WEB2PY

Python Web Framework

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

About the Tutorial

web2py is defined as a free, open-source web framework for agile development which involves database-driven web applications. It is written and programmable in Python. It is a full-stack framework and consists of all the necessary components a developer needs to build fully functional web applications.

Audience

This tutorial is primarily meant for software professionals who work on Python and are required to create scalable, secure and portable database-driven web-based applications. web2py provides all the functionalities to create, modify, deploy, and manage an application from anywhere using your browser.

Prerequisites

Before you start proceeding with this tutorial, we are assuming that you are already aware of the basics of Python programming. A basic understanding of Model-View-Controller is also equally important. If you are not well aware of these concepts, then we will suggest you to go through our short tutorial on Python.

Copyright & Disclaimer

Table of Contents

web2py – Introduction

web2py is defined as a free, open-source web framework for agile development which involves database-driven web applications; it is written in Python and programmable in Python. It is a full-stack framework; it consists of all the necessary components, a developer needs to build a fully functional web application.

web2py framework follows the **Model-View-Controller** pattern of running web applications unlike traditional patterns.

- **Model** is a part of the application that includes logic for the data. The objects in model are used for retrieving and storing the data from the database.

- **View** is a part of the application, which helps in rendering the display of data to end users. The display of data is fetched from Model.

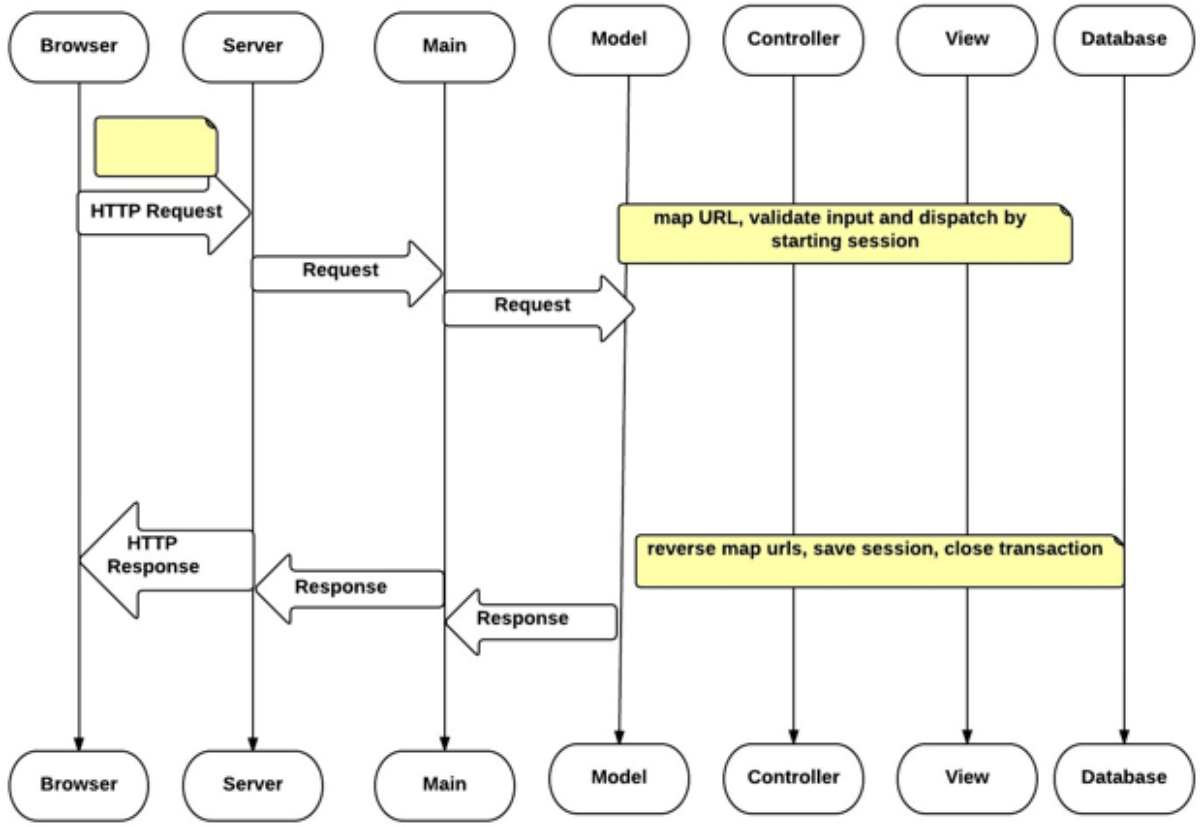
- **Controller** is a part of the application, which handles user interaction. Controllers can read data from a view, control user input, and send input data to the specific model.

web2py has an in-built feature to manage cookies and sessions. After committing a transaction (in terms of SQL), the session is also stored simultaneously.

web2py has the capacity of running the tasks in scheduled intervals after the completion of certain actions. This can be achieved with **CRON**.

web2py – Workflow

Take a look at the workflow diagram given below.



