CS 433 – Advance Programming

Lecture#03
Version Control System
Phase-1 (DevOps)

National University
Of Computer and Emerging Sciences

quest lab

Google Classroom link

https://classroom.google.com/c/Mzg5MDM1MjM0NTkx?cjc=eqj6hr7

Version Control with Git

Version control systems

- Version control (or revision control, or source control) is all about managing multiple versions of documents, programs, web sites, etc.
  - Almost all "real" projects use some kind of version control
  - Essential for team projects, but also very useful for individual projects
- Some well-known version control systems are CVS, Subversion, Mercurial, and Git
  - CVS and Subversion use a "central" repository; users "check out" files, work on them, and "check them in"
  - Mercurial and Git treat all repositories as equal
- Distributed systems like Mercurial and Git are newer and are gradually replacing centralized systems like CVS and Subversion

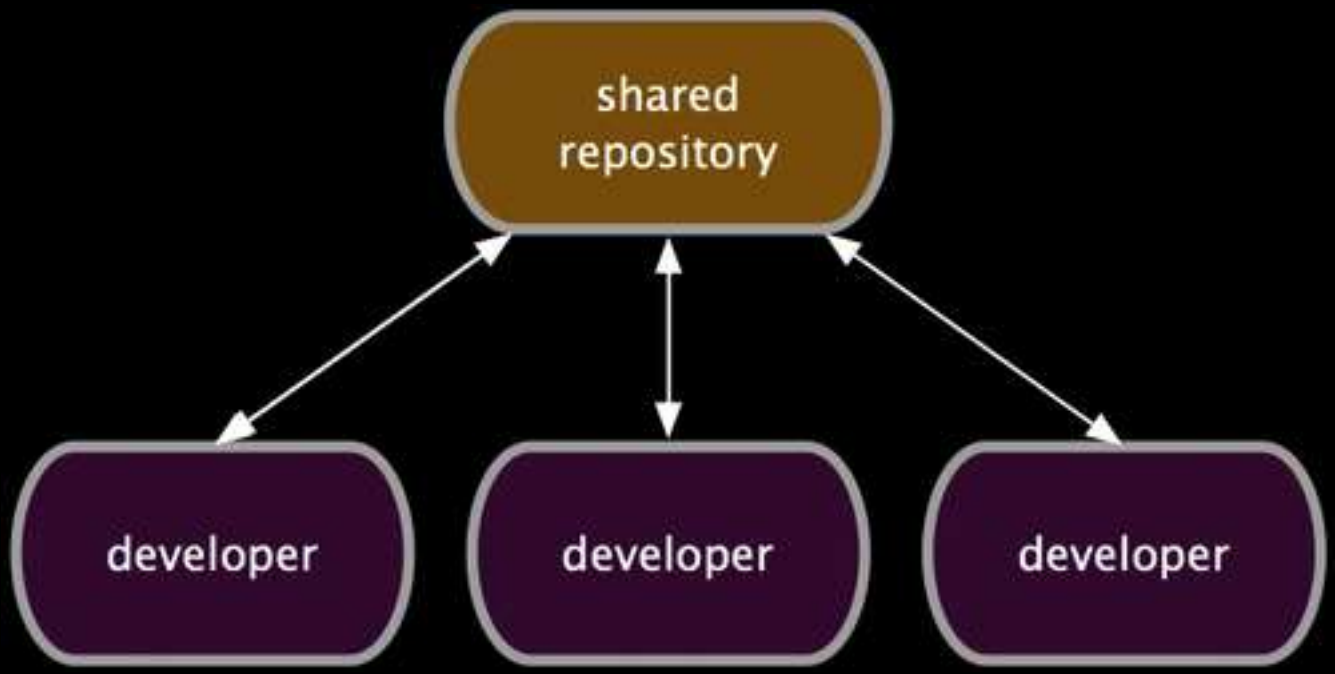
Why version control?

- For working by yourself:
  - Gives you a “time machine” for going back to earlier versions
  - Gives you great support for different versions (standalone, web app, etc.) of the same basic project
- For working with others:
  - Greatly simplifies concurrent work, merging changes

Why Git?

- Git has many advantages over earlier systems such as CVS and Subversion
  - the most commonly used version control system.
  - Git tracks the changes you make to files, so you have a record of what has been done, and you can revert to specific versions should you ever need to.
  - Git also makes collaboration easier, allowing changes by multiple people to all be merged into one source.

Collaborate: Work in parallel with teammates



http://git-scm.com/book/en/Distributed-Git-Distributed-Workflows

Version Control Systems (VCSs)

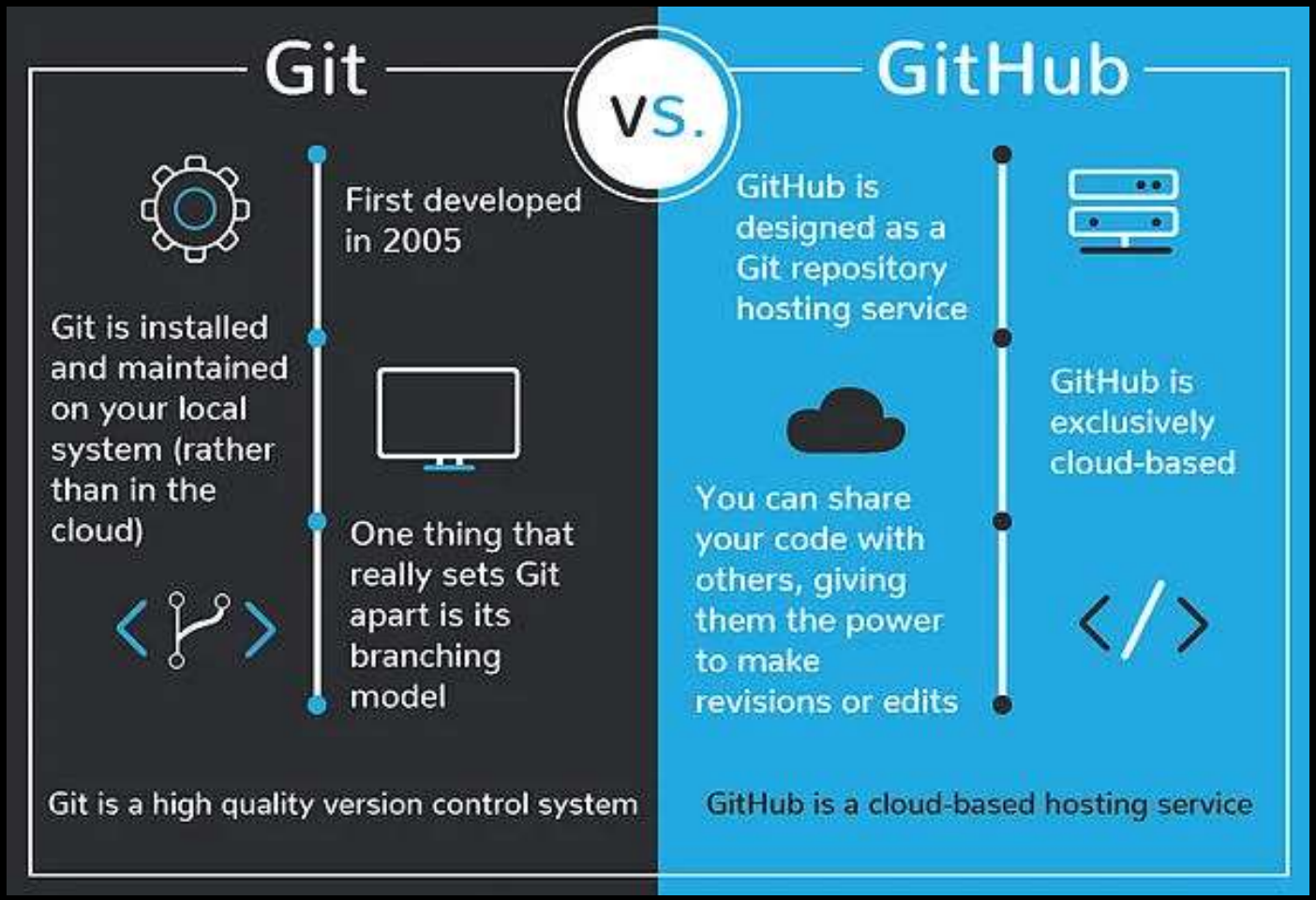
- Help you track/manage/distribute revisions
- Standard in modern development
- Examples:
  - Revision Control System (RCS)
  - Concurrent Versions System (CVS)
  - Subversion (SVN)
  - Git

older

newer

Our focus

Git and GitHub



Git and GitHub

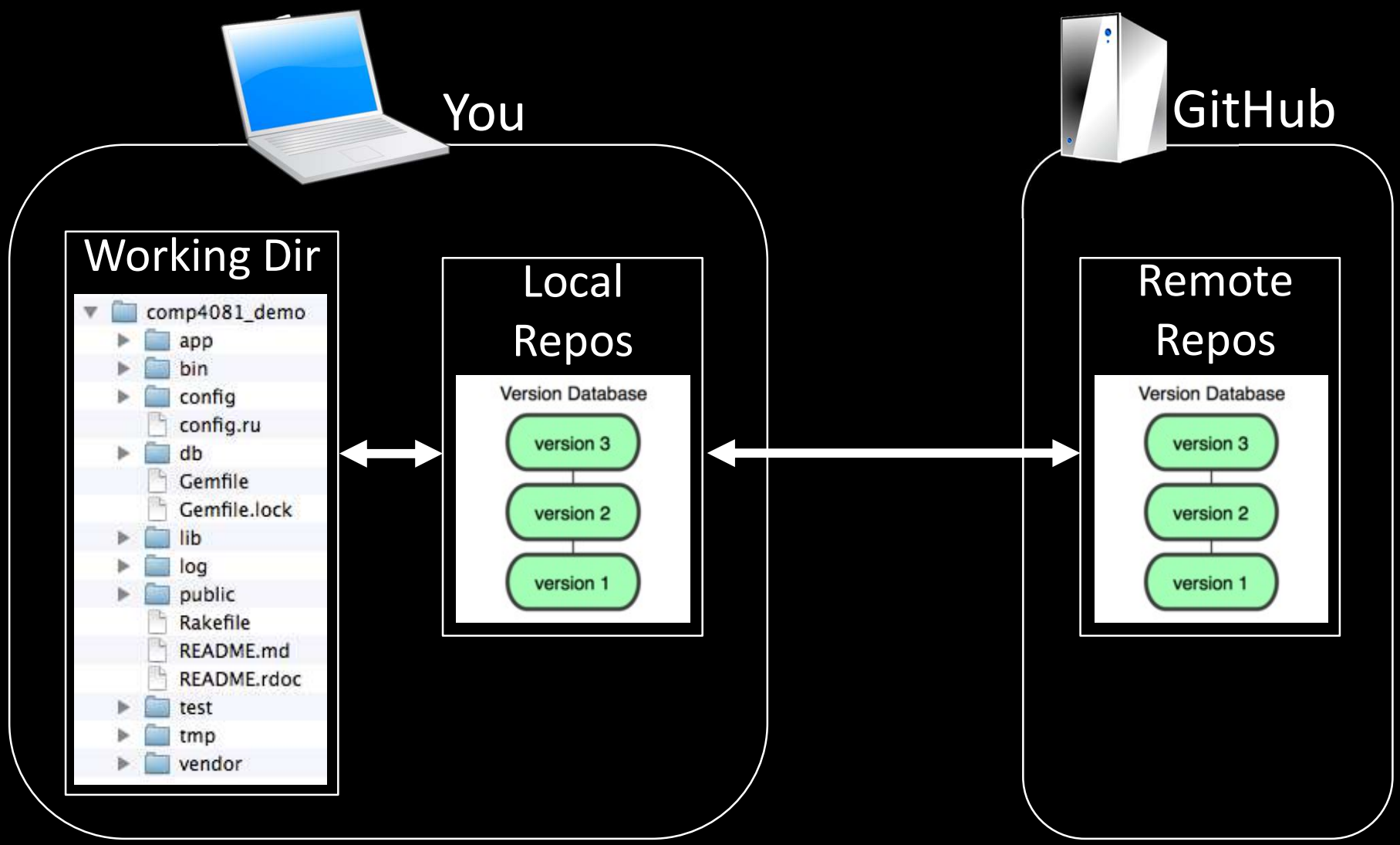
Git

- Git is a version control system that lets you manage and keep track of your source code history.
- Git is an open-source, version control tool created in 2005 by developers working on the Linux operating system.

