# Software Development: GIT Version Control

**CPSC 233: Introduction to Computer Science for Computer Science Majors II**
**Winter 2022**

Jonathan Hudson, Ph.D.
Instructor
Department of Computer Science
University of Calgary

UNIVERSITY OF CALGARY

# Not an acronym

UNIVERSITY OF CALGARY

# Not an acronym

- (from the source code read-me of GIT)
- "git" can mean anything, depending on your mood.
  - random three-letter combination that is pronounceable, and not actually used by any common UNIX command. The fact that it is a mispronunciation of "get" may or may not be relevant.
  - stupid. contemptible and despicable. simple. Take your pick from the dictionary of slang.
  - "global information tracker": you're in a good mood, and it actually works for you. Angels sing, and a light suddenly fills the room.
  - "g*dd**n idiotic truckload of s**t": when it breaks

UNIVERSITY OF
CALGARY

# The Rise of Git

- *Git* is the most popular implementation of a distributed version control system.
- Development started in 2005 by Linus Torvalds.
  - Linux kernel source host dispute with BitKeeper
  - Same reason resulted in another Distribute Version Control Sys. -> Mercurial
- It is used by many popular open source projects as well as many commercial organizations.

# Why Git?

- Git's the most popular version control system in the industry.

- Most popular VCS are similar to Git

## VersionControl Preference



Source: Stackoverflow 2018 survey
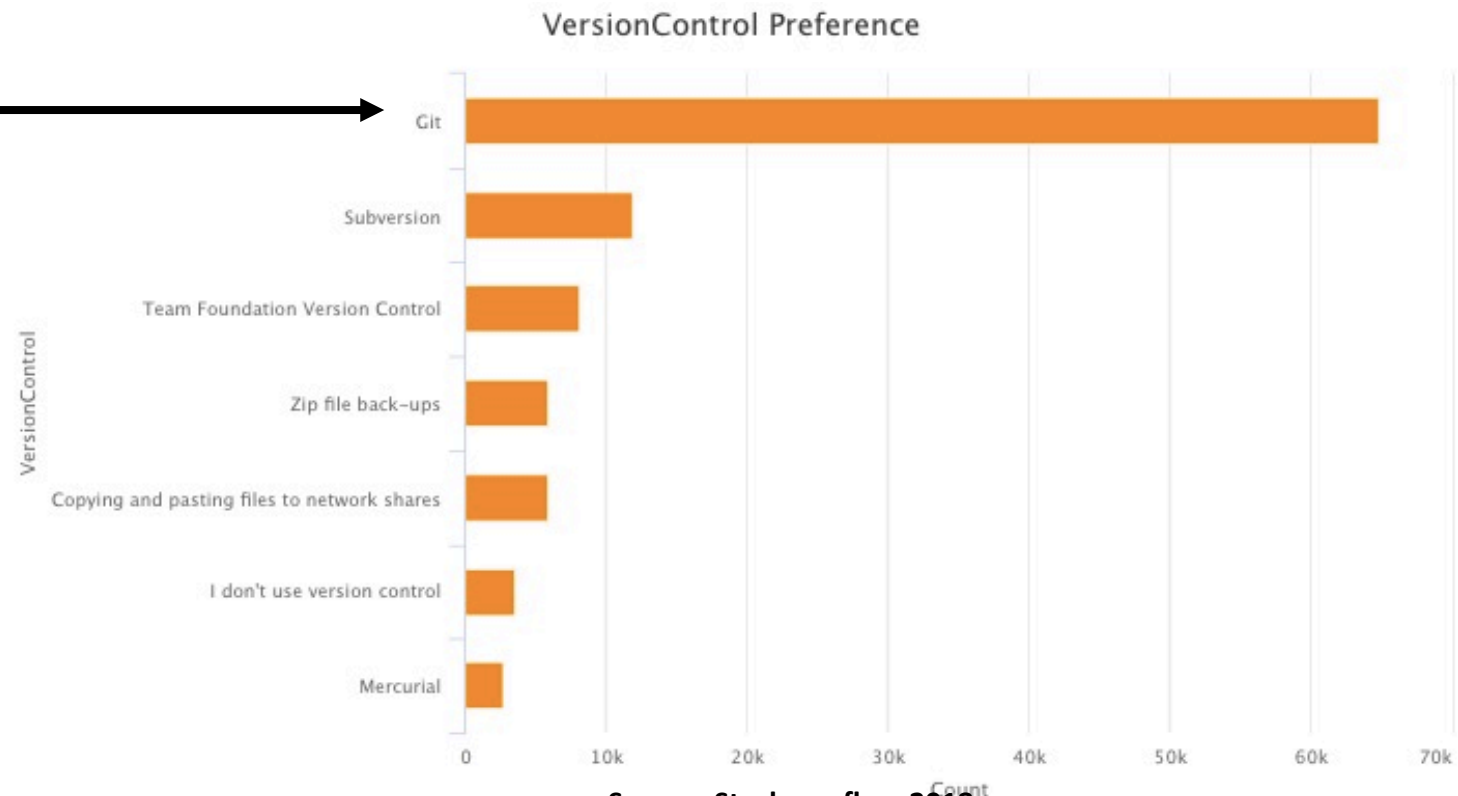
UNIVERSITY OF CALGARY

# Why Git?

- Git is distributed

- i.e. there is generally are remote repo and a local repo on your own machine
  - GIT lets each developer have their own version of repo
  - Each developer can make changes and make commits to own repo and periodically push/pull from remote to bring together development
  - Frees programmer, code on a plane and still do multiple local commits

**Source: Stackoverflow 2018 survey**

UNIVERSITY OF
CALGARY

# Terminology

UNIVERSITY OF
CALGARY

# Version Control

- **Version control**:

1. Stores source code files for a project in a **central** place
   - Allows multiple developers to work on the same code base in a controlled way

2. Keeps a **record of changes** made to source code files over time
   - You can recall any version of a file based on a date or version number

3. Allows you to maintain **multiple**, concurrent **releases** of your software
   - i.e. the mainline (or trunk) plus one or more branch releases

UNIVERSITY OF
CALGARY

# Version Control: Repository

- **Repository**: the place where source code files for projects are stored
  - Will contain all versions of the files
  - Actually stored as differences
    - much smaller than full copies
    - but means you need to history to recreate a full file
  - Can be local but often network accessible

# Version Control: Repository

- **Often stores non-code project artifacts** such as:
  - Ant/Maven files, Makefiles, etc.
  - External documentation (analysis, design, etc.)
- Generally **does not to store generated artifacts**
  - E.g. Object code, .class files, linking files, executables, temp files, etc

# Version Control: Basic Terms

**Workspace**: the place where you work on a copy of a project's files

Files in the *repository* are not changed by you directly

**Checking out**: populates your *workspace* with up-to-date copies of files and directories from the *repository*

**Committing**: saves your changes back into the repository

Sometimes called checking in

The repository keeps track of changes using revision numbers

**Updating/pulling**: repopulates your workspace with the latest versions of files

Useful when other developers are also working concurrently on the same project

UNIVERSITY OF
CALGARY

# Concepts

UNIVERSITY OF CALGARY

# Version Control: Versioning

## Revisions

1. Retrieve a specific revision of a file or set of files (i.e. a directory or a project)
2. List the differences between revisions
3. Retrieve all source code as it appeared at some date in the past

