Programming in Python

Course Overview

(Elective course offered at PESIT Bangalore for 6th Semester BE Computer Science and Information Science students)

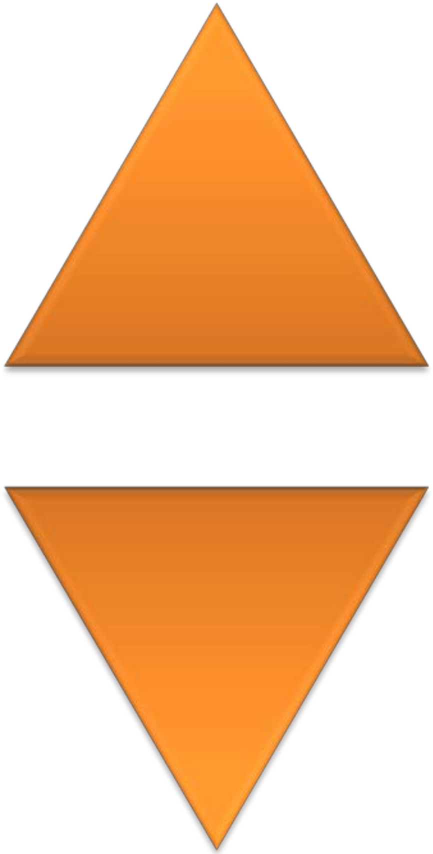
Anantharaman Narayana Iyer

19 Jan 2013

narayana.anantharaman@gmail.com

Learning a programming language: Two Approaches

- Different but complementary nature of two approaches:
  1. Start with the goal of learning a programming language and its associated tools, frameworks and solve appropriate problems
  2. Start with the class of problems you want to solve, choose the most appropriate language and platform, learn the platform.
- In our course we take the latter approach
  - Solve problems pertaining to web analytics and unstructured data processing and choose Python as an implementation vehicle



Sample Project Domains

- LinkedIn analysis
- Wiki People database
- Financial Information
- Tourism database
- Device database (look at WURFL)
- HTML Responsive design
- Articles database in ZDNet

Edit search terms: Franklin D. Roosevelt

uses, see FDR n).

'rouzavalt/ ROE-za-valt;[1] also known by his initials, FDR) was the 32nd President of the United States and a central figure in world events during the mid-20th century, leading the United States during a time of worldwide economic crisis and world war. The only American president elected to more than two terms, he forged a durable coalition that realigned American politics for decades. FDR defeated incumbent Republican Herbert Hoover in November 1932, at the depths of the Great Depression. FDR's combination of

Course Structure

- **Unit 1**: Python language features overview
- **Unit 2**: Programming with Python Standard Library
- **Unit 3**: Web Programming in Python
- **Unit 4**: Programming for the cloud using Python
- **Unit 5**: Web Analytics in Python

Instruction Methodology

- Classroom lectures for language overview, some standard library APIs and worked out examples for other topics

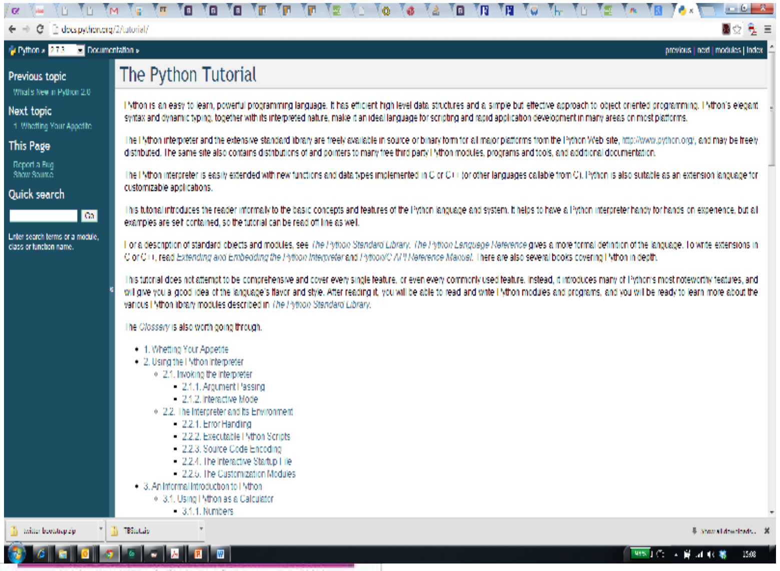
- Hands on project for each unit – each project is independent but all of them can be combined in to a wholesome project

