

LEARN PYTHON 3

Practical Course For Beginners To Programming
In One Week. A Complete Introduction Guide
To Learn Python Step By Step,
With Examples, Tips & Tricks And Simple
Exercises For Everybody



WILLIAM GRAY

LEARN PYTHON 3

PRACTICAL COURSE FOR BEGINNERS TO PROGRAMMING IN ONE WEEK. A COMPLETE INTRODUCTION GUIDE TO LEARN PYTHON STEP BY STEP, WITH EXAMPLES, TIPS & TRICKS AND SIMPLE EXERCISES FOR EVERYBODY

Written By **WILLIAM GRAY**

*Congratulation on downloading this ebook and thank
You for doing so.*

Please enjoy !

TABLE OF CONTENTS

<u>CHAPTER 1</u>	<u>9</u>
<u>LEARN PYTHON 3</u>	<u>9</u>
<u>INTRODUCTION</u>	<u>9</u>
<u>WHAT IS PYTHON?</u>	<u>12</u>
<u>How Python has been evolving over the years</u>	<u>14</u>
<u>CONCEIVED AS A HOBBY PROGRAMMING PROJECT</u>	<u>14</u>
<u>VERSION 1 OF PYTHON</u>	<u>14</u>
<u>VERSION 2 OF PYTHON</u>	<u>15</u>
<u>VERSION 3 OF PYTHON</u>	<u>15</u>
<u>LATEST VERSIONS OF PYTHON</u>	<u>16</u>
<u>VERSION 4 OF PYTHON</u>	<u>17</u>
<u>REASONS FOR LEARNING PYTHON</u>	<u>18</u>
<u>REASONS WHY THE MASSIVE POPULARITY OF PYTHON WILL REMAIN INTACT IN THE FUTURE</u>	<u>20</u>
<u>SUPPORTS MULTIPLE PROGRAMMING PARADIGMS</u>	<u>20</u>
<u>DOESN'T REQUIRE PROGRAMMERS TO WRITE LENGTHY CODE</u>	<u>20</u>
<u>PROVIDES A COMPREHENSIVE STANDARD LIBRARY</u>	<u>21</u>
<u>EFFECTUATES WEB APPLICATION DEVELOPMENT</u>	<u>21</u>
<u>FACILITATES DEVELOPMENT OF HIGH QUALITY GUI, SCIENTIFIC AND NUMERIC APPLICATIONS</u>	<u>22</u>
<u>SIMPLIFIES PROTOTYPING OF APPLICATIONS</u>	<u>22</u>
<u>CAN ALSO BE USED FOR MOBILE APP DEVELOPMENT</u>	<u>23</u>
<u>OPEN SOURCE</u>	<u>23</u>
<u>CHAPTER 2</u>	<u>24</u>
<u>THE BENEFITS OF PYTHON</u>	<u>24</u>

CHARACTERISTICS OF WEB PROGRAMMING
LANGUAGES 26

ASP.NET - ACTIVE SERVER PAGES 27

PHP 28

Java/JSP 29

Perl 29

Coldfusion 30

Ruby and Ruby on Rails 31

PYTHON FRAMEWORKS THAT WILL REMAIN
POPULAR 33

Kivy 33

Qt 33

PyGUI 34

WxPython 34

Django 34

CherryPy 35

Flask 35

Pyramid 36

Web.py 36

TurboGears 37

Why Django is Popular Among Existing Python
Programmers: 37

SHORTER AND CLEANER CODE 38

OPTIONS TO CUSTOMIZE WEB APPLICATIONS 39

BUILT-IN TOOLS FOR ACCOMPLISHING COMMON
TASKS 39

A VARIETY OF PACKAGES 39

OBJECT-RELATIONAL MAPPER (ORM) 40

HUMAN READABLE URLS 40

DYNAMIC ADMIN INTERFACE 41

OPTIMIZED SECURITY 41

OPTION TO EXCHANGE IDEAS 42

<u>CHAPTER 3</u>	<u>43</u>
<u>PYTHON GLOSSARY</u>	<u>43</u>
<u>CHAPTER 4</u>	<u>69</u>
<u>HOW TO GET UP AND RUNNING WITH PYTHON</u>	<u>69</u>
<u>HOW TO LEARN PROGRAMMING LANGUAGE</u>	<u>69</u>
<u>HOW TO SET UP PYTHON</u>	<u>78</u>
<u>DOWNLOADING PYTHON</u>	<u>79</u>
<u>CHAPTER 5</u>	<u>83</u>
<u>ADVANTAGES OF DOING WEB DEVELOPMENT WITH PYTHON</u>	<u>83</u>
<u>PYTHON IS EASY.</u>	<u>84</u>
<u>PYTHON LETS YOU BUILD MORE FUNCTIONS WITH FEWER LINES OF CODE.</u>	<u>85</u>
<u>PYTHON PROVIDES A STEPPING STONE TO LEARNING OTHER CODE.</u>	<u>85</u>
<u>IT'S HARD TO MESS UP WITH PYTHON.</u>	<u>86</u>
<u>PYTHON IS PERFECT FOR BUILDING PROTOTYPES.</u>	<u>86</u>
<u>PYTHON AND DJANGO ARE HUGE IN FINTECH.</u>	<u>87</u>
<u>PYTHON IS FLEXIBLE.</u>	<u>88</u>
<u>IT HAS A TON OF RESOURCES.</u>	<u>89</u>
<u>THERE'S A ROBUST STACK OF FRAMEWORKS WAITING FOR YOU.</u>	<u>89</u>
<u>DJANGO, A HIGH-LEVEL PYTHON WEB FRAMEWORK, IS FLAT-OUT AMAZING.</u>	<u>90</u>
<u>PYTHON IS GREAT IF YOU'RE ON A BUDGET.</u>	<u>91</u>
<u>PYTHON IS A HOT COMMODITY IN THE ERA OF INTERNET OF THINGS (IOT).</u>	<u>92</u>
<u>PYTHON IS A CORE TECHNOLOGY IN BLUE CHIP SITES AND SERVICES.</u>	<u>92</u>
<u>TECH GIANTS LOVE PYTHON.</u>	<u>93</u>
<u>THERE'S GOOD MONEY IN IT.</u>	<u>93</u>
<u>PYTHON IS PERFECT FOR GETTING INVOLVED IN HIGHER</u>	

<u>EDUCATION.</u>	<u>94</u>
<u>PYTHON MAKES SYSADMIN DUTIES A BREEZE.</u>	<u>94</u>
<u>PYTHON IS OPEN-SOURCE.</u>	<u>95</u>
<u>DJANGO SUPPORTS BEST PRACTICES FOR SEO.</u>	<u>96</u>
<u>DJANGO IS SECURE.</u>	<u>97</u>
<u>CHAPTER 6</u>	<u>98</u>
<u>FULL INSTRUCTIONS OF HOW TO CODE</u>	<u>98</u>
<u>TIPS TO HELP PHP PROGRAMMERS IMPROVE THEIR PROGRAM CODE</u>	<u>100</u>
<u>IMPORTANCE OF CODING STANDARDS</u>	<u>103</u>
<u>MAINTENANCE COSTS</u>	<u>104</u>
<u>ENFORCING THE STANDARD</u>	<u>105</u>
<u>READABLE CODE</u>	<u>105</u>
<u>BLOCKS OF CODE</u>	<u>106</u>
<u>CODE INDENTATION</u>	<u>106</u>
<u>VARIABLE NAMES</u>	<u>106</u>
<u>PROCEDURE NAMES</u>	<u>106</u>
<u>CODE DOCUMENTATION</u>	<u>107</u>
<u>COMPLEX CONSTRUCTS</u>	<u>107</u>
<u>KEEP THE CODING SIMPLE</u>	<u>107</u>
<u>ISOLATE COMPLEX CODE</u>	<u>108</u>
<u>REUSABLE CODE</u>	<u>108</u>
<u>HOW TO PROPERLY DEBUG YOUR CODE</u>	<u>108</u>
<u>REUSABLE CODE</u>	<u>112</u>
<u>CODE IGNITER WEB DEVELOPMENT</u>	<u>115</u>
<u>CHAPTER 7</u>	<u>118</u>
<u>BASICS OF THE OBJECT ORIENTED PROGRAMMING</u>	<u>118</u>
<u>HOW TO DEVELOP ADVANCED OBJECT-ORIENTED PROGRAM IN PHP</u>	<u>124</u>
<u>POPULAR OOP CONCEPTS IMPLEMENTED IN PHP</u>	<u>125</u>
<u>INHERITANCE</u>	<u>125</u>

<u>PUBLIC, PRIVATE, AND PROTECTED</u>	<u>125</u>
<u>OVERLOADING</u>	<u>126</u>
<u>PRINCIPLES OF OBJECT ORIENTED PROGRAMMING</u>	<u>132</u>
<u>OBJECT-ORIENTED DEVELOPMENT METHODS</u>	<u>136</u>
<u>PYTHON TECHNOLOGY USED FOR APPLICATION ORIENTED FIELDS</u>	<u>137</u>
<u>USES OF PYTHON TECHNOLOGY FOR APPLICATION DEVELOPMENT</u>	<u>138</u>
<u>FEATURES OF PYTHON</u>	<u>139</u>
<u>PYTHON DEVELOPMENT SERVICES</u>	<u>139</u>
<u>CHAPTER 8</u>	<u>140</u>
<u>REAL WORLD EXAMPLES OF PYTHON</u>	<u>140</u>
<u>WEB AND INTERNET DEVELOPMENT</u>	<u>140</u>
<u>APPLICATIONS OF PYTHON PROGRAMMING IN DESKTOP GUI</u>	<u>141</u>
<u>SCIENCE AND NUMERIC APPLICATIONS</u>	<u>142</u>
<u>SOFTWARE DEVELOPMENT APPLICATION</u>	<u>143</u>
<u>PYTHON APPLICATIONS IN EDUCATION</u>	<u>143</u>
<u>PYTHON APPLICATIONS IN BUSINESS</u>	<u>143</u>
<u>DATABASE ACCESS</u>	<u>144</u>
<u>NETWORK PROGRAMMING</u>	<u>144</u>
<u>AMES AND 3D GRAPHICS</u>	<u>145</u>
<u>OTHER PYTHON APPLICATIONS</u>	<u>146</u>
<u>PYTHON FRAMEWORKS AND THEIR REAL WORLD USAGE</u>	<u>147</u>
<u>Why should you go for Python?</u>	<u>147</u>
<u>Django framework:</u>	<u>148</u>
<u>Flask:</u>	<u>148</u>
<u>Pyramid:</u>	<u>149</u>
<u>WHY PYTHON HAS BECOME AN INDUSTRY FAVORITE AMONG PROGRAMMERS</u>	<u>149</u>
<u>What Makes Python a Preferred Choice Among</u>	

<u>Programmers?</u>	<u>150</u>
<u>First Steps in the World of Programming</u>	<u>150</u>
<u>Simple and Easy to Understand and Code</u>	<u>151</u>
<u>Getting Innovative</u>	<u>151</u>
<u>Python also Supports Web Development</u>	<u>151</u>

CHAPTER 1

LEARN PYTHON 3



INTRODUCTION

Python is one of the most popular coding languages. Along with being a high-level and general-purpose programming language, Python is also object-oriented and open source. At the same time, a good number of developers across the

world have been making use of Python to create GUI applications, websites and mobile apps. The differentiating factor that Python brings to the table is that it enables programmers to flesh out concepts by writing less and readable code. The developers can further take advantage of several Python frameworks to mitigate the time and effort required for building large and complex software applications.

