1495 UNIVERSITY OF ABERDEEN

JC2002 Java Programming

Lecture 3: Version control

References for this session

- Evans, B. and Flanagan, D., 2018. *Java in a Nutshell: A Desktop Quick Reference 7th edition.* O'Reilly Media.
- Deitel, H., 2018. *Java How to Program, Early Objects, Global Edition, 11th Edition.* Pearson.
- Ben Lynn, Git Magic, 2007, Available online: http://www-cs-students.stanford.edu/~blynn/gitmagic/book.html
- Tom Preston-Werner, The Git Parable, 2009, Available online: https://tom.preston-werner.com/2009/05/19/the-git-parable.html
- Johan Herald, NDC TechTown, 2008, Available online: https://docs.google.com/presentation/d/1u0cM0r07iL9v7Myo6RWGWRR3o2IYlebDIayG8rMagVw/edit#slide=id.g3bcc4c1a94_0_6

JAR files

- Compiled JAVA programs (i.e., the `.class` files) can be packed into `.jar` files
- Third party libraries are often distributed as `.jar` files (via Maven)
  - We will look at Maven later when we start automating management of project dependencies
- You can also pack your own application as executable `.jar`
  - Running .jar files: `java -jar example.jar`
- More info on using `.jar` files: https://docs.oracle.com/javase/tutorial/deployment/jar/basicsindex.html

Version control

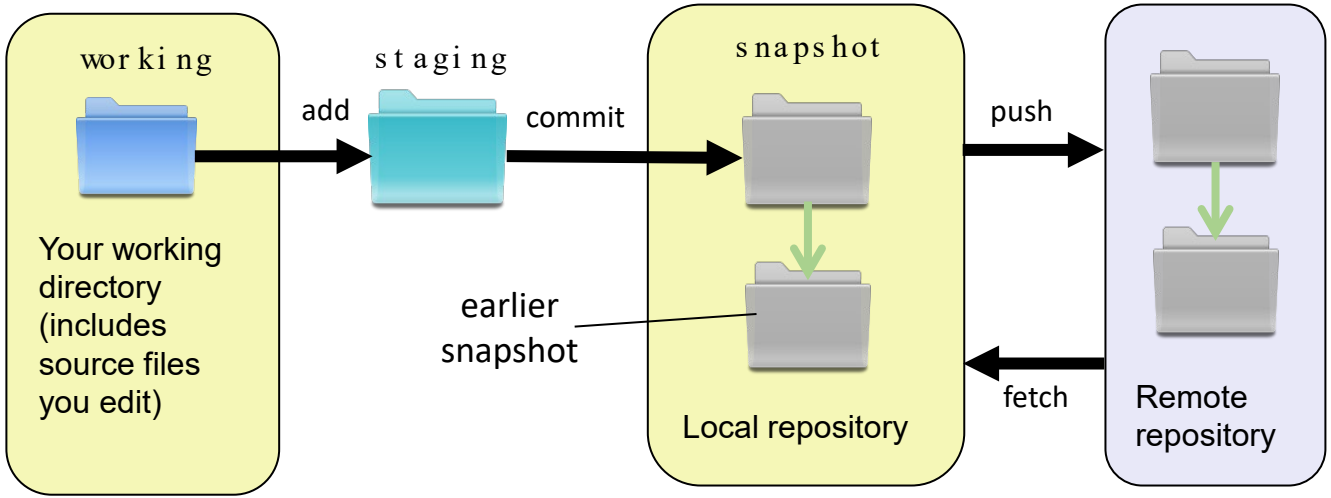
- Version control provides an undo for us so we can make changes with a safety net
- There are many types of version control:
  - Local (e.g., RCS)
  - Server based (e.g., Subversion)
  - Distributed (e.g., Git)
- On this course, we focus on Git

What is Git?

- An open-source distributed version control system
  - Original author Linus Torvalds
  - Created for the development of Linux Kernel
  - Most popular version control system today
  - Supported by many popular service providers such as *github.com*, *bitbucket.com*, etc.
- Git provides means for both:
  - A global “what if”
  - A global “undo” in our projects

