INTRO TO WEB DEVELOPMENT

Shannon Turner

Twitter: @svt827

GitHub: @shannonturner

HEAR ME CODE

OBJECTIVE

- Copy a project from Github
- Make changes to that project
- Test those changes
- Upload your new version to Github
- Learn about virtual environments
- Learn about web development in Django

OVERVIEW: COPYING A PROJECT

- **Forking** means to make an exact copy of another person's Github repository and save it to your account

- **Cloning** means to download an exact copy of a Github repository and save it to your computer

HOW TO FORK A REPO

- Fork this: https://github.com/shannonturner/digital-resolution

GitHub, Inc. [US] https://github.com/shannonturner/digital-resolution

This repository Search | Explore | Gist | Blog | Help | shannonturner

shannonturner / digital-resolution | Unwatch 1 | Star 1 | Fork 2

Description | Website

Short description of this repository | Website for this repository (optional) | Save or Cancel

Code | Issues 0 | Pull Requests 0 | Wiki | Pulse | Graphs

Python 62.0% | HTML 38.0%

branch: master | digital-resolution / +

Added requirements.txt

shannonturner authored 22 minutes ago | latest commit bb288bce09

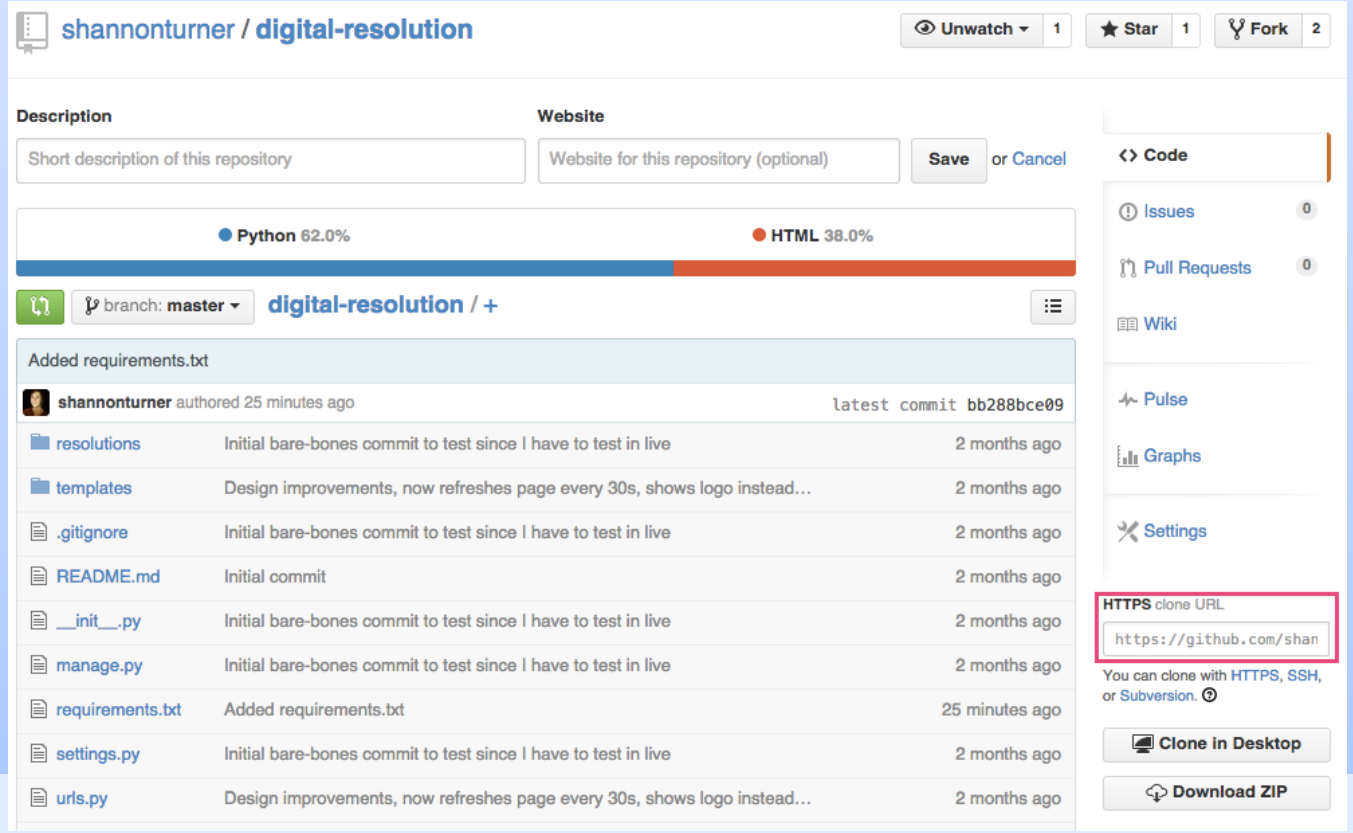
resolutions	Initial bare-bones commit to test since I have to test in live	2 months ago
templates	Design improvements, now refreshes page every 30s, shows logo instead…	2 months ago

WHY FORK?

