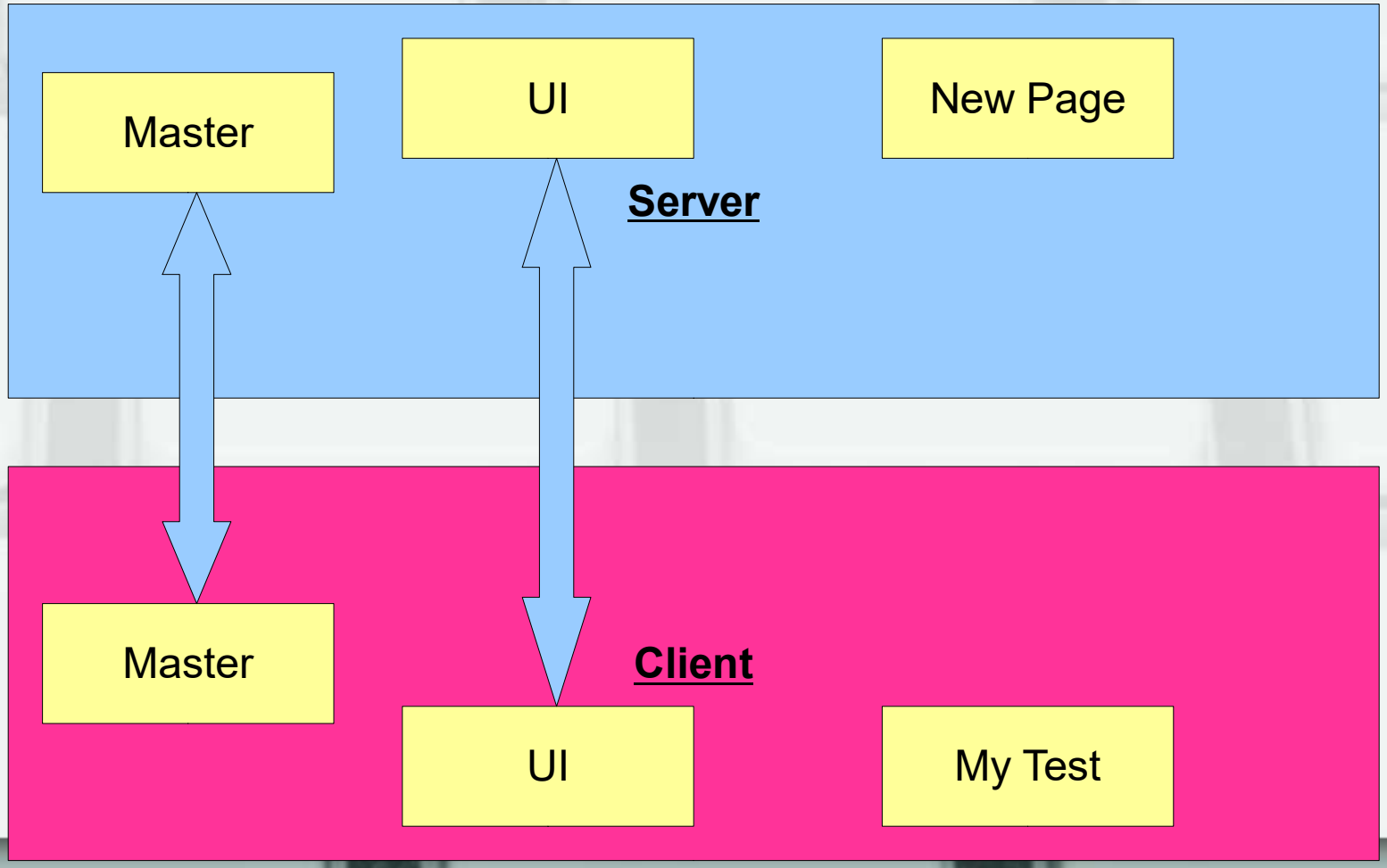
GIT – What is it?