UNIVERSITY OF CALGARY

# Version Control: Branching

- A **branch** is a separate, independent line of development
  - Is like a separate repository for the same project
  - Allows parallel development on the same code base
  - Useful for creating a release branch
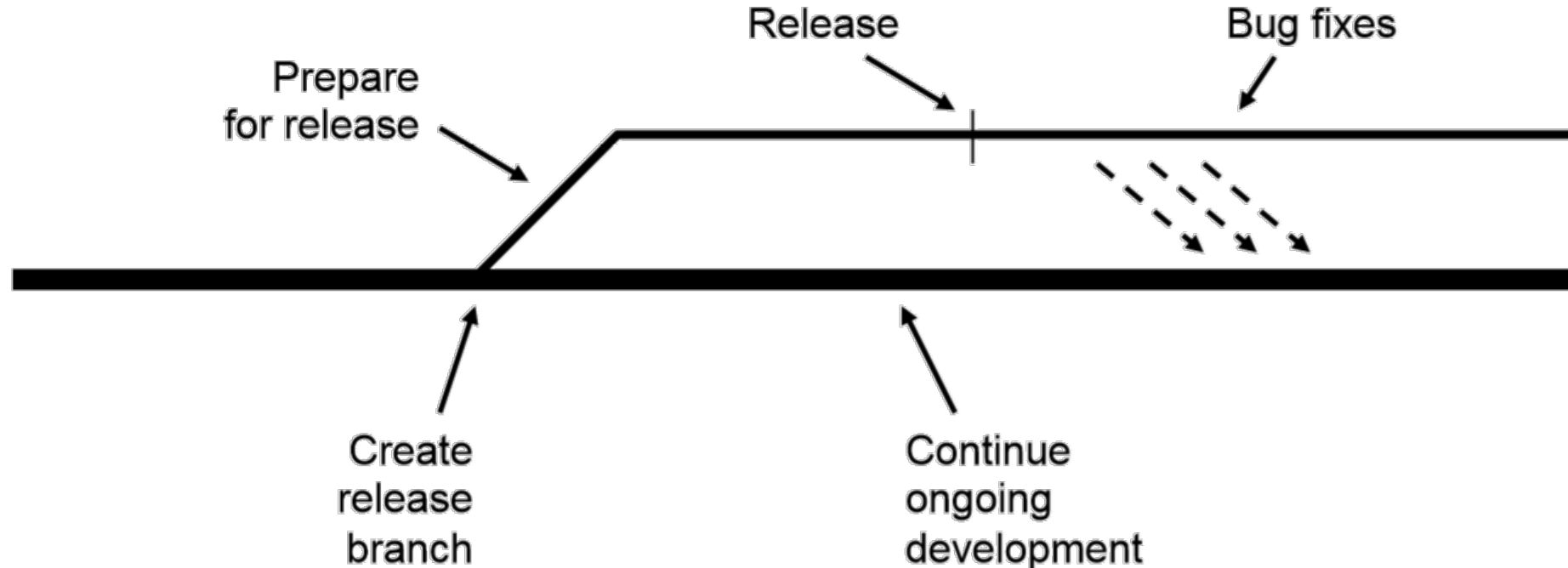
# Basic Concepts: Merging

- **Merging** allows you to apply changes made in a release branch back into the mainline
  - E.g.  Bug fixes, **Refactorings!!!**
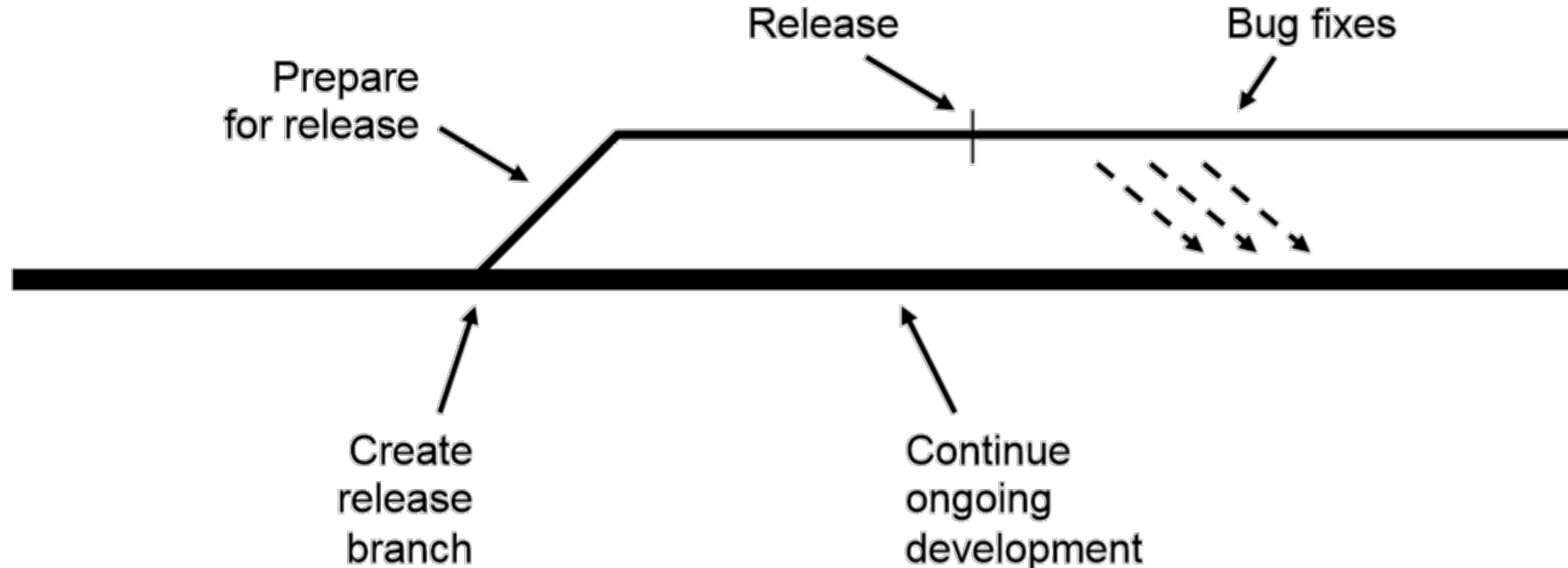
# Basic Concepts : Conflicts

- Two or more developers editing the same file can lead to **conflicts**
  - Strict locking allows only one person at a time to have write access to the file (gen 1)
- GIT
  - **Will attempt to do merge itself,** even within files
  - Will have 'conflict' if file is gone, or same line is edited
  - Will produce file with both lines and you'll have to pick (or to make more changes)

UNIVERSITY OF
CALGARY

# Git

UNIVERSITY OF CALGARY

# Git: *New* Version Control Terminology

**SHA**

- A SHA is basically an ID number for each commit.

- Ex. E2adf8ae3e2e4ed40add75cc44cf9d0a869afeb6

- Instead of version numbering

**Staging Area**

- You can think of the staging area as a prep table where Git will take the next commit.

- Files on the Staging Index are ready to be added to the repository.