Evaluation Model

Test	Approach	Marks	FINAL SEE
Test 1	Regular Written Test	50	10
Test 2	5 mini projects (each carrying 12 marks)	60	60
SEE	Regular Written Exam	100	25
	Attendance	5	5
	TOTAL		100

Reference books/URLs for the course

- T1: Core Python Applications Programming by Wesley J Chun, Third Edition, Pearson publications
- UNIT – I : Python Tutorial http://docs.python.org/2/tutorial/index.html
- UNIT – II : T1 Chapters 1-4
- UNIT – III : T1 Chapters 9-11 (Web frameworks like Django excluded)
- UNIT – IV : T1 Chapters 12-13
- UNIT – V : BeautifulSoup (http://www.crummy.com/software/BeautifulSoup/bs4/doc/), NLTK (http://nltk.org/), Orange (http://orange.biolab.si/doc/ofb/)


docs.python.org/2/tutorial/
Python » 2.7.3 Documentation »
previous | next | modules | index
Previous topic
What's New in Python 2.0
Next topic
1. Whetting Your Appetite
This Page
Report a Bug
Show Source
Quick search
Go
Enter search terms or a module, class or function name.

The Python Tutorial

Python is an easy to learn, powerful programming language. It has efficient high level data structures and a simple but effective approach to object oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, http://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

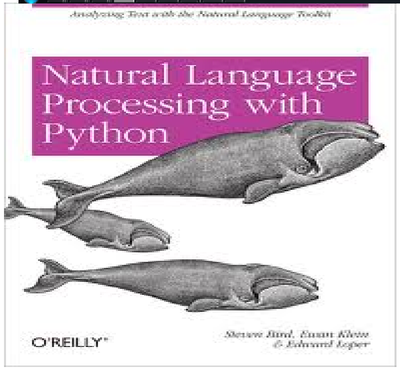
This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands on experience, but all examples are self contained, so the tutorial can be read off line as well.

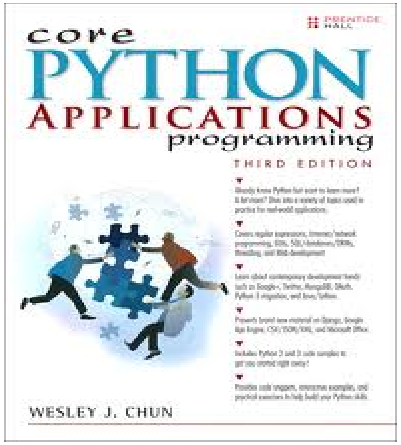
For a description of standard objects and modules, see *The Python Standard Library*. *The Python Language Reference* gives a more formal definition of the language. To write extensions in C or C++, read *Extending and Embedding the Python Interpreter* and *Python/C API Reference Manual*. There are also several books covering Python in depth.

This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in *The Python Standard Library*.

The *Glossary* is also worth going through.

- 1. Whetting Your Appetite
- 2. Using the Python Interpreter
  - 2.1. Invoking the Interpreter
    - 2.1.1. Argument Passing
    - 2.1.2. Interactive Mode
  - 2.2. The Interpreter and Its Environment
    - 2.2.1. Error Handling
    - 2.2.2. Executable Python Scripts
    - 2.2.3. Source Code Encoding
    - 2.2.4. The Interactive Startup File
    - 2.2.5. The Customization Modules
- 3. An Informal Introduction to Python
  - 3.1. Using Python as a Calculator
    - 3.1.1. Numbers


Analyzing Text with the Natural Language Toolkit
Natural Language Processing with Python
O'REILLY
Steven Bird, Ewan Klein & Edward Loper



```
random.html')
>>> soup=BeautifulSoup(f.
read())
>>> print soup.first('a')
<a href="http://posnteo.
com">link inside</a>
>>>
```

Example of Course Project

- Build an in-memory database using Python data structures (such as dictionary, lists etc)
- Expose it as a socket server that caters to multiple clients simultaneously
- Enhance the above in to a web server and service, expose a REST interface
- Process the unstructured/structured data from the web and covert that in to a database
- Port the above to Google App Engine

People Database

- Objective: Build a specialized database of famous people (example: Mahatma Gandhi, C.V. Raman, Thomas Alva Edison, Sachin Tendulkar) and information about them. We would like to answer queries such as:
  - Who are the Nobel laureates born in India?
  - What are the literary works of Kalidasa?
  - Who invented the light bulb?
  - Where did Sachin score his hundredth hundred?