GitHub

- GitHub is a cloud-based hosting service that lets you manage Git repositories.
- GitHub is a company founded in 2008 that makes tools which integrate with git.

GitHub-User Perspective



Configure your Git client
(Tutorial 2.33.0)

- Install Git (https://git-scm.com/downloads)

- Check config info:

```
$ git config --list
user.name=Scott Fleming
user.email=Scott.Fleming@memphis.edu
```

- Fix if necessary:

```
$ git config --global user.name "John Doe"

$ git config --global user.email jdoe@memphis.edu
```

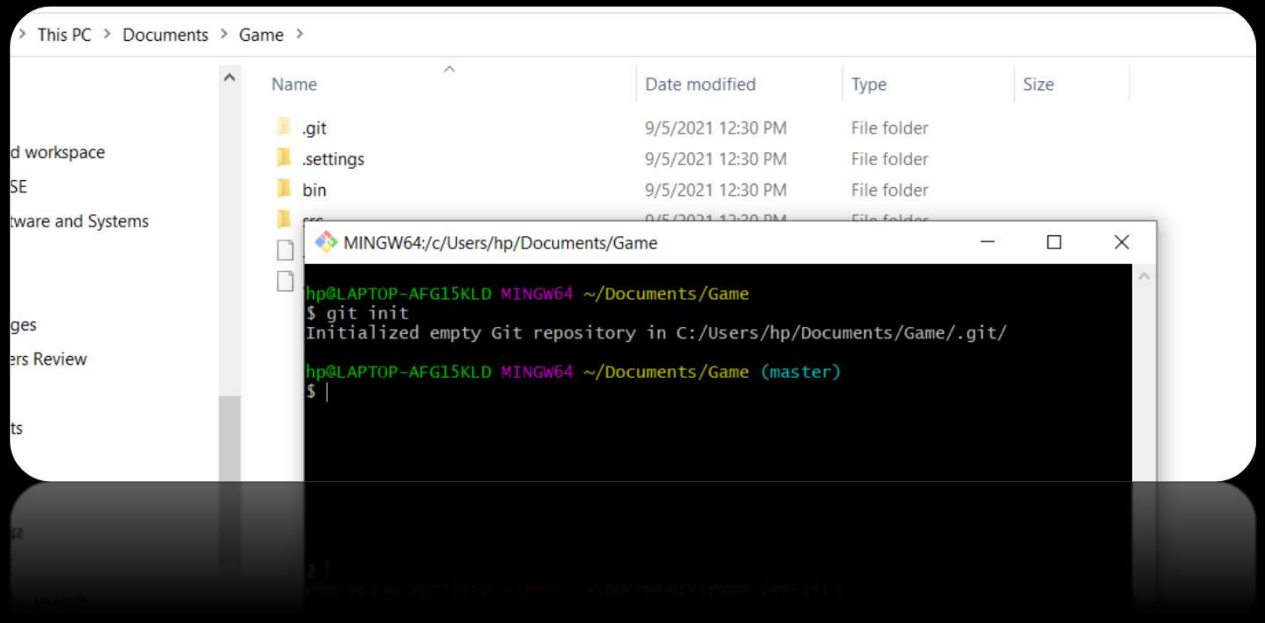
Steps

- Step 1: Create a local git repository
- Step 2: Add a new file to the repo
- Step 3: Add a file to the staging environment
- Step 4: Create a commit
- Step 5: Create a new branch
- Step 6: Create a new repository on GitHub
- Step 7: Push a branch to GitHub
- Step 8: Create a pull request (PR)
- Step 9: Merge a PR
- Step 10: Get changes on GitHub back to your computer

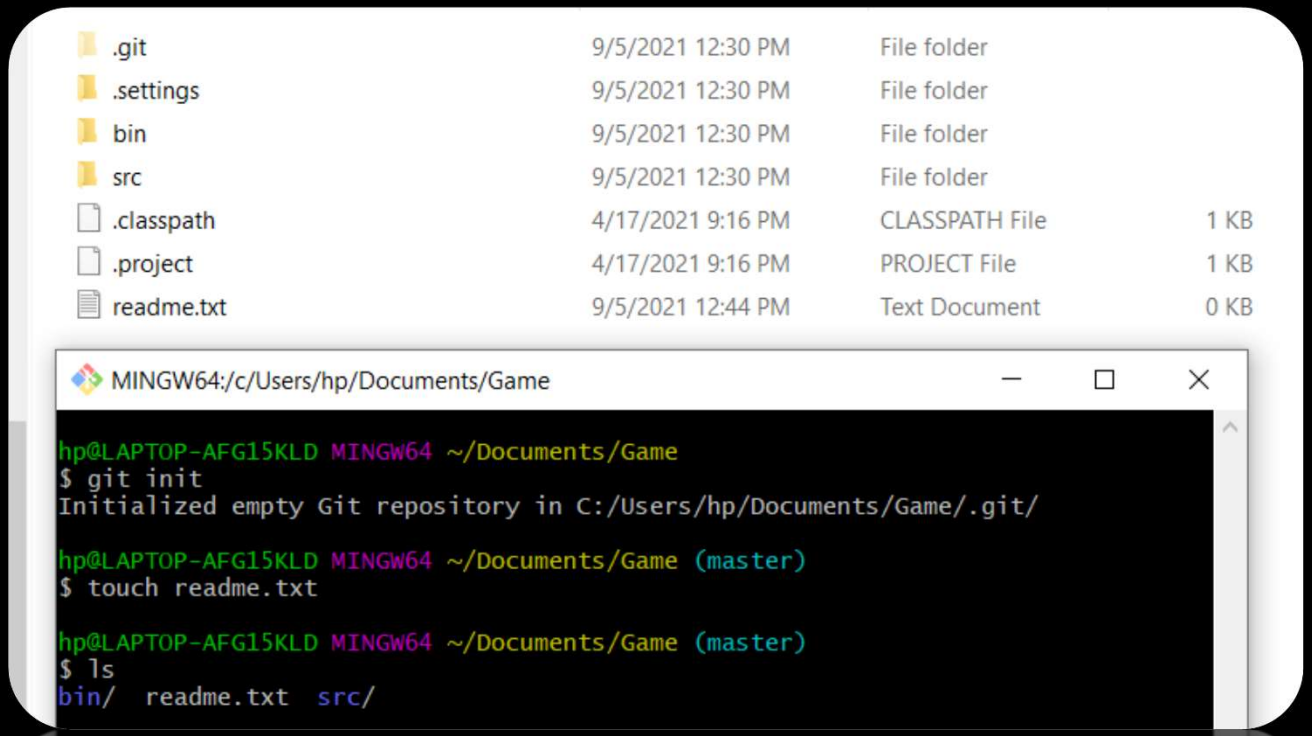
Step 1: Create a local git repository

- When creating a new project on your local machine using git, you'll first create a new repository.

1. cd to the project directory you want to use
2. Type in `git init`
   - This creates the repository (a directory named `.git`)



Step 2: Add a new file to the repo



Git status command

The `git status` command displays the state of the working directory and the staging area.

MINGW64:/c/Users/hp/Documents/Game

```
hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .classpath
        .project
        .settings/
        bin/
        readme.txt
        src/

nothing added to commit but untracked files present (use "git add" to track)

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ |
```

Step 3: Add a file to the staging environment

1. Type in `git add .`
   - This adds all your current files to the repository

```
MINGW64:/c/Users/hp/Documents/Game
hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .classpath
        new file:   .project
        new file:   .settings/org.eclipse.jdt.core.prefs
        new file:   bin/FireMonster.class
        new file:   bin/Monster.class
        new file:   bin/StoneMonster.class
        new file:   bin/TestMonster.class
        new file:   bin/WaterMonster.class
        new file:   readme.txt
        new file:   src/FireMonster.java
        new file:   src/Monster.java
        new file:   src/StoneMonster.java
        new file:   src/TestMonster.java
        new file:   src/WaterMonster.java


hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$
```

Step 4: Create a commit

1. Type in `git commit -m "Initial commit"`
   - You can use a different commit message, if you like

MINGW64:/c/Users/hp/Documents/Game

```
hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git commit -m "ReadMe.txt file is added!"
[master (root-commit) 1dbf500] ReadMe.txt file is added!
 14 files changed, 229 insertions(+)
 create mode 100644 .classpath
 create mode 100644 .project
 create mode 100644 .settings/org.eclipse.jdt.core.prefs
 create mode 100644 bin/FireMonster.class
 create mode 100644 bin/Monster.class
 create mode 100644 bin/StoneMonster.class
 create mode 100644 bin/TestMonster.class
 create mode 100644 bin/WaterMonster.class
 create mode 100644 readme.txt
 create mode 100644 src/FireMonster.java
 create mode 100644 src/Monster.java
 create mode 100644 src/StoneMonster.java
 create mode 100644 src/TestMonster.java
 create mode 100644 src/WaterMonster.java

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ |
```

Step 5: Create a new branch

- You want to make a new feature but are worried about making changes to the main project while developing the feature. This is where git branches come in.

- For instance, if you want to add a new page to your website you can create a new branch just for that page without affecting the main part of the project. Once you're done with the page, you can merge your changes from your branch into the primary branch.

- When you create a new branch, Git keeps track of which commit your branch 'branched' off of, so it knows the history behind all the files.

Create New Branch

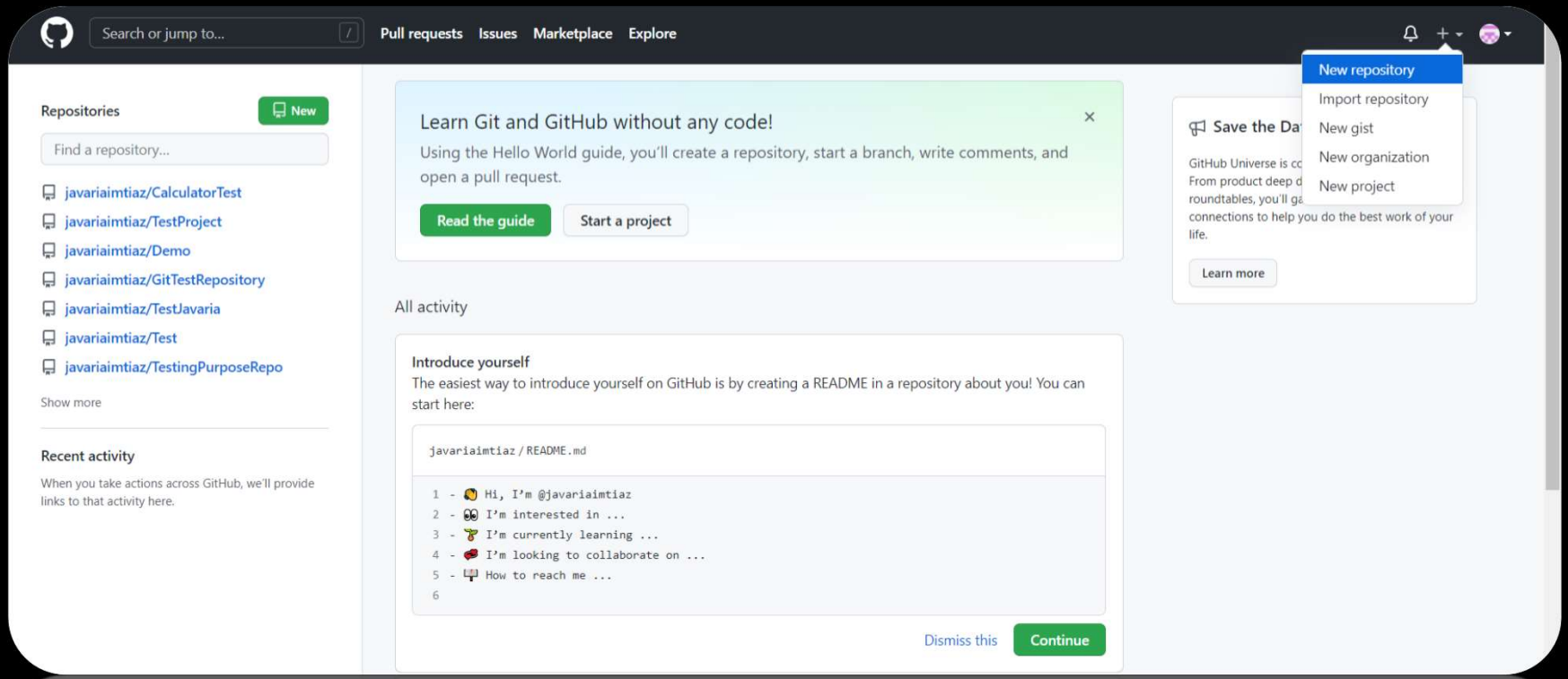
- Run `git checkout -b <my branch name>`.

