CSCI E-33a

CS50’s Web Programming
with Python and JavaScript

Spring 2020

Barbara Karakyriakou · Teaching Fellow

- Email: karakyriakou@fas.harvard.edu
- Phone: 617 820 6930
- Section Meetings: Wednesdays 8:30pm-10:00pm EST
- Office Hours: Saturdays 1:00pm - 2:30pm EST

Section 2: Django

Agenda:

- Django
- Django for Project 1
- Python
- Python for Project1
- HTML for Project1
- Q&A

Django

Django is a Python web framework that allows users to launch their web applications. It’s free and open source, but it requires installation.

https://www.djangoproject.com/

Install Django

To install:

```
pip3 install django
```

To check the django version installed:

```
python3 –m django --version
```

Django Entries – create a project

- Create a project: `$ django-admin startproject myproject`

```
myproject/
    manage.py
    myproject/
        __init__.py
        settings.py
        urls.py
        asgi.py
        wsgi.py
```

Explanation of *startproject* created files:

- Outer myproject/ root directory: project container; can be renamed (it will not affect Django)
- manage.py: A command-line utility; required for interacting with the Django project
- Inner myproject/ directory: actual Python package you'll need to use to import anything inside it
- myproject /__init__.py: An empty file that tells Python that this directory is considered a Python package
- myproject /settings.py: Django configuration settings
- myproject /urls.py: The URL declarations for your project
- myproject /asgi.py: An entry-point for ASGI-compatible web servers to serve your project. ASGI stands for asynchronous server gateway interface
- myproject /wsgi.py: An entry-point for WSGI-compatible web servers to serve your project. WSGI stands for web server gateway interface

Django Entries – development server

- Start the development server: $ `python3 manage.py runserver`

  By default, the development server starts at http://127.0.0.1:8000
- You can select a different port by declaring it at your command:

  ```
  $ python manage.py runserver 8080
  ```

  This will start the development server at http://127.0.0.1:8080

Django Entries – create an app

- While in *myproject* directory (where the manage.py file is located) create an app: $ `python manage.py startapp myapp`

```
myapp/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    views.py
```

Django Entries – create an app cont'd

- Open views.py and write python function to create views. Example:

```
def index(request):
    return HttpResponse("Hello, world!")
```

- Next you need to map views to a URL. To do this you need to create urls.py file within your app folder, which will have the following code:

```
from django.urls import path

from . import views

urlpatterns = [
    path("", views.index, name="index")
]
```

Django for Project 1

As per the instructions download the distribution code

from https://cdn.cs50.net/web/2020/spring/projects/1/wiki.zip and unzip it

Open the wiki folder (outer wiki directory) and you will find the following folders and files:

- encyclopedia : this is your app folder
- entries : this is a folder that contains information about the project
- manage.py : this is your manage.py file of your project
- wiki : this is your inner project directory