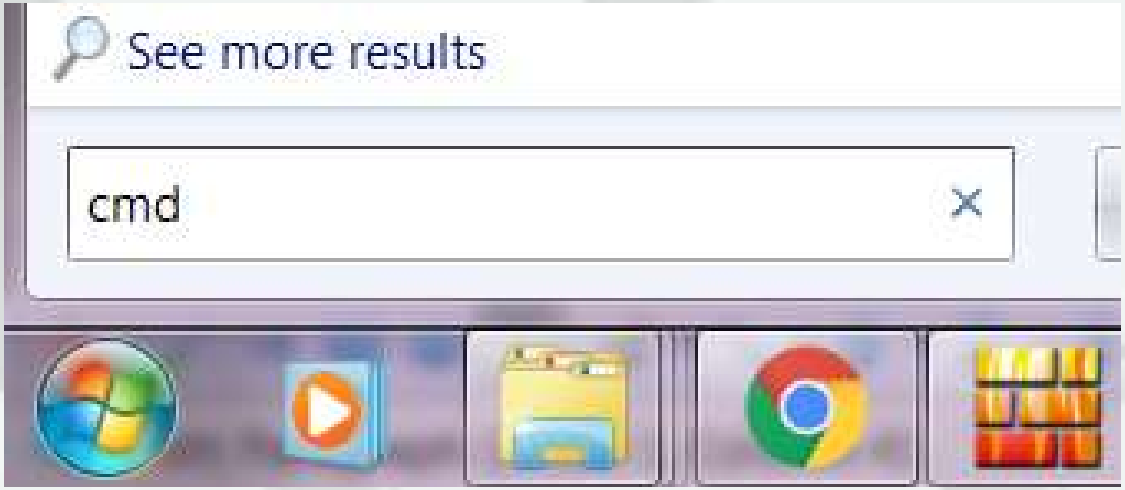
- It's used to share source code (Java code)
- It can version the code to make merging and rolling back changes easy
- You can develop code locally and push it to a remote server
- Remember: It's for “**source**” control, not “any random crap I have in my directory” control

GIT – What is it?



GIT – A Local repo

- Click Start, enter “cmd” and press Enter



GIT – A Local repo

- Create our repo
- mkdir hello_git
- cd hello_git

C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Andy>mkdir hello_git

C:\Users\Andy>cd hello_git

C:\Users\Andy\hello_git>_
```

GIT – A Local repo

- git init
- git status

```
C:\Users\Andy\hello_git>git init
Initialized empty Git repository in C:/Users/Andy/hello_git/.git/

C:\Users\Andy\hello_git>git status
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
```

GIT – Who am I?

- As GIT allows you to share code, you need to tell GIT who you are
- We wouldn't want people to confuse the code that Rob writes with the code I write
- git config --global user.email xxx
- git config --global user.name xxx

```
C:\Users\Andy\hello_git>git config --global user.email "turner.andy@gmail.com"

C:\Users\Andy\hello_git>git config --global user.name "Andy Turner"
```

GIT – Adding content

- echo “My first file” > file.txt

```
C:\Users\Andy\hello_git>echo "My first file" > file.txt

C:\Users\Andy\hello_git>dir
 Volume in drive C has no label.
 Volume Serial Number is 2619-73C0

 Directory of C:\Users\Andy\hello_git

08/07/2016  02:46 PM    <DIR>          .
08/07/2016  02:46 PM    <DIR>          ..
08/07/2016  02:46 PM                18 file.txt
               1 File(s)             18 bytes
               2 Dir(s)  28,306,505,728 bytes free
```

GIT – Adding content

- We now have a file in the directory, but it is not versioned by GIT
- We need to **add** the file to GIT
- git status

```
C:\Users\Andy\hello_git>git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        file.txt

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

GIT – Adding content

- We can add one file at a time, or add all of the files
- git add .
- git status

```
C:\Users\Andy\hello_git>git add .

C:\Users\Andy\hello_git>git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   file.txt
```

GIT – Adding content

- Our file is now in GIT, but we have not "committed" our work, so currently it is not versioned
- A "**commit**" takes a snapshot of the current files
- git commit -m "I added a text file"