MINGW64:/c/Users/hp/Documents/Game

```
 create mode 100644 bin/StoneMonster.class
 create mode 100644 bin/TestMonster.class
 create mode 100644 bin/WaterMonster.class
 create mode 100644 readme.txt
 create mode 100644 src/FireMonster.java
 create mode 100644 src/Monster.java
 create mode 100644 src/StoneMonster.java
 create mode 100644 src/TestMonster.java
 create mode 100644 src/WaterMonster.java

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git branch
* master

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git checkout -b TestBranch
Switched to a new branch 'TestBranch'

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (TestBranch)
$ git branch
* TestBranch
  master
```

Step 6: Create a new repository on GitHub

- if you want to work with a team, you can use GitHub to collaboratively modify the project's code.



Create New Repository

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
Import a repository.

Owner * javariaimtiaz / Repository name * My-GitHub-Project

Great repository names are short and memorable. Need inspiration? How about symmetrical-chainsaw?

Description (optional)

- (●) **Public**
  Anyone on the internet can see this repository. You choose who can commit.
- ( ) **Private**
  You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

- [ ] **Add a README file**
  This is where you can write a long description for your project. Learn more.
- [ ] **Add .gitignore**
  Choose which files not to track from a list of templates. Learn more.
- [ ] **Choose a license**
  A license tells others what they can and can't do with your code. Learn more.

Create repository

Step 7: Push a branch to GitHub

- Type `git remote add origin remote-repo-URL`
- Type `git push -u origin master`

MINGW64:/c/Users/hp/Documents/Game

```
hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git remote add origin https://github.com/javariaimtiaz/My-GitHub-Project.git

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git push -u origin master
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 12 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (19/19), 5.55 KiB | 1.85 MiB/s, done.
Total 19 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/javariaimtiaz/My-GitHub-Project.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ |
```

GitHub Repository View

javariaimtiaz / My-GitHub-Project

Unwatch 1 | Star 0 | Fork 0

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

master | 1 branch | 0 tags | Go to file | Add file | Code

javariaimtiaz ReadMe.txt file is added! 1dbf500 41 minutes ago 1 commit

.settings	ReadMe.txt file is added!	41 minutes ago
bin	ReadMe.txt file is added!	41 minutes ago
src	ReadMe.txt file is added!	41 minutes ago
.classpath	ReadMe.txt file is added!	41 minutes ago
.project	ReadMe.txt file is added!	41 minutes ago
readme.txt	ReadMe.txt file is added!	41 minutes ago

About

No description, website, or topics provided.

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Push Branch to GitHub

- Type `git push origin Branch-Name`

MINGW64:/c/Users/hp/Documents/Game

```
hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git push -u origin master
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 12 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (19/19), 5.55 KiB | 1.85 MiB/s, done.
Total 19 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/javariaimtiaz/My-GitHub-Project.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git push origin TestBranch
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'TestBranch' on GitHub by visiting:
remote:      https://github.com/javariaimtiaz/My-GitHub-Project/pull/new/TestBranch
remote:
To https://github.com/javariaimtiaz/My-GitHub-Project.git
 * [new branch]      TestBranch -> TestBranch
```

GitHub Repository View

javariaimtiaz / My-GitHub-Project

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

TestBranch had recent pushes less than a minute ago — Compare & pull request

master | 2 branches | 0 tags | Go to file | Add file | Code

javariaimtiaz ReadMe.txt file is added! — 1dbf500 1 hour ago — 1 commit

.settings	ReadMe.txt file is added!	1 hour ago
bin	ReadMe.txt file is added!	1 hour ago
src	ReadMe.txt file is added!	1 hour ago
.classpath	ReadMe.txt file is added!	1 hour ago
.project	ReadMe.txt file is added!	1 hour ago
readme.txt	ReadMe.txt file is added!	1 hour ago

Step 8: Create a pull request (PR)

- A pull request (or PR) is a way to alert a repo's owners that you want to make some changes to their code. It allows them to review the code and make sure it looks good before putting your changes on the primary branch.

GitHub Repository View

javariaimtiaz / My-GitHub-Project

Unwatch 1 Star

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

base: master ← compare: TestBranch ✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in this comparison with others. Learn about pull requests

Create pull request

1 commit | 2 files changed | 0 comments | 1 contributor

Commits on Sep 05, 2021

Test file is updated e14ccef

Showing 2 changed files with 1 addition and 0 deletions.

Unified Split

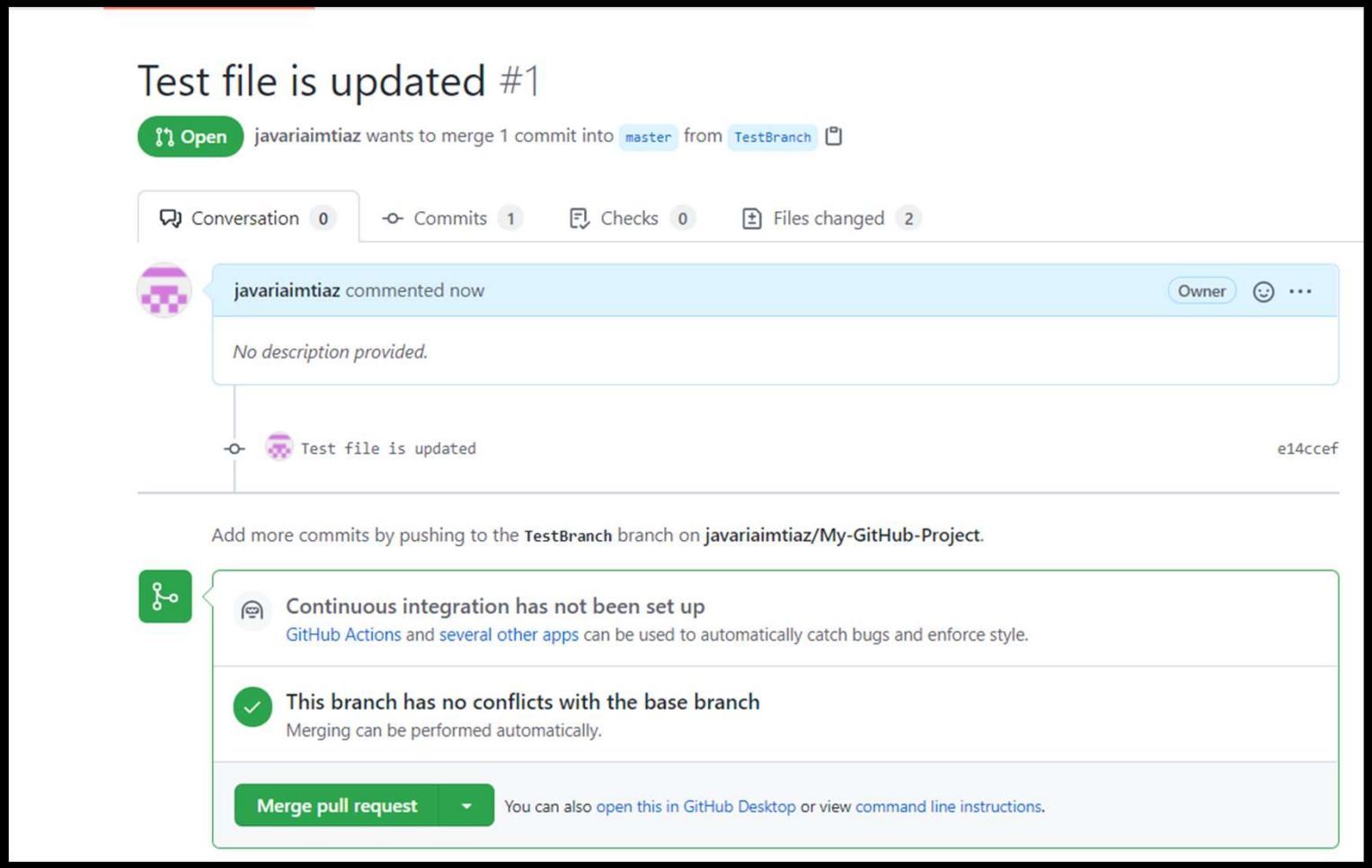
0 config.txt

Empty file.

1 testfile.txt

```
@@ -0,0 +1 @@
+ Hello Students!
```

Step 9: Merge a PR



GitHub Repository View

Test file is updated #1

Merged javariaimtiaz merged 1 commit into master from TestBranch now

Conversation 0 | Commits 1 | Checks 0 | Files changed 2

javariaimtiaz commented 2 minutes ago Owner

No description provided.

Test file is updated e14ccef

javariaimtiaz merged commit 04cdca4 into master now Revert

Pull request successfully merged and closed

You're all set—the TestBranch branch can be safely deleted.

Delete branch

GitHub Repository View

javariaimtiaz / My-GitHub-Project

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

master | 2 branches | 0 tags | Go to file | Add file | Code

javariaimtiaz Merge pull request #1 from javariaimtiaz/TestBranch ... 04cdca4 4 minutes ago 3 commits

.settings	ReadMe.txt file is added!	1 hour ago
bin	ReadMe.txt file is added!	1 hour ago
src	ReadMe.txt file is added!	1 hour ago
.classpath	ReadMe.txt file is added!	1 hour ago
.project	ReadMe.txt file is added!	1 hour ago
config.txt	Test file is updated	8 minutes ago
readme.txt	ReadMe.txt file is added!	1 hour ago
testfile.txt	Test file is updated	8 minutes ago

Help people interested in this repository understand your project by adding a README. Add a README

Step 10: Get changes on GitHub back to your computer

- The repo on GitHub looks a little different than what you have on your local machine. For example, the commit you made in your branch and merged into the primary branch doesn't exist in the primary branch on your local machine.

- In order to get the most recent changes that you or others have merged on GitHub, use the git pull origin master command (when working on the primary branch).