The workflow diagram is described below.

- The **Models**, **Views** and **Controller** components make up the user web2py application.
- Multiple applications can be hosted in the same instance of web2py.
- The browser sends the HTTP request to the server and the server interacts with **Model**, **Controller** and **View** to fetch the necessary output.
- The arrows represent communication with the database engine(s). The database queries can be written in raw SQL or by using the web2py Database Abstraction Layer (which will be discussed in further chapters), so that **web2py** application code is independent of any database engine.
- **Model** establishes the database connection with the database and interacts with the **Controller**. The **Controller** on the other hand interacts with the **View** to render the display of data.
- The **Dispatcher** maps the requested URL as given in HTTP response to a function call in the controller. The output of the function can be a string or a hash table.

- The data is rendered by the **View**. If the user requests an HTML page (the default), the data is rendered into an HTML page. If the user requests the same page in XML, web2py tries to find a view that can render the dictionary in XML.

- The supported protocols of web2py include HTML, XML, JSON, RSS, CSV, and RTF.

Model-View-Controller

The **model-view-controller** representation of web2py is as follows:

Model

```
"db.py" is the model:
db = DAL('sqlite://storage.sqlite')
db.define_table(employee,
   Field('name'),
   Field('phone'))
```

The **Model** includes the logic of application data. It connects to the database as mentioned in the figure above. Consider SQLite is being used and is stored in **storage.sqlite** file with a table defined as employee. If the table does not exist, web2py helps by creating the respective table.

Controller

The program **"default.py"** is the **Controller**.

```
def employees():
    grid=SQLFORM.grid(db.contact, user_signature=False)
    return locals()
```

In **web2py**, URL mapping helps in accessing the functions and modules. For the above example, the Controller contains a single function (or "action") called employees.

The action taken by the **Controller** returns a string or a Python dictionary, which is a combination of key and value including a local set of variables.

View

"default/contacts.html" is the **View**.

```
{{extend 'layout.html'}}
<h1>Manage My Employees</h1>
{{=grid}}
```

For the given example, **View** displays the output after the associated controller function is executed.

The purpose of this **View** is to render the variables in the dictionary, which is in the form of HTML. The **View** file is written in HTML, but it embeds Python code with the help of **{{ and }}** delimiters.

The code embedded into HTML consists of Python code in the dictionary.

Start with web2py

web2py comes in binary packages for all the major operating systems like Windows, UNIX and Mac OS X.

It is easy to install web2py because:

- It comprises of the Python interpreter, so you do not need to have it pre-installed. There is also a source code version that runs on all the operating systems.
- The following link comprises of the binary packages of **web2py** for download as per the user's need: http://www.web2py.com/init/default/download
- The **web2py** framework requires no pre-installation unlike other frameworks. The user needs to download the zip file and unzip as per the operating system requirement.
- The **web2py** framework is written in Python, which is a complete dynamic language that does not require any compilation or complicated installation to run.
- It uses a virtual machine like other programming languages such as Java or .net and it can transparently byte-compile the source code written by the developers.

Operating System	Command
Unix and Linux (source distribution)	python web2py.py
OS X (binary distribution)	open web2py.app
Windows (binary web2py distribution)	web2py.exe
Windows (source web2py distribution)	c:/Python27/python.exe web2py.py

Python can be defined as a combination of object-oriented and interactive language. It is an open source software. Guido van Rossum conceived python in the late 1980s.

