Version control with Git
(or: get distributed or svn will eat your face)

Andy Newton
Pat McSweeney
March-ish 2014
Version control overview

- Keeps a history of your code (or other files)
- Useful as a backup
- Track who did what and why
- Collaboration: work together and merge changes!
Concepts

- **Version** – A snapshot of your code at a point in time
- **History** – ability to view and check out old versions
- **Branch** – ability to create multiple versions based on the same starting point
- **Merge** – converge your branches
- **Tag** – A pointer to a specific version e.g. current release
- **Diff** – Show differences between versions
- **Distribution** – make code available to other users or to a system
Traditional VC with SVN/CVS/...

Repository (SVN server)

svn commit: Create version; Distribute version in one go

svn checkout: Bob gets a copy of the code (distribute to Bob)

svn update: Get others' changes (distribute to Alice)

Alice's Working Copy

Bob's Working Copy

Alice

Bob
Distributed VC: Git/hg/bzr/...

- **git clone**: Bob gets a copy of the repository
- **git push**: Distribute versions to remote repo
- **git pull**: Distribute versions to Alice
- **git commit**: Create version in Alice's repo
- **git commit**: Create version in Bob's repo

Other repositories (e.g. Git repo on shared server)

Alice's local repo

Bob's local repo

Can pull between Alice and Bob's repositories at will!
What to (not) check in

- Any text content! (code, documentation, your latest novel, …)
- Not binary files (Visual Basic beware!) - use a .gitignore file to automate ignoring files
- Images etc. are OK, but binaries that change a lot are a bad idea e.g. word docs
- No site-specific config (especially passwords, especially if checking into a public repo!) - makes it harder to pull in changes
- No confidential or personal information! (yeah, let's not put our bank account details on github)
Branching and merging
while(1){branch();}
Demo time!

Grab laptops if you got 'em...
SourceKettle and GitHub

- SourceKettle – agile project management tool, available to all University members, uses git
  - https://sourcekettle.ecs.soton.ac.uk/
- GitHub – public repository for open source projects, more comprehensive features, need to pay for private repositories
  - https://github.com/
Demo time!

You know the drill...
Advanced branching: git-flow

http://nvie.com/posts/a-successful-git-branching-model/
iSolutions projects?

- Develop
  - Pull latest for testing
  - Bug fixed
  - Pull latest for testing
  - Pull fix for testing
- Preprod
  - Pull new release
  - Bug found during testing
- Production
  - Pull new release
  - Pull latest for testing