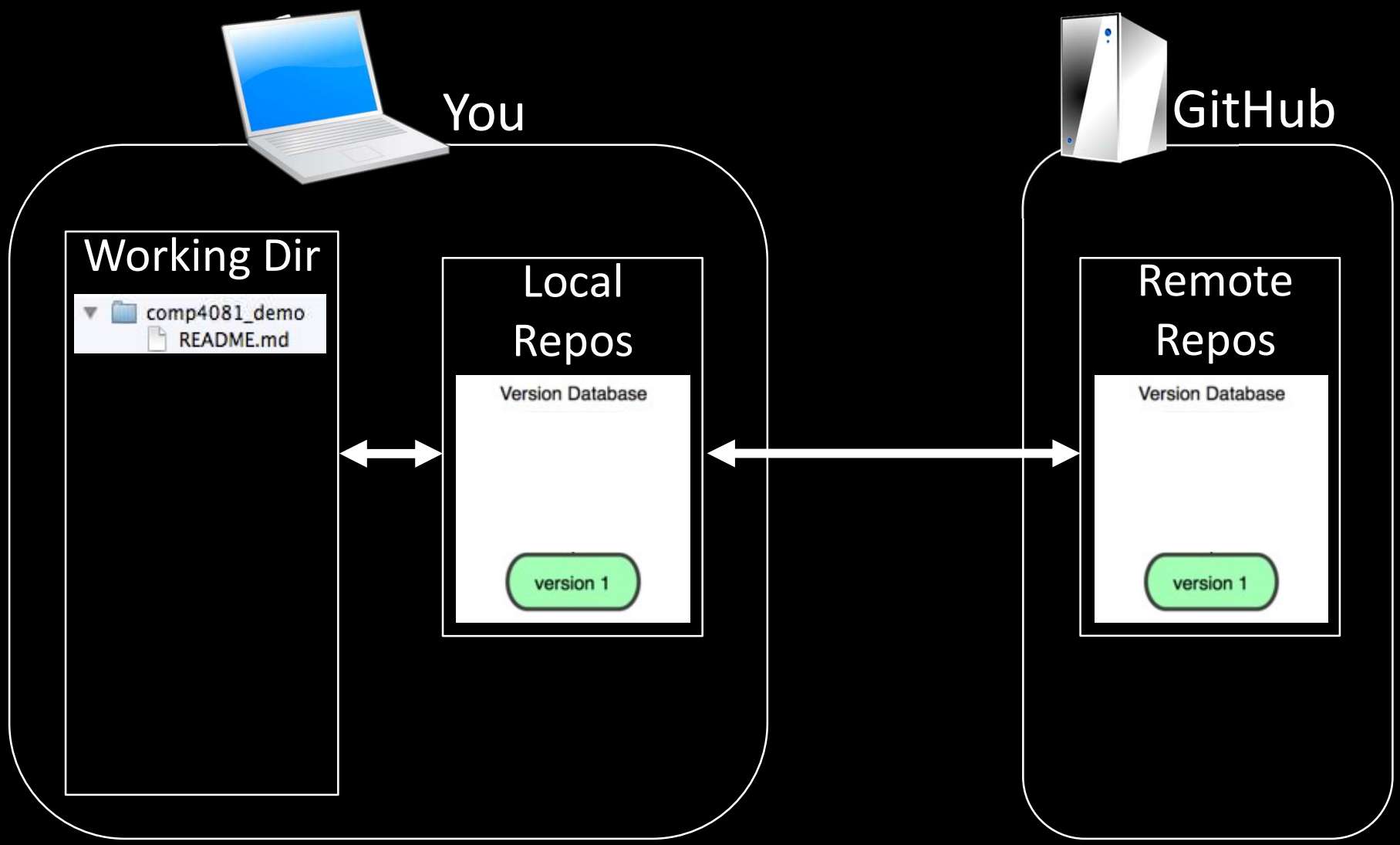
Get back changes to computer

MINGW64:/c/Users/hp/Documents/Game

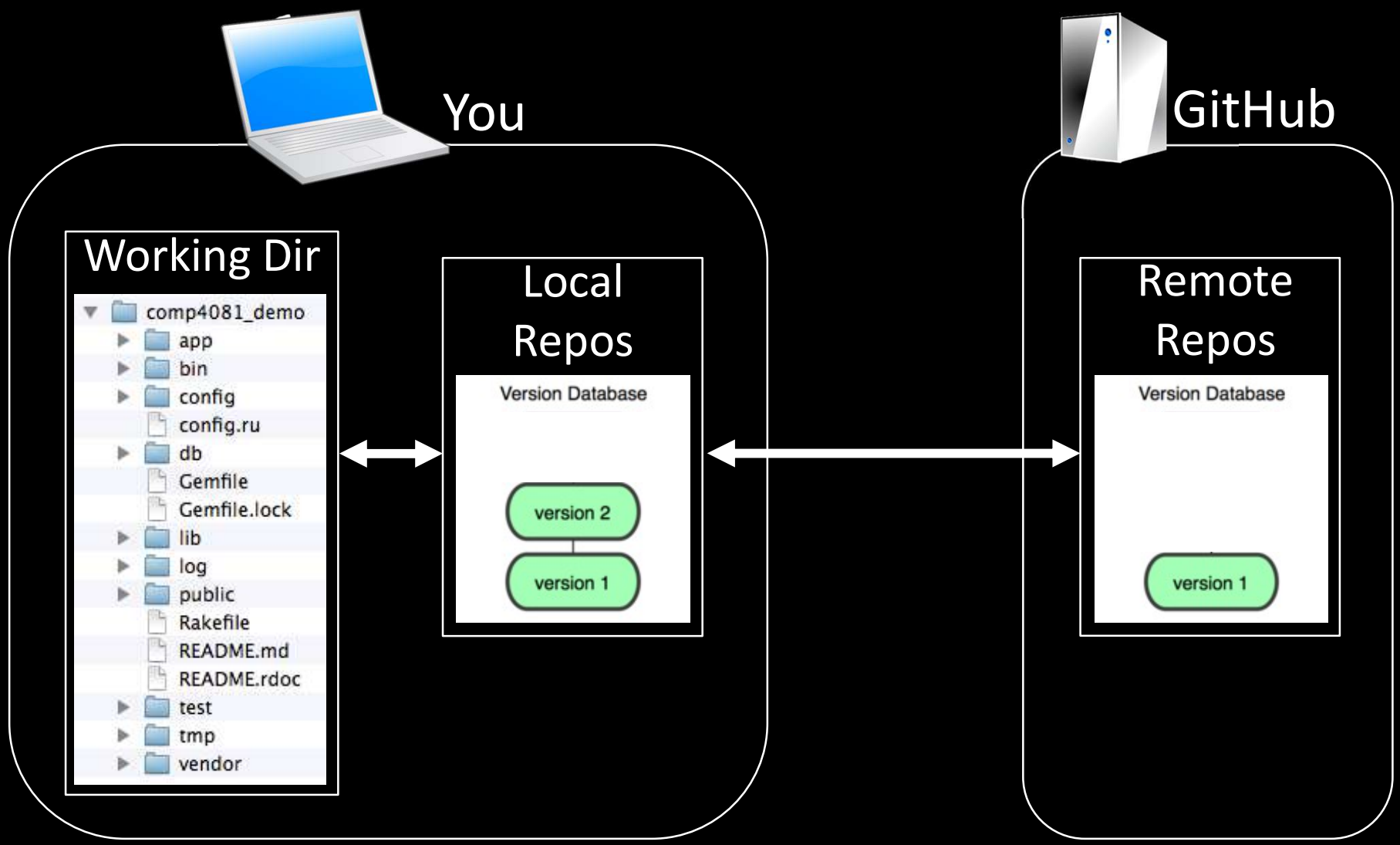
```
hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ ls
bin/  readme.txt  src/

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ git pull origin master
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), 637 bytes | 318.00 KiB/s, done.
From https://github.com/javariaimtiaz/My-GitHub-Project
 * branch            master     -> FETCH_HEAD
   1dbf500..04cdca4  master     -> origin/master
Updating 1dbf500..04cdca4
Fast-forward
 config.txt   | 0
 testfile.txt | 1 +
 2 files changed, 1 insertion(+)
 create mode 100644 config.txt
 create mode 100644 testfile.txt

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ ls
bin/  config.txt  readme.txt  src/  testfile.txt

hp@LAPTOP-AFG15KLD MINGW64 ~/Documents/Game (master)
$ |
```

Clone existing Repository

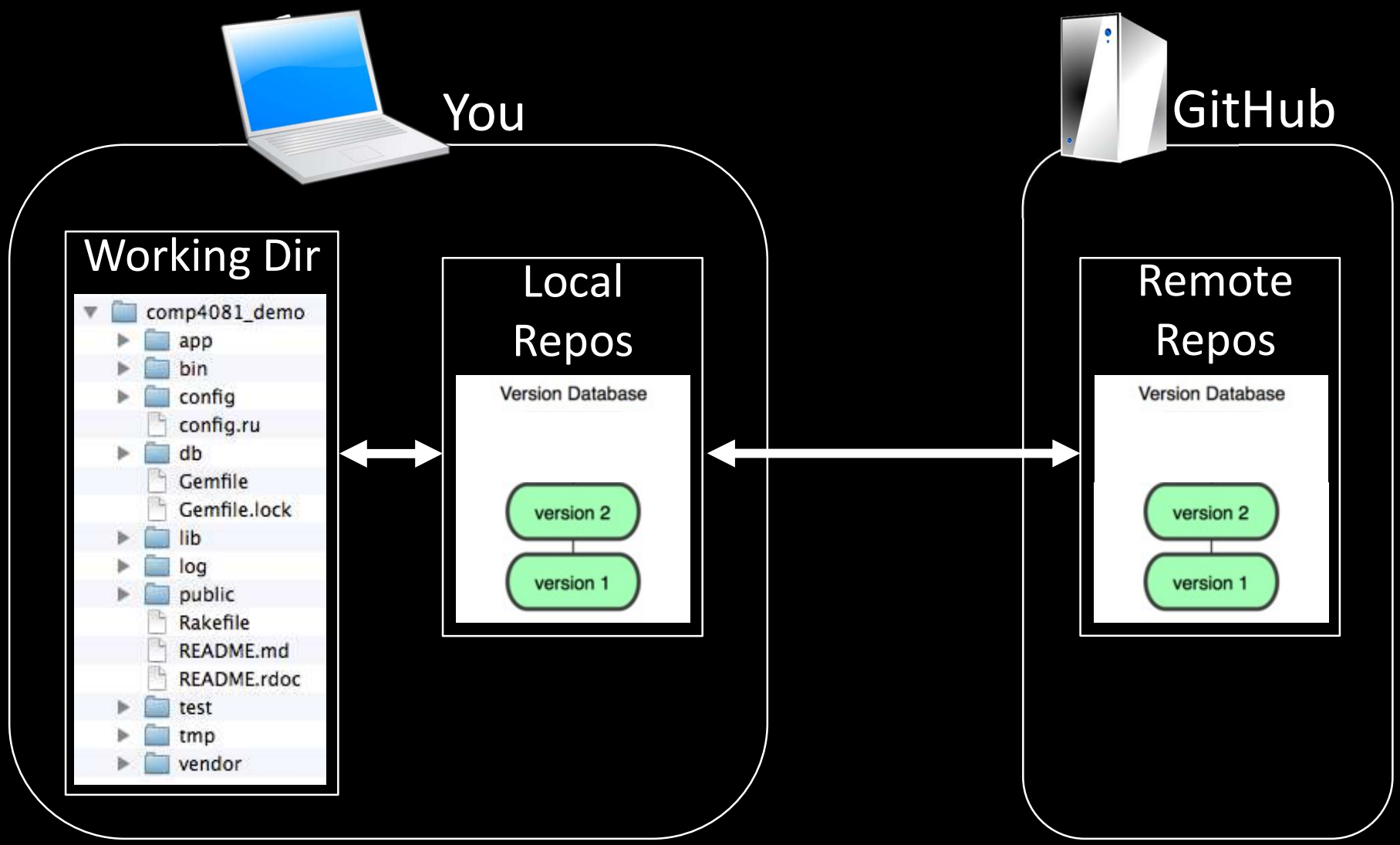
```
$ git clone https://github.com/sdflem/comp4081_demo.git
```



```
$ cd comp4081_demo
$ git add -A
$ git commit -m "Created java project"
```



```
$ git push
```


You
GitHub
Working Dir
comp4081_demo
app
bin
config
config.ru
db
Gemfile
Gemfile.lock
lib
log
public
Rakefile
README.md
README.rdoc
test
tmp
vendor
Local Repos
Version Database
version 2
version 1
Remote Repos
Version Database
version 2
version 1

