CSCI E-33a (Web50) Section 1+2

Ref: Lectures 1+2 (Git + Python)

Vlad Popil

Feb 3, 2021

Welcome!

About me:

Vlad Popil
Master’s **(**ALM) in Software Engineering
Principal DA, Enterprise Data Management, Capital One

Email: vlad@cs50.harvard.edu

Sections: Wed 6:30-8:00pm EST
Office Hours: Thu 8:30-10:30 pm EST

Agenda

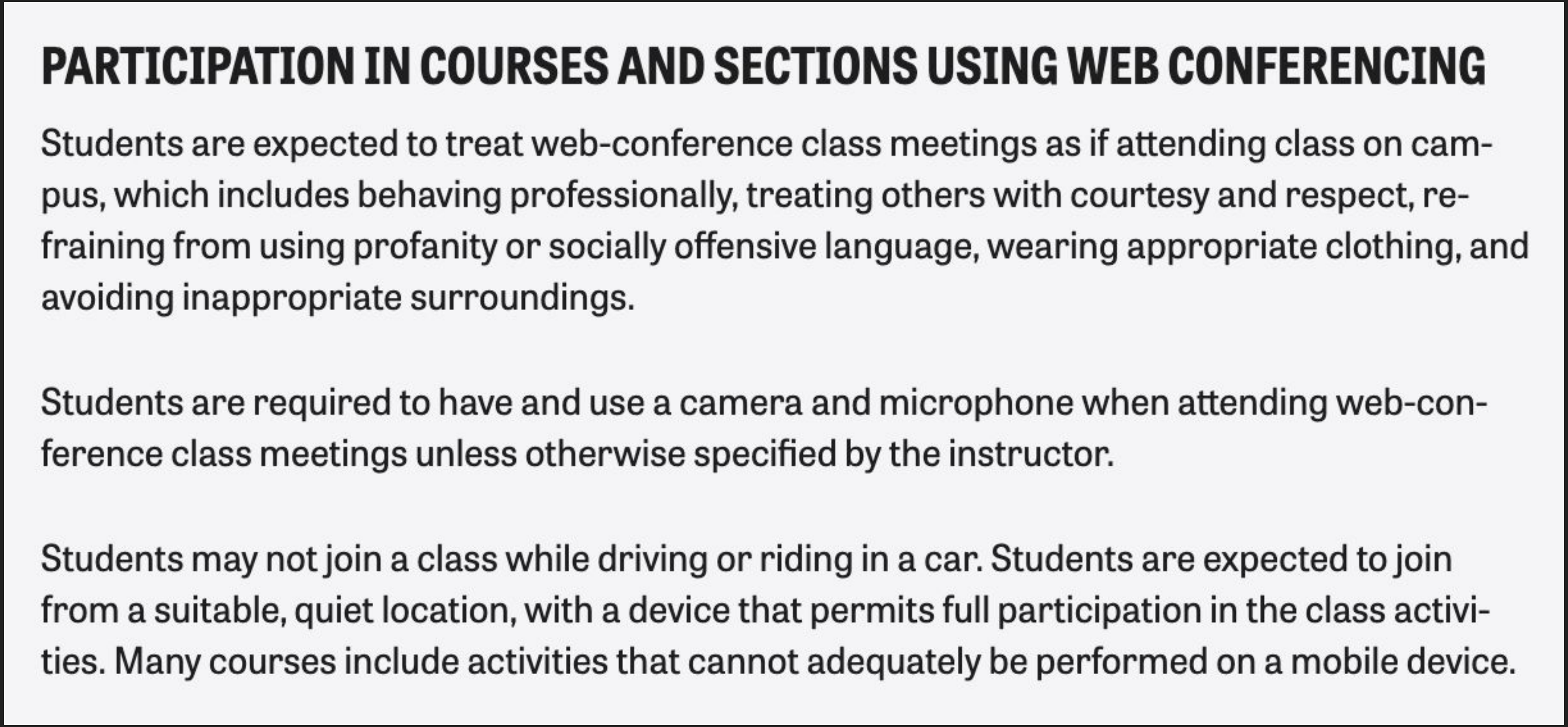
- Zoom
- Intro/Suggestions
- Sections 10,000 foot overview
- Git / GitHub
- Python
- Chrome Developer Tools
- Project 0 + submit50
- Grading criteria (not exhaustive)
- IDEs (very quickly)
- Tips
- Q&A

Logistics

Zoom

- Use zoom features like raise hand, chat and other
- Video presence is **strongly** encouraged
- Mute your line when not speaking (enable temporary unmute)
- Let’s review other features...