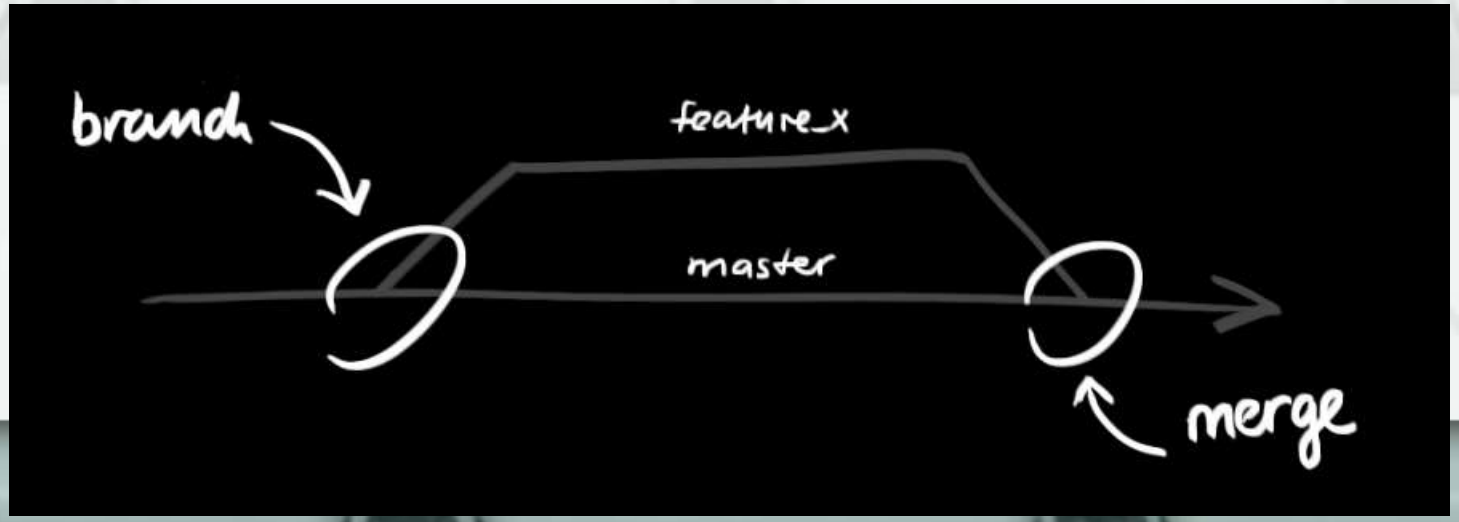
```
C:\Users\Andy\hello_git>git commit -m "I added a text file"
[master (root-commit) 33c74ef] I added a text file
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
```

GIT - Branching

- This is what GIT is all about
- **Branch** for EVERYTHING
- A branch is also called a feature, every feature you add to your program should be in its own branch
- Branch and commit EVERYDAY (that you program)
- A branch allows features to be added/removed easily and quickly. A big blob of 3 weeks of code is a PITA to merge

GIT - Branching

- The “**master**” is the main code base
- You create a branch to write/test your code
- Then **merge** your **branch** when you're finished



GIT - Branching

- git branch changing_text
- We created a branch, but we are not on it!
- git status

```
C:\Users\Andy\hello_git>git branch changing_text

C:\Users\Andy\hello_git>git status
On branch master
nothing to commit, working tree clean
```

GIT - Checkout

- **Checkout** changes the files on the disk to be the version that you checkout
- git checkout changing_text
- git status

```
C:\Users\Andy\hello_git>git checkout changing_text
Switched to branch 'changing_text'

C:\Users\Andy\hello_git>git status
On branch changing_text
nothing to commit, working tree clean
```

GIT – Change the file

- echo "Changed texttttt" > file.txt
- git status

```
C:\Users\Andy\hello_git>echo "Changed texttttt" > file.txt

C:\Users\Andy\hello_git>git status
On branch changing_text
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   file.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

GIT - reset

- I miss-typed, let's **reset** and try again
- type file.txt
- git reset –hard
- type file.txt

```
C:\Users\Andy\hello_git>type file.txt
"Changed texttttt"

C:\Users\Andy\hello_git>git reset --hard
HEAD is now at 33c74ef I added a text file

C:\Users\Andy\hello_git>type file.txt
"My first file"
```

GIT - commit

- **Add** the new files
- **Commit** the changes (to THIS branch)
- echo “Changed text” > file.txt
- git add .
- git commit -m “I updated the text”

```
C:\Users\Andy\hello_git>echo "Changed text" > file.txt

C:\Users\Andy\hello_git>type file.txt
"Changed text"

C:\Users\Andy\hello_git>git add .