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

Make changes in local branch

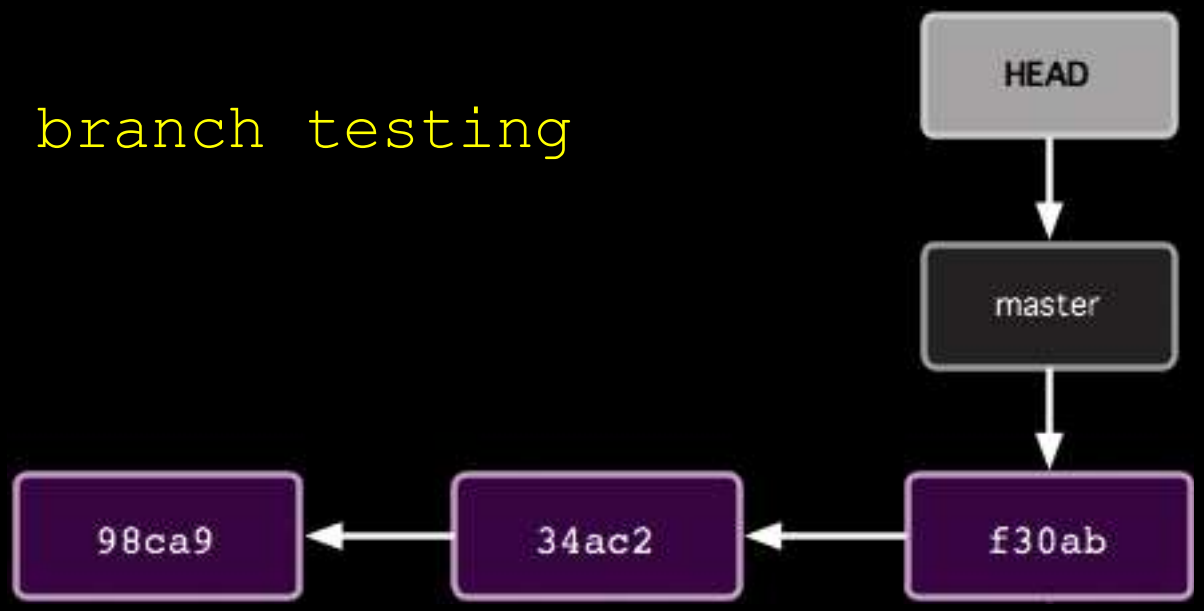
Merge with GitHub repos

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git branch works

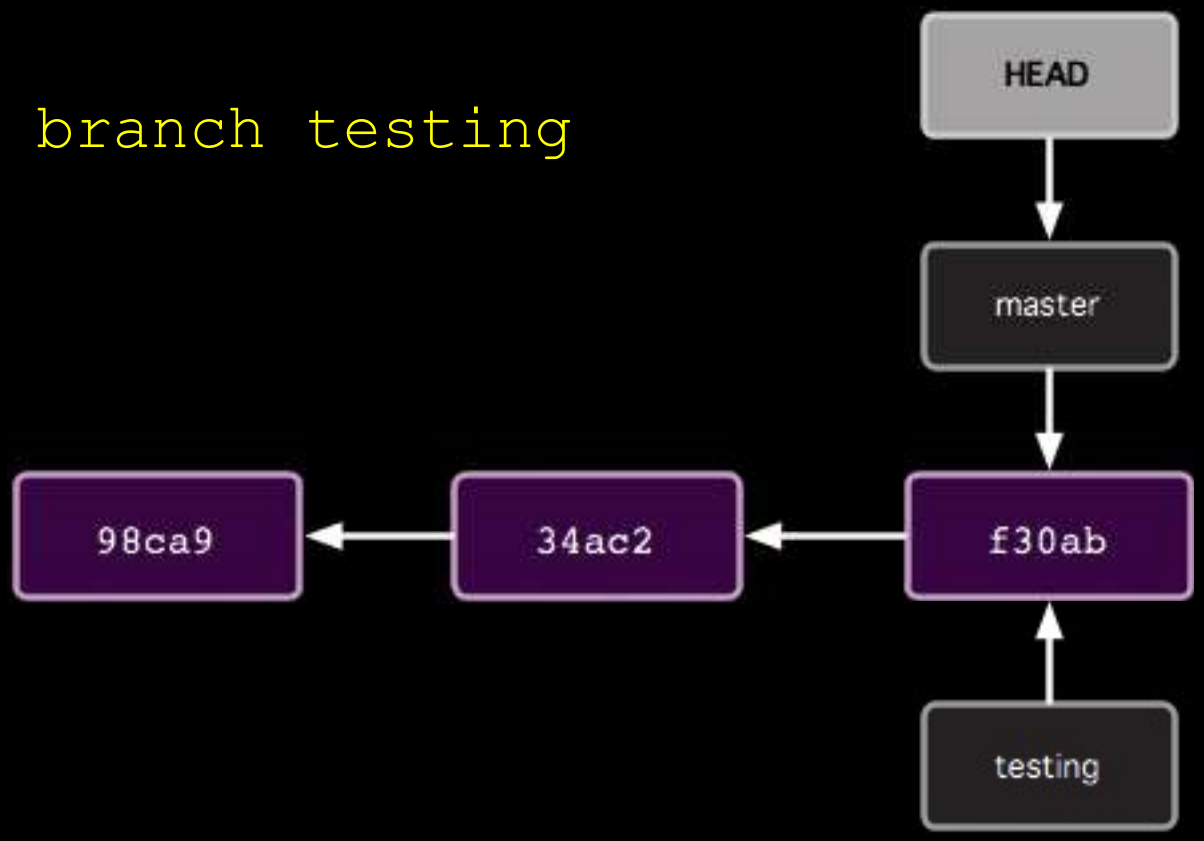
```
$ git branch testing
```



Before

How git branch works

```
$ git branch testing
```



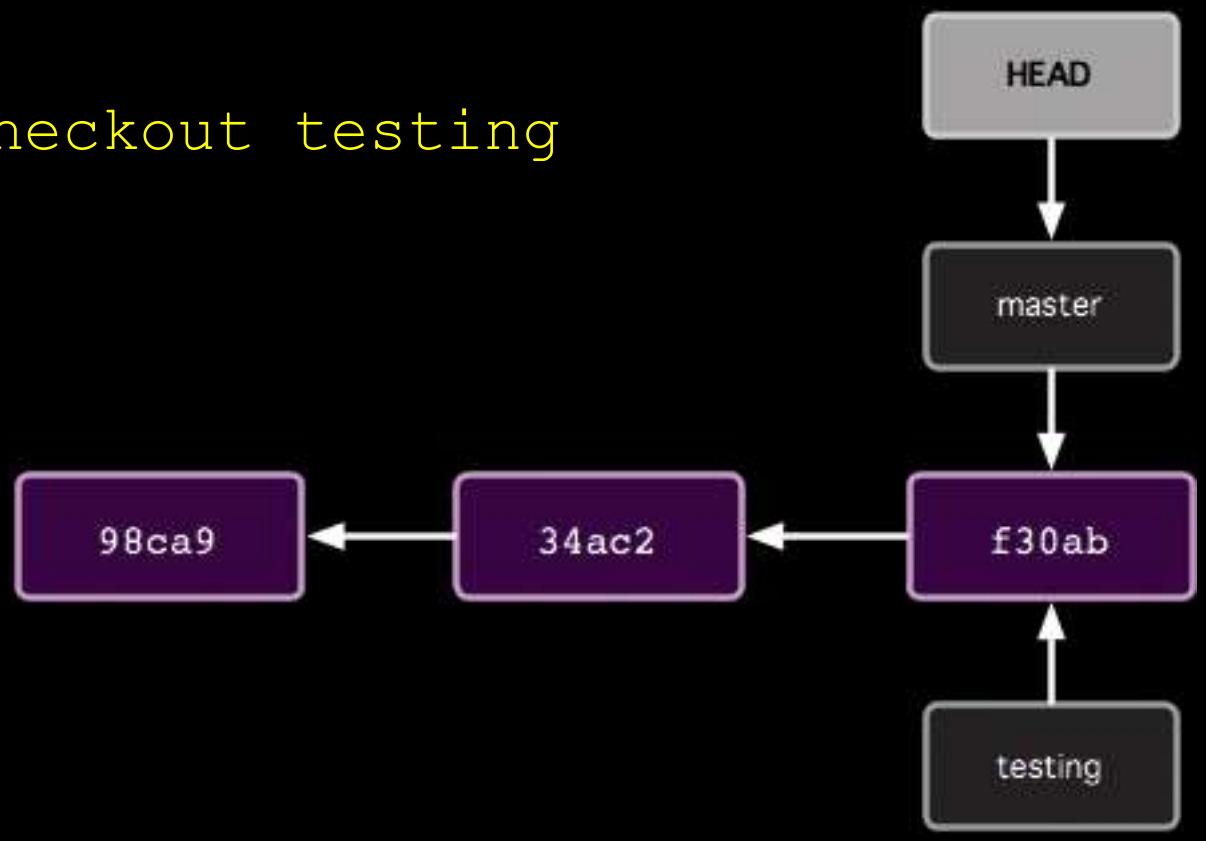
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git checkout works

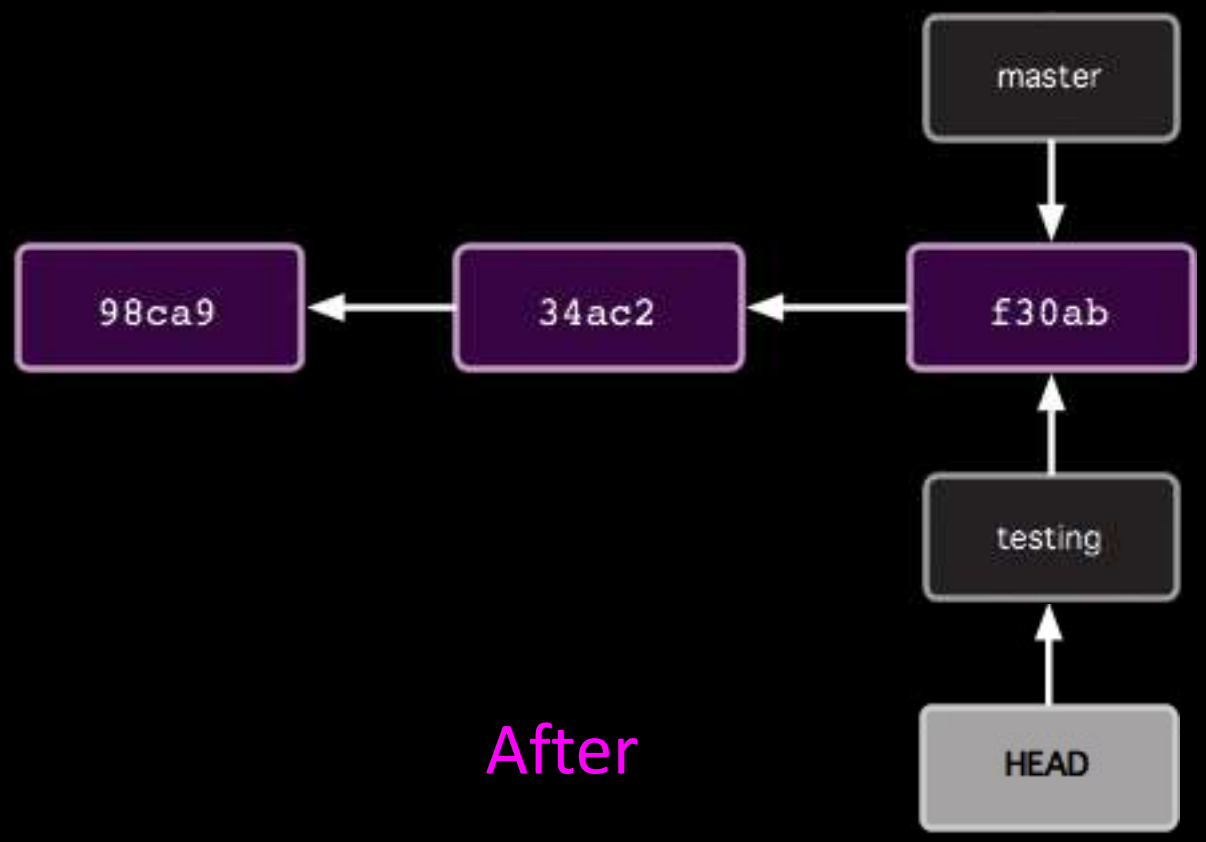
```
$ git checkout testing
```



Before

How git checkout works

```
$ git checkout testing
```



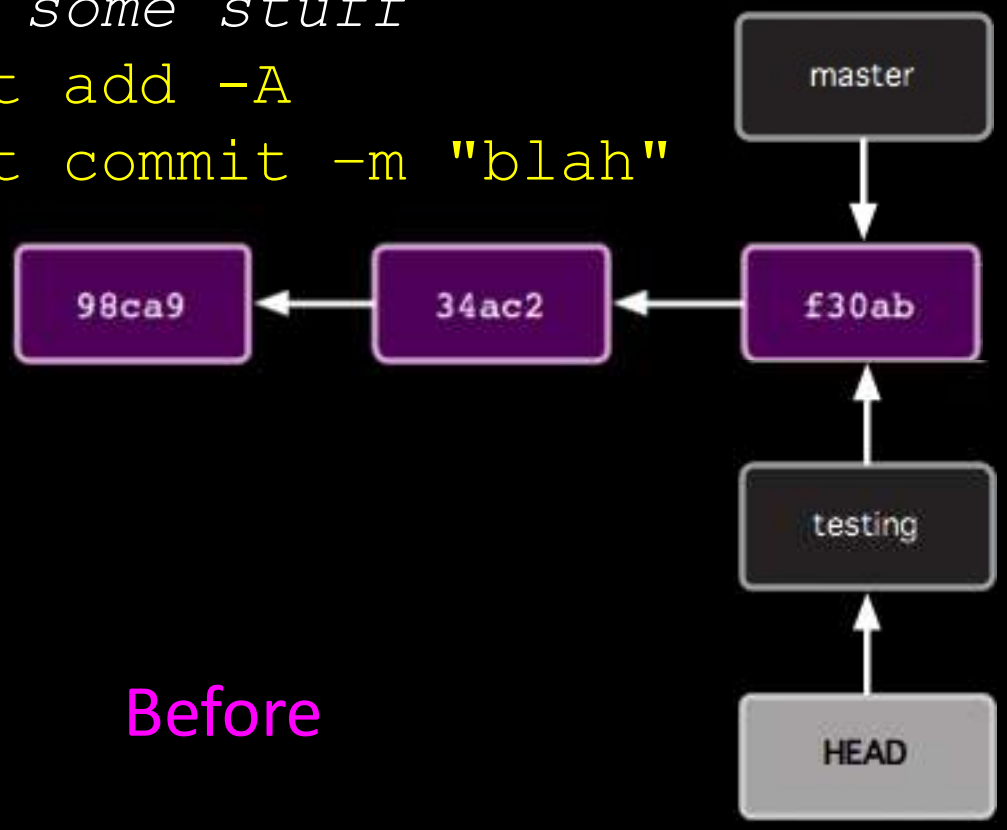
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git commit works with multiple branches