Assignment #1

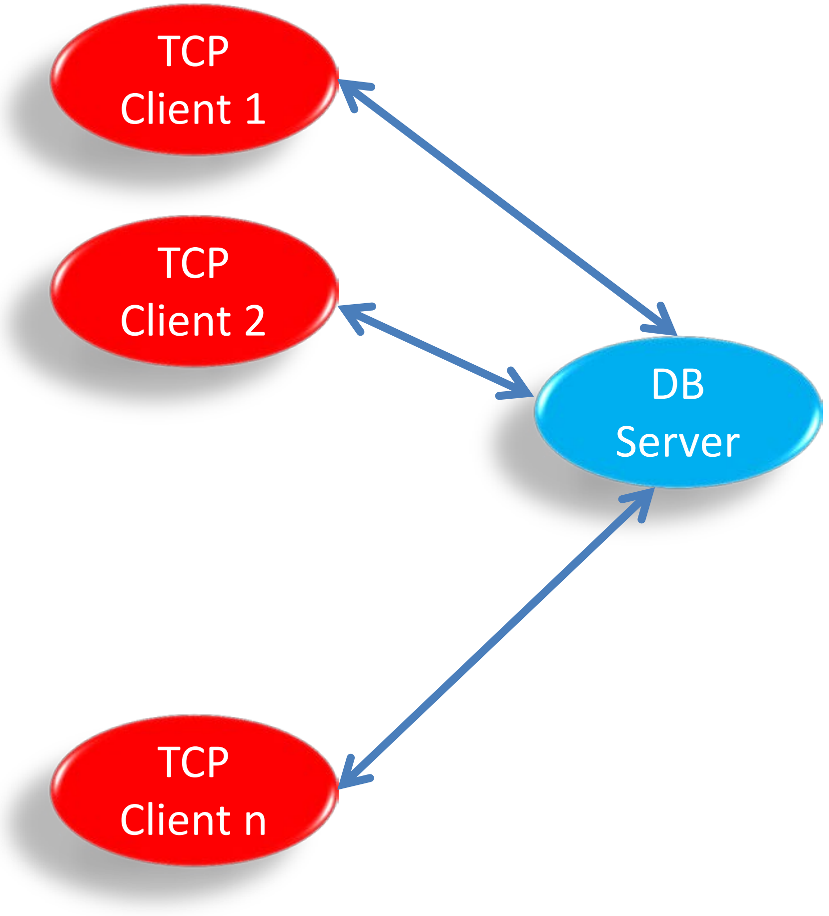
Build an in-memory database, support queries on the following:

- Name of the person
- Gender
- Country of birth
- Date of birth (Day, Month, Year)
- Date of Expiry (if no longer alive)
- Profession

Boolean combination of the above – eg: All people whose profession is scientist born in India after 1900