Zoom Policy



PARTICIPATION IN COURSES AND SECTIONS USING WEB CONFERENCING

Students are expected to treat web-conference class meetings as if attending class on campus, which includes behaving professionally, treating others with courtesy and respect, refraining from using profanity or socially offensive language, wearing appropriate clothing, and avoiding inappropriate surroundings.

Students are required to have and use a camera and microphone when attending web-conference class meetings unless otherwise specified by the instructor.

Students may not join a class while driving or riding in a car. Students are expected to join from a suitable, quiet location, with a device that permits full participation in the class activities. Many courses include activities that cannot adequately be performed on a mobile device.

https://www.extension.harvard.edu/resources-policies/student-conduct/expectations?_ga=2.263232908.1411291337.1537393563-2093574759.1491937594#attendance

Intro

- Refer to website: https://cs50.harvard.edu/extension/web/2021/spring
- Sections and office hours schedule on website, sections posted within 48hrs
- Get comfortable with command line
- Text editor is usually sufficient to write code, BUT!

- Six projects:
  - Start early (or even better RIGHT AWAY!!!)
  - Post questions on Ed platform
  - Remember: bugs can take time to fix
  - Grade -> 3 × Correctness + 2 × Design + 1 × Style (Project 0 may be an exception)
  - Lateness policy - 0.01 per minute x 60 x 24 x 7 days ~ 100
  - Set a reminder to submit the Google Form for each project
  - Project 0 - Due Sunday, Feb 7 at 11:59pm EST

Troubleshooting Tools

- Chrome Developer Tools
- Online Q&A Forums (Stack Overflow, GitHub, etc)
- Ed (ask, explore, contribute)
- Office Hours!
- Peers (adhere to Academic Policies please)

Are sections a good use of my time?

- First section and Project 0 may seem a bit introductory, but beware!!!...
- Tons of tips and hints that ~~can~~ **will** save hours or even days
- Debugging approaches not covered in lectures
- Supplemental material which complement lectures well
- And more...

YES!!!

10,000 foot overview

- *Section 0 - SKIPPED*
- *Section 1+2 (Git + Python)* - Chrome Dev Tools (Inspector), CDT (Network), Project 0, Grading aspects
- *Section 3 (Django)* - Env Config, Markdown, RegEx, IDEs, pycodestyle, Debugging, Project 1
- *Section 4 (SQL, Models, Migrations)* - IDE's, linting, DB modeling, Project 2
- *Section 5 (JavaScript)* - cURL/Postman, jshint, CDT + IDE's Debugging, Project 3
- *Section 6 (User Interfaces)* - Animations, DB modeling, Pagination, Project 4
- *Section 7 (Testing, CI/CD)* - Test Driven Development, DevOps, Final Project
- *Section 8 (Scalability and Security)* - Cryptography, CAs, Attacks, App Deployment (Heroku)

Most sections: material review, logistics, project criteria review, reminders, hints, etc.

Burning Questions?

Please ask questions, or topics to cover today!

Topics:

- *OH?*
- *Are projects built on top of each other?*
- *Smaller projects examples?*

Git/GitHub

Git/GitHub

- **Git:** Version control software for tracking changes locally
- **GitHub:** Place to store Git repositories remotely and share them with others
- **Repository:** A directory containing all files/directories associated with a project
- **Branch:** A version on your project where you can work on new features
- **Commit:** Save a snapshot of your repository
- **Push:** Upload your local repository to a remote website
- **Fork:** Create a copy of your another person’s repository that you can edit
- **Merge:** Combine two branches
- **Merge Conflict:** When manual intervention is required to merge two branches
- **Pull Request:** An attempt to merge two branches that must be approved

Git

- Track changes of code
- Sync the code between different people or projects
- Revert to old versions of code

Most popular commands:

- clone
- add
- status
- commit
- push
- pull

Git

Let’s create new repo and try the basics!

Git

- .gitignore / .gitkeep
- Branching:
  - branch
  - checkout
  - merge
- Pull Requests
- Demo Forking: https://github.com/octocat/Spoon-Knife

Python

Interesting Concepts

Running Python Programs

- Files should be in the form `file_name.py`
- Run a file by running `python file_name.py` in the terminal.