The programming language is currently being used by a number of high-traffic websites including Google, Yahoo Groups, Yahoo Maps, Linux Weekly News, Shopzilla and Web Therapy. Likewise, Python also finds great use for creating gaming, financial, scientific and educational applications. However, developers still use different versions of the programming language. According to the usage statistics and market share data of Python posted on W3techs, currently Python 2 is being used by 99.4% of websites, whereas Python 3 is being used only by 0.6% of websites. That is why, it becomes essential for each programmer to understand different versions of Python, and its evolution over many years.

Python has been under active development since the late 1980s and is considered a mature programming language. The developers of the Python language conduct extensive functionality and regression testing to ensure the language remains bug-free and stable with each new release.

© Copyright 2019 by **WILLIAM GRAY**

All rights reserved

WHAT IS PYTHON?

The python programming language is a modern web programming language that was originally conceived and developed by Guido van Rossum in the 1980s. Since that time, Python has evolved into a high-performance programming language that is modular and extensible. Some of the largest websites in the world are utilizing Python such as YouTube, Disqus, and Reddit. Python offers a number of features that make it an attractive programming platform including stability, portability, object-oriented development, a powerful standard library and a wealth of third-party modules or packages.

Different people with a wide range of backgrounds, ages and educational levels are involved in Python development today. You can become one of them, no matter whether you are a student, a computer designer, a housewife, or a retiree. There is always a number of thorough instructions to make your introduction into the matter easier, and your success more sustainable.

Python programming offers a number of features that make it an attractive option for web application development. Python applications are portable due to the fact that python interpreters are available for all modern operating systems and some embedded computing systems.

The object-oriented nature of Python makes it an ideal first language for new programmers and easy to learn for programmers migrating to Python from other object-oriented languages. Python programming is intuitive and reinforces good program structure and object-oriented methodologies.

The standard Python library offers developers a plethora of features comparable to more complex languages such as C++ while maintaining simple and approachable language syntax. Comprehensive file-based I/O, database interactivity, advanced exception handling and a host of built in data types make Python appropriate for both web applications and general purpose programming. This makes python web programming an easy task for application developers seeking to transition to web application development.

Python is known for being a comprehensive language with extensive functionality included in the standard library. However, the growing popularity of python programming has led to a vast array of third-party packages, or modules, that extend Python's functionality and allow the language to deal with unique programming challenges. For example, modules are available for handling non-standard database interactions and advanced cryptography functionality. There are also modules available for dealing with common tasks such as reading file metadata, rendering charts and compiling Python applications into standardized executable applications. Python web programming is made easier due to the availability of many web-centric modules to handle tasks such as e-mail, maintaining HTTP state, interacting with JavaScript, and other common web development tasks.

In today's workplace, Python training is an increasingly important part of a programmer's education. As a dynamic language whose design philosophy revolves around readability and conciseness, Python is a popular choice for use as a scripting language. Like other interpretative languages, it is more flexible than compiled languages, and it can be used to tie disparate systems together. Indeed, Python is a versatile language with many applications in growing fields.

For example, Python is a popular programming language for educational software. Raspberry Pi, the single-board computer project for teaching students computer programming, uses Python as its primary programming language. In addition, much of the software for the One Laptop per Child XO is written in Python. At the other end of the educational spectrum, Python is also a very effective language for scientific computing and mathematical software for theoretical mathematics. As educational software development continues to grow, Python will become a more and more important language to know.

In addition to educational software, Python is also a favored language for use in AI tasks. Because Python is a scripting language with rich text processing tools, module architecture, and syntax simplicity, it is a natural choice for applications involving natural language processing. Programs like Wolfram Alpha and Siri are just beginning to penetrate the end-user market and many such programs yet to come will be written in Python.

Moreover, Python is often used as a scripting language for web applications. For example, Google has adopted Python as one of the available languages in its Google App Engine, a cloud computing platform for developing and hosting web applications. Python is also used as a framework to program communications between computers for web applications like Dropbox. As web application development is a fast-growing field, programmers would do well to acquire some Python training to keep their skills up-to-date.

Python is also quite useful as a modern scripting language similar to Perl, which can be used to tie disparate systems together. Because of this, because Python is a

standard component for many Linux and Unix based operating systems, and because Python is used extensively in the information security industry, Python is an important tool for systems administrators to learn, as well as programmers.

Python training is becoming an increasingly vital programming language. Due to its versatility, Python has a wide variety of uses in many growing fields. Both programmers and systems administrators would do well to pick up some Python savvy in order to keep their skills up-to-date.

How Python has been evolving over the years

CONCEIVED AS A HOBBY PROGRAMMING PROJECT

Despite being one of the most popular coding languages, Python was originally conceived by Guido van Rossum as a hobby project in December 1989. As Van Rossum's office remained closed during Christmas, he was looking for a hobby project that will keep him occupied during the holidays. He planned to create an interpreter for a new scripting language, and named the project as Python. Thus, Python was originally designed as a successor to ABC programming language. After writing the interpreter, Van Rossum made the code public in February 1991. However, at present the open source programming language is being managed by the Python Software Foundation.

VERSION 1 OF PYTHON

Python 1.0 was released in January 1994. The major release included a number of new features and functional programming tools including lambda, filter, map and reduce. The version 1.4 was released with several new features like keyword arguments, built-in support for complex numbers, and a basic form of data hiding. The major release was followed by two minor releases, version 1.5 in December 1997 and version 1.6 in September 2000. The version 1 of Python lacked the features offered by popular programming languages of the time. But the initial versions created a solid foundation for development of a powerful and futuristic programming language.

VERSION 2 OF PYTHON

In October 2000, Python 2.0 was released with the new list comprehension feature and a garbage collection system. The syntax for the list comprehension feature was inspired by other functional programming languages like Haskell. But Python 2.0, unlike Haskell, gave preference to alphabetic keywords over punctuation characters. Also, the garbage collection system effectuated collection of reference cycles. The major release was followed by several minor releases. These releases added a number of functionality to the programming language like support for nested scopes, and unification of Python's classes and types into a single hierarchy. The Python Software Foundation has already announced that there would be no Python 2.8. However, the Foundation will provide support to version 2.7 of the programming language till 2020.

VERSION 3 OF PYTHON

Python 3.0 was released in December 2008. It came with a several new features and enhancements, along with a number of deprecated features. The deprecated features and backward incompatibility make version 3 of Python completely different from earlier versions. So many developers still use Python 2.6 or 2.7 to avail the features deprecated from last major release. However, the new features of Python 3 made it more modern and popular. Many developers even switched to version 3.0 of the programming language to avail these awesome features.

Python 3.0 replaced print statement with the built-in print() function, while allowing programmers to use custom separator between lines. Likewise, it simplified the rules of ordering comparison. If the operands are not organized in a natural and meaningful order, the ordering comparison operators can now raise

a TypeError exception. The version 3 of the programming language further uses text and data instead of Unicode and 8-bit strings. While treating all code as Unicode by default it represents binary data as encoded Unicode.

As Python 3 is backward incompatible, the programmers cannot access features like string exceptions, old-style classes, and implicit relative imports. Also, the developers must be familiar with changes made to syntax and APIs. They can use a tool called "2to3" to migrate their application from Python 2 to 3 smoothly. The tool highlights incompatibility and areas of concern through comments and warnings. The comments help programmers to make changes to the code, and upgrade their existing applications to the latest version of programming language.

LATEST VERSIONS OF PYTHON

At present, programmers can choose either version 3.4.3 or 2.7.10 of Python. Python 2.7 enables developers to avail improved numeric handling and enhancements for standard library. The version further makes it easier for developers to migrate to Python 3. On the other hand, Python 3.4 comes with several new features and library modules, security improvements and CPython implementation improvements. However, a number of features are deprecated in both Python API and programming language. The developers can still use Python 3.4 to avail support in the longer run.

VERSION 4 OF PYTHON

Python 4.0 is expected to be available in 2023 after the release of Python 3.9. It

will come with features that will help programmers to switch from version 3 to 4 seamlessly. Also, as they gain experience, the expert Python developers can take advantage of a number of backward compatible features to modernize their existing applications without putting any extra time and effort. However, the developers still have to wait many years to get a clear picture of Python 4.0. However, they must monitor the latest releases to easily migrate to the version 4.0 of the popular coding language.

The version 2 and version 3 of Python are completely different from each other. So each programmer must understand the features of these distinct versions, and compare their functionality based on specific needs of the project. Also, he needs to check the version of Python that each framework supports. However, each developer must take advantage of the latest version of Python to avail new features and long-term support.

REASONS FOR LEARNING PYTHON

One of the most robust and dynamic programming languages being used today is Python. It stresses a lot on code readability, and because of its syntax as well as implementation, programmers have to write lesser codes in comparison to Java and C++. Memory management in Python is done automatically and several standard libraries are available for the programmer here. After completing a certification course in Python training, a programmer can gain experience in various top IT companies.

Python programming supports numerous styles such as functional programming, imperative and object-oriented styles. Here are the top five reasons why a

computer programmer must learn the Python language:

Ease of learning- Python has been created with the newcomer in mind. Completion of basic tasks requires less code in Python, compared to other languages. The codes are usually 3-5 times shorter than Java, and 5-10 times smaller than C++. Python codes are easily readable and with a little bit of knowledge, new developers can learn a lot by just looking at the code.

Highly preferred for web development- Python consists of an array of frameworks which are useful in designing a website. Among these frameworks, Django is the most popular one for python development. Due to these frameworks, web designing with Python has immense flexibility. The number of websites online today are close to 1 billion, and with the ever-increasing scope for more, it is natural that Python programming will continue to be an important skill for web developers.

Considered ideal for start-ups- Time and budget are vital constraints for any new product or service in a company, and more so if it is a startup. One can create a product that differentiates itself from the rest in any language. However, for quick development, less code and lesser cost, Python is the ideal language here. Python can easily scale up any complex application and also can be handled by a small team. Not only do you save resources, but you also get to develop applications in the right direction with Python.

Unlimited availability of resources and testing framework- Several resources for Python are available today, and these are also constantly being updated. As a result, it is very rare that a Python developer gets stuck. The vast standard library

provides in-built functionalities. Its built in testing framework enables speedy workflows and less debugging time.

Fat paycheques- Today top IT companies such as Google, Yahoo, IBM and Nokia make use of Python. Among all programming languages, it has had amazing growth over the last few years.

It is clear that Python is a vital language for web-based programmers. More can be learnt at a reputed Python training institute.