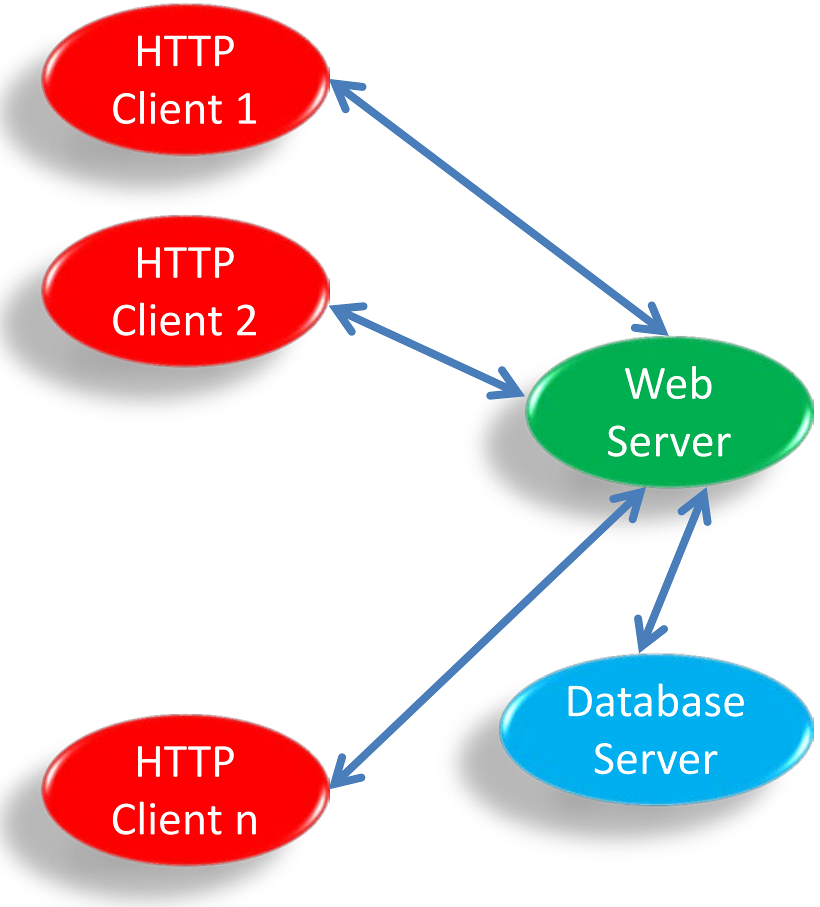
Assignment #2

- Expose the database program developed in assignment 1 in to a server application exposed through socket interface
- Write the client application that can take inputs from interactive console or file and can query the server through socket
- It should be possible for multiple clients to access the server simultaneously (Use Python multithreading)



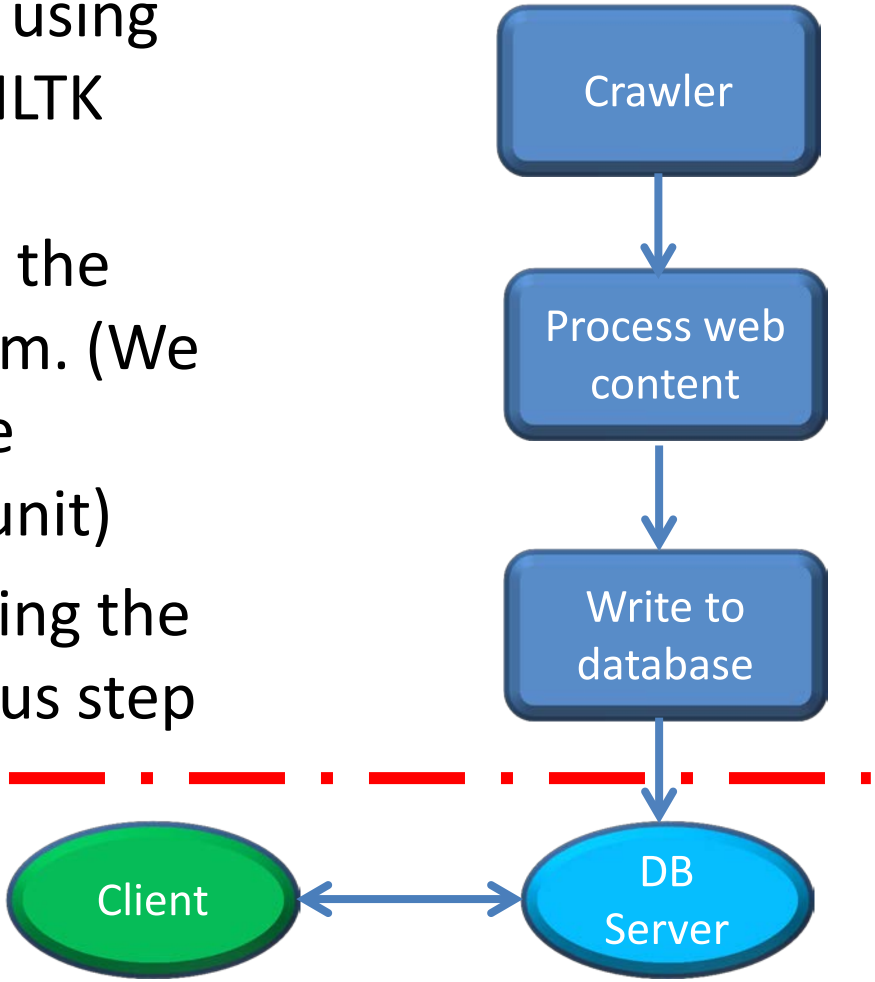
Assignment #3

- Expose the server developed in the Assignment#2 as a database server accessible through HTTP Server
- Build a typical HTML based web page serving as well as a REST interface, support one of JSON or XML
- Build a crawler to fetch the pages from the specified domain (you can repurpose freely available crawlers written in Python from open source)



