# Git

Git version control quick reference.

## git send-email

Setup email server:

```
[sendemail]
    smtpserver = mail.example.org
    smtpuser = you@example.org
    smtpencryption = ssl
    smtpserverport = 465
```

```
git config --global user.email "you@example.org"
git config --global user.name "Your Name"
```

Make some changes, then commit

```
echo 'hi' > stuff.txt
git add stuff.txt
git commit -m "my cool commit"
```

Send the Patch:

```
git send-email --to="dev@example.org" HEAD^
```

On getting feedback, amend commit, then send (`--annotate` edits the email before sending):

```
git commit -a --amend
git send-email --to"dev@example.org" --annotate -v2 HEAD^
```

When annotating an email, add prose under the `---` for details that *should not* be included in the git log.

```
Subject: [PATCH v2] erm what the σ + jω

---
oh my god its 's'!!!

your-name | 1 +
1 file changed, 1 insertion(+)
```

### Patches with Cover Letter

Not all changes are straightforward and self-contained. When a change requires several commits, a cover letter can be added as the starting point of the email thread.

```
git send-email --cover-letter origin/master
```

This will generate an extra email template with placeholders for the subject and main body:

```
Subject: [PATCH 0/3] *** SUBJECT HERE ***

*** BLURB HERE ***

your-name (3):
  core: Move reusable foo logic to a new function
  core: Add new flag to the aforementioned foo()
  cmd: New foo subcommand similar to bar

 core.c  |  38 ++++++++------
 cmd.c   |  13 ++++
 2 files changed, 34 insertions(+), 17 deletions(-)

--
```

In the example above, the subject could become:

```
Subject: [PATCH 0/3] Derive a foo subcommand from bar
```

The cover letter could then explain why a new subcommand is needed, and why it cannot or should not be implemented as an option to the existing subcommand that shares some aspects. A cover letter offers a convenient place to provide the rationale behind a change.

Since the default template includes a short log, it can also be useful to add a summary of the structure of a patch series. It could for example start with refactoring commits to prepare the actual change, and that change could also span multiple commits. A complicated change is likely easier to implement and review when decomposed into smaller logical chunks with a clear progression.

Other relevant information that does not necessarily belong in a commit log can be added to a cover letter, such as benchmark results, links to discussions on other forums or literature on the topic. As such, even a single patch submission may benefit from a cover letter.

# Settings of interest

```
git config format.subjectPrefix "PATCH foobar"
git config --global sendemail.annotate yes
git config format.signOff yes
git config sendemail.to "dev@example.org"
```

# Push local branch to named remote branch

Pass in `-u` to set the origin default.

```
git push <remote> <local branch>:<remote branch>
```