REASONS WHY THE MASSIVE POPULARITY OF PYTHON WILL REMAIN INTACT IN THE FUTURE

SUPPORTS MULTIPLE PROGRAMMING PARADIGMS

Good developers often take advantage of different programming paradigms to reduce the amount of time and efforts required for developing large and complex applications. Like other modern programming languages, Python also supports a number of commonly used programming styles including object-oriented, functional, procedural and imperative. It further features automatic memory management, along with a dynamic type system. So programmers can use the language to effectuate development of large and complex software applications.

DOESN'T REQUIRE PROGRAMMERS TO WRITE LENGTHY CODE

Python is designed with complete focus on code readability. So the programmers can create readable code base that can be used by members of distributed teams. At the same time, the simple syntax of the programming language enables them to express concepts without writing longer lines of code. The feature makes it easier for developers to large and complex applications within a stipulated amount of time. As they can easily skip certain tasks required by other programming languages, it becomes easier for developers to maintain and update their applications.

PROVIDES A COMPREHENSIVE STANDARD LIBRARY

Python further scores over other programming languages due to its extensive standard library. The programmers can use these libraries to accomplish a variety of tasks without writing longer lines of code. Also, the standard library of Python is designed with a large number of high use programming tasks scripted into it. Thus, it helps programmers to accomplish tasks like string operations, development and implementation of web services, working with internet protocols, and handling operating system interface.

EFFECTUATES WEB APPLICATION DEVELOPMENT

Python is designed as a general-purpose programming language, and lacks built-in web development features. But the web developers use a variety of add-on modules to write modern web applications in Python. While writing web applications in Python, programmers have option to use several high-level web frameworks including Django, web2py, TurboGears, CubicWeb, and Reahl. These web frameworks help programmers to perform a number of operations, without writing additional code, like database manipulation, URL routing, session storage and retrieval, and output template formatting. They can further use the web frameworks to protect the web application from cross-site scripting attacks, SQL injection, and cross-site request forgery.

FACILITATES DEVELOPMENT OF HIGH QUALITY GUI, SCIENTIFIC AND NUMERIC APPLICATIONS

Python is currently available on major operating systems like Windows, Mac OS X, Linux and UNIX. So the desktop GUI applications written in the

programming language can be deployed on multiple platforms. The programmers can further speedup cross-platform desktop GUI application development using frameworks like Kivy, wxPython and PyGtk. A number of reports have highlighted that Python is used widely for development of numeric and scientific applications. While writing scientific and numeric applications in Python, the developers can take advantage of tools like Scipy, Pandas, IPython, along with the Python Imaging Library.

SIMPLIFIES PROTOTYPING OF APPLICATIONS

Nowadays, each organization wants to beat competition by developing software with distinct and innovative features. That is why; prototyping has become an integral part of modern software development lifecycle. Before writing the code, developers have to create prototype of the application to display its features and functionality to various stakeholders. As a simple and fast programming language, Python enables programmers to develop the final system without putting any extra time and effort. At the same time, the developers also have option to start developing the system directly from the prototype simply by refactoring the code.

CAN ALSO BE USED FOR MOBILE APP DEVELOPMENT

Frameworks like Kivy also make Python usable for developing mobile apps. As a library, Kivy can be used for creating both desktop applications and mobile apps. But it allows developers to write the code once, and deploy the same code on multiple platforms. Along with interfacing with the hardware of the mobile device, Kivy also comes with built-in camera adapters, modules to render and play videos, and modules to accept user input through multi-touch and gestures.

Thus, programmers can use Kivy to create different versions of the same applications for iOS, Android and Windows Phone. Also, the framework does not require developers to write longer lines of code while creating Kivy programs. After creating different versions of the mobile app, they can package the app separately for individual app store. The option makes it easier for developers to create different versions of the mobile app without deploying separate developers.

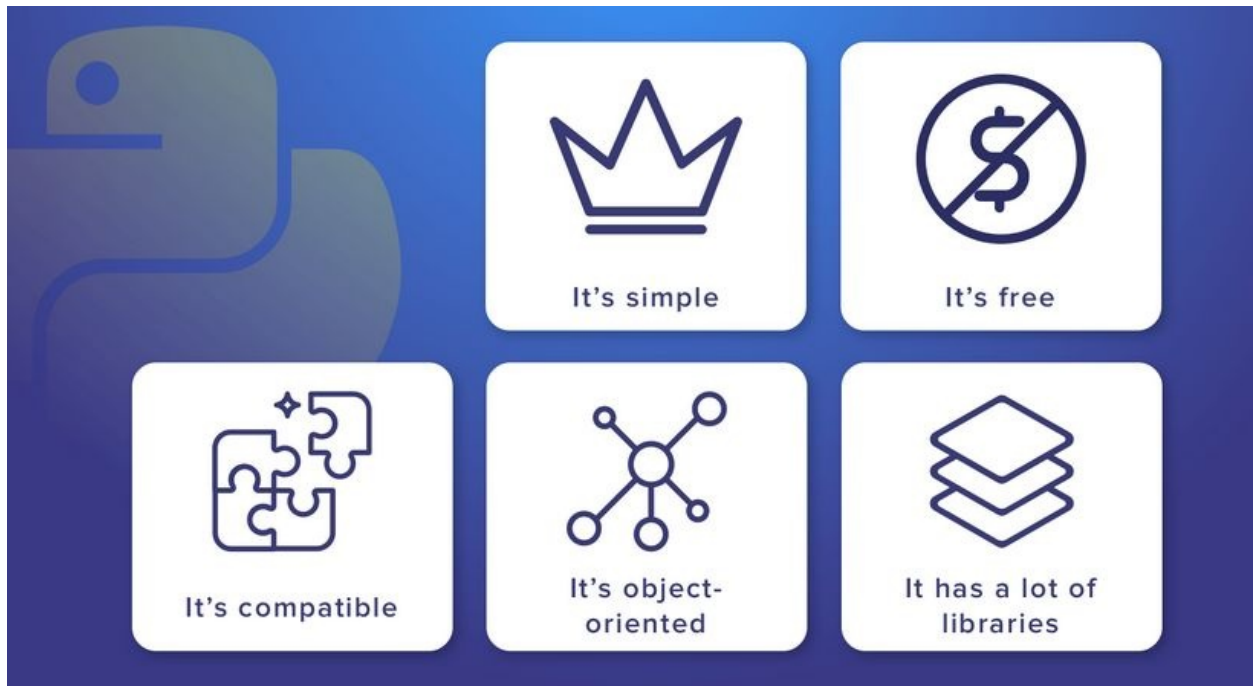
OPEN SOURCE

Despite being rated as the most popular coding language of 2015, Python is still available as open source and free software. Along with large IT companies, the startups and freelance software developers can also use the programming language without paying any fees or royalty. Thus, Python makes it easier for businesses to reduce development cost significantly. At the same time, the programmers can also avail the assistance of large and active community to add out-of-box features to the software application.

The last major release of Python took place in December 2008. Python 3 was released as a backward-incompatible version with most of the major features back ported to Python 2.6 and 2.7. However, the programming language is being updated by the community at regular intervals. The community released Python 3.4.3 on 23rd February with several features and patches. So the developer can always use the most recent version of the Python programming language to effectuate development of various software applications.

CHAPTER 2

THE BENEFITS OF PYTHON



Firstly let's discuss about the benefits of all web programming languages:

It all began with Tim Berners-Lee at CERN and the good old HTML. CERN, which is now mostly famous for its Large Hadron Collider, has recently gained major publicity by attracting the attention of frivolous news chasing the possibility of a black hole. I think even cooler than a black hole marking the end of humanity, is the virtual world that has largely impacted her evolution. The internet happened partly because of the outcome of CERN's research and now we have a world beside the real world, functioning in parallel and becoming the flesh and blood of a terrestrial specy who was one day drawing on cave walls.

HTML later became lingua franca of a world that was becoming bigger while making our real world smaller and smaller.

HTML is the basic language understood by all WWW (World Wide Web) clients. It can execute on a PC under any operating system such as Windows, Mac, Linux, or on a Unix workstation. However, it is limited in its computational power intentionally because it can prevent the execution of dangerous programs on the client machine. Web programmers, who are now much more sophisticated in their applications, provide different type of services to a growing demand of interactive content. Today, most users have competent client machines which are capable of doing much more than HTML allows. Fortunately, there is steady development in the field, and today the number of capable applications is expanding. We can easily build database-driven websites with various scripting languages such as PHP, ASP.NET, JSP, Perl, Cold Fusion and etc. This gives programmers a head ache. They generally fall into two main groups - proprietary and open-source, both have their own share of the market.

The languages mentioned below are all attempts to create the "ideal" Web programming language, which is usually done by extending and restricting existing languages. Web programming languages have a variety of ancestors: scripting languages, shell languages, mark-up languages and conventional programming languages. This document attempt to present a short introduction of the most important languages being used in Web today.

CHARACTERISTICS OF WEB PROGRAMMING LANGUAGES

There is a diverse range of languages available and suitable for Web

programming. There is no reason to believe that any one language will completely prevail and monopolize the Web programming scene. When you are developing a simple website, the question of which programming language and framework to choose can come up for things such as contact submission forms, photo galleries, jQuery Slider or any other dynamic content components that is generated by the web-server. When you are shopping for a web designer or web developer, you will hear them say "We program in PHP" or "We develop in ASP.NET/C#" and you might wonder what they are talking about and which one is better than the other.

As you can see, there is a fairly good selection of languages for web developers to choose to develop their web applications. There are many more languages available, some of them more specialized such as VRML, but discussing them is out of the scope of this article. Most developers have a working knowledge for at least a few of the frameworks, but tend to specialize in one.

New languages and language extensions are being developed to increase the usability of the Internet on a daily basis. Here we will look at some of the important languages that have shaped the Internet over the years and give a brief overview of each one of them.

ASP.NET - ACTIVE SERVER PAGES

The .NET Framework is the infrastructure for the Microsoft .NET platform. It is an environment that can be used for building, deploying, and running Web applications and Web Services with easier and quicker programming.

ASP.NET runs inside IIS which comes free with Windows servers. It enables scripts to be executed by an Internet server. This web development technology is built into the .NET Framework and utilizes full featured programming languages such as C# or VB.NET to build web applications. I personally like C# very much and I think it is a great language to code with and extremely easy to use.

ASP.NET drastically reduces the amount of code required to build large applications. With built-in Windows authentication and pre-application configuration, security and safety are guaranteed. But the main disadvantage is that ASP.NET applications can only run on Windows platform.

PHP

As an open-source alternative, PHP is steadily developed by an active and very dedicated international community. This is a great example of strength in numbers. Another strength of PHP is of course its low costs. Since it is open-source software, PHP can be compiled and customized for any operating system. In fact, there are always pre-compiled versions available for your OS of choice. In an open, collaborative and non-hierarchical environment, suggested improvements can be adopted quickly. You can count on its fast paced updates and improvements more than other languages.