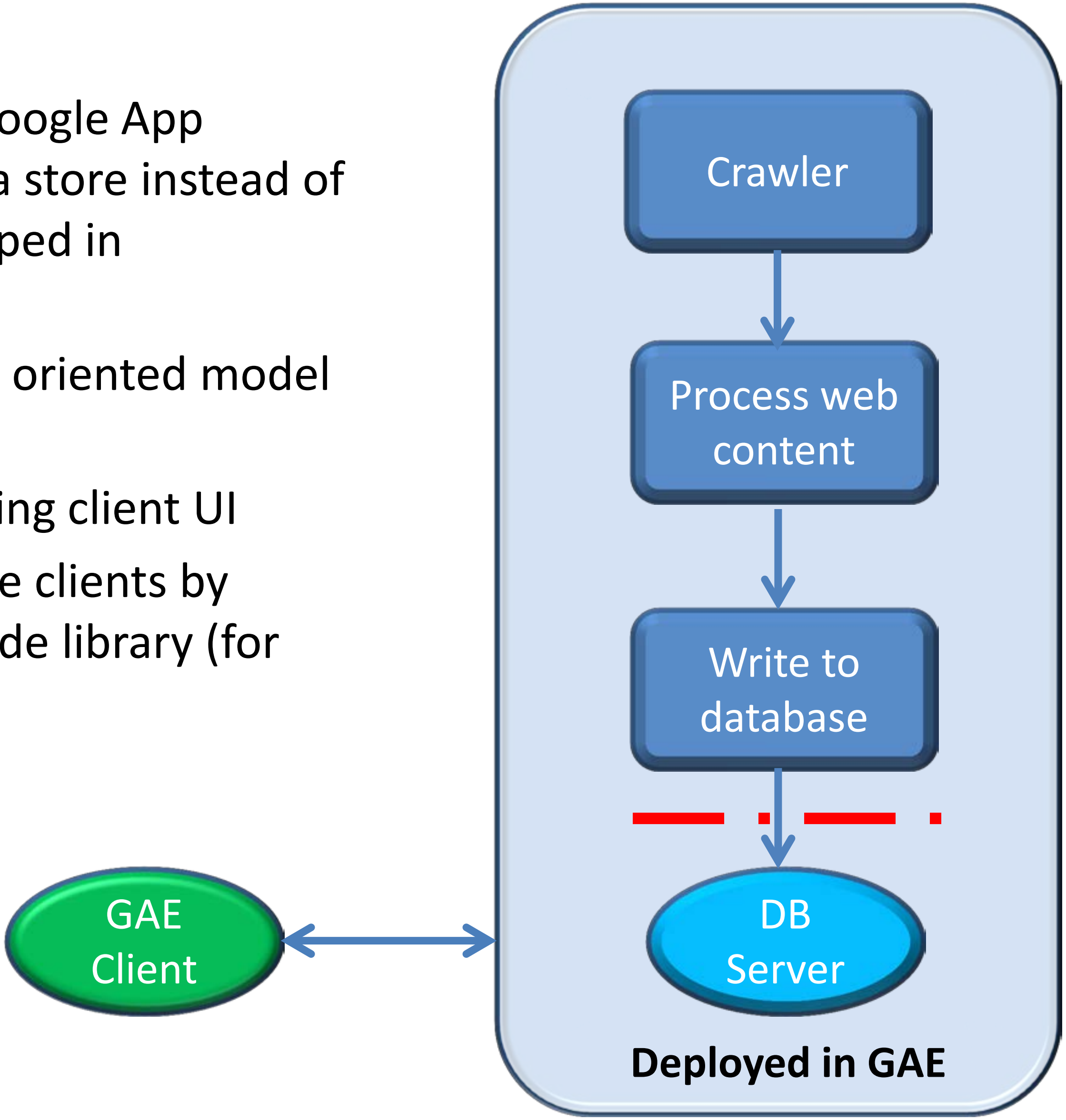
Assignment #4

- Perform HTML text processing using tools such as Beautiful Soup, NLTK and Orange to dig out the information needed to answer the queries specified in the problem. (We will specify these queries more precisely when we get to this unit)
- Build a structured database using the data retrieved from the previous step



Assignment #5

- Port the application to Google App Engine. Use the GAE data store instead of the database you developed in Assignment #1
- Support both the service oriented model as well as page serving
- (optional) Write compelling client UI
- (optional) Support mobile clients by building suitable client side library (for example Android SDK)


Crawler
Process web content
Write to database
DB Server
GAE Client
Deployed in GAE