UNIVERSITY OF
CALGARY

# Git: Getting Started

- Three trees of Git
  - The HEAD
    - last committed snapshot
  - Index (Staging Area)
    - Proposed next commit snapshot
  - Working directory
    - Sandbox where you are making changes to code

UNIVERSITY OF CALGARY
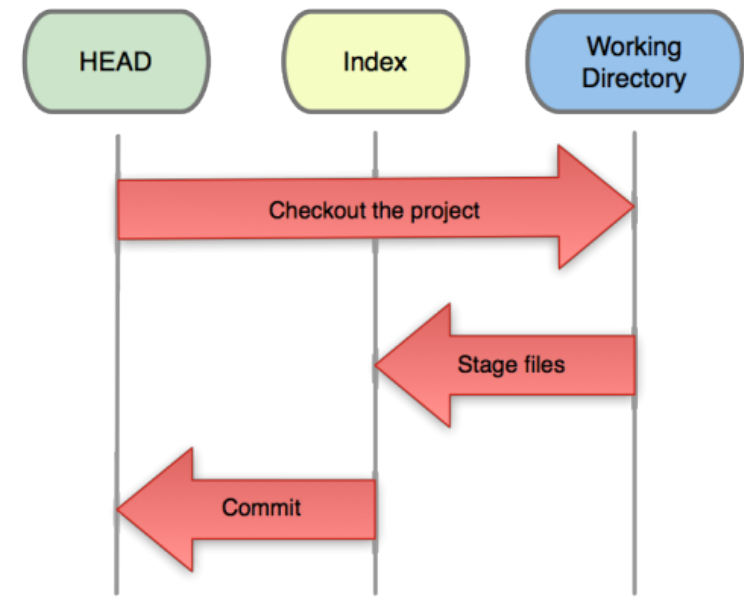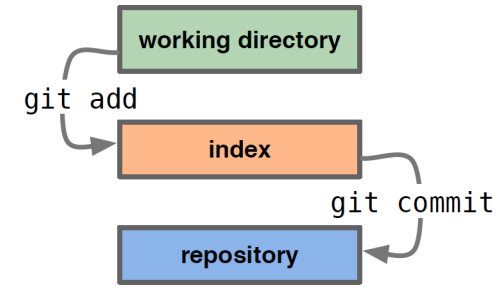
# Git: Basic Commands

- **git init** – *Initialize a Git repository/working directory*
  - *git init NAME*

- **git clone** – *Create an identical copy*

UNIVERSITY OF
**CALGARY**

# Git: A Basic Workflow

- A basic workflow
  - Init a repo (or clone an existing one)
  - Edit files
  - Stage the changes
  - Review your changes
  - Commit the changes

UNIVERSITY OF
CALGARY

# Git: A Basic Workflow

- A basic workflow
  - Edit files
  - Stage the changes
  - Review your changes
  - Commit the changes

UNIVERSITY OF CALGARY
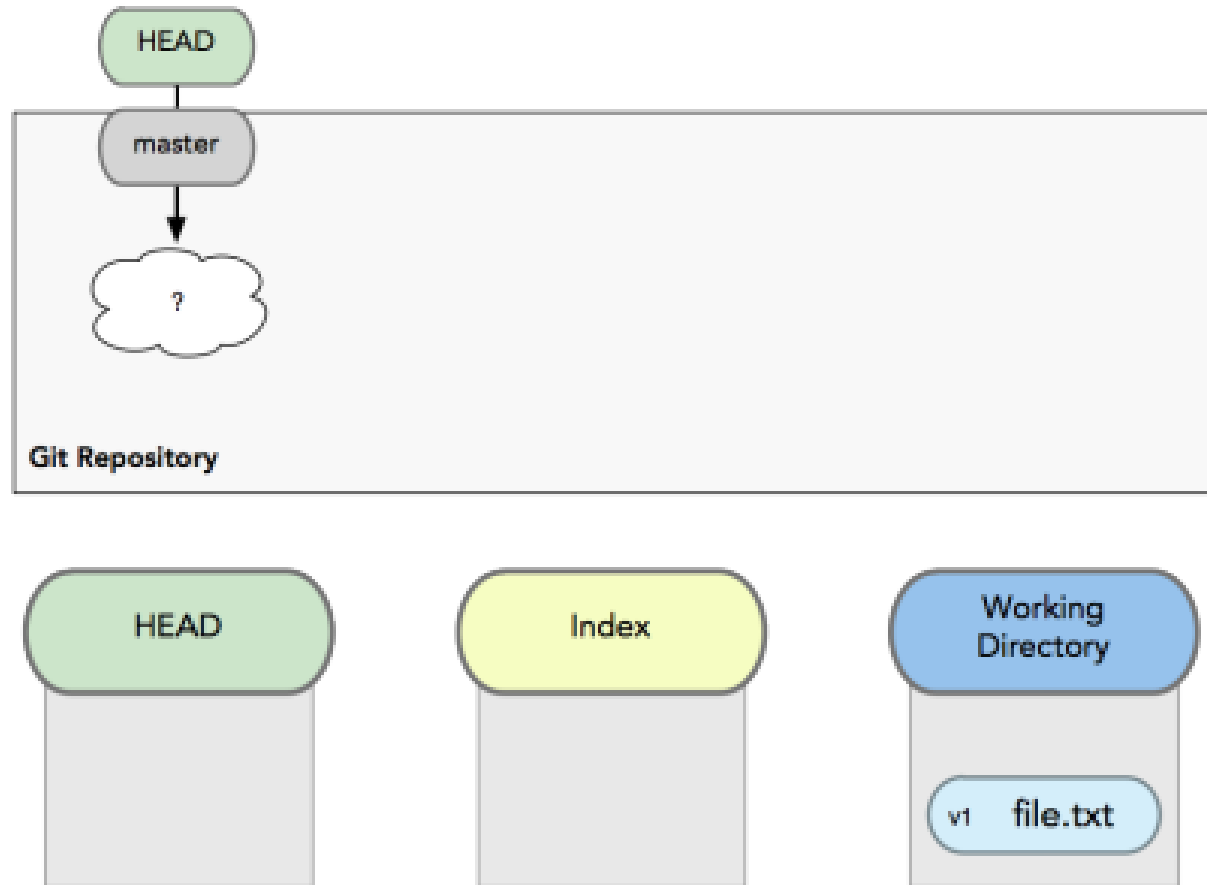
# Git: A Basic Workflow