- You can't make changes to my repos without my permission.
- But since my repos are public, you can fork (copy) mine and make changes to your copy all you want!
- If you make really cool changes, I could merge your changes into my original!

HOW TO CLONE A REPO

- In your newly-created fork, copy the Clone URL to the clipboard.



HOW TO CLONE A REPO

- In your newly-created fork, copy the Clone URL to the clipboard.
- In your terminal, navigate to the folder you'd like to store your local copy.
- **git clone** https://github.com/username/repository-you-want-to-clone.git

```
Shannons-MacBook-Air:github shannon$ git clone https://github.com/shannonturner/digital-resolution.git
```

HOW TO CLONE A REPO

- Did that fail? You'll need to install **git** onto your computer.
- Did that work? Congratulations! You just downloaded an exact copy of a repo onto your computer! (Now help a neighbor)
- Cloning creates a folder (the name of the folder is the name of the repo) inside your current folder, with the entire repo's contents.

WHY CLONE?

- Cloning creates a copy of the project on your computer.
- You can then run your local copy, make changes, and test those changes, all without affecting the original project.
- This is great for testing and for creating new features and designs!

VIRTUAL ENVIRONMENTS

- Before we can run the project, we'll need to set up a **virtual environment**.
- Virtual environments allow you to separate your different installations of python and python libraries from one another.
- This helps you avoid version conflicts and run projects after you've cloned them.

GET VIRTUALENV

- In the terminal, run **pip install virtualenv**
- If you're on a Mac, you'll probably need to run **sudo pip install virtualenv**
- This will install **virtualenv**, which will allow you to create new virtual environments.

VIRTUALENV

- We want to create a virtual environment for our new project.
- In my example, I was in the **github** folder when I cloned my **digital-resolution** repo.
- I'll want to navigate back to the **github** folder now. If you haven't changed folders since cloning, good!

CREATING A VIRTUALENV

- From the github folder, I'm going to run the following command in the terminal:
- **virtualenv digital-resolution**
- This creates a new virtualenv in the **digital-resolution** folder

WHAT DOES IT DO?

- When you run **virtualenv**, it creates a new, separate python installation in that folder.
- This installation creates new folders (**bin**, **include**, **lib**) inside **digital-resolution**
- You need to **activate** your virtualenv before you can use it for this project.

ACTIVATING YOUR VIRTUALENV

- To activate your virtualenv, change directory into the digital-resolution folder.
- From there, in the terminal, run:
- **source bin/activate**
- You'll notice your command prompt has changed a bit! This is a cue that your virtualenv is active.

INSTALLING REQUIREMENTS

- All of the necessary libraries needed to run **digital-resolutions** have been saved in a file called **requirements.txt**

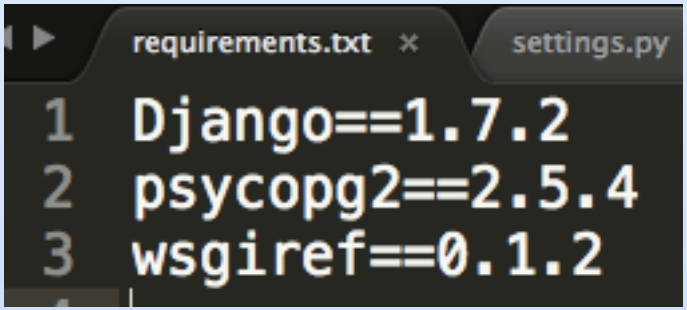
- Make sure your virtualenv is active, then in the terminal, run:

- **pip install -r requirements.txt**

INSTALLING REQUIREMENTS

- **pip install -r requirements.txt**
- This installs all of the python libraries this project needs into your virtual environment.
- **requirements.txt** is a special file that I included in the **digital-resolutions** repo to make it easier to install all the libraries you need.

