Redirecting output can result in altered program behavior

devblogs.microsoft.com/oldnewthing/20060519-09

May 19, 2006

Raymond Chen

Consider a program whose output to the console goes like this. (I've prefixed each line with the output stream.)

```
stdout: line 1
stderr: err 1
stdout: line 2
stderr: err 2
```

You want to capture both the normal and error streams, so you run the program and append "`>output 2>&1`" to capture both streams into a single file. But when you look at the resulting output file, you get this:

```
line 1
line 2
err 1
err 2
```

What happened?

Most programs change their output behavior depending on whether the output stream is a file or a device. If the output stream is a device (such as the screen), then buffering is disabled and every print statement goes to the screen immediately. On the other hand, if the output stream is a file, then buffering is enabled and print statements do not go directly into the file but rather into an application-managed memory buffer. When the buffer fills, it is written to the file, and then the buffer is emptied so it can accept more output.

This explains the behavior we see above. The program generates its output to both stdout and stderr, and the results are buffered. When the program exits, the buffers are flushed, first stdout and then stderr. That's why you see all the stdout output grouped together and all the stderr output grouped together.

"But I don't do any of this in my programs; why is it happening anyway?"

If you use the C runtime for your output, then your program does behave this way whether you realize it or not. The default behavior of the C runtimes is to perform unbuffered I/O for devices and buffered I/O for files. You can override this behavior in your own programs by calling `setvbuf` to force buffering on or off for a file handle.

(A classmate in college became legendary for fixing a bug in one of the programs used in the VLSI class, all of which were notoriously buggy. He patched the binary to disable buffered I/O.)

Raymond Chen

Follow