Frequently Asked Questions

1. General

1.1. What is this project about, in general?

Suppose you want to create an application which reads and writes files over the Internet. The remote file system is in a certain host, while your application resides in a local host, separated by a network. Your goal is to make it possible for your local application to be able to read and write data in a file system at a remote site.

1.2. What basic components are to be made? What are their requirements in brief?

Assuming the network runs TCP/IP (which is usually the case), you can achieve your goal by making the following components:

1. A file server (which directly reads and writes data to files, based on commands sent over the network). This component can use standard Java APIs for file operations and for network communication, and, once run, will always be listening to a request at a certain port on the host in which it is running.
2. A client-side API that accepts requests from a client, and translates them into commands sent over a network to the file server, receives response from the file server and returns it to the client. This part may also involve a caching mechanism. This is realized as an API which you implement as a Java class which is imported by a client in order to use the file server.

The main thrust of the project is for you to realize these two components so that they can offer basic file operations over the Internet, however far away a remote file resides.

We ask for the file server to be STATELESS. In stage 2, the client-side API should incorporate cache mechanisms. For details, see the project specification.

2. Read and Write Operations.

2.1. How does a pointer move when reading/writing?

It simply advances by the number of bytes you have read/written.

2.2. Can a client do read and write to the same file handle?

Yes. The program FileSystemTest.java actually reads from and writes to the same file.

2.3. The *read/write* for the Client-side API and the one exported by the server differ. Why?

The *read/write* in the client-side API is stateful - it just reads/writes a given number of bytes from the current file pointer. The read/write exported by the server is stateless - it reads/writes a given number of bytes from a given offest in the file.

3. Testing.

3.1. Should I definitely use the provided test programs and data files?

You can use your own test programs and data, but you should implement the same API for a client as specified in *FileSystemAPI*. The bottom line is that your API should work with any client which uses it, with any data provided.