C:\Users\Andy\hello_git>git commit -m "I updated the text"
[changing_text 552fc66] I updated the text
 1 file changed, 1 insertion(+), 1 deletion(-)
```

GIT – What is in the master?

- The original content is still there. We only changed our branch
- This allows us to develop
- type file.txt
- git checkout master
- type file.txt

```
C:\Users\Andy\hello_git>type file.txt
"Changed text"

C:\Users\Andy\hello_git>git checkout master
Switched to branch 'master'

C:\Users\Andy\hello_git>type file.txt
"My first file"
```

GIT – Where is our change?

- Still in our branch
- Err... what was the branch name?
- git branch
- git checkout changing_text
- type file.txt

```
C:\Users\Andy\hello_git>git branch
  changing_text
* master

C:\Users\Andy\hello_git>git checkout changing_text
Switched to branch 'changing_text'

C:\Users\Andy\hello_git>type file.txt
"Changed text"
```

GIT – Merge our change

- Checkout the branch you want to merge INTO
- git checkout master
- type file.txt
- git merge <the other branch>
- type file.txt

```
C:\Users\Andy\hello_git>git checkout master
Switched to branch 'master'

C:\Users\Andy\hello_git>type file.txt
"My first file"

C:\Users\Andy\hello_git>git merge changing_text
Updating 33c74ef..552fc66
Fast-forward
 file.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\Andy\hello_git>type file.txt
"Changed text"
```

GIT – We need the old version!

- I changed the text, but it's wrong
- We can **reset** to old versions

```
C:\Users\Andy\hello_git>git status
On branch master
nothing to commit, working tree clean

C:\Users\Andy\hello_git>type file.txt
"Changed text"

C:\Users\Andy\hello_git>git reset --hard HEAD~1
HEAD is now at 33c74ef I added a text file

C:\Users\Andy\hello_git>type file.txt
"My first file"
```

- ~1 is 1 version back, we can rollback ~n

GIT – How to share code

- My text file is so good everyone needs it
- We need to **push** to a remote server
- git push master

```
C:\Users\Andy\hello_git>git push master
fatal: The current branch master has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream master master
```

- We don't have anywhere to push to...

GitHub to the rescue!

- GitHub stores your code so that others can download it
- It supports all of GIT's features
- It has a UI to make merge etc. easier
- You can push your own branches to it as a backup

GIT – The Robot Project

- Make a new directory
- **Clone** the existing project into it
- cd %HOMEPATH%
- mkdir mr_roboto
- git clone https://github.com/Team2559/Normality-Zero-2016.git

```
C:\Users\Andy>mkdir mr_roboto

C:\Users\Andy>cd mr_roboto

C:\Users\Andy\mr_roboto>git clone https://github.com/Team2559/Normality-Zero-201
6.git
Cloning into 'Normality-Zero-2016'...
remote: Counting objects: 1595, done.
remote: Compressing objects: 100% (4/4), done.
emote: Total 1595 (delta 0), reused 0 (delta 0), pack-reused 1591
Receiving objects: 100% (1595/1595), 189.46 KiB | 0 bytes/s, done.
Resolving deltas: 100% (912/912), done.
Checking connectivity... done.
```

GIT – Don't want other to steal our code

- Login to get the code

GIT – We have the robot code

- We are currently on the master branch
- cd Normality-Zero-2016
- git status

```
C:\Users\Andy\mr_roboto>dir
 Volume in drive C has no label.
 Volume Serial Number is 2619-73C0