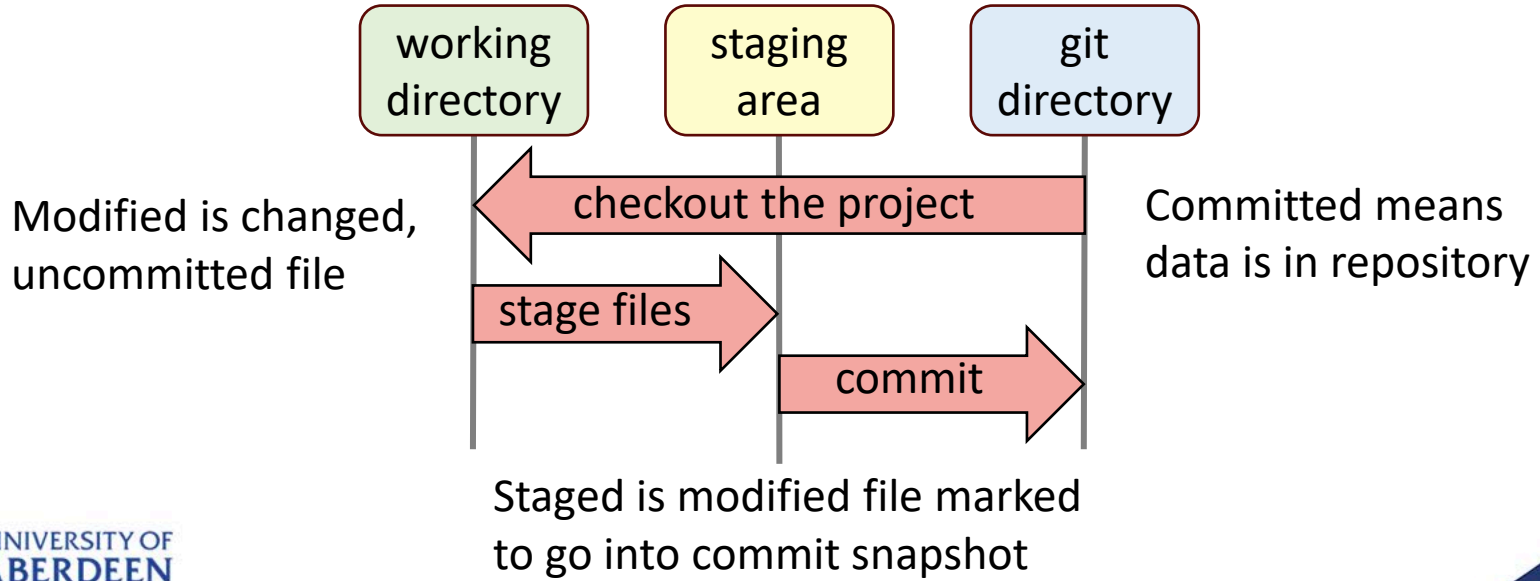
Git snapshots

- In Git, snapshot is a record of the state of your project files at a specific point in time



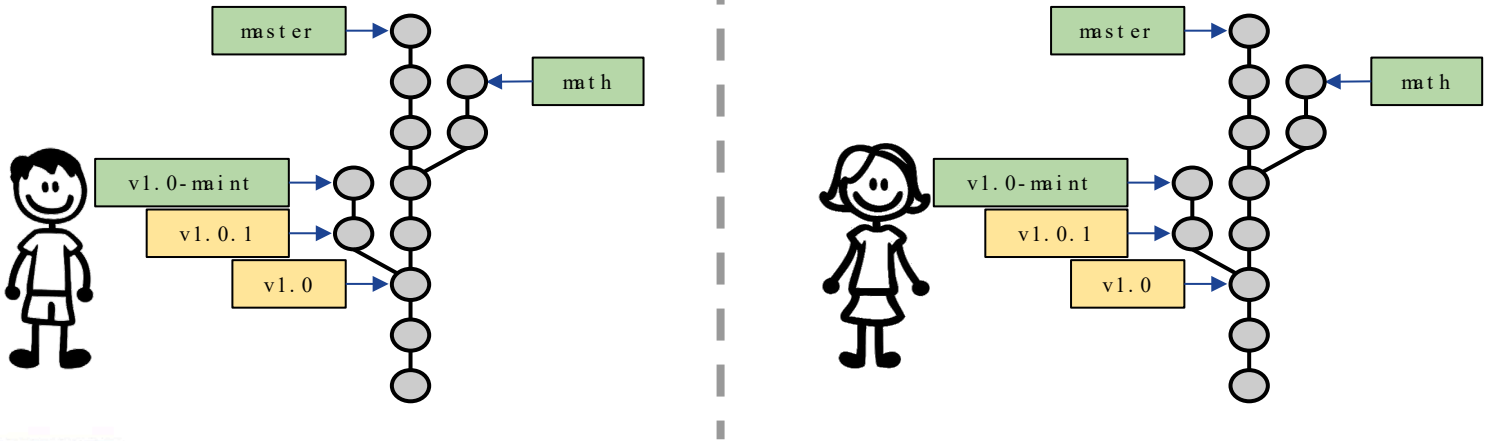
Git areas of operation

- Git has three areas of operation: working directory, staging area, and git directory (repository)



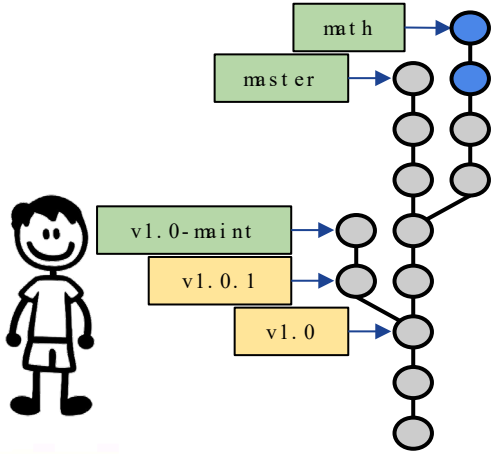
Snapshots and branches

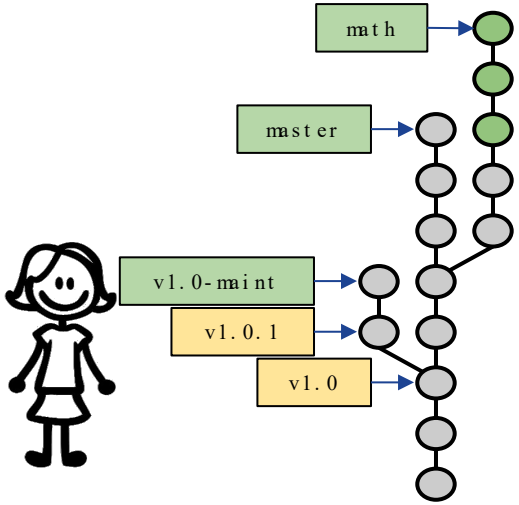
- Bob and Alice start with the same local directory