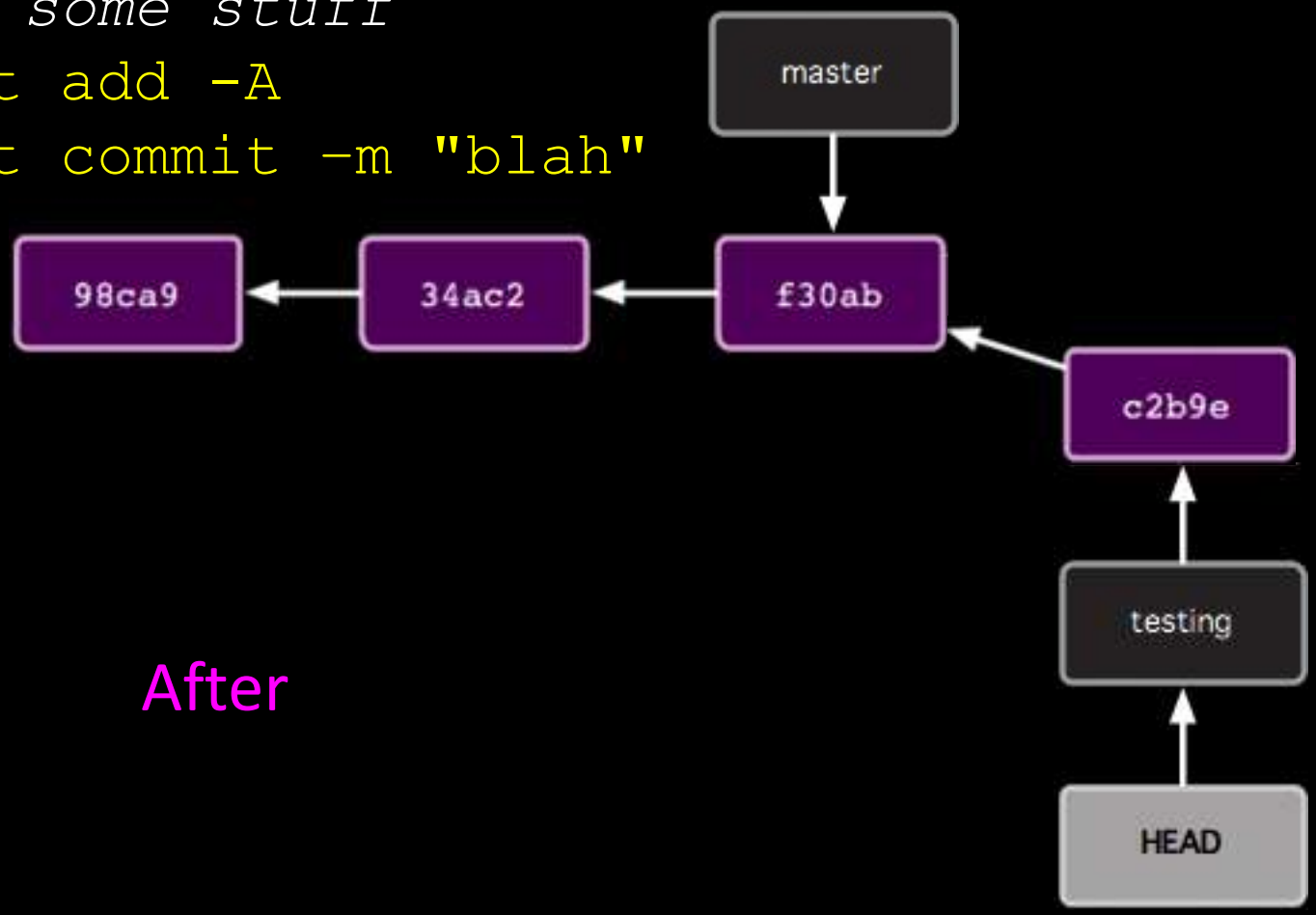
```
Edit some stuff
$ git add -A
$ git commit -m "blah"
```



Before

How git commit works with multiple branches

```
Edit some stuff
$ git add -A
$ git commit -m "blah"
```



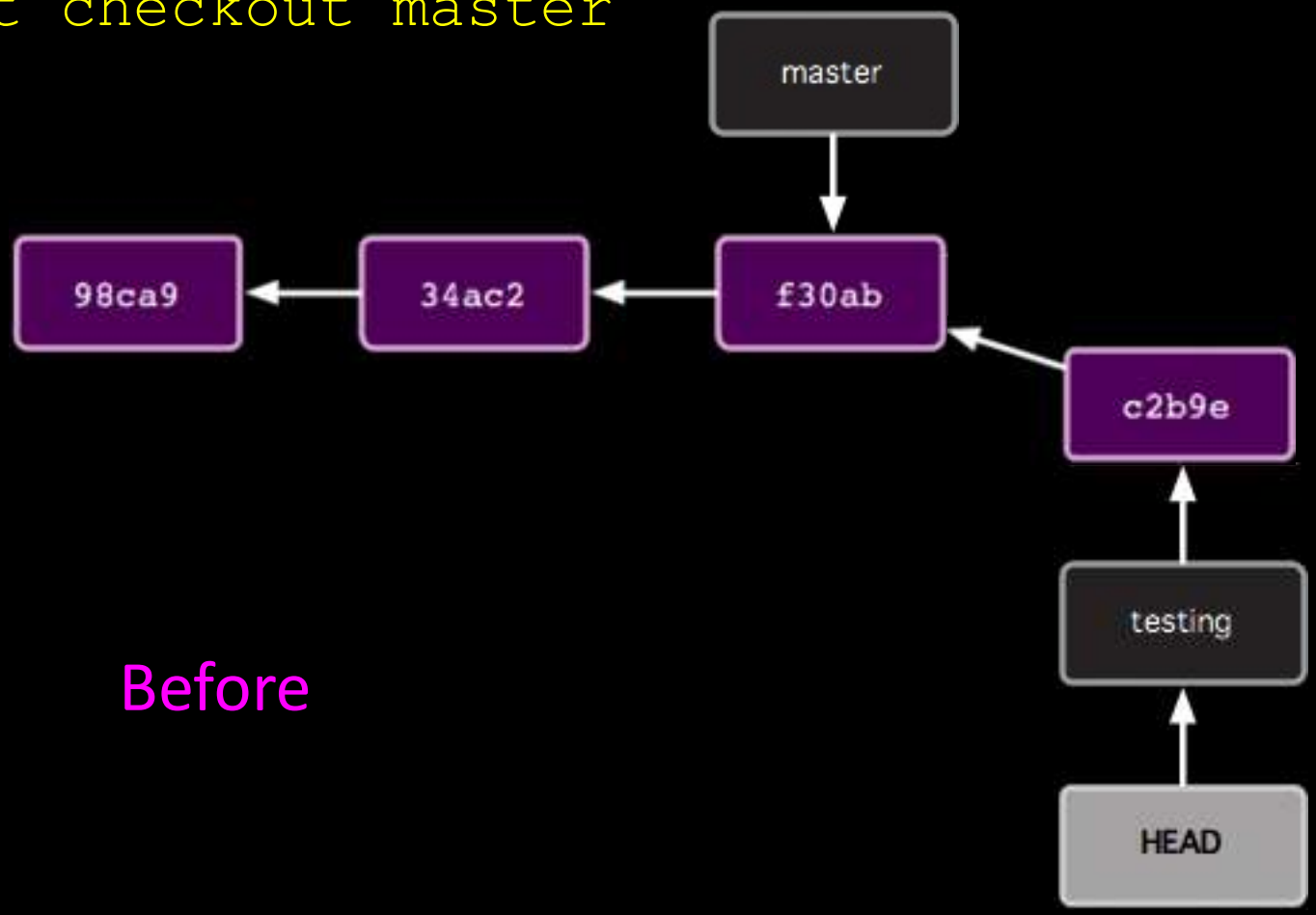
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git checkout works

