieves the size of the file. Returns True if the call succeeded, False otherwise.
StreamState **GetState**()	Returns the current state of the file.
bool **Open**(const char **filename*, const char **mode*)	Creates or opens the file named *filename*. Values for *mode* are the same values supported by stdio::fopen.
bool **OpenShare**(const std::string& *filename*, const char* *mode*, int *shflag*)	Creates or opens the file named *filename* with the sharing mode specified by *shflag*.
StreamResult **Read**(void **buffer*, size_t *buffer_len*, size_t **read*, int **error*)	Reads a specified number of bytes from the stream. Does not return the number of bytes read if any error occurs. • *buffer* A caller-allocated buffer to hold the data to read. • *buffer_len* The size, in bytes, of *buffer*. • *read* The number of bytes written to *buffer*. • *error* Details about any error that occurred. Error values are defined in talk/base/socket.h.
bool **Rewind**()	Resets the current read/write position to the start of the stream. Returns

	True if successful, False otherwise.
bool **SetPosition**(size_t *position*)	Tries to move the pointer to the position *position* in the file. Returns True if successful, False otherwise.
StreamResult Write(const void **data*, size_t *data_len*, size_t **written*, int **error*)	Writes a specified number of bytes to the stream. Does not return the number of bytes written if any error occurs. • *data* The data to write. • *data_len* The size, in bytes, of *data*. • *written* The number of bytes written to the stream. • *error* Details about any error that occurred. Error values are defined in talk/base/socket.h

Attributes: public

Declaration file: talk/base/stream.h

Last updated March 23, 2012.