Using Github for research and life | **Part II**

Matt Malishev

@darwinanddavis

But why?

Reproducible

Unlimited

Transparent

Shareable

Why use the command line?

- Complete control and ease of workflow

- Transparent history

- Automate your commands, e.g. schedule an automatic daily

But why?

Reproducible

Unlimited

Transparent

Shareable

Why use the command line?

- Complete control and ease of workflow

- Transparent history

- Automate your commands, e.g. schedule an automatic daily

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 to your Github

1. Create a repo

Initialise your new local repo

```
# initialise your local git
### <b>
```

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 to your Github

1. Create a repo

Initialise your new local repo

```
# initialise your local git
### <b>
```

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 to your Github

1. Create a repo

Initialise your new local repo

```
# initialise your local git
### <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/UsefulCo
```

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.

Workflow example

Open an '.Rproj' with self-contained files

Troubleshooting

Common errors

fatal: remote origin already exists
The remote origin already exists, so you can't add it again

```
git remote rm origin # if origin already exists, remove it
git remote add origin "your github repo" # then re-add
git push origin master # then push again
```

! [rejected] master -> master (non-fast-forward) Someone else has made changes since your latest ones and git refuses to lose the commit, so won't push your new changes

```
git pull origin master # fetches any updates to online repo
```

Common errors (cont ...)

fatal: refusing to merge unrelated histories Usually associated with a README file on the Github repo

Troubleshooting

Common errors

fatal: remote origin already exists
The remote origin already exists, so you can't add it again

```
git remote rm origin # if origin already exists, remove it
git remote add origin "your github repo" # then re-add
git push origin master # then push again
```

! [rejected] master -> master (non-fast-forward) Someone else has made changes since your latest ones and git refuses to lose the commit, so won't push your new changes

```
git pull origin master # fetches any updates to online repo
```

Common errors (cont . . .)

fatal: refusing to merge unrelated histories Usually associated with a README file on the Github repo

Troubleshooting

Common errors

fatal: remote origin already exists
The remote origin already exists, so you can't add it again

```
git remote rm origin # if origin already exists, remove it
git remote add origin "your github repo" # then re-add
git push origin master # then push again
```

! [rejected] master -> master (non-fast-forward) Someone else has made changes since your latest ones and git refuses to lose the commit, so won't push your new changes

```
git pull origin master # fetches any updates to online repo
```

Common errors (cont ...)

fatal: refusing to merge unrelated histories Usually associated with a README file on the Github repo

Troubleshooting

Common errors

fatal: remote origin already exists

The remote origin already exists, so you can't add it again

```
git remote rm origin # if origin already exists, remove it
git remote add origin "your github repo" # then re-add
git push origin master # then push again
```

! [rejected] master -> master (non-fast-forward) Someone else has made changes since your latest ones and git refuses to lose the commit, so won't push your new changes

```
git pull origin master # fetches any updates to online repo
```

Common errors (cont . . .)

fatal: refusing to merge unrelated histories Usually associated with a README file on the Github repo

Troubleshooting

Common errors

fatal: remote origin already exists
The remote origin already exists, so you can't add it again

```
git remote rm origin # if origin already exists, remove it
git remote add origin "your github repo" # then re-add
git push origin master # then push again
```

! [rejected] master -> master (non-fast-forward) Someone else has made changes since your latest ones and git refuses to lose the commit, so won't push your new changes

```
git pull origin master # fetches any updates to online repo
```

Common errors (cont . . .)

fatal: refusing to merge unrelated histories Usually associated with a README file on the Github repo

Troubleshooting

Common errors

fatal: remote origin already exists
The remote origin already exists, so you can't add it again

```
git remote rm origin # if origin already exists, remove it
git remote add origin "your github repo" # then re-add
git push origin master # then push again
```

! [rejected] master -> master (non-fast-forward) Someone else has made changes since your latest ones and git refuses to lose the commit, so won't push your new changes

```
git pull origin master # fetches any updates to online repo
```

Common errors (cont ...)

fatal: refusing to merge unrelated histories Usually associated with a README file on the Github repo