I think PHP is a good investment because it is not going to disappear anytime soon. It is used by a very large group of fantastic programmers who come out with amazing websites combining PHP with other technologies. Also it is a good idea that if you have invested in an online business and you have lost one of your precious programmers, it will be the easiest to find a replacement programmer.

Java/JSP

Developed by Sun Microsystems, Java is a very powerful object-oriented language. JavaServer Pages (JSPs) are web pages with embedded Java code. The embedded Java code is executed on the server, then the page is returned to the browser for display.

Unlike ASP, JSP is a lot less platform-specific and it doesn't rely as heavily on Microsoft for support or performance improvements. Java programs for Unix can be made to run on Windows or the Mac system with little or no effort. Many development projects are taking place on the Java platform and it is getting richer by day.

You should not confuse JSP with Javascript. Note that Javascript is a programming language that runs on a web browser and does not require any server software. Since all execution takes place on the browser, Javascript is responsible for most of the interactivity on a web page. Image change or text color change on mouseover, creating mouse trails are all possible through Javascript.

Perl

Perl doesn't offer the graphics and security desired for Internet programming on its own. However, those features are now available through Perl/Tk. Perl/Tk which extends Perl with access to the Tk GUI library and makes it fully available for web programming.

Perl is an open-source language that is both mature and powerful. It offers web developers every tool they need to create dynamic websites. Like other open-source languages, it benefits tremendously from ongoing development, and the support offered by its international community is amazing. Perl is particularly good for creating single websites quickly, cleanly and elegantly. Its major weakness is that it may be unnecessarily complicated. If you are not comfortable switching gears between different syntaxes, then you should choose from one of the other alternatives.

Coldfusion

Built by Allaire in 1995, purchased by Macromedia in 2001 and finally ended up in the hands of Adobe in 2005, This ColdFusion might not be as cool as cold fusion nuclear energy, but it did introduce an exciting technology which hold an active developer community together up to this day.

ColdFusion enables programmers to create dynamic and database-powered Web applications. With its own markup language (CFML) and tags to connect to the database, it is relatively easy to create forms and dynamic pages. Using ColdFusion, Building websites can't possibly get easier! You can deploy powerful web applications and services with minimal training. It provides an application development platform suitable for network software. It is now at version 10 and supports most major databases from Oracle to Microsoft SQL Server.

Ruby and Ruby on Rails

Ruby is a dynamic, object-oriented and open-source programming language. It has a very clean syntax which makes it a lot of fun to use. Python is ease to learn and use, and closely enjoys the Perl's pragmatism. Ruby's simplicity, productivity and ease of use has spread its usage very quickly in a matter of few years.

Ruby on Rails is its open-source Web application framework written in Ruby which follows the Model-View-Controller (MVC) model. It is a highly-productive and can empower industrial-strength web applications. It scales from the simplest expense tracking application to full-featured applications with thousands of users.

Now let's discuss why Python is more important:

As a dynamic, general purpose and object-oriented programming language, Python is used widely by developers across the world for building a variety of software applications. Unlike other modern programming languages, Python enables programmers to express concepts with less and readable code. The users also have an option to integrate Python with other popular programming languages and tools seamlessly. But it cannot be used directly for writing different types of software.

Often Python developers have to use a variety of frameworks and tools to build high quality software applications within a shorter amount of time. The resources provided by the Python frameworks help users to reduce the time and effort required for modern applications. They also have an option to choose

from a number of frameworks according to the nature and requirements of individual projects

Like Java, Python is a full featured, object-oriented language developed, but extremely easy to use, by Guido van Rossum at CWI in the Netherlands. Similar to almost every other cool language, it is initially developed in a Unix environment but is available on PCs and Macs, and applications are portable across platforms. Python's syntax resembles C and C++, but doesn't stick too closely to those languages.

Python offers several portable GUI libraries. Python is simpler and easier to program than Java and very suited for Internet programming.

PYTHON FRAMEWORKS THAT WILL REMAIN POPULAR

Kivy

As an open source Python library, Kivy makes it easier for programmers to build multi-touch user interfaces. It supports a number of popular platforms including Windows, Linux, OS X, iOS and Android. So the cross-platform framework enables users to create the app for multiple platforms using the same code base. It is also designed with features to take advantage of the native inputs, protocols and devices. Kivy further includes a fast graphic engine, while allowing users to choose from more than 20 extensible widgets.

Qt

The open source Python framework is written in C++. Qt enables developers to build connected applications and UIs that run on multiple operating systems and devices. The developers can further create cross-platform applications and UIs without making any changes to the code. Qt further scores over other frameworks due to its comprehensive library of APIs and tools. The programmers have option to use Qt either under the community license or the commercial license.

PyGUI

PyGUI is considered to be simpler than other Python frameworks. But it enables developers to create GUI API by taking advantage of the language features of

Python. PyGUI currently supports Windows, OS X and Linux. So the developers can use it for creating lightweight GUI APIs that can be implemented on these three platforms. They can further document the API comprehensively without referring to the documentation of any third-party GUI library.

WxPython

The GUI toolkit for Python helps programmers to create applications with highly functional graphical user interfaces. As wxPython supports Windows, Linux and OS X, it becomes easier for developers to run the same program in multiple platforms without modifying the code. The users can write the programs in Python, while taking advantage of the 2D path drawing engine, standard dialogs, dockable windows and other features provided by the framework.

Django

Django is the most popular high-level web application development framework for Python. Despite being open source, Django provides a simple and rapid development environment for building a variety of websites and web applications rapidly. It further helps programmers to create web application without writing lengthy code. It further comes with features to prevent some of the common security mistakes made by the developers.

CherryPy

As a minimalist web framework, CherryPy enables programs to create websites and web applications just like writing other object-oriented Python programs. So

it becomes easier for developers to build web applications without writing lengthy code. CherryPy further comes with a clean interface, while allowing developers to decide the right frontend utilities and data storage option. Despite being the oldest Python web application development framework in the market, CherryPy is still being used by programmers to create a variety of modern websites.

Flask

Flask is one of the micro web frameworks available for Python. Its core is simple and easy to use, but highly extensible. It also lacks many features provided by other web frameworks including database abstraction layer and form validations. Also, it does not allow users to add common functionality to the web application through third-party libraries. However, Flask enables programmers to create website rapidly by using extensions and code snippets. The snippets and patterns contributed by other members help developers to accomplish common tasks like database access, caching, file upload and authentication without writing any additional code.

Pyramid

Despite being a lightweight and simple Python web framework, Pyramid is hugely popular among programmers due to its high and rapid performance. The open source framework can be used for creating a variety of applications. Once the standard Python development environment is set up, the developers can use Pyramid to build the applications rapidly. Pyramid further allows users to take advantage of an independent Model-view-controller (MVC) structure. At the same time, they can further take advantage of other frameworks by integrating

them with Pyramid.

Web.py

As a simple but powerful web framework for Python, web.py helps programmers to build a variety of modern web applications rapidly. The combination of simple architecture and impressive development potential further helps users to overcome some of the common restrictions and inconveniences in web development. It still lacks many features provided by other modern web frameworks. But developers can easily integrate web.py with other frameworks to avail a number of advanced features and functionality.

TurboGears

As a highly-scalable web application development framework for Python, TurboGears helps users to eliminate restrictions and limitations within the development environment. It can be used as a micro-framework or full-stack framework. It further provides a flexible object relationship mapper (ORM), along with supporting several databases, multiple data exchange formats, and horizontal data partitioning. The developers can further use the new widget system provided by TurboGears to effectuate development of AJAX-heavy web applications.

On the whole, the Python developers have option to choose from many frameworks. Some of these frameworks effectuate development of GUI desktop applications, whereas others help programmers to build modern websites and web application rapidly. At the same time, the developers also have option to use

certain frameworks to write mobile apps in Python. That is why; it becomes essential for the developer to assess the suitability of each framework for his project based on its features and functionality. The user can also consider integrating the framework with other frameworks and tools to avail more advanced features and functionality.

Why Django is Popular Among Existing Python Programmers:

As a powerful server side scripting language, Python makes it easier for developers to build high-performing websites rapidly. The object-oriented programming language supports modules and packages. So the developers can divide the code into different modules, and reuse these modules across different projects. They can further reduce overall development time and efforts significantly by using a Python web framework.

As highlighted by several surveys, existing Python developers across the world prefer Django to other popular Python web frameworks like TurboGears, Falcon, Pyramid, web2py and web.py. Along with being a high-level web framework, Django is also flexible and extensible, and comes with features that help developers to create customized internet applications. There are also a number of reasons why Django is hugely popular among both beginners and existing Python programmers.

What Makes Django Popular Among Existing Python Programmers?

SHORTER AND CLEANER CODE

The existing Python programmers understand the long-term benefits of a shorter and cleaner code base. As Python enables those to express common concepts with less code, they can always avoid creating longer code. At the same time, Django supports model-view-controller (MVC) pattern. The pattern makes it

easier for programmers to organize their code efficiently by keeping the business logic, user interface and application data separate. The combination of Python and Django helps experienced developers to create readable, shorter and cleaner code.

OPTIONS TO CUSTOMIZE WEB APPLICATIONS

Nowadays each business wants its website to deliver distinct and rich user experience. Python developers look for options to customize pieces of websites without putting any extra time and effort. As a flexible web framework, Django enables them to customize different pieces of a website. Instead of using pre-built web applications, the programmers are required to focus only on customizing pieces of the website according to client's specific requirements. The focus enables them to create applications that deliver relevant content or information according to the specific needs of user.

BUILT-IN TOOLS FOR ACCOMPLISHING COMMON TASKS

Django is being updated regularly with new features and built-in tools. It includes a variety of built-in tools that help users to accomplish common web development tasks without writing lengthy code. These built-in tools help programmers to reduce the amount of time required for developing large websites.

A VARIETY OF PACKAGES

The existing Python programmers further boost performance of their web application using Django packages. The Django packages include reusable tools, apps, and sites. Many developers frequently use apps like Django Extensions, Django Celery, Django Rest Framework and South. They also effectuate development of ecommerce websites by using django SHOP, django-oscar, Satchmo, satchless or Cartridge. They also have option to choose from a variety of reusable tools, apps and sites according to the nature and needs of the web application. These packages make it easier for them to boost the website's performance without writing extra code.

OBJECT-RELATIONAL MAPPER (ORM)

The choice of database differs from one client to another. The experienced Python developers prefer using object-relational mapper to write database queries without using SQL. Django comes with an ORM that enables developers to manipulate database without writing lengthy SQL queries. The framework implements the ORM by default to allow programmers to describe the database layout as a Python class. At the same time, they also have option to use a Python API to access data in a more efficient way. As the API is generated on the fly, the developers are not required to generate any additional code. That is why; Django is used widely for development of data-driven websites.

HUMAN READABLE URLS

The beginners often ignore the significance of human readable URLs. But existing Python developers understand the benefits of human readable URLs for the web application. The website visitors can understand and remember the URL more easily. Also, the human readable URLs will make the web pages rank

higher on search engine results pages. Django makes it easier for programmers to create simple, readable and easy-to-remember URLs for both website visitors and search engine bottoms.