- A basic workflow
  - Edit files
  - Stage the changes
  - Review your changes
  - Commit the changes

git add file.txt
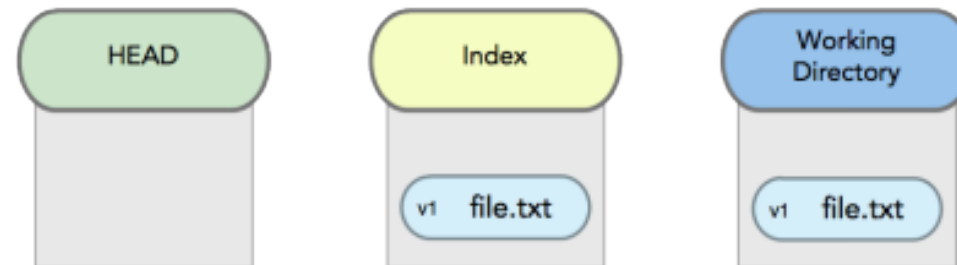


git add

UNIVERSITY OF CALGARY

# Git: A Basic Workflow

- A basic workflow
  - Edit files
  - Stage the changes
  - Review your changes
  - Commit the changes

git status
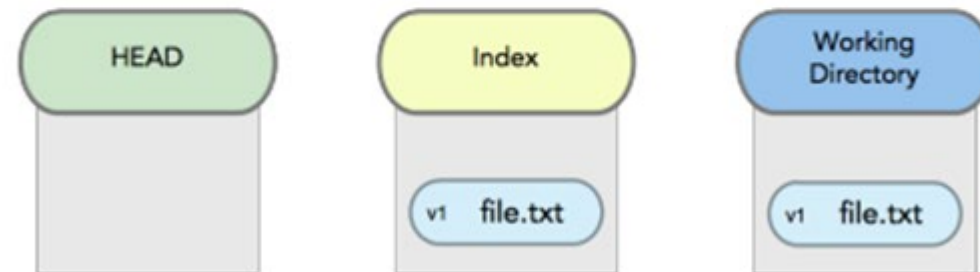


git status

ERSITY OF
LGARY

# Git: A Basic Workflow

- A basic workflow
  - Edit files
  - Stage the changes
  - Review your changes
  - Commit the changes

git commit



git commit

UNIVERSITY OF CALGARY
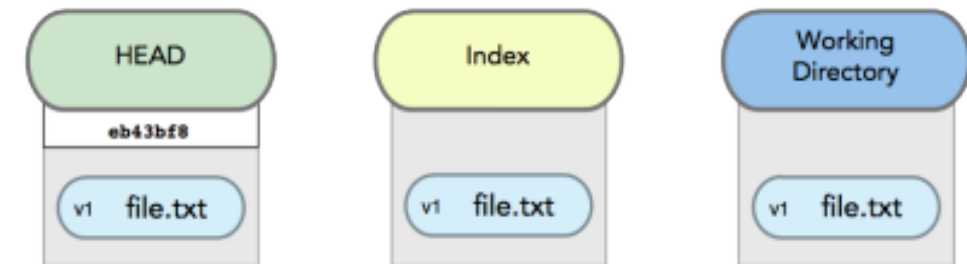
# Git: Informational

- View changes
  - **git diff**
    - Show the difference between working directory and staged
  - **git diff --cached**
    - Show the difference between staged and the HEAD

- View history
  - **git log**

UNIVERSITY OF
CALGARY

# Git: Revert

- Revert changes (Get back to a previous version)
  - **git checkout commit_hash**