INSTALLING REQUIREMENTS



- What's in requirements.txt?
- It's a list of names of python libraries and which versions to install.
- Python will auto generate this file when you run **pip freeze > requirements.txt** (don't do this now, but remember for later!)

MAKING A CHANGE

- My version of digital-resolutions uses a PostgreSQL database.
- Setting that up on our computers would take about 45 minutes by itself, so let's use a simpler solution.
- There's a type of database called SQLite that creates a small database file.

MAKING A CHANGE

- To use SQLite instead of PostgreSQL, we need to edit our **settings.py** file. Find the DATABASES variable (line 60) and change it to:

```
# Database
# https://docs.djangoproject.com/en/1.6/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

GIT NOTICES OUR CHANGES

- **Git** keeps track of any changes you've made to your files.

- In the terminal, **git status** will tell you which files you have modified since the time you cloned or last committed.

GIT STATUS

- **git status** also tells us
  - the branch we're on
  - hints on how to add files or to discard changes we've made.

```
(digital-resolution)Shannons-MacBook-Air:digital-resolution shannon$ git status
# On branch master
# 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:   settings.py
#
no changes added to commit (use "git add" and/or "git commit -a")
```

GIT DIFF

- To see the changes we've made in detail, you'll run **git diff**.
- **git diff** will show you all of the changes you've made to all of your files in this repo since your last commit.
- **git diff settings.py** will show you the changes you've made just to the **settings.py** file since your last commit.

GIT DIFF

```
(digital-resolution)Shannons-MacBook-Air:digital-resolution shannon$ git diff settings.py
diff --git a/settings.py b/settings.py
index 9bef4b4..2ca3f3d 100644
--- a/settings.py
+++ b/settings.py
@@ -59,12 +59,8 @@ WSGI_APPLICATION = 'wsgi.application'

 DATABASES = {
     'default': {
-        'ENGINE': 'django.db.backends.postgresql_psycopg2',
-        'NAME': 'digitalresolution',
-        'USER': 'digitalresolution',
-        'PASSWORD': 'digitalresolution',
-        'HOST': 'localhost',
-        'PORT': '5432',
+        'ENGINE': 'django.db.backends.sqlite3',
+        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
     }
 }
```

DJANGO SETUP

- Make sure your virtualenv is active and you're in the **digital-resolutions** folder.
- In the terminal, run **./manage.py syncdb**
- This will create the database tables Django needs to run.
- This will also prompt you to create a superuser account for your Django admin. This is important, so write down your username and password!

DJANGO SETUP

- Make sure your virtualenv is active and you're in the **digital-resolutions** folder.
- In the terminal, run **./manage.py runserver**
- This will tell Django to run a web server locally on your computer. Only you can access it.

IS DJANGO RUNNING?

- If Django failed, it'll give us an error. If it succeeded, you should see a message something like:

```
(digital-resolution)Shannons-MacBook-Air:digital-resolution shannon$ ./manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
February 28, 2015 - 01:56:11
Django version 1.7.2, using settings 'settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

- Open a web browser and go to **http://127.0.0.1:8000/admin** to see if it worked!

IS DJANGO RUNNING?

- If it looks like this, we're in good shape!

- Log in with your Django admin superuser credentials you just created.

