Class 02: An Introduction to Python

Held: Thursday, January 28, 2010

Summary: We begin to explore the Python programming language.

Related Pages:

- EBoard.

Notes:

- It appears that few of you noticed the “instructions for readings”. Make sure to send me a short question on each of the two readings for Tuesday.
- As you’d expect, I will not respond to your introductory surveys until (at least) Tuesday.
- Topical reading for Tuesday: Learn a little bit about the UML from the Web.
- Don’t forget tomorrow’s CS table. Noon. JRC Day PDR. If on meal plan, please use your plan. If not, charge the department.

Overview:

- Beautiful Code: Python Dictionaries.
- Python: Important Characteristics.
- Python: The Basics.

Beautiful Code: Python Dictionaries

- What was the purpose of the code described in this chapter?
- What puzzled you about the chapter?
- What parts of the code are beautiful?
- What parts of the code are less than beautiful?

Python: Important Characteristics

- Python is a “scripting language”
  - What does that mean to you?
- Python has made an interesting syntactic decision: The structure of a program is given by the indentation of code, rather than by begin/end (open-brace/close-brace) pairs.
  - We indent for clarity anyway
  - Helps ensure that the "clear" indentation meets the actual structure
- Python supports multiple paradigms:
  - It began as an imperative language.
○ It has a relatively rich object model, including multiple inheritance.
○ It includes some important functional capabilities, such as anonymous and higher-order procedures.
● Like many scripting languages, Python provides some useful built-in types.
  ○ Strings (obviously)
  ○ A list type, which combines the best aspects of arrays (quick access to individual elements) and traditional lists (easy to expand).
  ○ A dictionary type, which maps values to values.
● Like many scripting languages, Python is often run in an interactive interpretive mode.
● Python makes some clever decisions about parameters
  ○ As in many languages, some parameters are positional.
● You can also have a number of named parameters.

Disclaimer: I am not an expert Python programmer. There will be some things I do differently than more experienced programmers.

Python: The Basics

● We’ll be using Python 3, which should be available at /glimmer/bin/python3. (If it’s not, let me know which workstation it’s missing on.)
  ○ Some of the examples you see assume Python 2. There are a few differences in the print statement and the use of objects.
● Often, we type simple expressions directly in the interpreter.
  ○ Examples folows
  ○ Interactive Python is cool b/c it gives you easy access to help.
● When defining classes and functions, it’s useful to put them into a file and load it.
  ○ execfile(‘filename.py’)
  ○ import filename (components referred to as filename.component)
  ○ from filename import thing (import just part of a package/group)
● SamR’s preferred formatting guidelines (also standard Python guidelines):
  ○ Use four spaces as your indentation.
  ○ Make sure to include comment strings in anything that is commentable.
● We’ll use Python 101 as a partial guide to the rest of the basics. (I’m too lazy to cut-and-paste into this guide.)
● Some things not covered in Python 101:
  ○ Additional characteristics of classes
  ○ map
  ○ anonymous functions

Copyright © 2010 Samuel A. Rebelsky. This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc/2.5/ or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.