UNIVERSITY OF
CALGARY

# Git: Commit Tree

- Git sees commits this way...
- Branch annotate which commit we are working on
- (ref is the current head here)
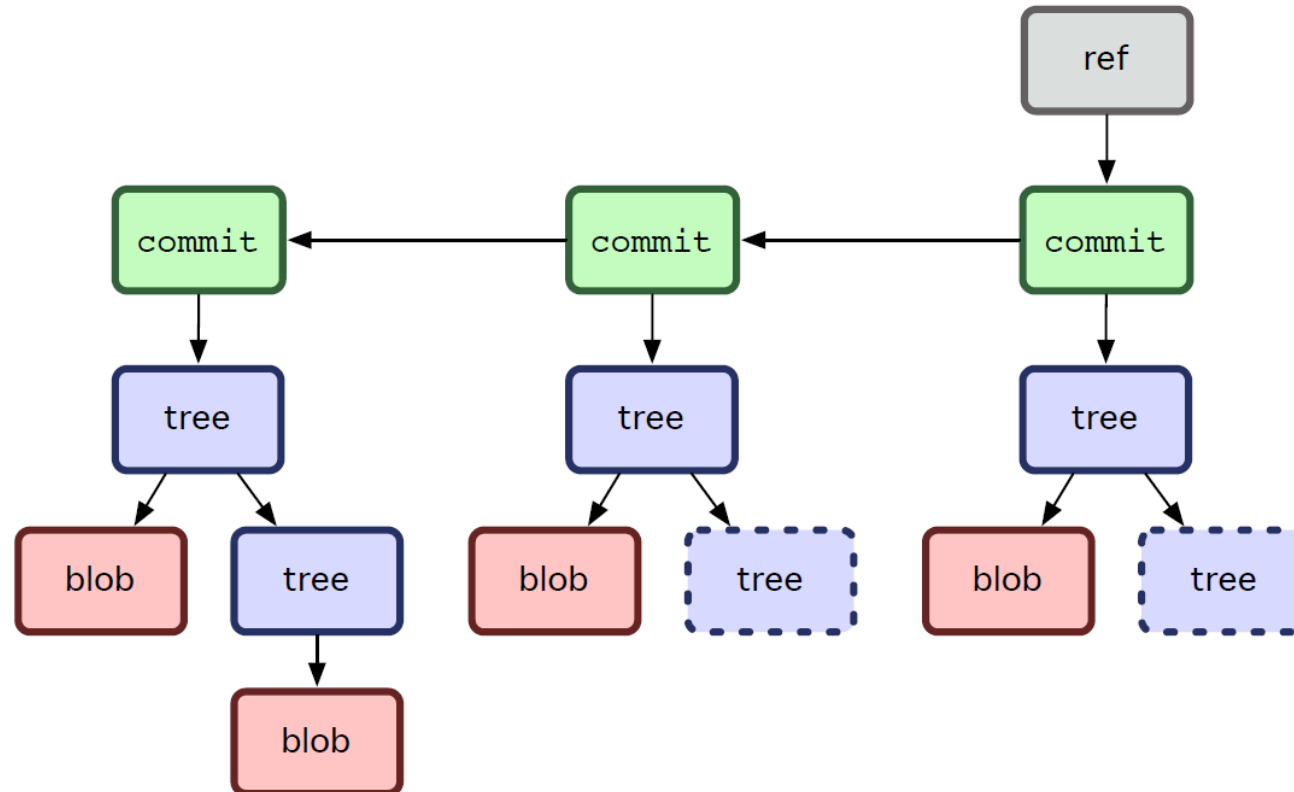
# Branching
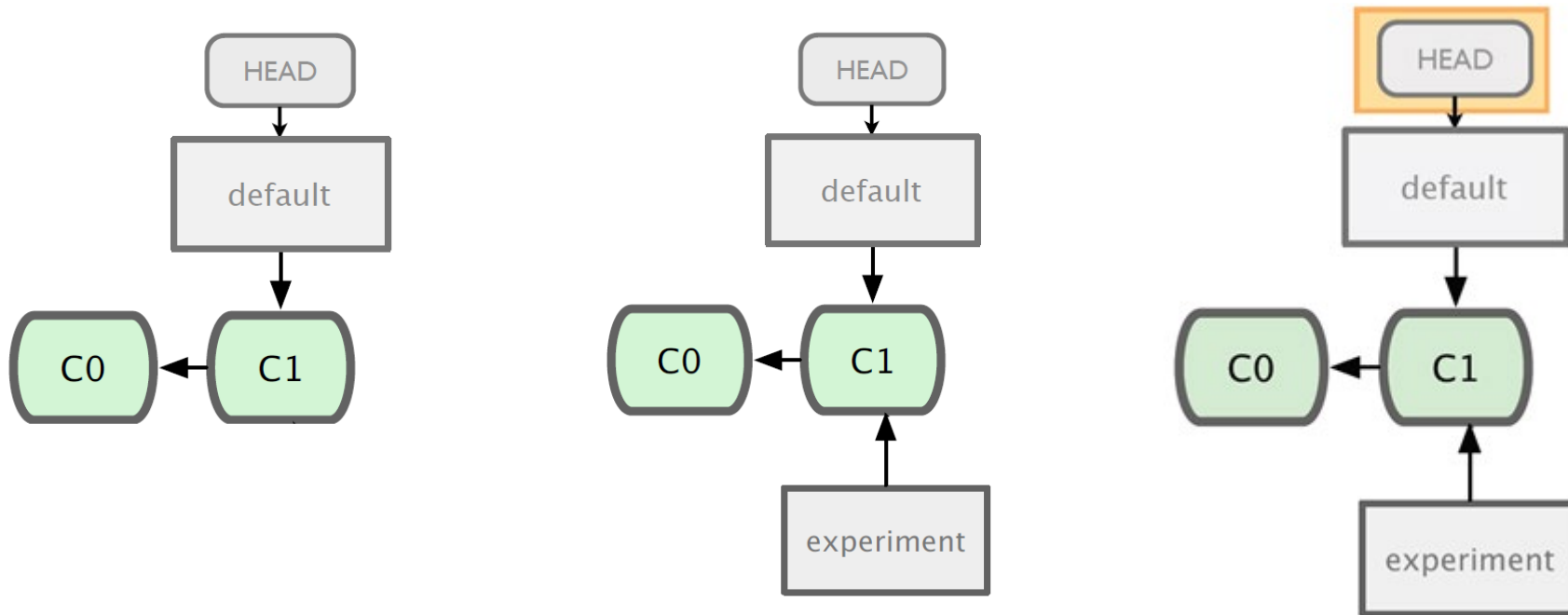
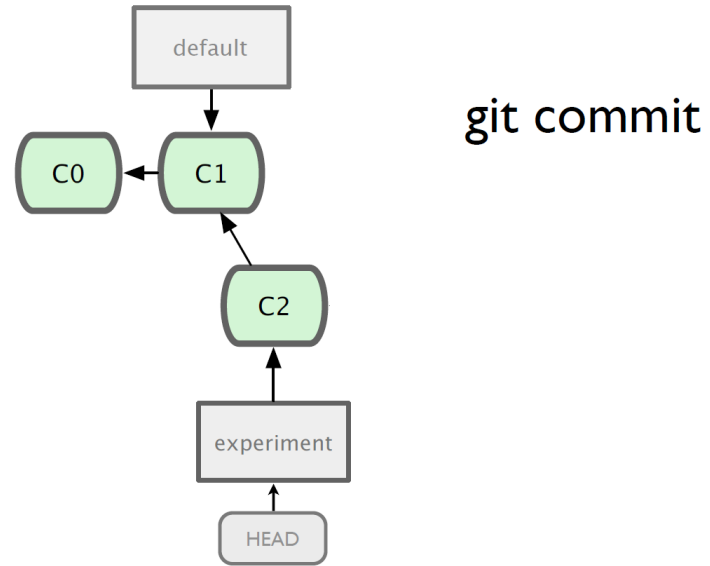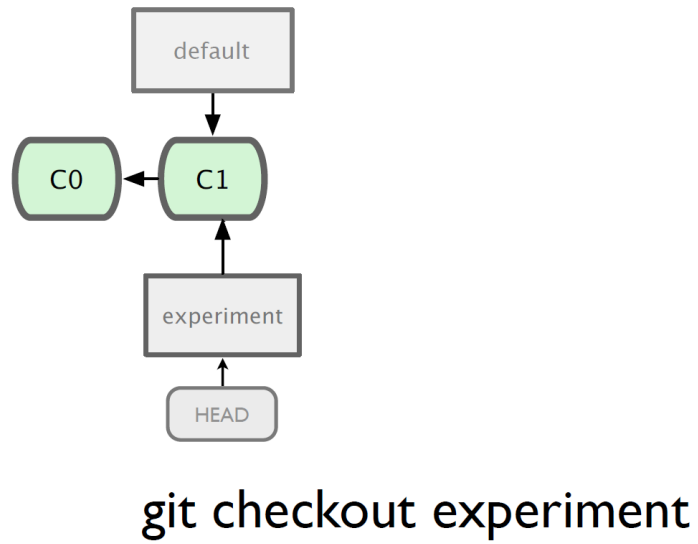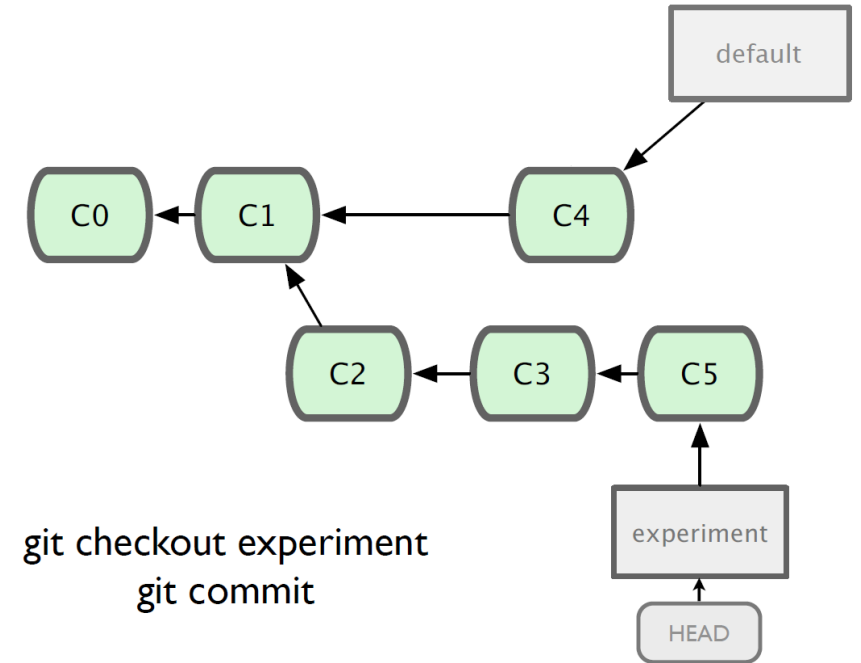# Git: Branching