Merges

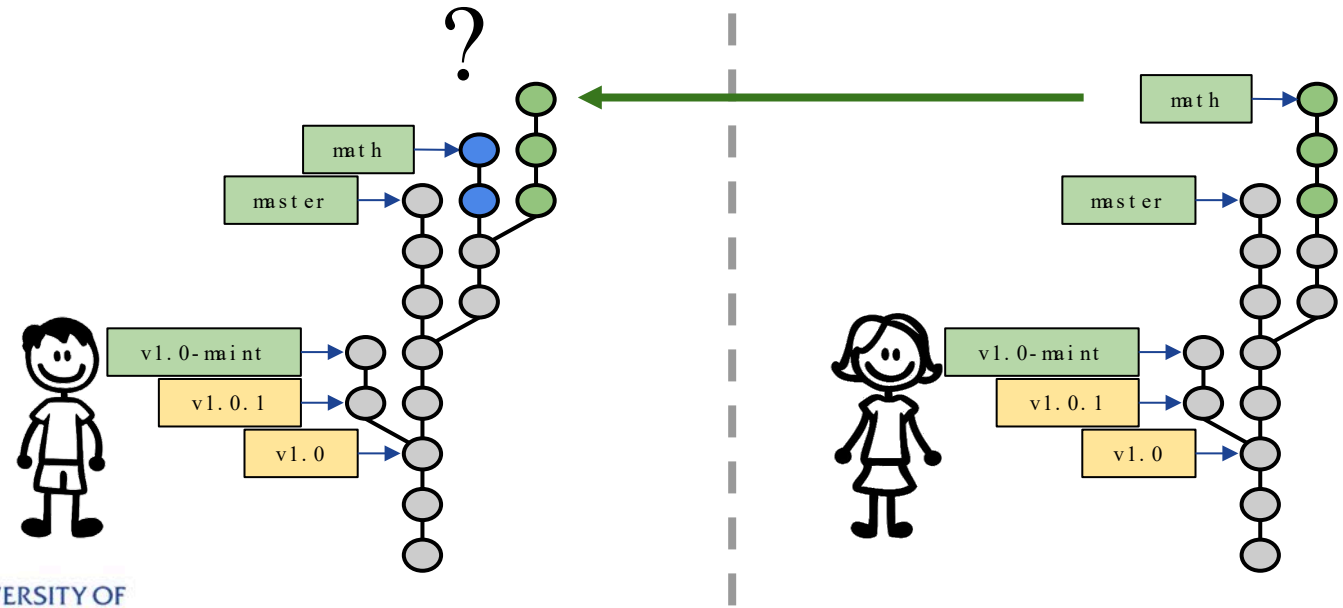
- Bob and Alice make local changes



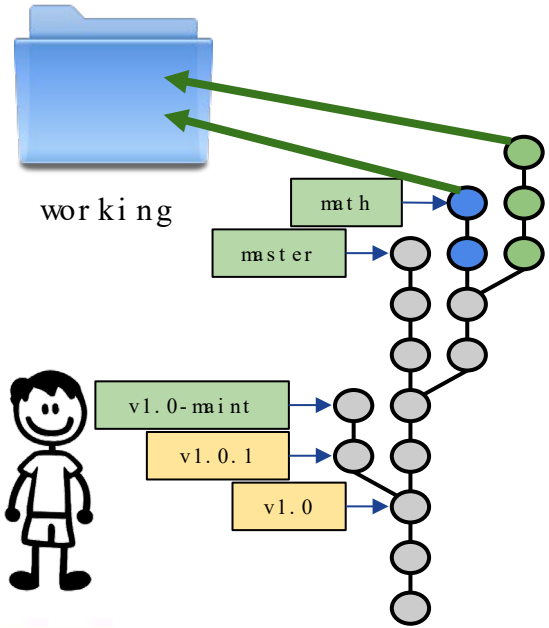


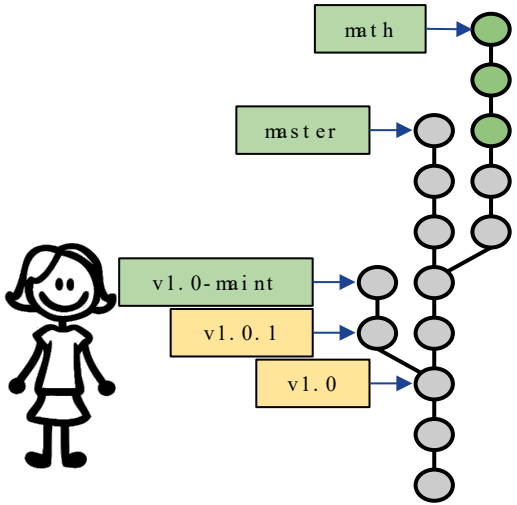
Merges

- How to merge the local changes by Bob and Alice?