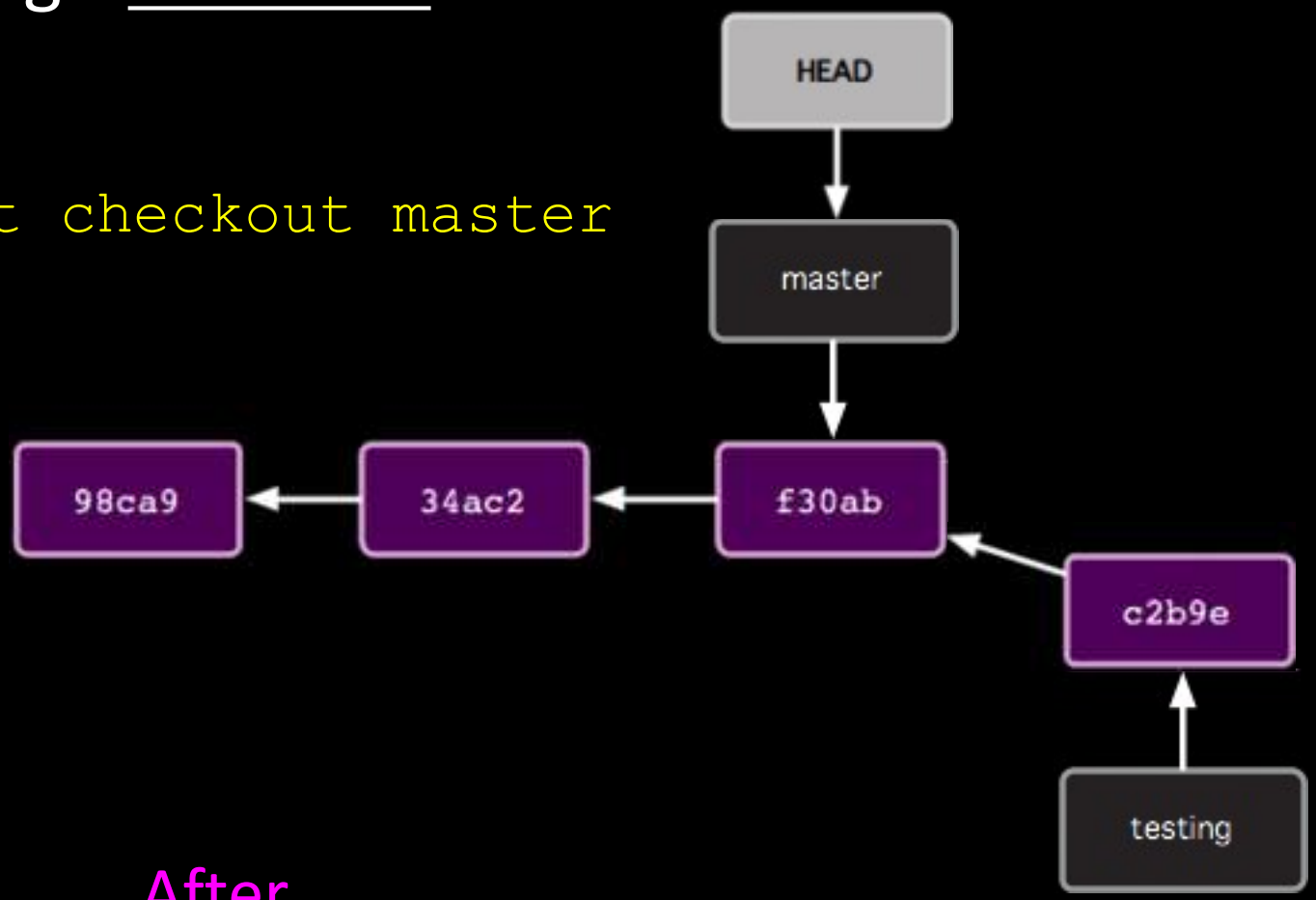
```
$ git checkout master
```



Before

How git checkout works

```
$ git checkout master
```



After

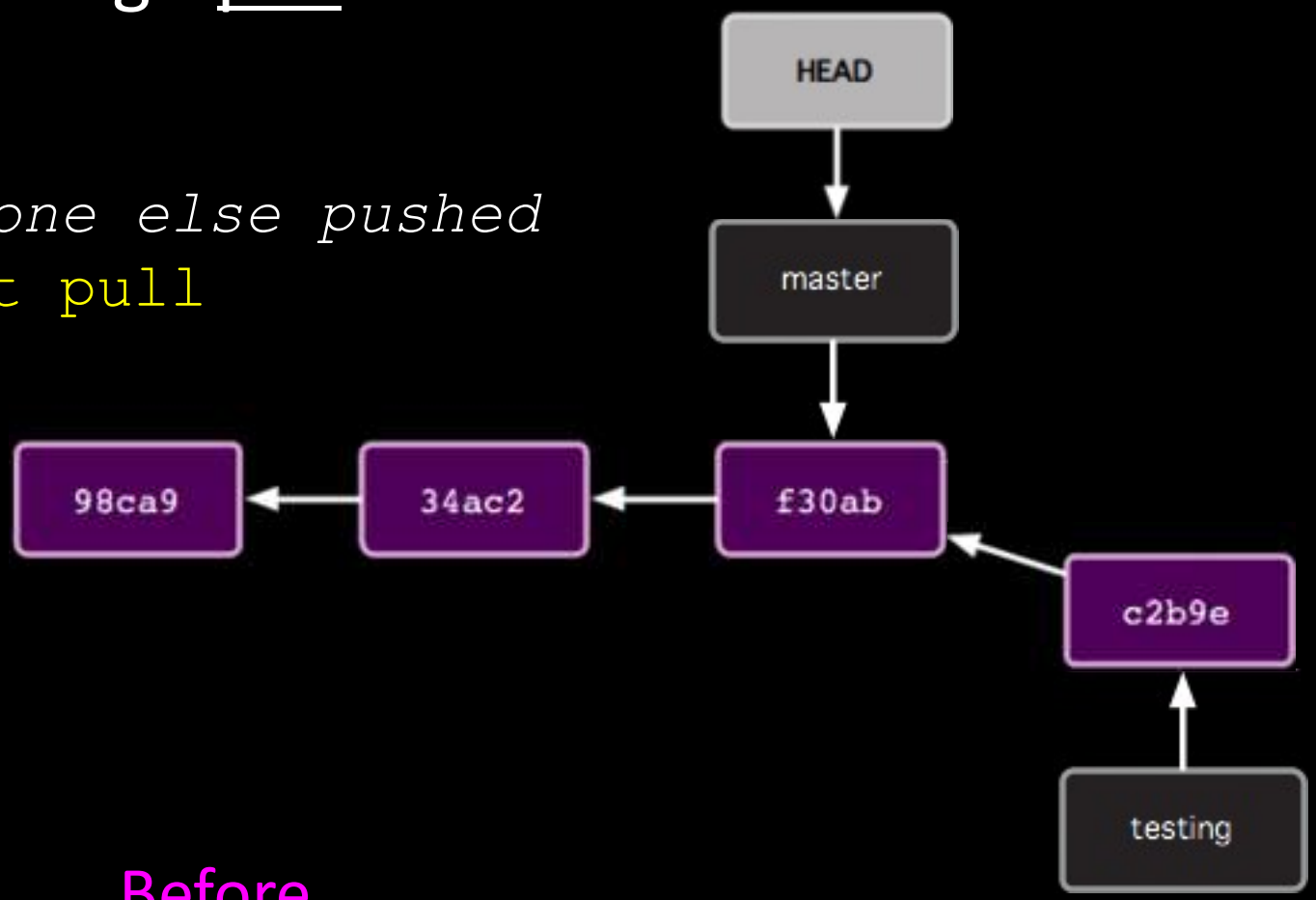
Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git pull works

Someone else pushed

```
$ git pull
```

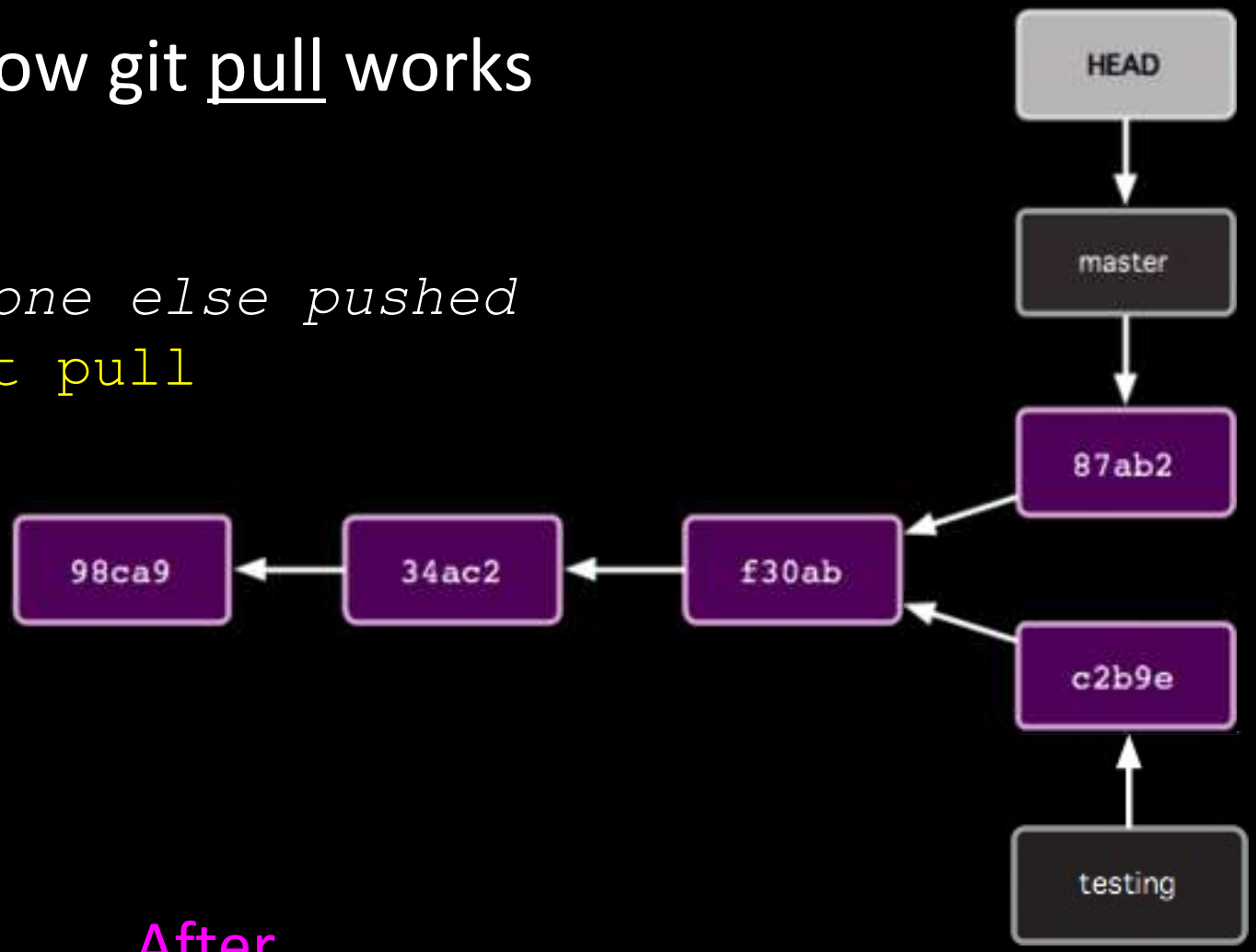


Before

How git pull works

Someone else pushed

```
$ git pull
```



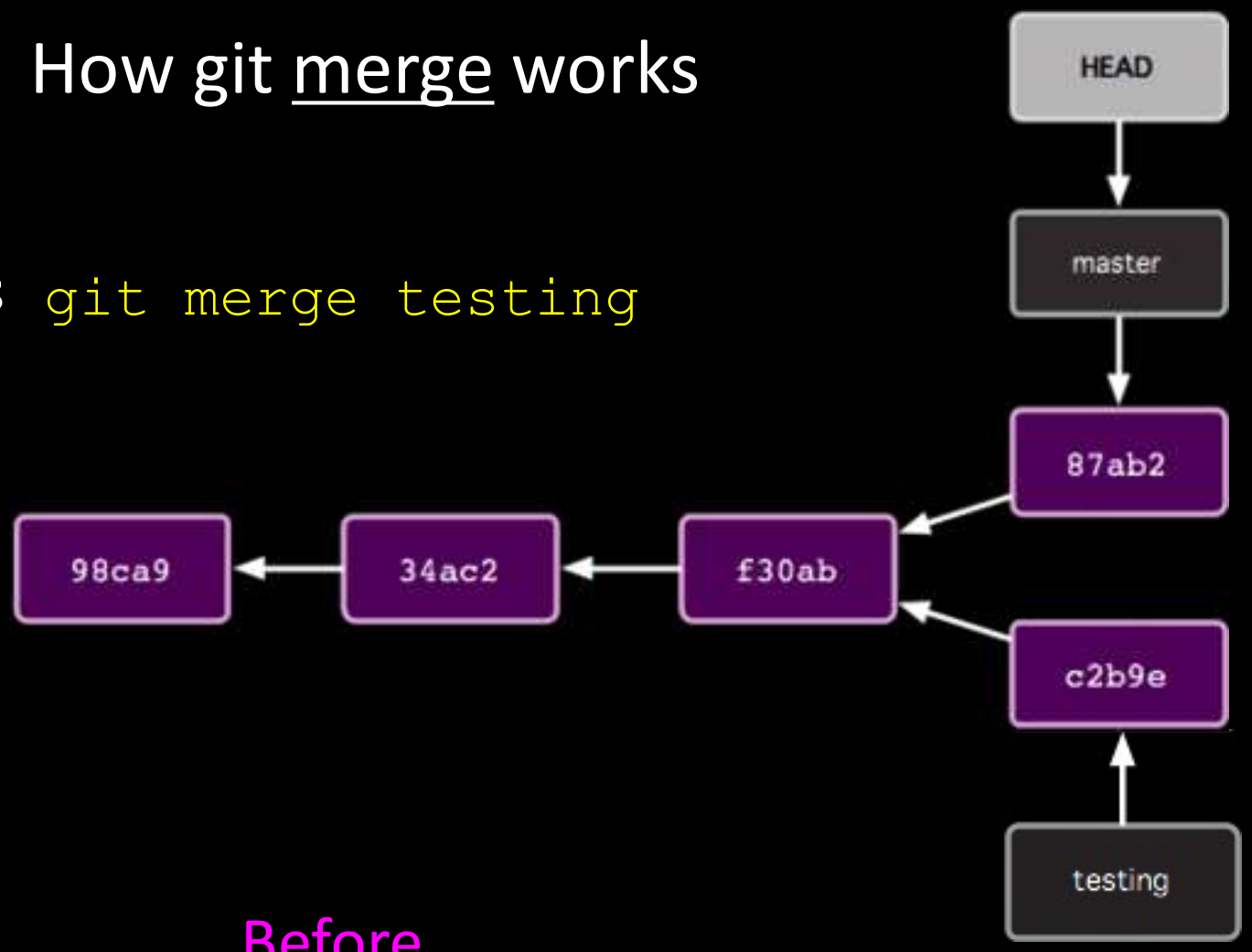
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git merge works