DYNAMIC ADMIN INTERFACE

Each client wants a simple and dynamic admin interface to manage the application smoothly. Django is designed with features to generate a production-ready admin interface. The dynamic admin interface allows authenticate users to add, delete and change objects. Thus, it makes it easier for the business to edit or update the website content, without using any backend interface. The existing Python programmers take advantage of this feature to setup and run admin sites while developing the models.

OPTIMIZED SECURITY

Python scores over other popular web programming language in the category of security. The existing Python developers also avail the features of Django to optimize the security of Python web application. Unlike other web frameworks, Django often generates web pages dynamically, and sends the content to web browsers through templates. So the source code remains hidden from both the web browser and end users. As the source code is not directly exposed to the end users, the internet application gets comprehensive security cover. At the same time, the developers can also use Django to prevent cross-site scripting attacks, SQL injection and other security threats.

OPTION TO EXCHANGE IDEAS

Like other open source technologies, Django is also supported by a large and active community. So the existing Python web developer often avail assistance of the community to handle new issues. At the same time, they also exchange ideas and best practices with other members of the community on a regular basis. The exchange makes it easier for them to keep track of the latest trends in web development, along with understanding how to implement these trends without any hassle.

The existing Python programmers also upgrade to the latest version of Django to avail new features and enhancements, along with a number of bug fixes. Further, they can avail regular security updates for the most recent version of the web framework to protect the application from latest security threats. Many programmers even upgrade to the latest version of Django to keep their code base relevant and up to date.

The default prompt of the Python interactive shell when entering code under an indented block or within a pair of matching delimiters. Delimiters may be parentheses, curly braces, or square brackets.

This is also called the ellipsis object.

3. 2to3

While most of the applications existing today have their base in Python 2.x, the future belongs to Python 3.x. But 2.x code isn't completely compatible with 3.x. Interestingly, we have a tool available that will help us convert Python 2.x code to Python 3.x.

2to3 handles the incompatibilities, detecting them by parsing the source and traversing the parse tree. The standard library has this as lib2to3.

4. Abstract Base Class

An abstract base class provides a way to define interfaces. This way, it complements duck typing. For this, we have the module abc. It introduces virtual subclasses (classes that are recognized by `isinstance()` and `issubclass()`, but do not inherit from another class. Python has several built-in ABCs for data structures (use the `collections.abc` module), numbers (use the `numbers` module), or streams (use the `io` module). You can also import finders and loaders (use the `importlib.abc` module). And to create our own ABCs, we use the `abc` module.

5. Python Argument

An argument is a value we pass to a function or a method when calling it. In Python, we have the following kinds of arguments:

a. Default Arguments

When defining a function, we can provide default values for arguments. This way, when we call it without any missing arguments, the default values will fill in for them. Default arguments can only follow non-default ones.

```
>>> def sayhello(name='User'):
```

```
    print(f"Hello, {name}")
```

```
>>> sayhello('Ayushi')
```

Hello, Ayushi

```
>>> sayhello()
```

Hello, User

b. Keyword Arguments Python

Keyword arguments pertain to calling a function. When we then call the

function, we can pass it arguments in any order.

```
>>> def subtract(a,b):  
    return b-a
```

```
>>> subtract(3,2)
```

```
>>> subtract(b=2,a=3)
```

c. Arbitrary Arguments

When we don't know how many arguments we'll get, we use an asterisk to denote an arbitrary argument.

```
>>> def sum_all(*nums):  
    total=0  
    for i in nums:  
        total+=i  
    return total
```

```
>>> sum_all(1,2,3,4)
```



```
>>> sum_all(1,2,3)
```

d. Positional Arguments Python

These are regular arguments that aren't keyword arguments. Python Positional Argument Example.

```
>>> def add(a,b):
```

```
    return a+b
```

```
>>> add(3,4)
```

We use a * before an iterable if we must pass it as an argument to a function.

```
>>> add(*(3,4))
```

7. Asynchronous Context Manager

ACM is an object that controls the environment observed in an async with statement. It does so by defining `__aenter__()` and `__aexit__()`.

8. Asynchronous Python Generator

We have seen about Generators in Python. They let us yield one object at a time.

An asynchronous generator is a function that returns an asynchronous generator iterator. We define it with 'async def', and it contains 'yield' expressions to produce a series of values. We can use these values in an async for-loop.

Such an asynchronous generator function may contain await expressions, and async for and async with statements.

9. Asynchronous Generator Iterator

An asynchronous generator function creates an asynchronous generator iterator.

When we call this iterator using the `__anext__()` method, it returns an awaitable object. This object executes the function's body until the next yield expression.

Actually, each yield suspends processing temporarily. It remembers the location execution state, and the local variables and pending try statements. On resuming with another awaitable returned by `__anext__()`, the generator iterator picks up where it left off.

10. Asynchronous Iterable

It is an object that we can use in an async for statement. It must return an

asynchronous iterator from its `__aiter__()` method.

Any Doubt yet in Python Glossary? Please Comment.

11. Asynchronous Iterator

An asynchronous iterator is an object that implements `__aiter__()` and `__anext__()` methods. `__anext__()` must return an awaitable object. `async` for resolves the awaitable returned from the iterator's `__anext__()` method until it raises a `StopAsyncIteration` exception.

12. Attribute

An attribute is a value an object holds. We can access an object's attributes using the dot operator (`.`). In our examples, we have done this as following:

```
orange.color
```

13. Awaitable

Any object in Python that we can use in an `await` expression is an awaitable. It can be a coroutine or any object with an `__await__()` method.

14. BDFL

Who other than Guido Van Rossum, the creator of Python, deserves to be called

Benevolent Dictator For Life?

15. Binary File

A file object that is able to read and write bytes-like objects is a binary file. When we open a file in a binary mode, we use the modes 'rb', 'wb', or 'rb+'.

More on File I/O.

16. Bytes-like Object

Any object that supports the Buffer Protocol, and is able to export a C-contiguous buffer, is a bytes-like object. Examples include bytes, bytearray, and array.array objects. It also includes many common memoryview objects.

We can use such objects for operations that deal with binary data (compression, saving to a binary file, sending over a socket, and more)

17. Bytecode

As you know, Python compiles its source code into bytecode. It is the internal representation of a Python program in the CPython interpreter. When we talked earlier of .pyc files, we mentioned that bytecode is cached into them. This lets the files execute faster the second time since they don't need to recompile.

In essence, bytecode is like an intermediate language that runs on a virtual machine. This virtual machine converts it into machine code for the machine to actually execute it on. However, one bytecode will not run on a different virtual machine.

If you're interested in finding out about bytecode instructions, you can refer to the official documentation for the `dis` module.

18. Python Class

A class, in Python, is a template for creating user-defined objects. It is an abstract data type, and acts as a blueprint for objects of a kind while having no values itself.

To learn how to create and use a class, refer to [Classes in Python](#).

19. Coercion

When we carry out operations like `2+3.7`, the interpreter implicitly converts one data type to another. Here, it converts `2` to `2.0` (int to float), and then adds, to it, `3.7`. This is called coercion, and without it, we would have to explicitly do it this way:

```
>>> float(2)+3.7
```

5.7

20. Complex Number

A complex number is made of real and imaginary parts. In Python, we use 'j' to represent the imaginary part.

```
>>> type(2+3.7j)
<class 'complex'>
```

An imaginary number is a real multiple of -1(the imaginary unit). To work with complex equivalents of the math module, we use cmath. For more on complex numbers, read up on Python Numbers.

These Python Glossary terms are very important to know before you dive into learning Python.

21. Context Manager

The context manager is an object that controls the environment observed in a with-statement. It does so with the `__enter__()` and `__exit__()` methods.

22. Coroutine

A subroutine enters at one point and exits at another. A coroutine is more

generalized, in that it can enter, exit, and resume at many different points. We implement them with the `async def` statement.

23. Coroutine Function

A coroutine function is simply a function that returns a coroutine object. We may define such a function with the `async def` statement, and it may contain the keywords `await`, `async for`, and `async with`.

24. CPython

CPython is the canonical implementation of Python in C. It is the one distributed on python.org.

25. Python Decorator

A decorator is a function that returns another function, or wraps it. It adds functionality to it without modifying it. For a simple, detailed view on decorators, refer to [Python Decorators](#).

26. Descriptor

If an object defines methods `__get__()`, `__set__()`, or `__delete__()`, we can call it a descriptor. On looking up an attribute from a class, the descriptor attribute's

special binding behavior activates. Using `a.b` looks up the object 'b' in the class dictionary for 'a'. If 'b' is a descriptor, then the respective descriptor methods is called.

27. Python Dictionary

A dictionary is an associative array that holds key-value pairs. Think of a real-life dictionary. Any object with `__hash__()` and `__eq__()` methods can be a key.

28. Dictionary View

A dictionary view is an object returned from `dict.keys()`, `dict.values()`, or `dict.items()`. This gives us a dynamic view on the dictionary's entries. So, when the dictionary changes, the view reflects those changes.

29. Docstring

A docstring is a string literal that we use to explain the functionality of a class, function, or module. It is the first statement in any of these constructs, and while the interpreter ignores them, it retains them at runtime. We can access it using the `__doc__` attribute of such an object. You can find out more about docstrings in Python Comments.

30. Duck-Typing

We keep saying that Python follows duck-typing. But what does this mean? This means that Python does not look at an object's type to determine if it has the right interface. It simply calls or uses the method or attribute. "If it looks and quacks like a duck, it must be a duck."

This improves flexibility by allowing polymorphic substitution. With duck-typing, you don't need tests like `type()` or `isinstance()`; instead, you use `hasattr()` tests or EAFP programming.

31. EAFP Programming

EAFP stands for Easier to Ask for Forgiveness than Permission.

This means that Python assumes the existence of valid keys or attributes, and catches exceptions on falsity of the assumption. When we have too many `try` and `except` statements in our code, we can observe this nature of Python.

Other languages like C follow LBYL (Look Before You Leap).

32. Python Expression

An expression is a piece of code that we can evaluate to a value. It is an aggregation of expression elements like literals, names, attribute access, operators, or function calls. All of these return a value. An `if`-statement is not an expression, and neither is an assignment, because these do not return a value.

33. Extension Module

An extension module is one written in C or C++, using Python's C API to interact with the core, and with user code.

34. f-string

An f-string is a formatted string literal. To write these, we precede a string with the letter 'f' or 'F'. This lets us put in values into a string.

```
>>> name,surname='Ayushi','Sharma'  
>>> print(f"I am {name}, and I am a {surname}")
```

I am Ayushi, and I am a Sharma

For more on f-strings, read up on Python Strings.

35. File Object

A file object, in Python, is an object that exposes a file-oriented API to an underlying resource. Such an API has methods such as read() and write().

We also call them file-like objects or streams, and have three categories:

Raw binary files

Buffered binary files

Text files

The canonical way to create a file object is to use the open() function. For help with reading and writing files, refer to Reading and Writing Files in Python.

36. File-Like Object

Like we said earlier, it is a synonym for file objects.

