Python development environments

Help Center

For this class, we assume that the students are experienced Python programmers. As a result, you are welcome to use whichever Python development environment suits your needs. However, OwlTest (the machine grader for this class) runs Python 2. Therefore, we require the use of Python 2 in this class. So, whatever environment you choose must support Python 2. You will also be asked to generate plots of various data sets as part of the Application component of each Module. Therefore, your development environment should support the creation of plots.

Suggested development environments

The download page at python.org lists many options for possible development environments. Here are few that we suggest you consider if your current environment doesn't suit the needs of this class.

- **IDLE**
  IDLE is an integrated development environment for Python, which has been bundled with the default implementation of Python. Note that IDLE does not include a default plotting package. If you decide to use IDLE, we recommend that you use the package matplotlib as your plotting package.
  To install `matplotlib`, you have several options. You may install the package using the page above. However, be warned that `matplotlib` depends on the following packages being installed: `numpy`, `dateutil`, `pytz`, `pyparsing`, and `six`. For Windows users, we have had success manually downloading and installing these packages from this site.

  For Mac users (who already have Python installed) and those who want to avoid lots of manual downloads and installs, another option is to install pip, which is a tool for installing and managing Python packages. pip supports quick installation of a wide range of Python packages. However, it requires the user to be comfortable with the command line interface.

- **Anaconda**
  Anaconda is a Python distribution for large-scale data processing, predictive analytics, and scientific computing. Anaconda is free and the default installation includes all of the popular Python packages including `matplotlib`. Several of the CTAs in this class use it and recommend it.
- **Canopy**
  Canopy is a Python distribution that supports scientific and analytic computing. The free version of Canopy, Canopy Express, has a default installation that includes standard Python plus 50 popular packages including `matplotlib`. (As a caveat, we have had some issues with installing Canopy on systems that use unicode in directory names.)
- **CodeSkulptor**
  CodeSkulptor is a browser-based environment (developed by one of the instructors) for Python. CodeSkulptor includes two custom (specific to CodeSkulptor) Python modules, `simpleplot` and `simplegui`, that support easy creation of plots and interactive programs. For this class,

CodeSkulptor has two advantages: there is no installation required and creating the required plots with `simpleplot` is easy.

However, due to the fact that all of your code is running in your browser, you will need to be careful as you implement the more computationally intensive programs in this class. Use of CodeSkulptor in this class is welcomed, but not particularly encouraged. If you've only used CodeSkulptor up to now, this class is the right time to move to running Python on the desktop.

As a final note, **remember to install the version of the software that matches the "bitness" of your computer**. If your computer is a 32-bit machine, install the 32-bit version. If your machine is a 64-bit machine, install the 64-bit version.

Other notes

For some of the Projects, we will import some provided code, *i.e.*, `import alg_provided`. To access this code, you simply need to go to the corresponding CodeSkulptor URL (by adding a hash tag, the imported file name, and the ".py" extension): http://www.codeskulptor.org/#alg_provided.py (note that this is just an example of how to construct the URL and that file does not exist). Once you have copied all of the provided files onto your machine (with the same names that we gave them), then you should be able to develop your code on your local machine. However, you will need to make sure you locate these files where your environment can find them.

OwlTest currently has two restrictions. First, the machine grader only supports the upload of a single file. So, even though most development environments make it easy to break your program up into multiple files, you must keep all of the code you write for an assignment in one file for this class. While this is not necessarily the best programming practice, it is necessary for us to deal with the complexities of a large online class like this. Second, the machine grader only supports the following Python libraries: `random`, `time`, `math`, `re`, `collections.defaultdict`, and `collections.Counter`. Again, this has to do with the complexities of providing a machine grader for a large online class like this. These modules are sufficient to complete all of the assignments in this class and CodeSkulptor also supports all of these modules.

Created Sun 17 Aug 2014 12:48 PM PDT

Last Modified Mon 27 Apr 2015 8:18 PM PDT