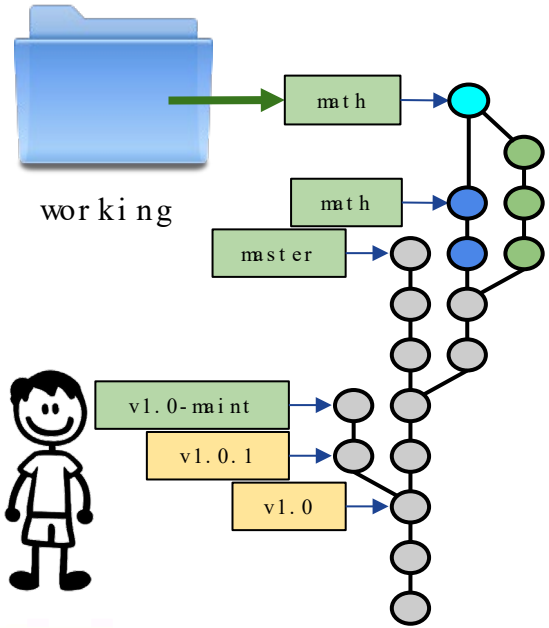


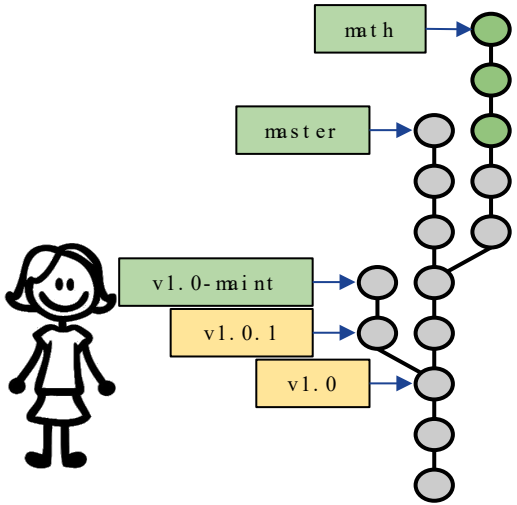
Merges


working
math
master
v1.0-maint
v1.0.1
v1.0

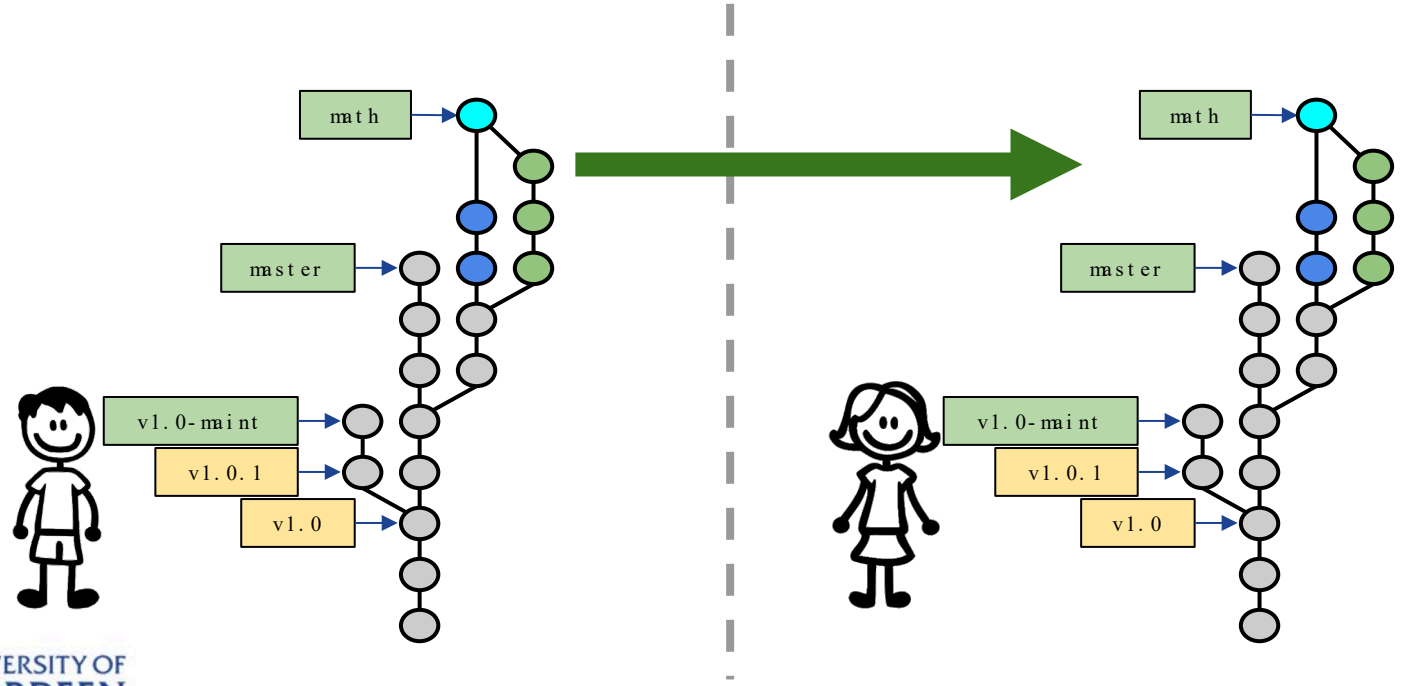

math
master
v1.0-maint
v1.0.1
v1.0

Merges

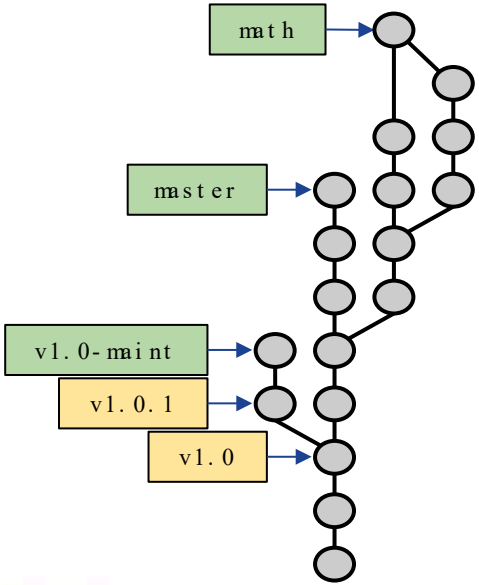

working
math
math
master
v1.0-maint
v1.0.1
v1.0

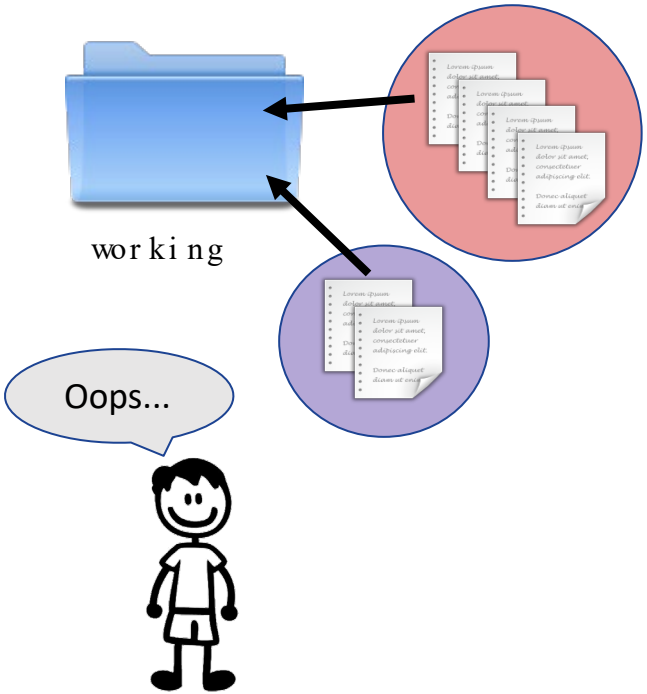