36. Finder

The finder is an object that attempts to find the loader for a module that we are importing.

With Python 3.3 and above, we have two types of finders:

Meta path finders- to use with `sys.meta_path`

Path entry finders- to use with `sys.path_hooks`

37. Floor Division

Floor division is division that rounds the result down to the nearest integer. For this, we use the `//` operator.

```
>>> 18//4
```

```
4
```

```
>>> -18//4  #-4.5 is rounded down to -5
```

```
-5
```

These are some of the terminologies from our Python Glossary. We have Python

Glossary Part II as well for more Python Glossaries. Link is Provided at the end of this article.

38. Python Function

A function is a sequence of statements that may return a value to the caller. It may take zero or more arguments. For more on functions, read up Functions in Python.

39. Function Annotation

An annotation to a function is an arbitrary metadata value associated with a parameter or return value. We can access a function's annotations using the `__annotations__` attribute. And while Python itself does not assign a meaning to an annotation, third-party libraries or tools make use of them.

40. `__future__`

Interestingly, in Python, we have a pseudo-module available that lets us enable new language features that aren't yet compatible with the current interpreter.

```
>>> import __future__
```

```
>>> __future__.division
```

```
_Feature((2, 2, 0, 'alpha', 2), (3, 0, 0, 'alpha', 0), 8192)
```

```
>>> __future__.absolute_import
_Feature((2, 5, 0, 'alpha', 1), (3, 0, 0, 'alpha', 0), 16384)
```

41. Garbage Collection

Memory must be freed when it isn't needed anymore. Using reference counting and a cyclic garbage collector that can detect and break reference cycles, Python collects its garbage. We can use the `gc` module additionally.

42. Python Generator

A generator is a function that 'yields' values one by one. It returns a generator iterator. We can use this function with a for-loop to retrieve one value at a time.

For more on generators, refer to [Generators in Python](#).

43. Generator Iterator

A generator iterator is an object created by a generator function.

44. Generator Expression

It is an expression that returns an iterator. Below is an example of the same.

```
>>> sum(i**2 for i in range(7))
```

```
91
```

45. Generic Function

A generic function is made of multiple functions that implement the same operation for different types. The dispatch algorithm decides which implementation to use during a call.

46. GIL

GIL stands for Global Interpreter Lock. This is the mechanism that the CPython interpreter uses to assure that only one thread executes Python bytecode at a time. This makes the object model implicitly safe against concurrent access, and this simplifies CPython.

47. Hashable

If an object has a fixed hash value for its entire lifetime, and is comparable to other objects, it is hashable. Two equal hashable objects have the same hash values.

While a dictionary itself is unhashable, it cannot hold unhashable types like itself. In fact, all immutable types are hashable. Mutables like lists and dictionaries are not. User-defined objects are hashable.

48. IDLE

The IDLE is an Integrated DeveLopment Environment for Python. It is a basic editor and interpreter environment that ships with Python.

49. Immutable

Any object with a fixed value is an immutable. Examples include numbers, strings, and tuples. If you must change a value, you need to create a new object. In places where we need a constant hash value, like a key in a dictionary, we use immutables.

50. Import Path

A list of locations searched by the path-based finder to import modules. During an import, this list comes from `sys.path`. For subpackages, it may come from the parent package's `__path__` attribute.

51. Importing

Importing is the process through which we make the Python code in one module available to another.

52. Importer

The importer is an object that finds and loads a module. Hence, it is both- a finder and a loader object.

53. Interactive

Being of an interpreted nature, Python lets you enter statements/expressions at the integer prompt, and immediately execute them and see results.

54. Interpreted

We couldn't highlight this more when we say Python is an interpreted language. However, because it does have a bytecode compiler, the distinction is a bit blurry. The source files can run without explicitly creating an executable. While this makes Python faster to develop/debug, it often results in slower execution.

55. Interpreter Shutdown

When we shut down the interpreter, it gradually releases all allocated resources. These include modules and different critical internal structures. Alongside, it makes several calls to the garbage collector. This may trigger execution of code in user-defined destructors or in weakref callbacks. Since the resources it relies on may not function anymore during the shutdown phase, the code executed can encounter various exceptions.

56. Python Iterable

Any object that can return its members one at a time is an iterable. Examples include lists, strings, tuples, dicts, and file objects.

For more on iterables, read up on Python Iterables.

57. Python Iterator

An iterator is an object that represents a stream of data. We can define an iterator using the `iter()` function/method, and get one object at a time with the `next()` function/method.

For a detailed introduction to iterators, refer to Python Iterators.

58. Key Function

It is a callable that returns a value that we can use for sorting or ordering. We also call it a collation function. Functions like `max()`, `min()`, and `sorted()` make use of them.

59. Keyword Argument

Refer to section 6 for this.

This is all about the Python Glossary Part I. For more More Python Glossary see Python Glossary Part II.

CHAPTER 4

HOW TO GET UP AND RUNNING WITH PYTHON



HOW TO LEARN PROGRAMMING LANGUAGE

Programming is a very useful and rewarding hobby. There are few better feelings than when someone sees you using a program you lashed together to make your life easier and says that it looks really useful. Most people have, at some point in their lives, really wanted to be able to do something on their computer or phone and been unable to. If you know a programming language, then there is often a fair chance that you can write a program to accomplish that task yourself. While there are a huge number of programming languages, many of them have a lot of similarities; this means that once you learn one language

quite well, in most cases you will be able to pick up a new one far quicker.

One thing that all new programmers must come to term with is the amount of time learning a programming language takes. Although when you have become an expert you will be able to write many programs quickly, you must remember that many programs have taken whole teams of expert developers years to create. So it is important to understand that knowing a programming language or even several is not enough to write some of the more complex programs you have seen. Don't look upon this new hobby as a way to save yourself a lot of money, as writing your own version of most of the programs that you need to pay for now will be out of your reach.

The most important thing that a new programmer needs to know is that the "Learn Programming in 24 hours" sort of books are simply not true. A more accurate title would be "Learn Programming in 10,000 hours". If you put 24 hours or a week into learning a language you will not be creating the next Windows or a new, state of the art game. It is possible to learn to write a program in 10 minutes, and really all you need to learn a new language is your favourite search engine, but you will not be an expert. The only way to become an expert is much like learning the violin; the answer is practice, practice and practice some more.

Now that we have examined the limitations and handled some of the more unrealistic expectations, those of you still wanting to learn to code will be happy to know that programming is not a hard thing to start learning and will not require you to pay out huge sums of money. If you are reading this article online, you already have the resources to start with some languages, so let us consider what your first language ought to be.

Traditionally the first language a programming newcomer learns is either Visual Basic or Python. The first thing to understand is that these two languages are very different. The simplest difference is one of price. Python is totally free; you can start writing python now with just a text editor on your computer, though if you are on Windows, you will probably need to install it first. However Visual Basic, often abbreviated to VB, is both free and not free. On the upside, VB can be simpler for newcomers to learn because it allows you to build the interfaces (the part of the program the user will see) by dragging and dropping the different parts much like designing it in some basic art application. The version of VB newcomers learn is usually Visual Basic 6, but this is rather outdated and has been discontinued. So these days the version learned is often VB.NET which can be considerably less simple for newcomers.

VB.NET must be developed inside what we call an IDE (Integrated Development Environment); this is basically a special program you use to write other programs. They also exist for Python, but their use is totally optional. The free VB.NET IDE is called Visual Studio Express. At the time of writing, the latest version is Visual Studio Express 2010. Unfortunately, by using the free version of the IDE you are restricted with what you can do, and any programs you create cannot be commercially sold on. Regretfully, the full paid version of the IDE is not cheap, and probably not appropriate for a hobbyist, but fortunately to learn VB the free version is enough. In practice, very few commercial programs are developed in VB these days, but the Visual Studio IDE allows you to use many other languages. The familiarity you will develop by using it will also allow you to use the power of the IDE for development in many other languages. Some will argue that almost every language can be developed in a text editor and that they are by far the most flexible way in which to code. While this is technically true (and I do suggest trying development in a text editor to

compare once you get a little better), I would strongly advise learning your first language with a proper IDE.

While traditionally, people learn Python or VB first and these are generally what is taught at schools, I would not suggest either of these. I am of the opinion that your first language should continue to be useful to you one it has served the purpose of helping you learn the fundamentals of programming. If I had to recommend one of these for newcomers, it would be VB.NET as often the most complex part of programming is the graphical side of things and in VB.NET this is very simple due to the drag and drop interface. These two languages are often used as introductions as they are very tolerant of mistakes, and allow you to become confident in programming principles without worrying about a lot of the more complex matters.

For those brave souls among you, I would actually suggest Java as your first language, even though it can be complex, and is therefore not a common choice for a first language. Java programs are different to most others in that they do not run on your computer. The user downloads Java, then your code runs on what is called a VM (Virtual Machine). This means that your code runs in a special place Java sets up for it - a fake copy of your computer - and handles the translation of this to the real machine for you. This means that Java programs are "cross-platform", meaning that they will for the most part run on Windows, Mac, Linux and most other operating systems.

Java is a good language to learn, as it is very widespread and useful. Furthermore, it is very powerful, and is available for free for both hobbyists and commercial uses. However, in contrast to VB and Python, it does not tolerate mistakes and requires you to be very specific about everything. It is also an

object-oriented programming language, which is a very complex issue which I will briefly try to summarise. Languages like Python and VB are what is known as procedural languages, meaning that the lines of code are run one after another, whereas Java is an object-oriented language. object-oriented development is a term thrown around a lot these days in the programming world, and while not always appropriate it is generally considered a good idea. At the most basic level, an object-oriented program is all about objects. An object is an "instantiation" of a "class". A class is a blueprint used to describe something like a cat. The class contains both the data about the cat such as its name, age and owner as well as "methods" which are essentially actions the cat can perform, such as miaow. An instance of the class "cat" would give you a particular cat. However, this is not a Java tutorial, so if you are brave enough to experiment with Java you will come across this yourself in more detail. It is worth noting that VB.NET and Python both have support for object-oriented development, and Java has the potential to be used procedurally, but these are not the languages' primary intended uses and are not often used. If you did not understand that comparison, don't worry about it too much. Object orientation is hard to get your head around, but any basic Java or other object-oriented language tutorial will have you understanding everything in that paragraph.

A final reason Java is a good first language is that it is similar in many ways to Javascript, which is an entirely different class of language. Javascript is a scripting language (as is Python), and learning Java will mean you understand Javascript reasonably well. The difference between scripting languages and normal programming languages is outside the scope of this article, but as a large generalisation scripts are generally used for automated tasks while programs are used interactively by users. This is not totally true, as both types of language are used for both tasks and most web programs are built in Javascript.

As for the actual language you pick, it is entirely up to you. Some may choose the traditional beginner languages or be brave and experiment with Java. Some of you may already have your eye on a language or fancy one of the more specialist languages like Scheme or Prolog. Whatever your choice, the way you will learn how to program is the same.

