Using Github for research and life

Matt Malishev

November 15, 2018

What the hell is Github? | And git

Applications

But why?

- Reproducible
- Unlimited
- Transparent
- Shareable

Best practice for git prep

Avoid spaces and CamelCase

- e.g. ‘my data.csv’, ‘My Data.csv’

Annotation

```
p <- rep(rnorm(100),20) # this is well annotated code
```

Tab is your friend

Useful syntax

`cd` set working dir

`pwd` print current working dir

`ls` list files in working dir

`mkdir newfolder` make new working dir

`touch text.txt` create new file

More useful syntax

`cp source destination`
copy files from *source* to *destination*. e.g. cp /Users/mydir/README.txt ~/Documents

`cp -R source destination`
copy all folders, subfolders, and files from *source* to *destination*

`mv source destination`
move files or folders from *source* to *destination* (no need for `-R`)

`cp ~/Desktop/*.rtf ~/Documents`
move multiple files with the * wildcard, which copies all .rtf files. The tilde (~) symbol is a shortcut for your Home folder, which contains '/Desktop'.

```
mv ~/Desktop/MyFile.rtf ~/Desktop/MyFile-old.rtf
cp ~/Desktop/MyFile.rtf ~/Documents/MyFile-old.rtf
```
rename files

Let's git it

Initialising and using your repo

1. **Create a repo**

2. **Create and stage your files**

- `add` and `commit` your files

3. **Push to a remote github repo**

- `push` your files

1. Create a repo

Navigate to your new project folder in the command line

```
cd ~/Documents/
ls
```

Make a new project on your local comp

```
# create new project
### <b>
cd ~/Documents/
### </b>
# make a new folder
### <b>
mkdir newgit
### </b>
# navigate to that folder
### <b>
cd newgit/
ls
```

1. Create a repo (cont . . .)

Create a new file in the command line

```
# create new file
### <b>
touch test.txt
### </b>

# navigate to your new git repo
### <b>
pwd
cd ~/Documents/newgit/
### </b>

# move the new file into the git repo
### <b>
mv ~/Documents/test.txt ~/Documents/newgit/
ls
### </b>
```

2. Create and stage your files

Add the files in your folder to the local git repo

```
# add the files to the git
### <b>
git add . # the '.' adds everything
### </b>
git add test.txt # adds individual files
```

Stage the files for the commit

```
# add the files to the git
### <b>
git commit -m 'init commit' # -m adds a message
### </b>
```

We've now added and staged files to a local repo. Version control!

Let's check the changes

```
### <b>
```

3. Push to a remote github repo

Now we push the changes we made from our local repo to our Github cloud

First, copy the Github repo link you want to push to. Select either **https** or **SSH** (requires key access).