math
master
v1.0-maint
v1.0.1
v1.0

Merges

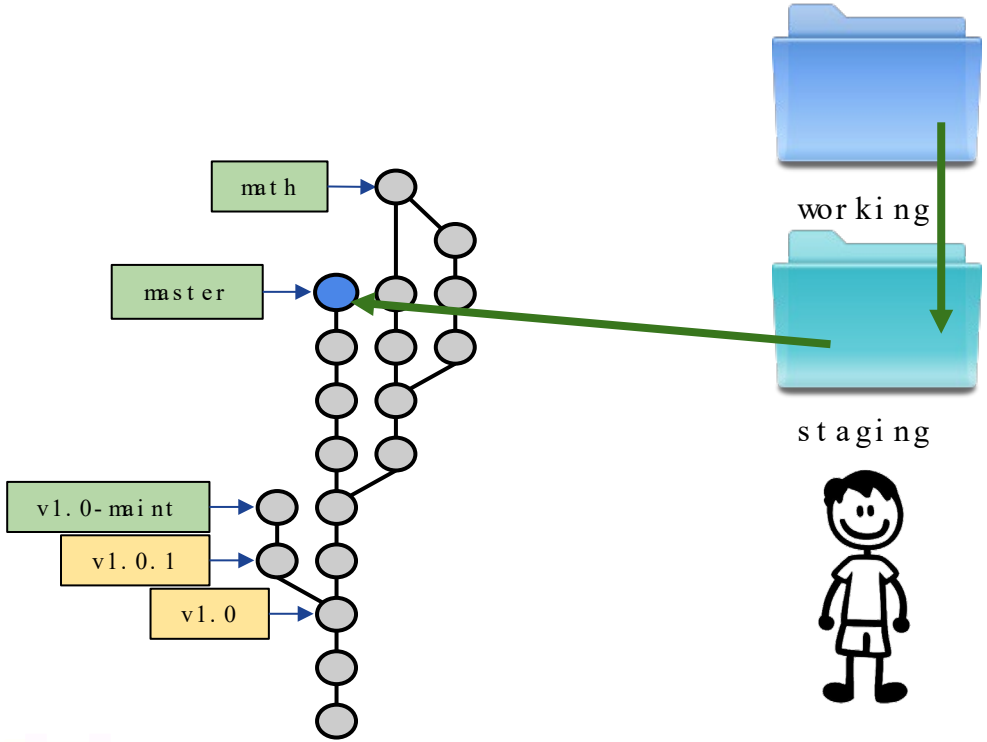

math
master
v1.0-maint
v1.0.1
v1.0
math
master
v1.0-maint
v1.0.1
v1.0

Staging area

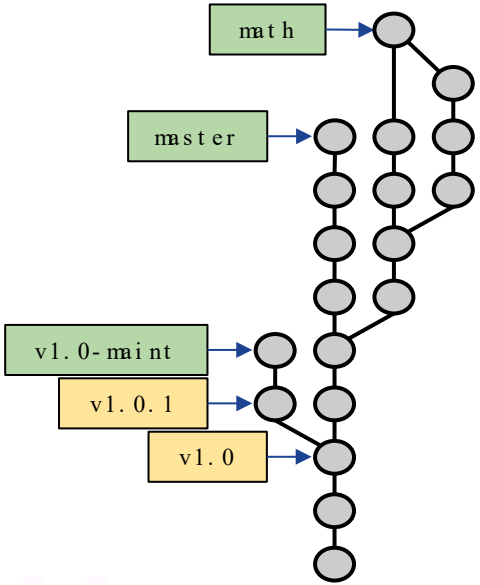


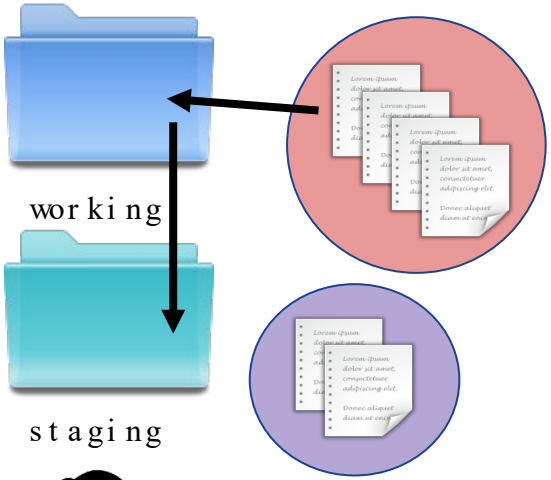


Staging area

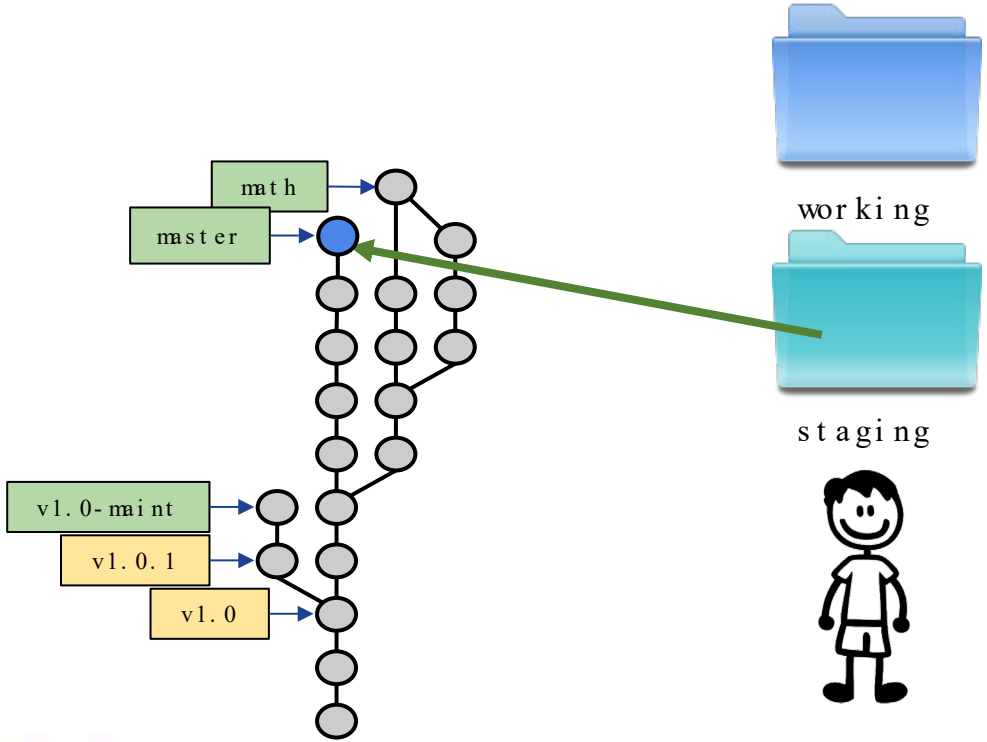


Staging area