Many of the purists say that IDEs are a bad idea, and are packed with unnecessary tools and menus that take up disk space and time to learn. While this is true, I feel that an IDE is definitely worthwhile. Many people offer free IDEs, such as Eclipse and Netbeans, for the more popular languages. There is also Visual Studio, which I mentioned previously; it is very intuitive, very powerful and it supports many languages (much as Netbeans and Eclipse do). If you chose to use Java I would suggest Netbeans, as there is a packaged version of Netbeans with the JDK (Java Development Kit). Most languages need an SDK (Software Development Kit) to work with them, and getting it installed properly and linked to the IDE is often the hardest part of the procedure. Visual Studio already comes with the development kits set up, which makes life easier, but other languages like Java and Python can be quite hard to set up properly. This is why I suggested the Netbeans + JDK bundle for those experimenting with Java, as it handles the complex set up for you, which will save you hours of suffering.

There are, in my opinion, three major advantages to using a fully featured IDE. Firstly, they are usually extensible, meaning that there are many free plug-ins that could make your life a lot easier when you get a little more advanced. Secondly, and most importantly, is the ease with which an IDE allows you to debug your code. Most IDEs let you set breakpoints in the code, which will make the program stop when it gets to that point and let you step through it line

by line, so you can examine the contents of all the variables at any time. (For those of you who do not know what a variable is, I will briefly explain. A variable is a bit like a train station locker. You ask for one big enough to hold what you want to store, and if what you want to store is the right shape, it can be stored there. When you write a program, any data you want to store temporarily will be held in one of these until you are done with it.) As the old programming saying goes, if you have not found any bugs, you are not looking hard enough. Almost no non-trivial program will work first time, and trying to work out where the problem lies without the use of a debugger is a pain I would not wish on anyone. Finally, an IDE will often give you advice on how to fix issues in the code. This can be very useful for fixing bugs, and saves you having to resort to Google every other minute.

Now that you have a language and an IDE, it is finally time to learn the language. This, as you may or may not be surprised to learn, is not complex at all - it is simply time consuming. To learn programming for the first time, there is no better way than exploration. Buying a book that walks you through steps will not teach you anything, as you will not understand the reasoning behind what they are doing, and people often get disheartened by the tedium.

The key to learning programming is to have a goal. Think of a task, such as a system to keep track of where you are in all the various TV shows you watch, or a system to let you look at all the books you own in a particular category, or, if you feel brave, try to replicate part of something that you use on a regular basis. My advice would be to start small, perhaps by making a sequence of message boxes that insults the user or a really simple calculator. It is important when you first start that your goals are interesting, challenging and entertaining. If you try to make really boring programs you will quickly get disheartened, so try to

inject some comedy into your program. The calculator is a very good introductory program, but after you get the general idea it is important to set quite ambitious goals, as if you keep doing simple things you will never learn anything new. It is important to try to incorporate some of the knowledge you have gained from previous work. One of the reasons most books fail to teach programming well is that they use small examples for each thing they introduce, whereas what you really need to do is plan the task without considering what you will need to accomplish it. This means you will be able to code some of it using what you already know, but most importantly, you will not know how to code some of it. The best way to learn is to learn by doing. Go for a full program that does a task you wanted to do on a computer in the past, work on it, and when you are finished you will have learned a lot and you will have a useful (or at least entertaining) program which is far better than some toy program demonstrating lists.

I have said that you learn by choosing to do projects where you are unable to do certain sections, thus requiring you to learn, but how do you go about finding out how to do them? It's simple, and most likely the way you found this article. Go to your favourite search engine (like Google) and search for what you want to do - for example, search "drop down list Java" to find some examples of using drop down lists in Java. Because you will need it for another task, and not just to re-do the same thing the examples did, you will have to play with the examples you find and try to get them to do what you want. Just search each bit you need, and before long you will find that most of the basics are as natural as waking up in the morning, and you did it all without spending a small fortune on books, without getting bored and hopefully while being entertained. To this day, if I am bored, I sometimes break out one of my very first programs which is just a list of boxes and a random number generator. It is your task to try to fill all the boxes such that the numbers the random number generator gives you are in ascending

order - if you don't leave space and can't fit a number in a hole then you lose and must start again. It's a simple program, but it took a lot of work when I first made it and I learned a lot from the experience.

Once you have a few decent sized programs under your belt, you will find that you know the language well. You will also find that it is rare, no matter how well you know a language, to be able to write a program without resorting to Google at least once just to check something. So with that in mind, it could be argued that you learned the language without ever actually trying to learn it. Clearly there are standards and good practices that you may not pick up on your own, but as you see more examples and read the comments you will find you adopt your own standards rather rapidly.

Once you have learned one language, whatever it may be, the most valuable thing you will have learned is all the key words for searches. When you want to do something in a new language, you need only search what you want to do and the language name. However, by now you will know the names used to refer to what you want to do, allowing your searches to be more effective and yield examples and answers much more quickly. As the fundamentals of programming are mostly the same, regardless of the language you use, you will hopefully be able to guess at the meaning of most of the code much more effectively once you locate an example, allowing you to pick up most of the language very quickly indeed.

If you take nothing else away from this article, remember that the best way to learn a skill is practice, practice and practice some more, so don't expect to become an expert overnight. Remember that programming is not something that can be learned overnight, and that to become a passable expert you probably

need to spend at least 10,000 hours programming, so you will need to find ways to remain motivated. Don't think of it as learning to program - rather, just start programming, and before you know it you will be an expert. Programming is a skill, and while it is quite simple once you have the feel of it, it can be quite daunting to see your little calculator that took you a week and then to consider a modern game like "Batman: Arkham City" and realise how far you have to go.

Programming is easy when you know how, but is not a trivial thing to learn, so it is important that you set yourself tasks. These tasks should preferably be interesting and, better yet, entertaining, as these will be what keeps you programming and learning more and more until, one day, you wake up and realise that you know quite a lot. You are your own best tutor and the key is simply to jump in and get started.

HOW TO SET UP PYTHON

- How to download and install python
- How to run python from the command line
- How to install python libraries using pip, a popular package manager for python (we'll explain what that means!)

DOWNLOADING PYTHON

Visit the official Python downloads page and find the most recent release of Python 2.7 that corresponds to your OS (operating system).

Make sure to download Python 2.x and not Python 3.x. Although Python 3 has a higher version number, it isn't completely compatible with Python 2, the version you learned in the Codecademy course. Both versions are widely used and actively updated.

After your download has completed, launch the installer. This may require that you unzip the file first depending on the version of the installer you downloaded. Follow the instructions provided by the installer to complete your Python installation.

WHAT DID I JUST INSTALL?

Computers read and execute code based on machine language which is stored in

hexadecimal format. It is virtually impossible for a human to write in machine language and each processor has its own version.

To make programming simpler, human-readable languages like Python were invented. The files you just installed include a Python interpreter. This interpreter converts your human-readable Python code into instructions that the computer can act on.

RUNNING PYTHON:

Once you have downloaded Python, you should be able to pull up your computer's terminal and start running it. On Windows, search for a program called "cmd" and then launch it. If you're on a Mac or a Linux environment look for and launch the program "terminal."

If you downloaded Python properly, you can just type python into the prompt and hit enter to get a result.

From here we can type in Python code and watch it be interpreted for us on the fly. Go ahead and type some simple stuff in and watch Python work right in front of you.

The command line is useful for checking simple code, but if you're looking to build something more involved, you'll benefit by starting with a text editor. Let's exit out. We'll make our own files and return to the command line to run those.

If you do not have a good text editor for editing code, we recommend Sublime Text, for which we have a step-by-step setup guide [here](#)(link). Any text editor will work but may require some additional setup.

Once you're set-up, write a simple piece of Python code like the below in your text editor. Save it, making sure the file name ends with a `.py` extension.

Now let's run your code and see what we get. Pull up your terminal again and locate the directory in which you saved your python file. Use the `cd` command to get there. (If you're rusty on how to locate and change directories, refer to our [Learn the Command Line](#) course). Then type `python first.py`.

The output of your script should print in your terminal window. If you're seeing errors or nothing at all, make sure you're in the same directory as your code, and that your code has a print statement that gets executed. (In other words, that you're telling your code to actually print something.)

If it printed, congratulations! You've now written and run a Python file all on your own! Now you're equipped to create those projects that you've dreamed of.

Making big projects takes lots of work. Lots and lots of work, so much so that it's useful to have a little help from others. Thanks to the very open culture of programming, there are many open source libraries out there for you to use. Using libraries help us work quickly, and allow us not to reinvent the wheel every time we sit down to code.

CHAPTER 5

ADVANTAGES OF DOING WEB DEVELOPMENT WITH PYTHON



Python is a favorite among many developers for its strong emphasis on readability and efficiency, especially when compared to other languages like Java, PHP, or C++.

Sure, it's old, but it's 1980s old, not Cobol or Fortran old. Besides, if something works, why change it, especially when there are a so many ways to improve it.

Actually, depending on how you view it, longevity is a good thing in itself—a sign of stability and reliability.

If you're like many people who first started out with Java, C, or Perl, the learning curve for Python is practically nonexistent. But the fact that it's easy to learn is also the reason why some people don't see Python as a necessary programming skill.

Why Doing Web Development with Python has many advantages:

PYTHON IS EASY.

Try asking programmers what programming language was easiest to learn for them, and I bet the majority of responses will probably be Python.

It's true: Python looks like it was designed for newbies. It reads like kindergarten math and is so easy to understand that you could teach its basic concepts to someone who doesn't know a lick of coding in one day.

Python's reliance on whitespace and common expressions trims out a lot of programming fat, allowing you to do more with fewer lines of code next to say, Java or C++.

Development is not an easy task. Why not make the job easier by using a language that is as simple as it gets?

Pro tip: If you're at a point where you think you're good with Python, dig

deeper. Pick up as many libraries as you can, and perhaps learn Django to make yourself more marketable.

PYTHON LETS YOU BUILD MORE FUNCTIONS WITH FEWER LINES OF CODE.

Python is a quick study for anyone. With practice, you could easily build a rudimentary game in two days tops (and that's coming from knowing absolutely nothing about programming).

Another factor that makes Python an attractive programming language for novices is its readability and efficiency.

Perhaps the best example to demonstrate this is the "Hello world!" program. That simplicity is key. Simplicity is what lets you do more with Python quickly and with fewer lines of code.

PYTHON PROVIDES A STEPPING STONE TO LEARNING OTHER CODE.

Starting with Python can serve as a stepping-stone for developers new to the world of programming. Python's object-oriented principles are compatible with other languages like Perl, JavaScript, Ruby, and C#.

Therefore, once you've mastered the concepts of Python, other languages grounded on similar principles should come naturally to you, allowing you to

focus on their syntax, which is really what matters.

IT'S HARD TO MESS UP WITH PYTHON.

The beauty of Python—besides its simplicity—lies in the highly established rules the language is built on.

These tenets include:

- * Readability is important
- * Less is more
- * Complex is fine, but not complicated
- * Clarity is better than implied

PYTHON IS PERFECT FOR BUILDING PROTOTYPES.