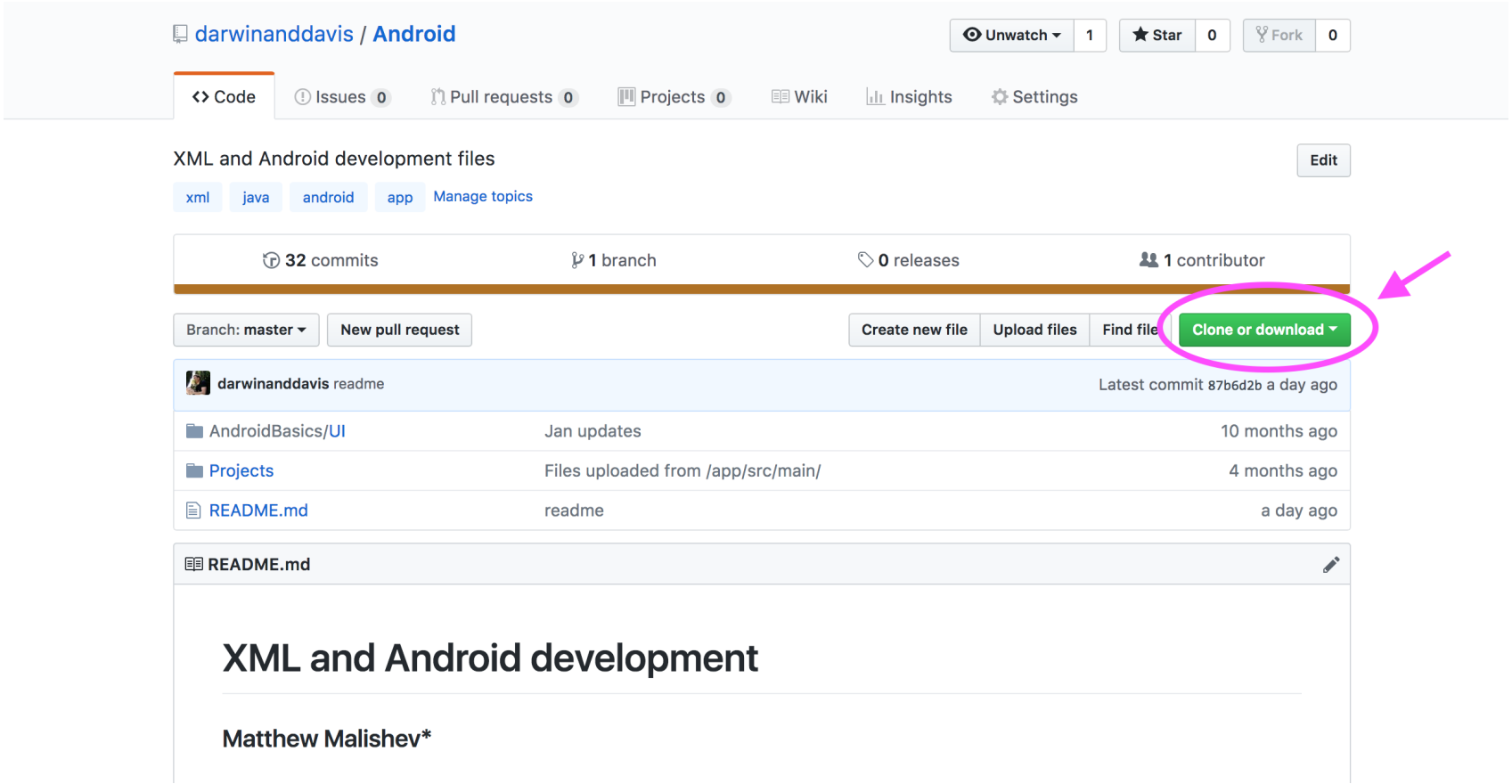

Figure 1: Copy the Github repo

3. Push to a remote github repo (cont . . .)

Then push your staged (`commit`) files from your local repo to the remote repo

```
# set the new remote repo
### <b>
git remote set-url origin "your github repo"
### </b>
# verify the remote repo
### <b>
git remote -v
### </b>
# push changes from local repo to remote repo
### <b>
git push -u origin master
### </b>
```

That's it! | Your data is now stored and version controlled in local and remote repos

Cloning an existing repo

Clone a remote repo to your local computer

This creates a git repository on your local machine complete with version control.

Every version of every file for the history of the project is grabbed by default when you run `git clone`.

```
git clone "github url" "new repo name (optional)"
# e.g. git clone https://github.com/darwinanddavis/UsefulC
```

Why clone?

You can dump the contents of any public repo, including its complete version history, onto your own computer, then upload it onto the cloud.

Now here's where it will git interesting

Troubleshooting

Staging and pushing files

Re-do a commit

```
git reset --soft HEAD~1
```

After pushing to your remote repo and this error appears:
`! [rejected] master -> master (fetch first)`

```
git fetch origin master # match the local repo commit stat
git merge master #
git push -u origin master

# for non-fast-forward error
git fetch origin master:tmp
git rebase tmp
git push origin HEAD:master
git branch -D tmp
git push -u origin master
```

Accessibility

If Github questions your user credentials.

```
git config --global user.email "<your email>"
git config --global user.name "<your github user name>"
```

When using SSH for your github remote repo, e.g.
`git@github.com:username/reponame.git`

Generating a new SSH key

Accessing your SSH key:
- In Mac, in *Terminal*, type

```
cat ~/.ssh/id_rsa.pub
```

- In Windows, in *cmd*, type

```
ls ~/.ssh/*.pub
```

Accessing commits

How to undo anything with Git

How to access recent commits to your local repo

```
git log # check recent activity and select commit e.g. 0df
git checkout "your commit"
git checkout master  # return to current branch
```

References