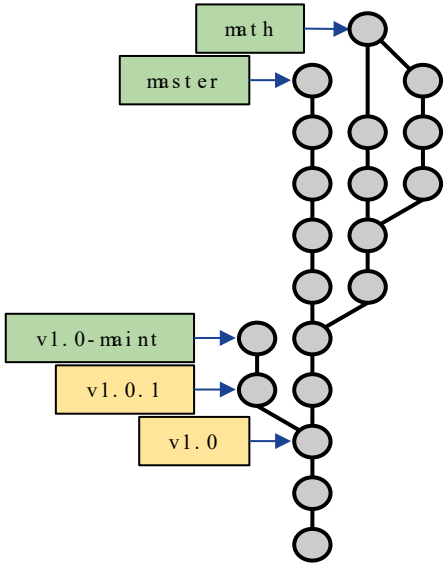


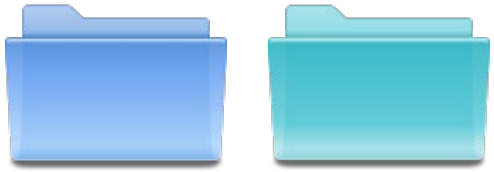


Staging area



Diffs

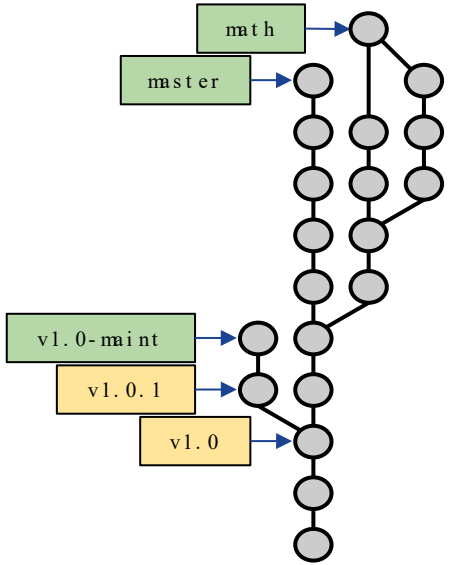


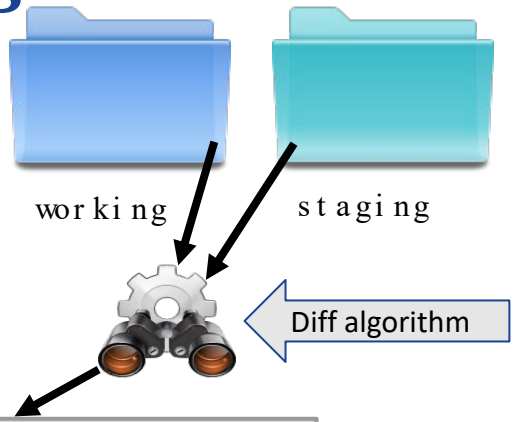


working staging

What are the changes?
- working vs. staging?
- working vs. snapshot X?
- staging vs. snapshot X?
- snapshot X vs. snapshot Y?

