Modules: An Overview

Kyra Balenzano, Lisanne de Groot

Basics

What's a module?

An external module is a **library of code** that provides extra capabilities and needs to be installed

What are some examples of Python modules?

- Pygame
  - Allows for smoother graphics compared to Tkinter, often used for coding games (shocker)
- OpenCV
  - Enables ability to use information gained from the computer's camera to dictate code behavior

- Others: Leap Motion, Beautiful Soup, Scikit-learn, Selenium, Matplotlib, TensorFlow, PyTorch

Pros and Cons

What are the benefits to using modules in my project?

- Certain projects may be easier
  - Pygame can make game collisions easier

- Makes more projects possible
  - OpenCV, Leapmotion get physical data and make it available to you

- You learn a new skill that you can apply in future projects

What are the risks to using modules in my project?

- The learning curve may be steep—and you have to show proficiency **by TPO**
  - It often takes more than a few days to get used to the syntax
  - You're expected to teach yourself the module
  - Sometimes the work to learn it isn't worth what the module will give you (see below)

- It could do the work for you
  - If one module call completes the most complex part of your project, you **do not** get credit for it!

Additional Information

Example Project → Module Pairings

- Game where you use your hand or object to move the player
  - OpenCV, Leap Motion for hand tracking

- Movie database and recommender
  - Beautiful Soup for web page scraping

- Rubik's Cube solver
  - None required!

How can I learn more about [insert module here]?

- Attend a TP mini-lecture!
- Surf the ‘net
- Watch YouTube videos

Make sure you cite any code or scaffolds that you take from other sources!

The End!

Still have questions?

Do not hesitate to contact your mentor; they are here to help! :)