git branch experiment

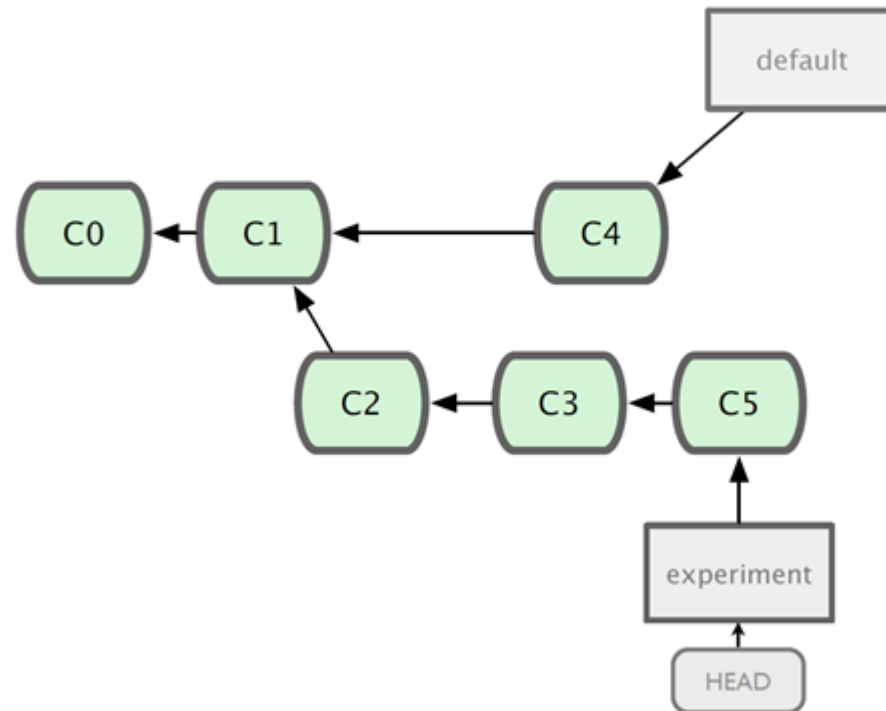UNIVERSITY OF CALGARY

# Git: Branching



git commit

git checkout experiment

git commit

git commit

# Git: Branching



git checkout default

git commit

git checkout experiment
git commit

UNIVERSITY OF
CALGARY

# Merging

UNIVERSITY OF
CALGARY
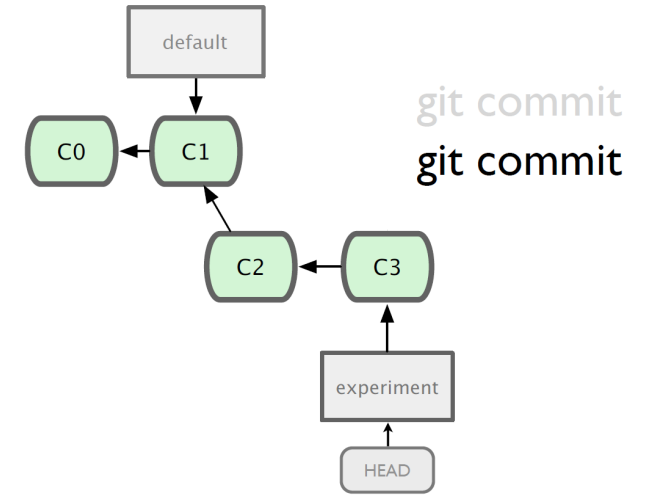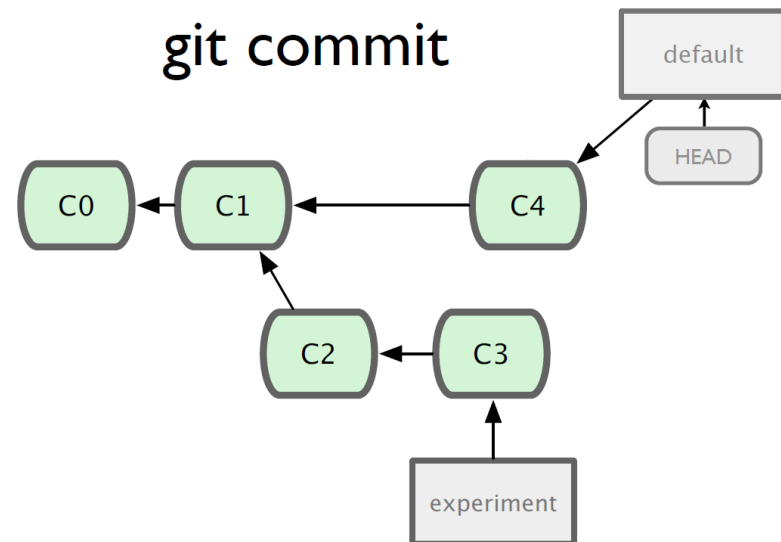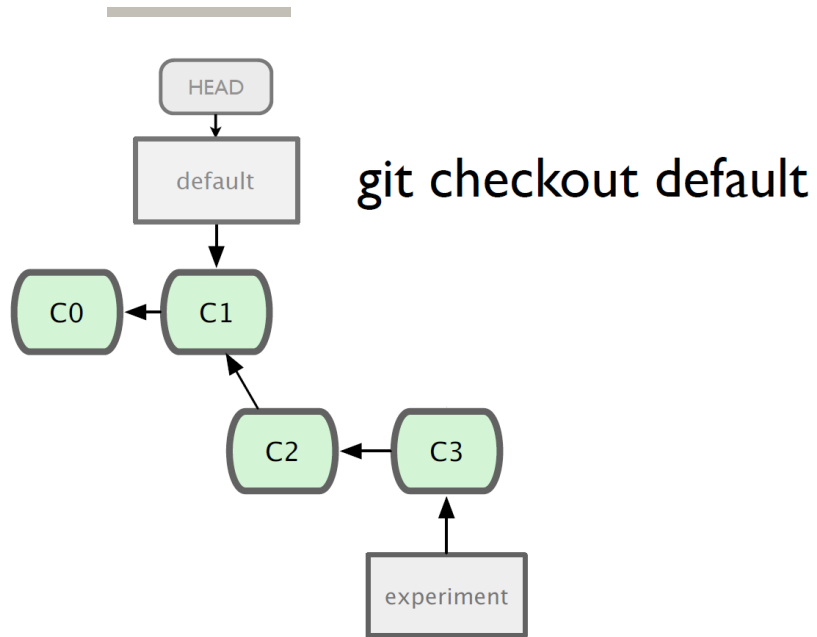
# Git: Merging

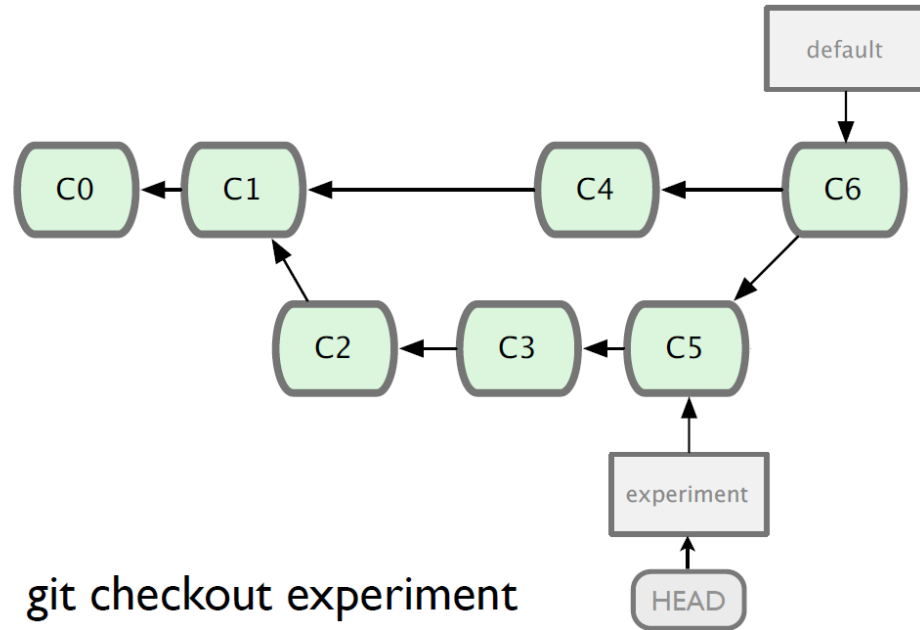- What do we do with this mess?
  - Merge them

