**Creating a changelog and tarball for a software release**

The [git-archive[1]](https://git-scm.com/docs/git-archive) command can create a tar or zip archive from any version of a project; for example:

$ git archive -o latest.tar.gz --prefix=project/ HEAD

will use HEAD to produce a gzipped tar archive in which each filename is preceded by project/. The output file format is inferred from the output file extension if possible, see [git-archive[1]](https://git-scm.com/docs/git-archive) for details.

Versions of Git older than 1.7.7 don’t know about the tar.gz format, you’ll need to use gzip explicitly:

$ git archive --format=tar --prefix=project/ HEAD | gzip >latest.tar.gz

If you’re releasing a new version of a software project, you may want to simultaneously make a changelog to include in the release announcement.

Linus Torvalds, for example, makes new kernel releases by tagging them, then running:

$ release-script 2.6.12 2.6.13-rc6 2.6.13-rc7

where release-script is a shell script that looks like:

#!/bin/sh

stable="$1"

last="$2"

new="$3"

echo "# git tag v$new"

echo "git archive --prefix=linux-$new/ v$new | gzip -9 > ../linux-$new.tar.gz"

echo "git diff v$stable v$new | gzip -9 > ../patch-$new.gz"

echo "git log --no-merges v$new ^v$last > ../ChangeLog-$new"

echo "git shortlog --no-merges v$new ^v$last > ../ShortLog"

echo "git diff --stat --summary -M v$last v$new > ../diffstat-$new"

and then he just cut-and-pastes the output commands after verifying that they look OK.

**Finding commits referencing a file with given content**

Somebody hands you a copy of a file, and asks which commits modified a file such that it contained the given content either before or after the commit. You can find out with this:

$ git log --raw --abbrev=40 --pretty=oneline |

grep -B 1 `git hash-object filename`

Figuring out why this works is left as an exercise to the (advanced) student. The [git-log[1]](https://git-scm.com/docs/git-log), [git-diff-tree[1]](https://git-scm.com/docs/git-diff-tree), and [git-hash-object[1]](https://git-scm.com/docs/git-hash-object) man pages may prove helpful.

Developing with Git

**Telling Git your name**

Before creating any commits, you should introduce yourself to Git. The easiest way to do so is to use [git-config[1]](https://git-scm.com/docs/git-config):