File Input & Output

Dylan Lane McDonald

CNM STEMulus Center
Web Development with PHP

November 11, 2015

Outline

Streams

Definition

A **stream** *is a mechanism for making data available over time.*

Streams can be likened to the counterparts in the natural world in the sense that both computer science and geophysical streams consist of:

- **Source**: where the material originates
- **Direction**: deterministic path the material must follow
- **Destination**: where the material will end up

In computer science all file and "file-like" operations are generalized as streams. Examples of streams include:

- **File Operations**: reading and writing local files
- **HTTP Requests**: connecting to and communicating with web servers

Examples of Streams

Table 1 compares different streams from both the computer science and geophysical world.

	Source	**Direction**	**Destination**
Río Grande	San Juan Mountains, CO	south	Cameron County, TX
HTTP Connection	`https://www.cnm.edu/`	read-only	web browser
Saving a File	`local-file.txt`	read/write	hard disk

Table 1: Examples of Streams

Note the "direction" of a computer science stream simply denotes whether the stream is readable, writable, or both.

PHP Stream Contexts

PHP stream contexts modify and enhance the stream opened in PHP. Contexts are created using the `stream_context_create()` and then can be added to any stream.

```
$options = array();
$options["http"] = array();
$options["http"]["method"] = "GET";
$options["http"]["header"] = "Cookie: foo=bar\r\n";
$context = stream_context_create($options);
$fd = fopen("http://www.cnm.edu/", "rb", false, $context);
fpassthru($fd);
fclose($fd);
```

Listing 1: Using Context to Create an HTTP Stream

Listing 1 uses context to create a HTTP stream.

General Considerations

Streams are external entities and **always** subject to failure. When opening a stream of any kind, **always** verify the stream was opened successfully. For instance, when opening a stream with `fopen()`, always compare to `false` to detected a failed stream before using it.

```
$fd = fopen("http://www.cnm.edu/", "rb", false, $context);
if($fd === false) {
  throw(new RuntimeException("Unable to open HTTP stream"));
}
```

Listing 2: Verifying a Stream

Listing 2 takes the `fopen()` from Listing 1 and throws an exception when it fails.

fopen() Modes

By far, the most common function to create a stream is `fopen()`. `fopen()`'s second parameter is the **mode** in which the stream is opened it.

Parameter	Mode	Comment	Create?
r	read only	start at the beginning	No
r+	read/write	start at the beginning	No
w	write only	overwrite the stream	Yes
w+	read/write	overwrite the stream	Yes
a	write only	start at the end of the stream	Yes
a+	read/write	start at the end of the stream	Yes
x	write only	fail if the file already exists	Yes
x+	read/write	fail if the file already exists	Yes

Table 2: Partial List of `fopen()` Modes

ASCII vs Binary Mode

`fopen()` and many other stream functions initialize in either ASCII or binary mode. ASCII mode translates newline characters from Windows (`\r\n`) to Unix (`\n`), or vice versa. This has the net effect of transparently modifying files in your program. **This is rarely what you want.**

`fopen()`'s default mode is PHP version/operating system/systems administrator/planet alignment dependent. **Never assume which mode you're in. Always specify!** The PHP documentation explicitly says, "For portability, it is strongly recommended that you always use the 'b' flag when opening files with `fopen()`." [1] More in-depth discussion of this topic can be found at the C FAQ. [2]

File System

In addition to the general considerations, the permissions of the Apache web server must also be considered. The Apache web server is typically, and by default, configured as an underprivileged user that can't access very many system resources.

In effect, all file operations run by PHP scripts are run by the Apache web server. Often, collaboration with the systems administrator will be necessary to grant the Apache web server access rights to the area(s) of the file system that need modification. While granting access to the Apache web server has very valid use cases, this opens up a great security hazard and must be handled with extreme caution both on the part of the systems administrator and web developer.

Network

Accessing a network stream presents three major concerns:

- **Reliability**: network connections are by their very nature slow and subject to failure. Even when network operations succeed, they are very slow. [3, 4]
- **Fragmentation**: network operations are conducted in **packets**, and sometimes packets are not transmitted in order or not at all.[1] Stream functions can handle fragmentation, but if and only if the developer properly accounts for fragmentation while using a network stream.
- **Security**: using external network data presents an element of risk. Assess carefully as to whether this data is necessary. If it is necessary, assess the trustworthiness of the network stream and **always** sanitize inputs derived from networked sources.

With these considerations in mind, a developer can effectively use network streams to enhance the web site's features, interactivity, and usefulness.

[1] This is known as **packet loss**.

Works Cited I


The PHP Group.
Php: fopen - manual.
http://php.net/manual/en/function.fopen.php, 2015.


Steve Summit.
comp.lang.c faq list question 12.40.
http://c-faq.com/stdio/textvsbinary.html, 2005.


Colin Scot.
Latency numbers every programmer should know.
http://www.eecs.berkeley.edu/~rcs/research/interactive_latency.html, 2012.

Works Cited II


P Stark.
Latency numbers every programmer should know.
https://gist.github.com/hellerbarde/2843375, 2012.