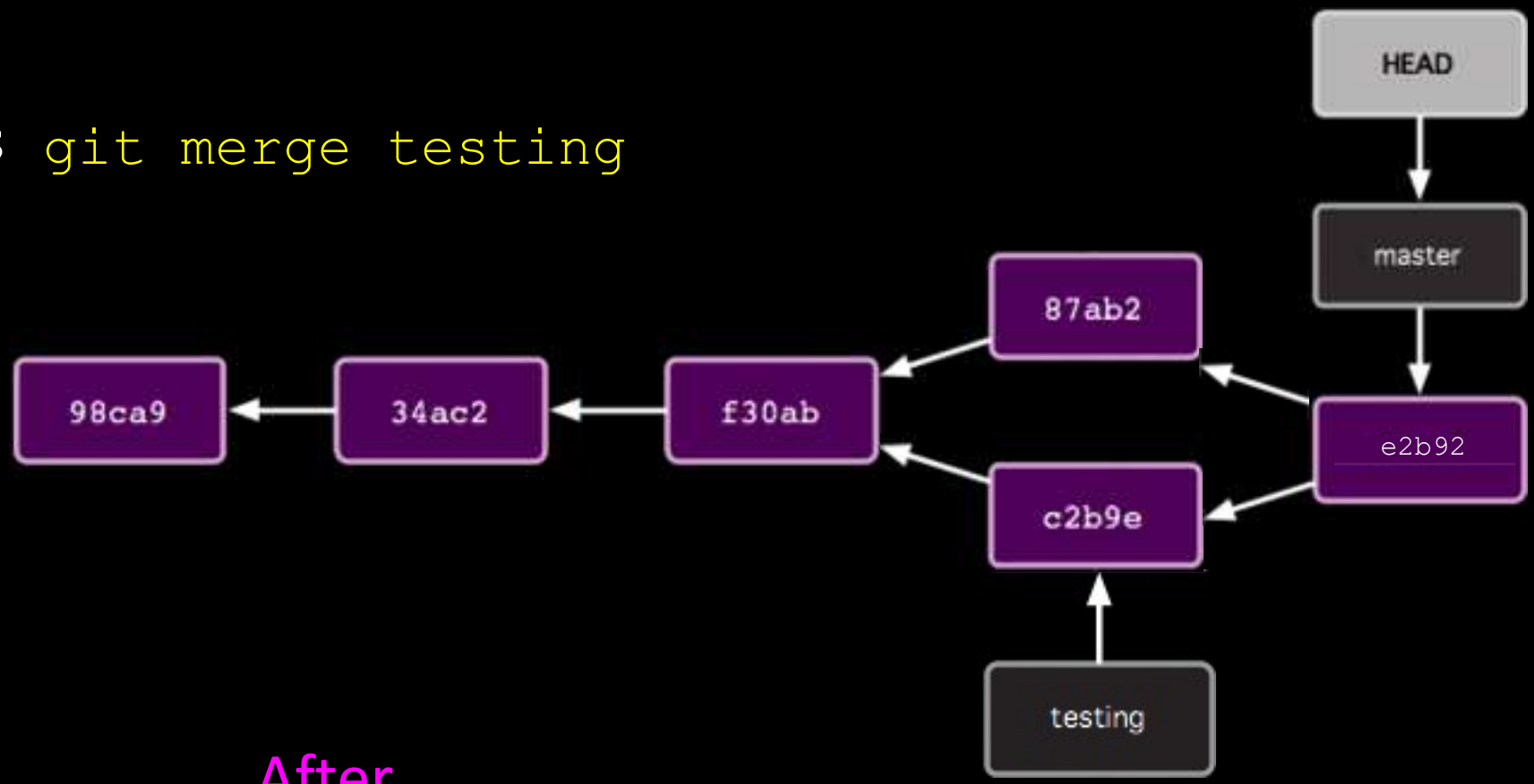
```
$ git merge testing
```



Before

How git merge works

```
$ git merge testing
```



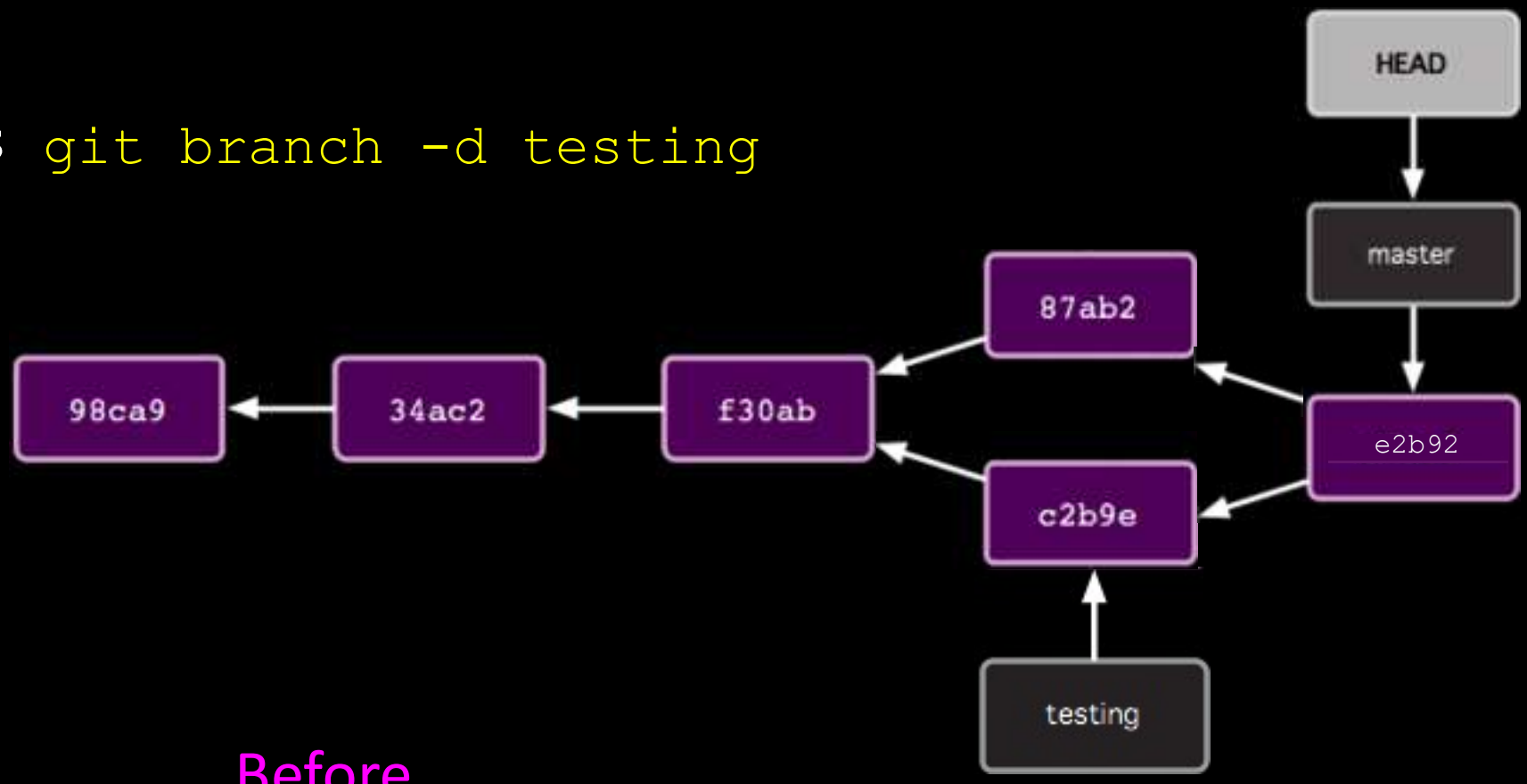
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How to delete branches

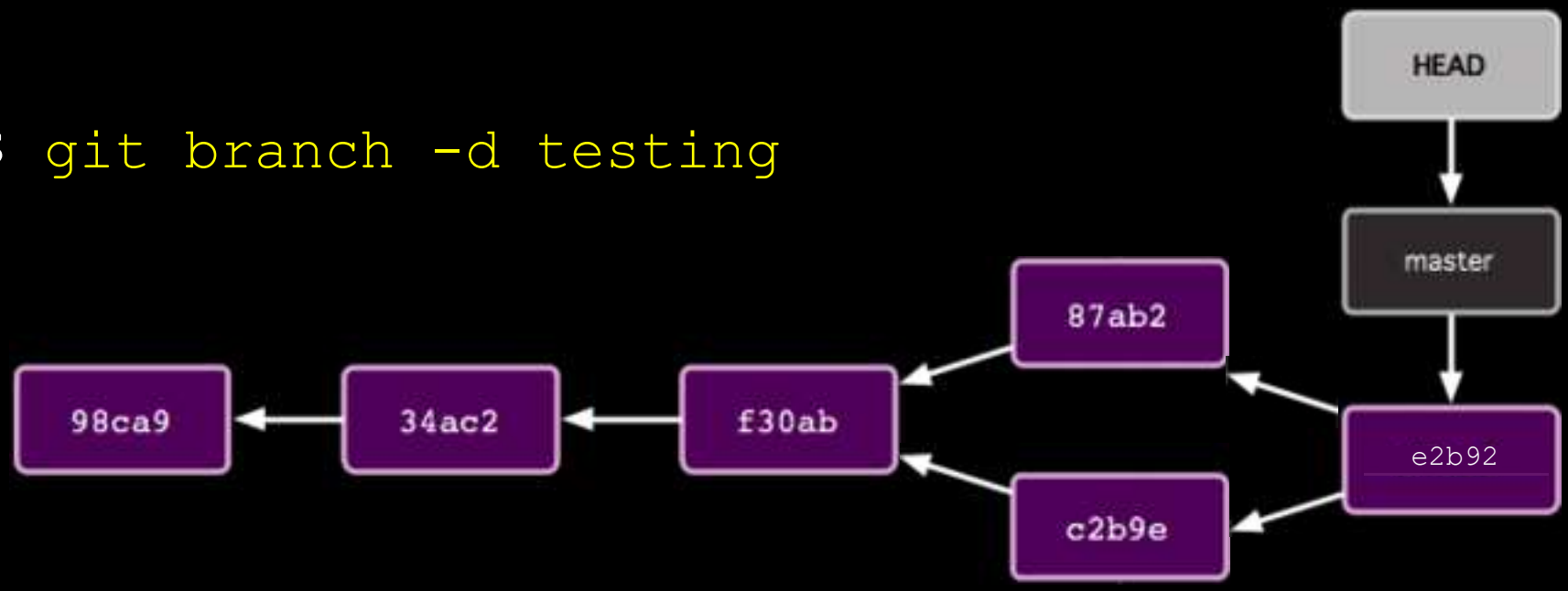
```
$ git branch -d testing
```



Before

How to delete branches

```
$ git branch -d testing
```



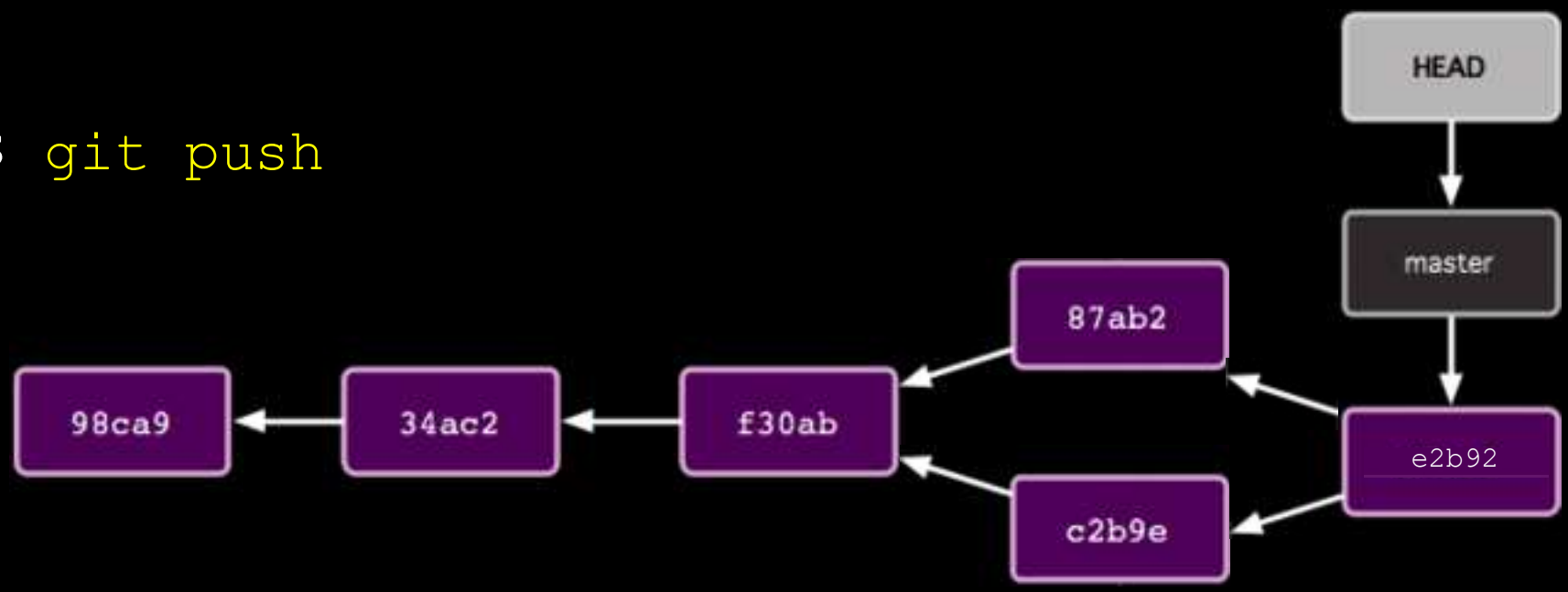
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git push works

```
$ git push
```



Should update server repos

(if no one else has pushed commits to master branch since last pull)

Task#01

- Configure Git on your machine
- Create a GitHub account and send me a link on Google Classroom (GC).
- Create a simple Java-based Account Management System and push it on your GitHub Account.
- Description of Account Management System is present on GC.