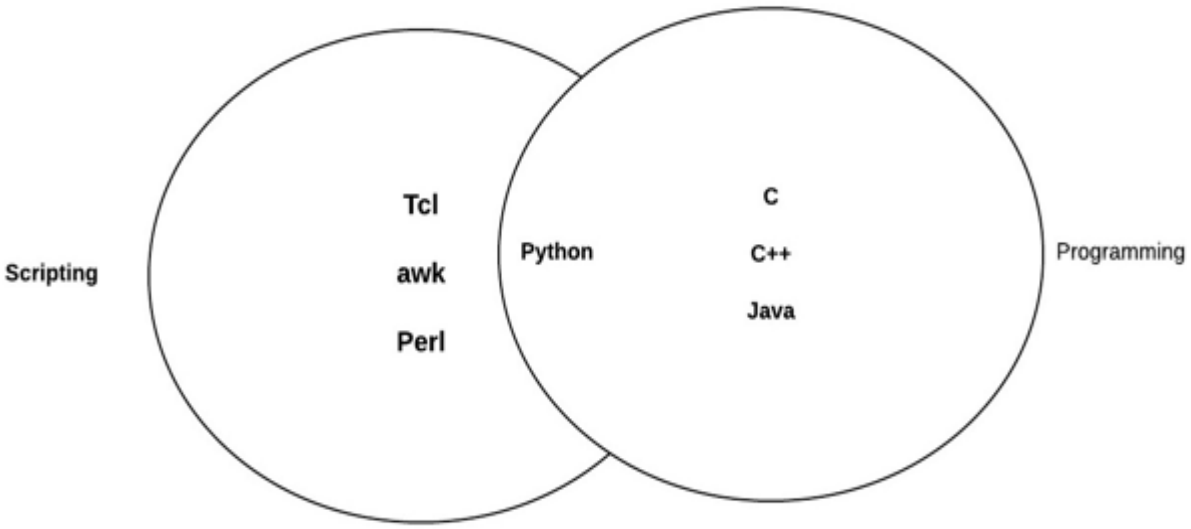
Python is a language similar to PERL (Practical Extraction and Reporting Language), which has gained popularity because of its clear syntax and readability.

The main notable features of Python are as follows:

- Python is said to be relatively easy to learn and portable. Its statements can be easily interpreted in a number of operating systems, including UNIX-based systems, Mac OS, MS-DOS, OS/2, and various versions of Windows.

- Python is portable with all the major operating systems. It uses an easy to understand syntax, making the programs, which are user friendly.

- It comes with a large standard library that supports many tasks.



From the above diagram, it is clearly visible that Python is a combination of scripting as well as programming language. They are interpreted within another program like scripting languages.

Versions of Python

Python has three production-quality implementations, which are called as CPython, Jython, and IronPython. These are also termed as versions of Python.

- **Classic Python** a.k.a **CPython** is a compiler, interpreter and consists of built-in and optional extension modules which is implemented in standard C language.
- **Jython** is a Python implementation for Java Virtual Machine (JVM).
- **IronPython** is designed by Microsoft, which includes Common Language Runtime (CLR). It is commonly known as .NET

Starting Up

A basic Python program in any operating system starts with a header. The programs are stored with **.py** extension and Python command is used for running the programs.

For example, **python_rstprogram.py** will give you the required output. It will also generate errors, if present.

Python uses indentation to delimit blocks of code. A block starts with a line ending with colon, and continues for all lines in the similar fashion that have a similar or higher indentation as the next line.

```
# Basic program in Python

print "Welcome to Python!\n"
```

The output of the program will be:

```
Welcome to Python!
```

Indentation

Indentations of the programs are quite important in Python. There are some prejudices and myths about Python's indentation rules for the developers who are beginners to Python.

The thumb rule for all the programmers is:

"Whitespace is significant in Python source code."

Leading whitespace, which includes spaces and tabs at the beginning of a logical line of Python computes the indentation level of line.

Note:

- The indentation level also determines the grouping of the statements.
- It is common to use four spaces i.e. tab for each level of indentation.

- It is a good policy not to mix tabs with spaces, which can result in confusion, which is invisible.

Python also generates a compile time error if there is lack of indentation.

```
IndentationError: expected an indented block
```

ControlFlow Statements

The control flow of a Python program is regulated by conditional statements, loops and function calls.

1. The **If** statement, executes a block of code under specified condition, along with else and elif(a combination of else-if).

2. The **For** statement, iterates over an object, capturing each element to a local variable for use by the attached block.

3. The **While** statement, executes a block of code under the condition, which is **True**.

4. The **With** statement, encloses a code block within the context manager. It has been added as a more readable alternative to the **try/finally** statement.

```
# If statement in Python
    x = int(raw_input("Please enter an integer: ")) #Taking input from the user
if x<0:
   print "1 - Got a negative expression value"
   print x
else:
   print "1 - Got a positive expression value"
   print x
print "Good bye!"
```

Output

```
sh-4.3$ python main.py
Please enter an integer: 4
1 - Got a positive expression value
4
Good bye!
```

Functions

The statements in a typical Python program are organized and grouped in a particular format called, **"Functions"**. A function is a group of statements that perform an action based on the request. Python provides many built-in functions and allows programmers to define their own functions.

In Python, functions are values that are handled like other objects in programming languages.