UNIVERSITY OF
CALGARY

# Git: Merging

- Steps to merge two branch
  - Checkout the branch you want to merge onto
  - Merge the branch you want to merge



git checkout default
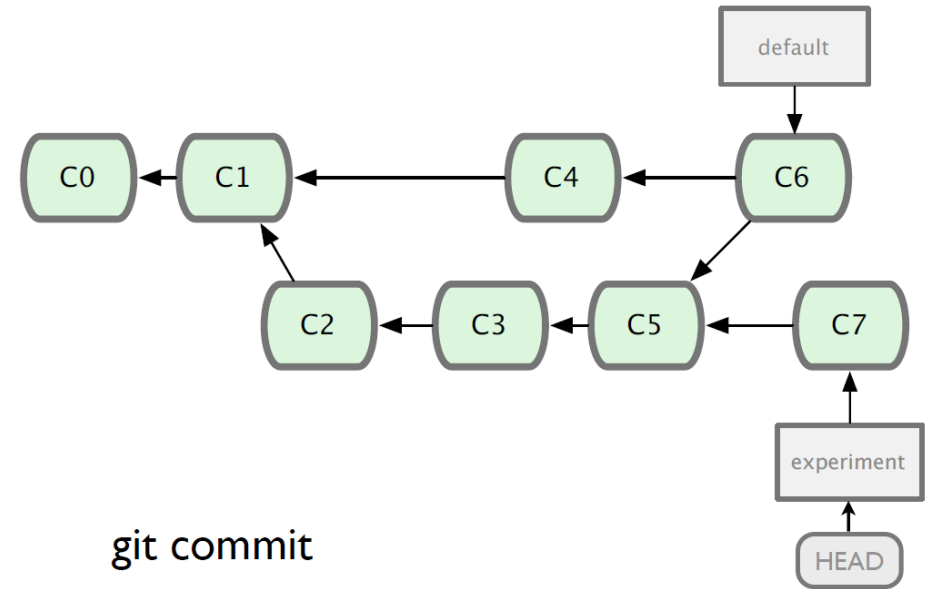


git checkout default
git merge experiment
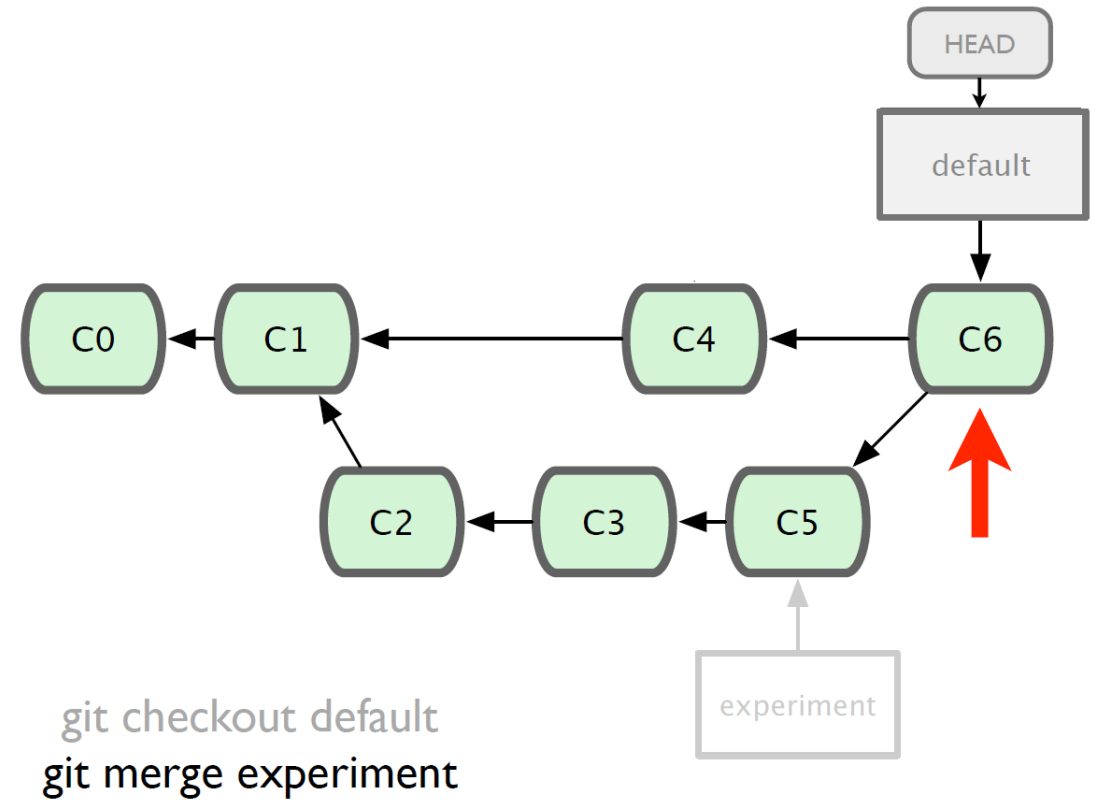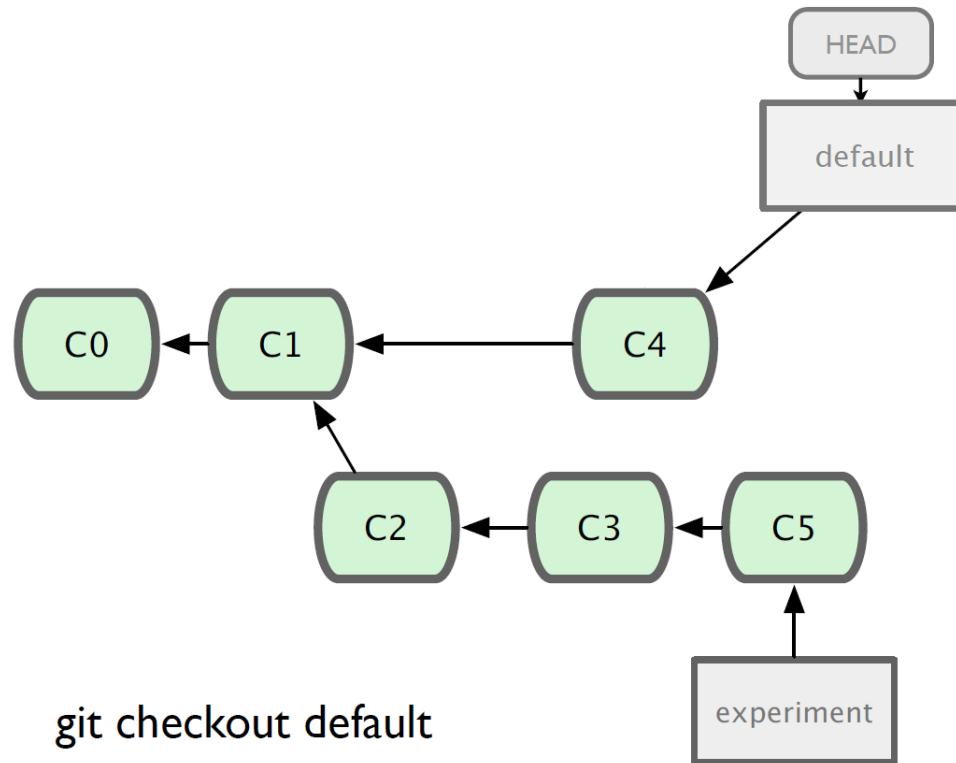
# Git: Merging

