Error Redirection

Type this command (exactly) in the shell, and press ENTER:

```
$ cat > output.txt < input.text
```

In this case, there **is no file** named `input.text`, so `output.txt` is **erased**. The `cat` filter prints an error message on the **standard error stream**, still connected to the screen. **Output redirection only redirects standard output, not standard error.**

You can the **redirect standard error** stream by using the symbol `2>` like this:

```
$ cat > output.txt 2> err.txt < input.text
$ cat < output.txt
$ cat < err.txt
bash: input.text: No such file or directory
$
```

Now `output.txt` is still empty, but `err.txt` contains the errors that originally appeared on the screen. **Combine both** into a single stream (which may be sent to a file) like this.

```
$ cat > combined.txt 2>&1 < input.text
```

Sometimes, you **don't want to see** either the error messages or any progress reports. For instance, if you try to remove a file which doesn't exist, the shell displays an error message like this:

```
$ rm filter.exe
rm: cannot remove 'filter.exe': No such file or directory
```

Instead of redirecting those messages to a file, you can send them to the **"bit bucket"** which has the name, (in Unix), `/dev/null`. (If you are using redirection on Windows, the name is `NUL:` with the trailing colon.) Anything redirected to `/dev/null` just disappears.