Diffs: working vs. staging

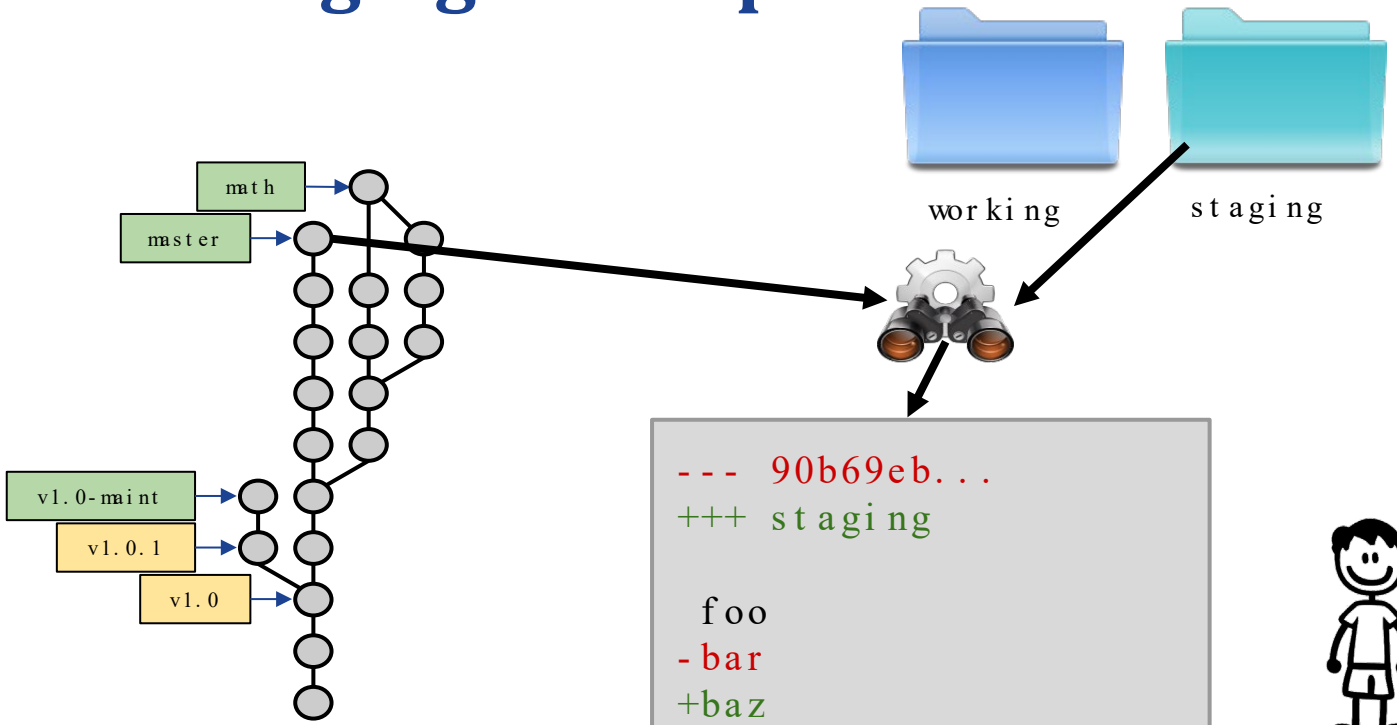




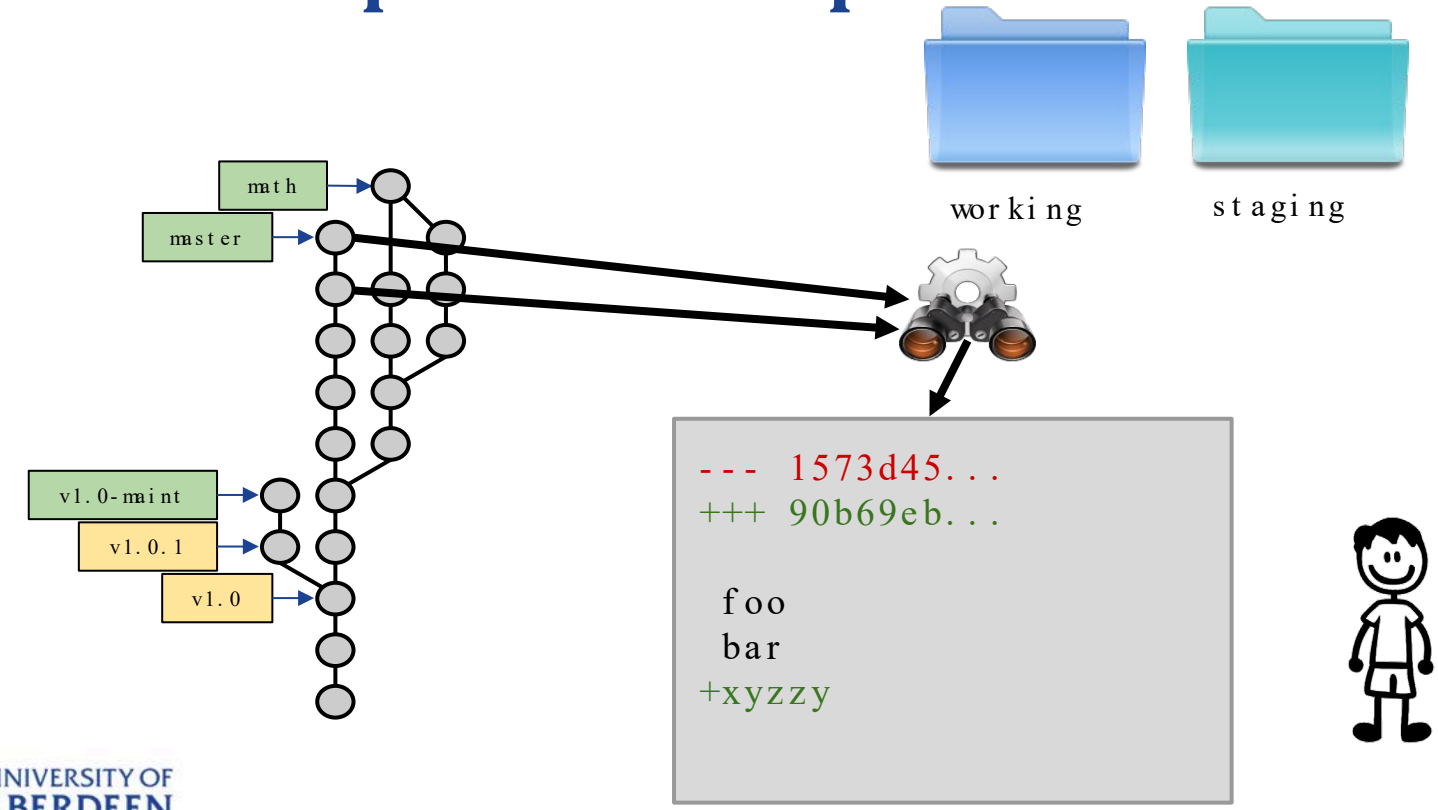

```
--- staging
+++ working

 foo
-baz
+bazzle
 xyzzy
```