 Directory of C:\Users\Andy\mr_roboto

08/07/2016  05:14 PM    <DIR>          .
08/07/2016  05:14 PM    <DIR>          ..
08/07/2016  05:14 PM    <DIR>          Normality-Zero-2016
               0 File(s)              0 bytes
               3 Dir(s)  28,062,937,088 bytes free

C:\Users\Andy\mr_roboto>cd Normality-Zero-2016

C:\Users\Andy\mr_roboto\Normality-Zero-2016>git status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working tree clean
```

GIT – Let's add code to the robot

- git branch andy_robot
- git checkout andy_robot
- echo “hello” > andy.txt
- git add .
- git commit -m “Added a text file”

```
C:\Users\Andy\mr_roboto\Normality-Zero-2016>git branch andy_robot

C:\Users\Andy\mr_roboto\Normality-Zero-2016>git checkout andy_robot
Switched to branch 'andy_robot'

C:\Users\Andy\mr_roboto\Normality-Zero-2016>echo "hello" > andy.txt

C:\Users\Andy\mr_roboto\Normality-Zero-2016>git add .

C:\Users\Andy\mr_roboto\Normality-Zero-2016>git commit -m "Added a text file"
[andy_robot 0849fb9] Added a text file
 1 file changed, 1 insertion(+)
 create mode 100644 andy.txt
```

GIT – Now I can push my code! - almost

- git push
- We tried to push to the remote server, but we didn't say what remote branch to put out local branch in

```
C:\Users\Andy\mr_roboto\Normality-Zero-2016>git push
fatal: The current branch andy_robot has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin andy_robot
```

GIT - “upsteam” push

- We want to push our local code “upstream” to the remote server
- Generally call the remote branch the same as your local one
- git push --set-upstream origin andy_robot

```
C:\Users\Andy\mr_roboto\Normality-Zero-2016>git push --set-upstream origin andy_
robot
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 284 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To https://github.com/Team2559/Normality-Zero-2016.git
 * [new branch]      andy_robot -> andy_robot
Branch andy_robot set up to track remote branch andy_robot from origin.
```

GIT – what branches?

- git branch
  - Only the local branches!

```
C:\Users\Andy\mr_roboto\Normality-Zero-2016>git branch
* andy_robot
  master
```

- git branch -r

```
C:\Users\Andy\mr_roboto\Normality-Zero-2016>git branch -r
  origin/HEAD -> origin/master
  origin/No-Shooter
  origin/andy_robot
  origin/drive_pid
  origin/example_scheduler_push
  origin/increase_max_i
  origin/master
  origin/new_pid_controller
```

GIT - **pull**

- If someone else "**push**"ed their code, we can "**pull**" their code to see what they did
- git branch -r
- git pull origin drive_pid
- git branch

```
C:\Users\Andy\mr_roboto\Normality-Zero-2016>git branch -r
  origin/HEAD -> origin/master
  origin/No-Shooter
  origin/andy_robot
  origin/drive_pid
  origin/example_scheduler_push
  origin/increase_max_i
  origin/master
  origin/new_pid_controller