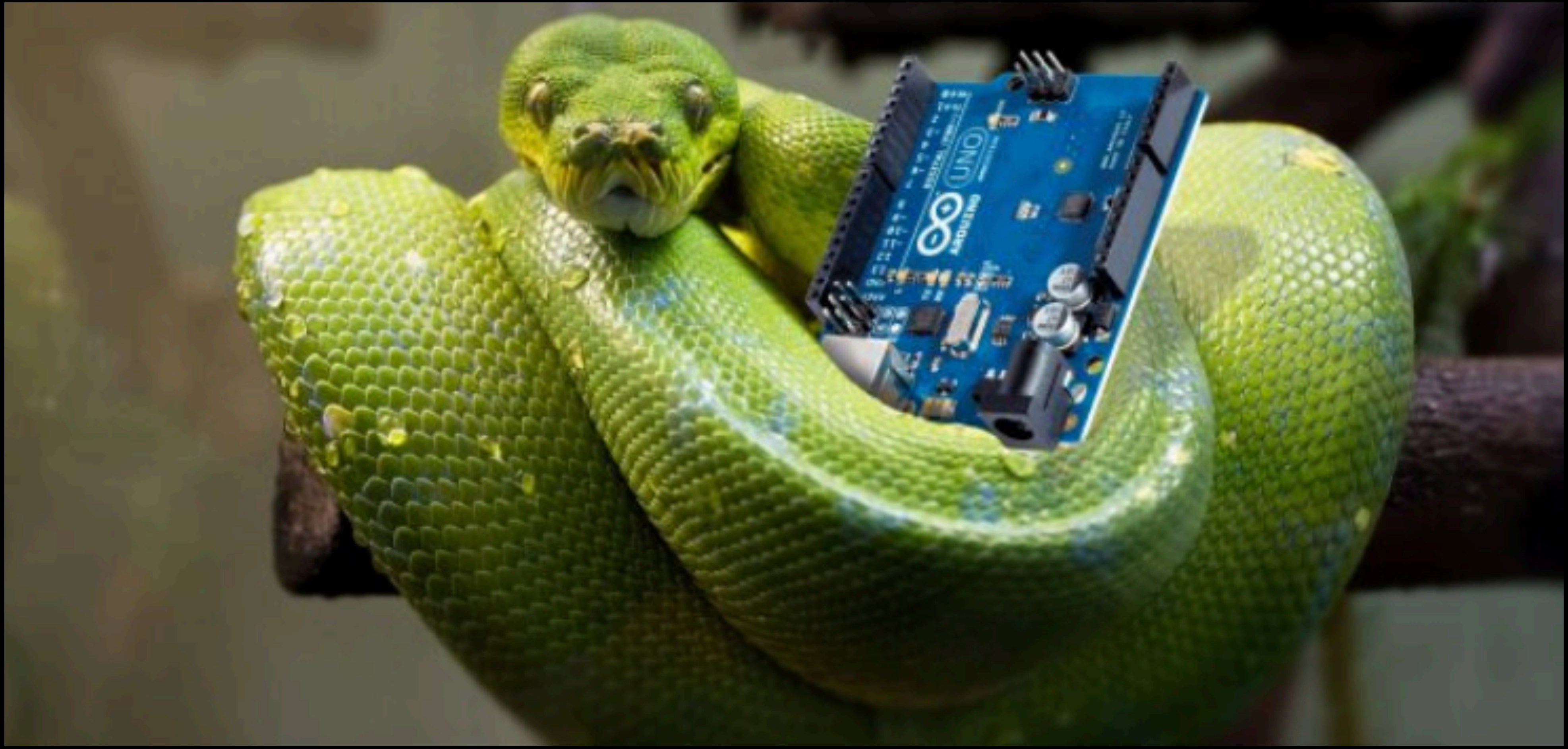
As you will realize, the project directories, the app, and the urls.py have been already created for you.

Django for Project 1

TO DO:

- Edit `settings.py` to add your app
- Edit `views.py` within your app folder and create your views
  - ➢ Hint: You need to import few packages in `views.py`
- Edit `urls.py` within your app folder and add paths to your created views
  - ➢ Hint: `from . import views`, where `.` stands for current directory, connects `views.py` to `urls.py`
- Edit `urls.py` within your project (outside of your app folder) and check if a path to your app exists
  - ➢ Hint: `path('', include(”app_name.urls"))`

Python



Python Function

- A function is a block of code that runs when the function is called.
- A function has parameters where you can pass data (arguments) into it.
- A function will process your data and return a result

Python Function

```
def myFunction(parameter(s)):

    return result

myFunction(argument(s))
```

Function Examples (Project 1)

- Entry page:

```
def entry(request, title):
    content = util.get_entry(title)
    # condition: if there is no content, return something
    # condition: if there is content, return something else
    # needs to handle Markdown (convert Markdown to HTML)
```

- Index page
- Search

```
def search(request):
    # allows user to type a query
    # conditions as per instructions
```

Function Examples (Project 1) cont’d

- New Page

```
def newpage(request):
    # title and content text areas, and a save button
    # condition: if title exists, render an error
```

- Edit page:

```
def edit(request, title):
    # user should be able to edit an existing page (entry)
    # save button
```

- Random Page

```
def random(request):
    # takes user on a random encyclopedia page
```

HTML for Project 1

HTML will be created by extending the existing `layout.html`

Example:

```
{% extends "encyclopedia/layout.html" %}

{% block title %}

    New Page – Encyclopedia

{% endblock %}

{% block body %}

    <form>
        {% csrf_token %}     <!-- Cross-site request forgery token -->
        <input></input>
        <textarea></textarea>
        <input></input>
    </form>

{% endblock %}
```

Project 1 is due by 11:59pm EST on Sun 2/23.

Q&A