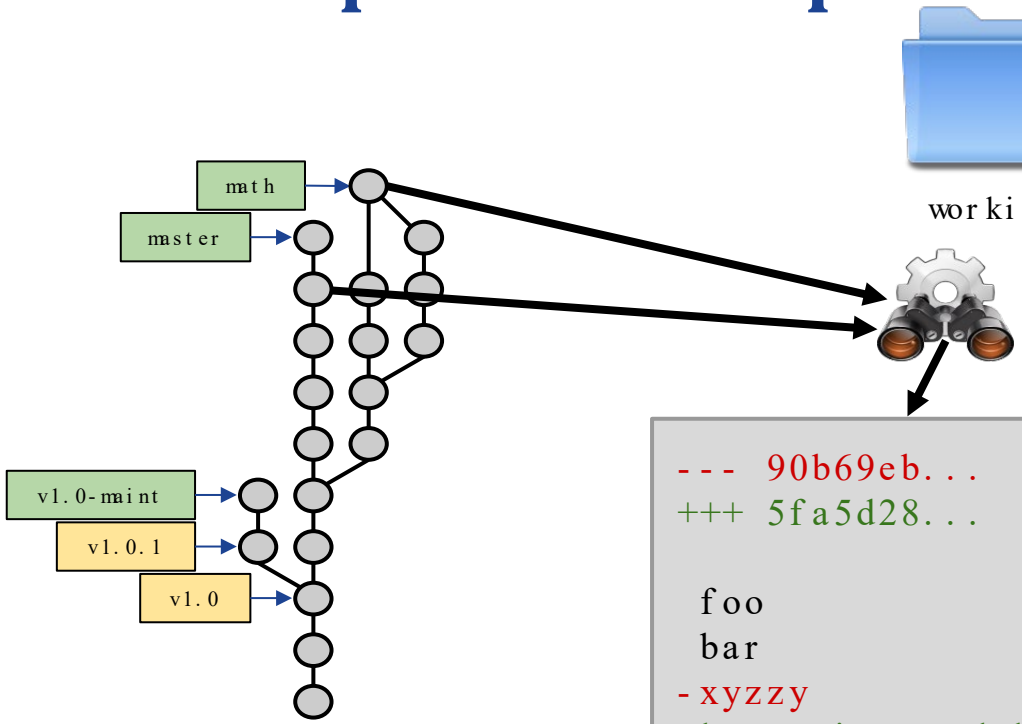
Diffs: staging vs. snapshot



Diffs: snapshot vs. snapshot



Diffs: snapshot vs. snapshot




```
--- 90b69eb...
+++ 5fa5d28...

 foo
 bar
-xyzzy
+lorem ipsum dolor
+sit amet, ...
```

Git commands: getting started

- First, tell Git who you are:

```
git config --global user.name ”My Name”
git config --global user.email ”my@email.address”
```

- Get help:

```
git <command> -h
git help <command>
```

- Start a new Git repository:

```
git init
```

Fetching, merging, pushing

```
git remote add <name> <URL>
git fetch <name>
git merge <name>/<branch>
```

`git pull`

```
git clone <URL> <project>
```

```
git init <project>
cd <project>
git remote add origin <URL>
git fetch origin
git checkout master
```

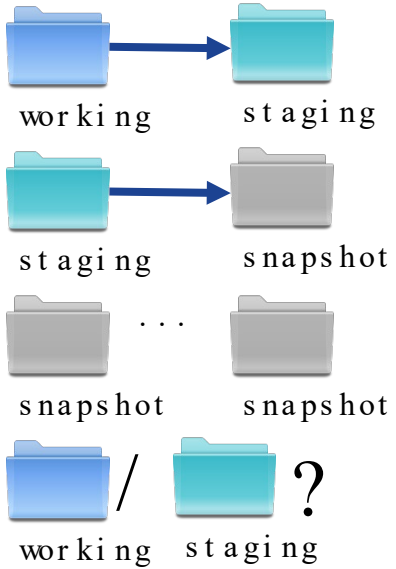
```
git push origin <name>
```

Adding branches and tags

```
git branch
git branch <branch>
git checkout <branch>
git tag -l
git tag <tag>
```

`git branch <branch>` + `git checkout <branch>` = `git checkout –b <branch>`

Making snaphots



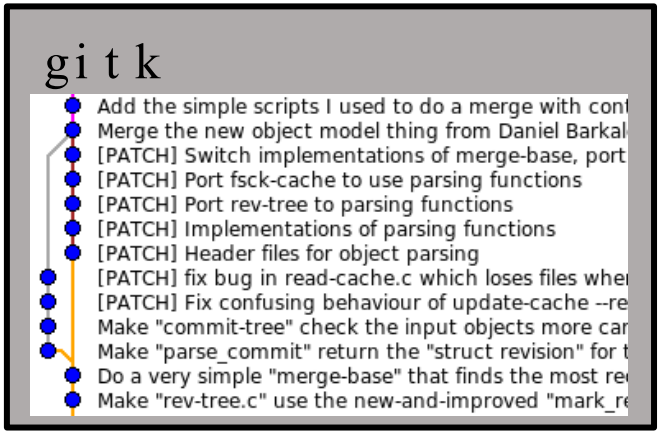
git add

git commit

git commit -a

git log

git status

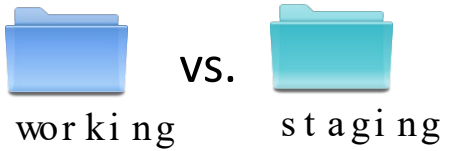


Conflicts

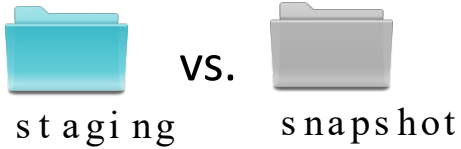
- Merging two branches that modified the same file independently will result in conflict
- If two people work on the same file within the same branch independently the second person's commit will create a conflict
- Conflicts can be resolved manually
- Avoid conflicts by pulling recent changes periodically

```
git pull <branch name>
```

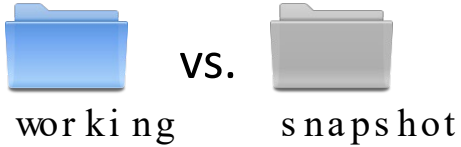
Commands for diffs



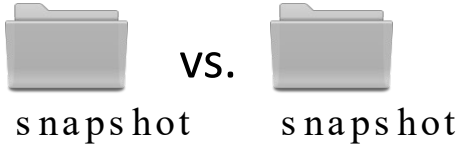

```
git diff
```




```
git diff --staged
```




```
git diff HEAD
```




```
git diff <from> <to>
```

Further reading about Git

- Oh My Git!: https://ohmygit.org (a game about learning Git)
- Git homepage: http://git-scm.com
- Pro Git: http://git-scm.com/book
- GitHub: http://github.com
- Learn Git Branching: https://learngitbranching.js.org
- Git Ready: http://gitready.com

Questions, comments?