Sequences

Sequence Type	**Ordered**	**Mutable**	**Example**
String	Yes	No	name = “Bob”
List	Yes	Yes	ls = [1, 3, “hi”, True, [1, 2, 3], None]
Tuple	Yes	No	coordinates = (4, 5)
Set	No	N/A	odds = {1, 3, 5, 7, 9}
Dictionary	No	Yes	person = {‘name’: ‘Bob’ s’, ‘age’: 20}

Modules

- Allow us to import code from other files
- Allow us to import code written by others
- For a file named **name.py**, we can write **import name**

Formatting Strings

- Used to more easily incorporate variables in strings
- Can include function calls and expressions in curly braces
- More formatting options here

```
f“Hello, {name}”
```

```
f“{x} times {y} is {x * y}”
```

Exceptions

- If we expect an exception to be thrown in a certain place, we can handle it without any crashes.

```
try:
    x = int(input(“Type a number to find it’s square root:
“))
    print(math.sqrt(x))
except ValueError:
    print(f“Could not take a sqrt of {x}”)
```

Optional Arguments

- Arguments can have default values if none are provided
- We can call a function using either the order of arguments or their names.

```
def square_root(x, truncate=False)
    if truncate:
        y = int(math.sqrt(x))
    return math.sqrt(x)


square_root(4)
square_root(4, True)
square_root(truncate=True, x=4)
~~square_root(True, 4)~~
```

Python

Interactive examples:

- Jupyter Notebook
- Demo...

Chrome Developer Tools (Inspect / Network)

In Chrome:

1. Right click
2. Inspect
3. → Demo

Extremely powerful! Let’s try...

Project 0

- Start early!!!
- Make a checklist of requirements and check off all before submission
- Don’t forget to include the .css / .scss file(s)
- Make sure there’s no bugs
- Google Form
- *Next page...*

Project 0

1. Requirement review
2. Chrome developer tools
3. Dual button review
4. Double key:value
5. `submit 50`

Mac or Windows or Linux?

1. Mac
2. Linux
3. Windows with WSL
4. ~~Windows~~
5. Chromebook? ¯_(ツ)_/¯

Integrated Development Environments (Intro)

- Text Editor or Heavy IDE?
- Options:
  - VS Code
  - PyCharm (Pro)
  - Atom
  - Sublime
  - vim/Emacs
  - And dozens more, including Notepad :)
- My suggestion: VS Code or PyCharm
- Benefits: Debugging, Autocomplete, Navigation, Find Usages, Refactoring, and much more.

HTML beautifiers/prettify

- Automatically formats your HTML (except line breaks)
- Most IDEs supports integration of marketplace beautifiers
- Demo...

Grading criteria generic suggestions (not limited to)

- Correctness:
  - All requirements + bugs
- Design (not limited to):
  - Simplest solution
  - Avoiding repetition (refactoring)
  - Structure (e.g separate files vs inline styling)
- Style (not limited to):
  - File naming/structure
  - Line breaks
  - Spacing / Indentation
  - Naming
  - Comments

Both Design and Style consider readability but from different perspective.

Random Tips

- Windows licence (https://harvard.onthehub.com/)
- GitHub Education Pack
- Spotify + Hulu + Showtime
- The Great Suspender + Chrome Tabs
- Video Speed Controller
- Leetcode / AlgoExpert

WATCHING THIS RECORDED? (**aka “Fruit of the Week”**)

<<< If you are watching this recorded >>

Please email the this name of the fruit in subject (no msg necessary): **APPLE**

To: **vlad@cs50.harvard.edu**

Thank you.

Q&A

Please ask any questions. Ideas:

- Anything discussed today
- Anything from lecture material
- About the project
- Logistics
- *Random*

Resources

- https://github.com/vpopil/e33a-sections-spring-2021

Anaconda Distribution

- Anaconda - World's Most Popular Python/R Data Science Platform
- Miniconda (lighter version):
  a. Download https://docs.conda.io/en/latest/miniconda.html
  b. Run in terminal in Downloads: `` `zsh Miniconda3-latest-MacOSX-x86_64.sh` ``
  c. Run `` `conda init` `` ONLY if not prompted during installation
  d. Create new environment: `
     - `` `conda create -n s33a python=3.7` ``
  e. See environments:
     - `` `conda env list` ``
  f. Deactivate/Activate environment:
     - `` `conda deactivate` ``
     - `` `conda activate s33a` ``
  g. Install more packages:
     - `` `conda install django` `` (preferred)
     - `` `pip install django` `` (if conda doesn't find), although
       It is better to `` `python -m pip install django` `` (to assure proper pip)

CSCI E-33a (Web50) Section 1+2

Ref: Lectures 1+2 (Git + Python)

Vlad Popil

Feb 3, 2021