- We can continue working one whichever branch we want (the trunk **default** or on **experiment**)



git checkout experiment

git commit

# Git: Branching and Merging

- Why this is cool?
  - Non-linear development

```
clone the code that is in production
create a branch for issue #53 (iss53)
work for 10 minutes
someone asks for a hotfix for issue #102
checkout 'production'
create a branch (iss102)
fix the issue
checkout 'production', merge 'iss102'
push 'production'
checkout 'iss53' and keep working
```

UNIVERSITY OF
CALGARY

# Remote

UNIVERSITY OF
CALGARY

# GitHub, UofC GitLab

- It's a hosting medium/website for your Git repositories

- Offers powerful collaborative abilities

- A good indicator of what you code/ow much you code/quality of your code

- Access on https://github.com/ or https://gitlab.cpsc.ucalgary.ca

UNIVERSITY OF
CALGARY

# Git: Working with a remote repository

- Remote?

The common central repository

By default, remote name is **origin** and default branch is **main** *(previously master)*.

UNIVERSITY OF
CALGARY

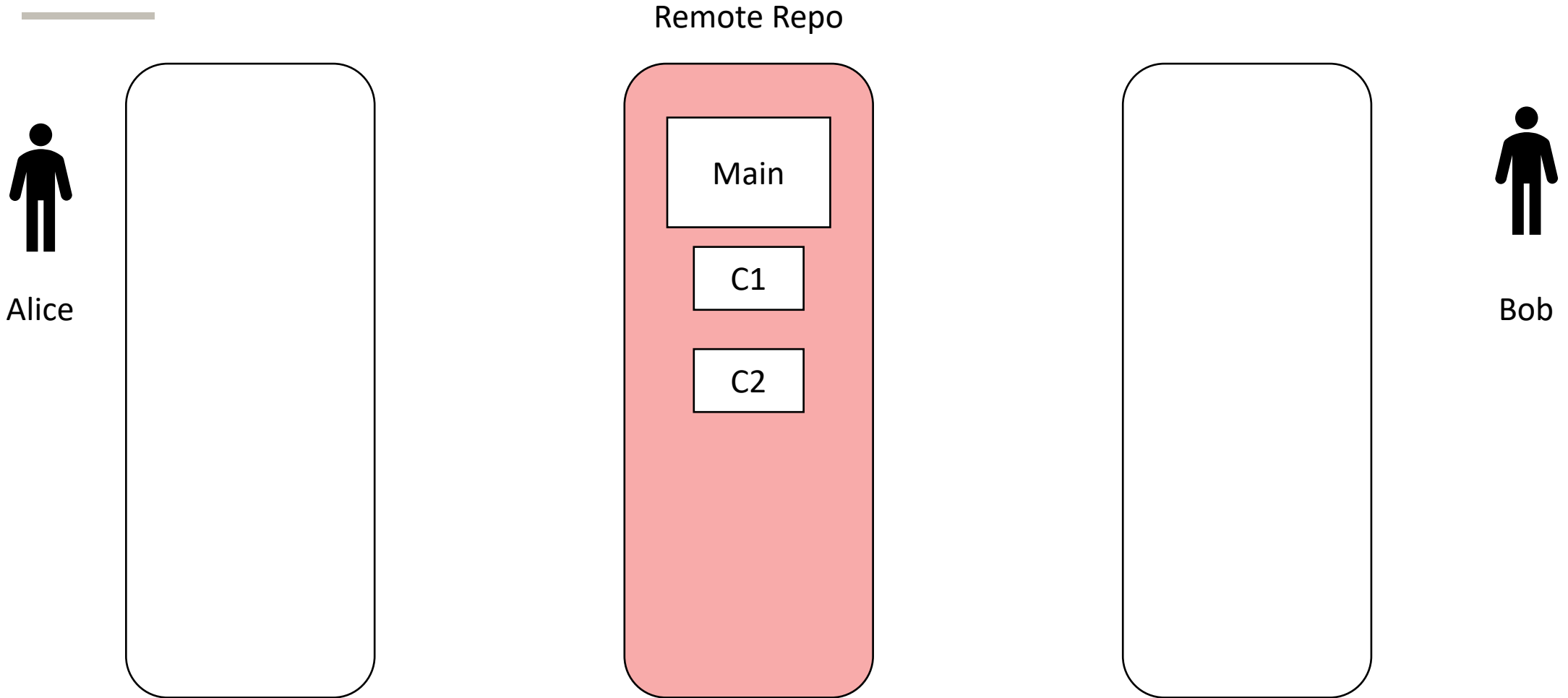# Git: Remote Commands

**git push**

   push your changes into the remote repository


**git pull**

   pull your latest changes from the remote repository

UNIVERSITY OF
CALGARY

# Collaborate via Remote

UNIVERSITY OF
CALGARY

# Git: Collaborate

Remote Repo

Main

C1

C2

Alice

Bob

UNIVERSITY OF CALGARY

# Git: Collaborate

Remote Repo

git clone

git clone

| Alice | | Bob |
|-------|--------|-----|
| Main | Main | Main |
| C1 | C1 | C1 |
| C2 | C2 | C2 |

UNIVERSITY OF CALGARY

# Git: Collaborate

Remote Repo

Alice

| Main |
| C1 |
| C2 |
| CA |

**git add**
**git commit**

| Main |
| C1 |
| C2 |

| Main |
| C1 |
| C2 |
| CB |

Bob

**git add**
**git commit**

UNIVERSITY OF CALGARY

# Git: Collaborate

Remote Repo

Alice

| Main |
| C1 |
| C2 |
| CA |

git push →

| Main |
| C1 |
| C2 |
| CA |

Bob

| Main |
| C1 |
| C2 |
| CB |

UNIVERSITY OF CALGARY

# Git: Collaborate

Alice

| Main |
| :---: |
| C1 |
| C2 |
| **CA** |

Remote Repo

| Main |
| :---: |
| C1 |
| C2 |
| **CA** |

**git fetch**

Bob

| Main |
| :---: |
| C1 |
| C2 |
| **CB** |

CA

UNIVERSITY OF CALGARY

# Git: Collaborate

Alice

Main

C1

C2

CA

Remote Repo

Main

C1

C2

CA

**git merge**

Bob

Main

C1

C2

CA

CB

UNIVERSITY OF CALGARY

# Git: Collaborate

Alice

Main

C1

C2

CA

Remote Repo

Main

C1

C2

CA

CB

git push

Main

C1

C2

C3

CB

Bob

UNIVERSITY OF CALGARY

# Git: Collaborate

Remote Repo

Alice

Bob

Main

C1

C2

CA

CB

git pull

Main

C1

C2

CA

CB

Main

C1

C2

CA

CB

UNIVERSITY OF CALGARY

# Onward to ... Testing

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~jwhudson/

UNIVERSITY OF
CALGARY