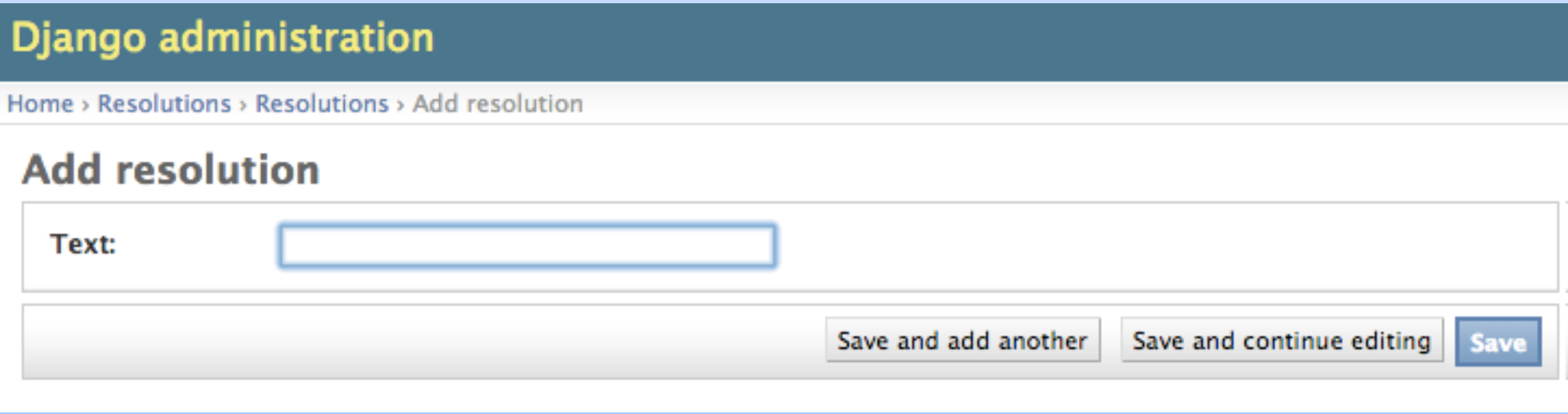
IN THE DJANGO ADMIN INTERFACE

- In the admin panel, you should see **Resolutions** and an **add** button.

IN THE DJANGO ADMIN INTERFACE

- Enter some text for your resolution and click **Save**.

- Example: I want to learn web development!



MOMENT OF TRUTH

- Did it work? Go to **http://127.0.0.1:8000**

NOW, LET'S MAKE SOME REAL CHANGES

- Open up **digital-resolution/templates/resolutions.html** in Sublime Text.

```
1  <html>
2  <head>
3      <link href='http://fonts.googleapis.com/cs
4      <style>
5
6          .resolutions
7          {
8              width: 80%;
9              color: #fff;
10             padding: 10%;
11             font-family: 'Oswald', sans-serif;
12         }
13
14         .pink {
15             background-color: #e54164;
16         }
```

NOW, LET'S MAKE SOME REAL CHANGES

- **resolutions.html**: Django uses this to display our resolutions on a webpage.

```
1  <html>
2  <head>
3      <link href='http://fonts.googleapis.com/cs
4      <style>
5
6          .resolutions
7          {
8              width: 80%;
9              color: #fff;
10             padding: 10%;
11             font-family: 'Oswald', sans-serif;
12         }
13
14         .pink {
15             background-color: #e54164;
16         }
```

HTML, CSS, PYTHON, ETC.

- Django's templates use a mix of HTML, CSS, logic, and variables all wrapped up together.
- See line 62: **{{ resolution.text }}**

```
57          <div id="count">
58          # {{ resolution_id }} out of {{ resolutions_count }}
59          </div>
60
61          <div id="resolution">
62              {{ resolution.text }}
63          </div>
64
65          <div id="footer">
66              What's your #digitalresolution? Text yours to ###-###-####
67          </div>
```

MAKE SOME CHANGES!

- Ideas of what you could change:
  - The "Whats your #digitalresolution" text
  - The CSS colors available
  - The page refresh time
  - Whether or not/how often to show logos
  - Show different logos
  - The layout of the page
  - Something else, get creative!

ONCE YOU'VE MADE CHANGES

- You've each made different changes to your own local copies.
- Now we're going to save those changes to our individual forks of the original project.
- Remember, changes you make to your fork (copy) don't affect anyone else's!

SAVING CHANGES TO YOUR REPO

- Step 1: **git status** to see all of the files you've edited. Should be **settings.py** and **templates/resolutions.html**

- Step 2: **git diff** to see line by line all of your deletions and insertions (changes) for each file. Make sure those look good! If not, edit the files, save, and **git diff** again.

SAVING CHANGES TO YOUR REPO

- Step 3: Use **git add** to add the files you changed.
  - **git add settings.py**
  - **git add templates/resolutions.html**

- There are other ways to add all changed files, but I personally like this method because I can choose which files I add.

SAVING CHANGES TO YOUR REPO

- Step 4: Commit your changes using
  git commit -m "Lowered page refresh time to 15 seconds"

- Substitute your own commit message inside the quotes! (Maybe you didn't adjust the page refresh time)

- Be descriptive! What changes to your files did you make? Say it here!

- In your commit message, you can't end with an exclamation point**!** If you do, you'll get an error.

SAVING CHANGES TO YOUR REPO

- Step 5: Push your changes to Github using **git push**
- Most likely, you'll be prompted for your Github username and password.
- Visit your fork on Github. Are your changes there? If so, celebrate!