C:\Users\Andy\mr_roboto\Normality-Zero-2016>git pull origin drive_pid
From https://github.com/Team2559/Normality-Zero-2016
 * branch            drive_pid  -> FETCH_HEAD
Already up-to-date.
```

GIT - CLI

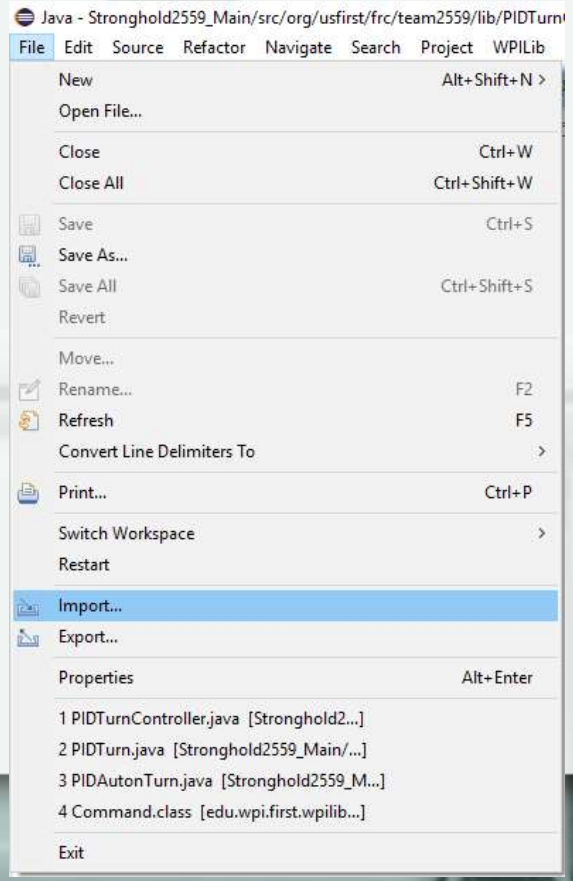

IF YOU DON'T LEARN GIT CLI
INSTRUCTOR
YOU'RE GONNA HAVE A BAD TIME

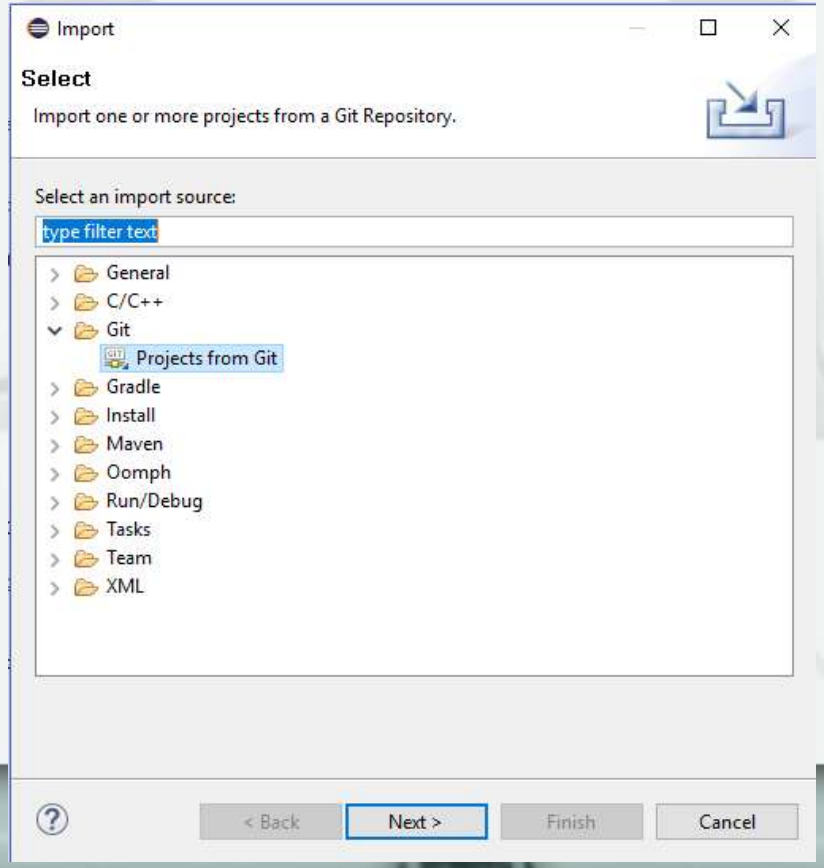
GIT - Eclipse

- When Eclipse GIT gets stuck:
  - Close Eclipse
  - Fix GIT via the CLI
    - Add/commit
    - Rollback etc.
  - Re-open Eclipse
  - Refresh the project

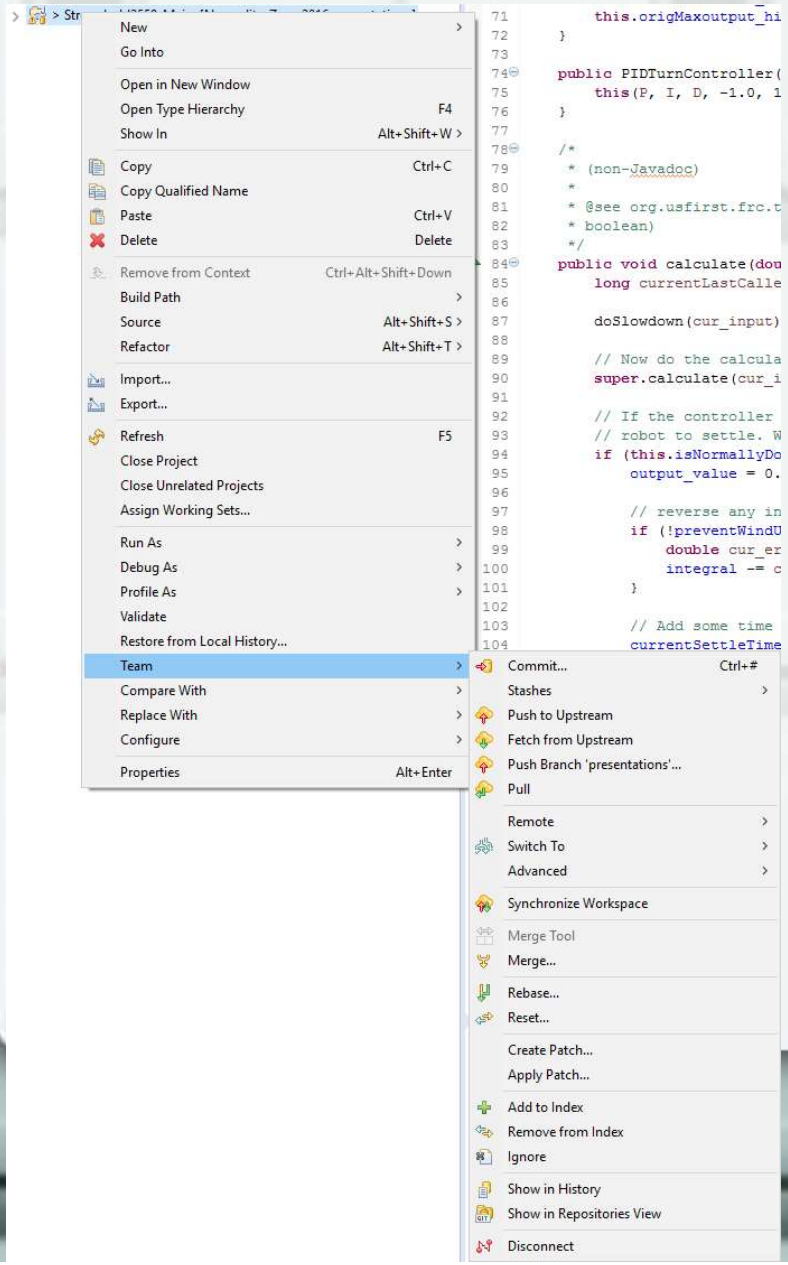
GIT - Eclipse

- File > Import





GIT - Eclipse



- Right-click project
- Team >

Conflict

- If both you and someone else edit and commit the same line of code there will be a conflict
- This is why it is best to branch, commit, and merge frequently
- To fix a conflict you need to manually edit each conflict and choose the code that you want to commit
  - That's why it's best avoided
- https://help.github.com/articles/resolving-a-merge-conflict-from-the-command-line/

Conflict

```
$ git status
# On branch branch-b
# You have unmerged paths.
#   (fix conflicts and run "git commit")
#
# Unmerged paths:
#   (use "git add ..." to mark resolution)
#
# both modified:      planets.md
#
no changes added to commit (use "git add" and/or "git commit -a")
```

Conflict

```
the number of planets are
<<<<<<< HEAD
nine
=======
eight
>>>>>>> branch-a
```

TestClass.java

```
package com.sap.text;

public class TestClass {

<<<<<<< OURS
    void doSomethingElse() {
=======
    void doSomething() {
>>>>>>> THEIRS

    }
}
```