The **def** statement is the most common way to define a function. **def** is a single-clause compound statement with the following syntax:

```
def function-name (parameters):statement(s)
```

The following example demonstrates a generator function. It can be used as an iterable object, which creates its objects in a similar way.

```
def demo ():
    for i in range(5):
        yield (i*i)
for j in demo():
    print j
```

Output

```
sh-4.3$ python main.py
0
1
4
9
16
```

Special Attributes, Methods, and Operators

The attributes, methods, and operators starting with double underscore of a class are usually private in behavior. Some of them are reserved keywords, which include a special meaning.

Three of them are listed below:

- **__len__**
- **__getitem__**
- **__setitem__**

The other special operators include __getattr__ and __setattr__, which defines the **get** and **set** attributes for the class.

File I/O Functions

Python includes a functionality to open and close particular files. This can be achieved with the help of **open(), write() and close()** functions.

The commands which help in file input and output are as follows:

Command	Functionality
open()	It helps in opening a file or document
write()	It helps to write a string in file or document
read()	It helps in reading the content in existing file
close ()	This method closes the file object.

Example

Consider a file named "**demo.txt**", which already exists with a text "This is a demo file".

```
#!/usr/bin/python
# Open a file
fo = open("demo.txt", "wb")
fo.write( “Insering new line \n");
# Close opend file
fo.close()
```

The string available after opening the file will be:

This is a demo file

Inserting a new line

web2py Framework – Overview

web2py is a full-stack web framework that can be used by a developer to completely develop a web application. It includes SQL database integration and multi-threaded web server for designing a program.

Web Interface for Designing a User's Program

Once the command is executed as per the operating system, web2py displays a startup window and then displays a GUI widget that asks the user to choose-

- a one-time administrator password,
- the IP address of the network interface to be used for the web server,
- and a port number from which to serve requests.

The administrator includes all the authority for addition and editing any new web application.

By default, web2py runs its web server on **127.0.0.1:8000** (port 8000 on localhost) but a user can run it on any available IP address and port as per the requirement.

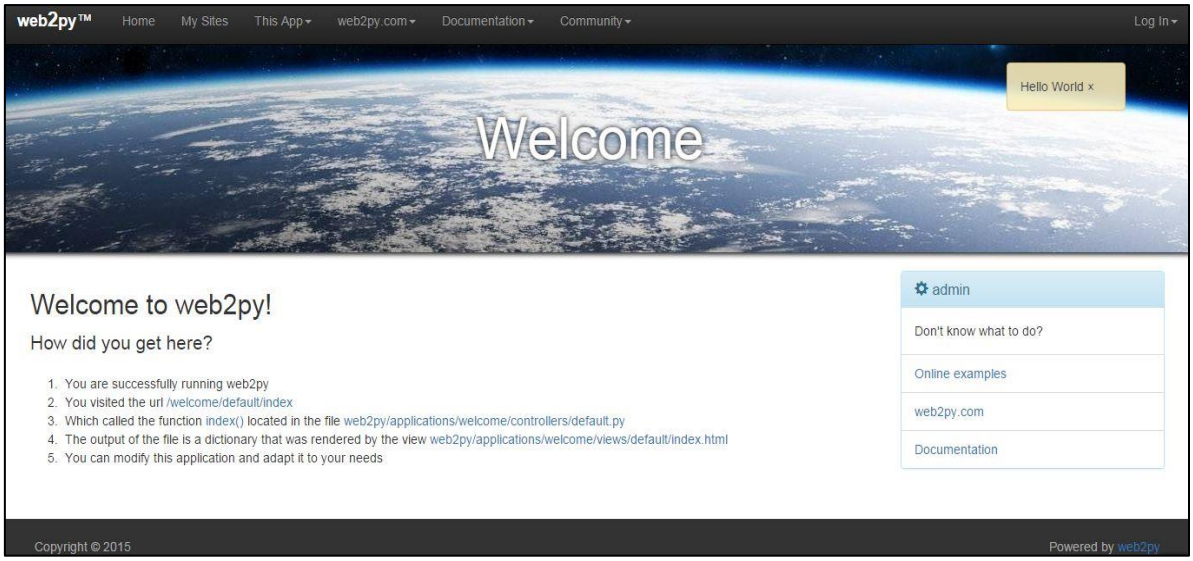
The web2py GUI widget will be displayed as shown below.

The password is used in the administrative interface for any changes in the new module.

After the user has set the administration password, web2py starts up the web browser at the page with the following URL:

http://127.0.0.1:8000/

The welcome page of the framework will be displayed as shown below.



End of ebook preview

If you liked what you saw…

Buy it from our store @ **https://store.tutorialspoint.com**