The fact that Python lets you do more with less code also lets you build prototypes and ideas quickly.

Ideation is an often-overlooked aspect of web development, and the ability to come up with functioning prototypes at a faster rate can help reduce time, save money, and satisfy clients.

PYTHON AND DJANGO ARE HUGE IN FINTECH.

Never heard of fintech?

Better start reading about it, because you might be working in the industry soon.

The fintech revolution has led to an explosion of new companies that combine Silicon Valley's innovations with the money machinery of Wall Street.

Basically, it's a new sector that combines money with technology.

But why should you care? You're a programmer, not a banker or financial analyst.

As it turns out, the fintech industry is one of the major contributors to the increase in demand for Python programmers.

Python is the fastest growing language in fintech, which might explain why Python now beats C++, PHP, and iOS in a survey on the most popular languages.

PYTHON IS FLEXIBLE.

There are several robust Python implementations integrated with other programming languages.

- * CPython, a version with C
- * Jython, or Python integrated with Java
- * IronPython, which is designed for compatibility with .NET and C#
- * PyObjc, or Python written with ObjectiveC toolkits
- * RubyPython, or Python combined with Ruby.

IT HAS A TON OF RESOURCES.

As a Python developer, it's practically impossible to get stuck in a rut with the huge number of resources that constantly keep getting refreshed.

It also boasts an extensive library with built-in functionality, which explains why so many programs are written in Python.

Python also has a built-in unittest framework to ensure your code works as intended.

THERE'S A ROBUST STACK OF FRAMEWORKS WAITING FOR YOU.

Python owes a great deal of its flexibility to the many programming environments and frameworks that make the development of specific applications quick and easy.

For example, web developers can turn to frameworks like Django or Flask,

which let you focus on writing the app or site rather than get bogged down by tedious legwork.

What this means is that frameworks and environments allow Web developers to be more productive and efficient on Python than with other languages. This is a critical factor when you need to bring applications to final deployment right away.

Django is the most popular web framework for Python. Flask and Pyramid are two other popular frameworks.

Other Python web frameworks include Zope2, Grok, web2py, and TurboGears.

DJANGO, A HIGH-LEVEL PYTHON WEB FRAMEWORK, IS FLAT-OUT AMAZING.

The ability to use Django is perhaps one of the biggest advantages of learning Python.

The Django framework lets you model your domain and code classes, and just like that, you already have an ORM.

Now you can focus your efforts on your user interface.

Django's ease of building templates, or using the already built-in template

language, makes it easy to build applications that are ready for deployment.

It's no surprise why Django is the foundation of sites and services like The New York Times, The Guardian, Pinterest, and Instagram.

As an open-source framework, Django is supported by an active community of users who continually contribute to updating the resources on DjangoProject.com.

PYTHON IS GREAT IF YOU'RE ON A BUDGET.

If you or your company wants to build a product, your choice of language may ultimately boil down to preference and expertise.

But if you're running on a budget and need a product rolled out right away, the choice of language then becomes more important. The bigger the project, the more important this choice becomes.

Python is an ideal option for bootstrappers and startups because of its quick deployment and—as mentioned earlier—lesser amount of required code next to Java, C, and PHP among others.

PYTHON IS A HOT COMMODITY IN THE ERA OF INTERNET OF THINGS (IOT).

The advent of the Internet of Things introduces countless opportunities for Python programmers.

Platforms like Raspberry Pi, a series of credit card-sized computers running Python, allow developers to build their own exciting devices like cameras, radios, phones, and even games through Python with ease.

With advanced Python programming concepts, developers can homebrew their own gadgets, and connect them with real world markets independently and on the cheap.

PYTHON IS A CORE TECHNOLOGY IN BLUE CHIP SITES AND SERVICES.

Given Python's affinity for scale, it shouldn't be surprising why it's the core language in many 'blue chip' sites and services. The list includes Dropbox, YouTube, Instagram, PayPal, eBay, Yelp, Reddit, Disqus, and games like EVE Online and Second Life among others.

For web developers, this means that mastering Python and its popular advanced frameworks like Django should ensure you're able to find work or even build your own product or service as a startup.

TECH GIANTS LOVE PYTHON.

Several IT giants, as well as the IT infrastructure of major organizations, rely

heavily on Python. These groups include NASA, JP Morgan, Google, Yahoo!, Disney, Nokia, and Mozilla among many others.

And as long as these companies and organizations exist, there will always be a demand for Python web developers.

But you might be wondering, “How much are they paying for it?”

THERE’S GOOD MONEY IN IT.

If creating cool, never-before-heard gadgets with Python and Raspberry Pi isn’t awesome enough of an incentive, then how does cold, hard cash sound?

A 2015 study by Computer Science Zone using data from WANTED Analytics shows that, after taking into vacancies and projections, the most popular and highest paying languages are SQL, Java, JavaScript, C# and Python.

PYTHON IS PERFECT FOR GETTING INVOLVED IN HIGHER EDUCATION.

If you’ve always wanted to moonlight as a teacher, being proficient with Python should give you a spot at the cool teachers’ table.

PYTHON MAKES SYSADMIN DUTIES A BREEZE.

By now, you already know Python can be used to develop practically any kind of software or web application. But one task that really brings out the best in Python is writing software for managing sysadmin tasks.

Extensions like Salt and Ansible carry dedicated system administration features to Python, complementing its scripting principles to create sysadmin tasks that control features and utilities within operating systems.

It's also for this reason that Python shines as an instructional language, as it allows students to create system utilities and learn the basic principles of system administration, all while learning the language.

PYTHON IS OPEN-SOURCE.

Python's community of programmers is one of the best in the world.

As an open-source platform built by thousands of contributors from all corners of the world, Python is a crowdsourcing success story. As mentioned earlier, it's robust, scalable, well designed, and easy to learn, a product of several years of the best minds coming together to build a language with clearly-defined features.

The fact that it has an open-source license also means Python can be implemented and modified in any way you please. It's possible to insert blocks of other code to make the language even more feature-rich, all at no cost.

You can also tap libraries to create applications for data analysis, language processing, and machine learning among many others.

Support is also free, so if you have any questions, getting answers is as easy as going to Python.org and asking or using the tutorials.

DJANGO SUPPORTS BEST PRACTICES FOR SEO.

SEOs and web developers aren't always known to play nice together. The task of a developer and optimization of an SEO sometimes seem to be at cross purposes.

Thankfully, when said developers are using Django, this is less of an issue.

For one, Python's Django framework supports the use of human-readable website URLs, which isn't only helpful from the actual user's perspective, but also to search engines, which use the keywords in the URL when ranking sites.

Your SEO team will thank you for using Django. Besides, it just makes more sense to ensure URLs mean something instead of being just a series of random numbers and letters.

When it comes to Django SEO, this easy tutorial will get you started on the right path.

DJANGO IS SECURE.

By default, Django prevents a number of common security mistakes better than say, PHP does.

For starters, Django 'hides' your site's source code (except CSS and html files) from direct viewing on the Internet by dynamically generating webpages and sending information to web browsers through templates.

CHAPTER 6

FULL INSTRUCTIONS OF HOW TO CODE



There are many programming languages available and each of them is suitable for another program or application. There are people who have learnt only a few programming languages and who use these because that is what they know, but most of the times software programmers will use the programming language that is required by the application they are creating. Java is one of the most frequently used programming language and writing in this language is somehow different from the usual Pascal or any C/C++ version but that does not mean that learning the java code is harder than learning Pascal or C++. Nowadays there are numerous applications written in Java and its terminology it may seem a bit harder in the beginning but anyone can write in this programming language,

that's for sure.

When looking into a new programming language, most people would like to know if it is easy to learn and work in. If you compare it to C or C++, you may discover that indeed, using it can be more straight forward. This is due to the fact that Java has far fewer surprises compared to C versions. C and C++ make use of a lot of peculiarities so learning and mastering them all can be a daunting task (for example, temporary variables hang around long after the function that created them has terminated). Being more straight forward, Java is a bit easier to learn and to work with. Java eliminates explicit pointer dereferences and memory allocation/reclamation, for example, two of the most complicated sources of bugs for C and C++ programmers. Out of range subscripts are easy to find, as Java is able to do add array bounds checking. Others may argue that it seems easier to work with because there are very few examples of extremely complicated projects done using it, but the general accepted idea is that it is somehow easier to master than C or C++.

Learning Java programming is not very difficult, especially if you are familiar with other, more basic, programming languages and you know for sure what you want to create using it and it has a series of benefits compared to C and C++. First of all, code written in this programming language is portable. Code written in C and C++ is not and this makes Java more practical (for example, in C and C++, each implementation decides the precision and storage requirements for basic data types).

When you want to move from one system to another, this is a source of problems because changes in numeric precision can affect calculations). On the other hand, Java defines the size of basic types for all implementations (for example,

an "int" on one system is the same size and it represents the same range of values as on every other given system).

The cases of programs that make use of floating point arithmetic requires a special attention: a program that uses floating point calculations can produce different answers on different systems (in this case, the degree of difference increases with the number of calculations a particular value goes through). But this is a thing specific to all floating point code, not only Java code which is also more portable than C or C++ in its object code. It compiles to an object code for a theoretical machine - in other words, the interpreter emulates that machine. This translates to the fact that code compiled on one computer will run on other computer machines that has a Java interpreter, but more on this subject you will find out while learning Java programming.

TIPS TO HELP PHP PROGRAMMERS IMPROVE THEIR PROGRAM CODE

Ever since it made humble beginnings around two decades ago, PHP programming has witnessed a rapid climb, going on to become the most popular programming language for all Web applications. Numerous websites are powered by PHP programming code while an overwhelming majority of Web projects and scripts rely on this popular language for their development.

Owing to the huge popularity of this development language, it is seemingly impossible any PHP programmer to say that he lacks even the basic working knowledge of PHP. In this tutorial we aim to help those at the learning stage of this programming language, willing to roll up their sleeves and let their hands get dirty with this language. These simple programming tips will aid in speeding up the proficiency of the development process while making the code cleaner, responsive and offering optimum levels of performance.

Implement a cheat sheet for SQL Injection: SQL Injection can prove to be a really nasty thing. This is a security exploit wherein a hacker can dive into your database by making the most of a vulnerability in the programming code. MySQL database is commonly implemented in PHP programming code so being aware of the things that you should avoid can prove to be really handy if you are planning to write a secure code.

Be able to differentiate between your comparison operators: An integral part of PHP programming is comparison operators. However, some programmers are

not as adept at sorting out the difference between them as they ought to be. In fact, many documentations lay claim to the fact that most programmers are incapable of telling the difference between the comparison operators at first glance! Failure to identify the difference between "==" and "===" is a big blot on the reputation of any programmer.

Favour the use of `str_replace()` over `preg_replace()` and `ereg_replace()`: As far as the overall efficiency is concerned, `str_replace()` proves to be much more efficient as opposed to some of the regular string replace expressions. In fact, statistics show that `str_replace()` improves the efficiency of code by as much as 61% compared to some of the regular expressions such as `preg_replace()` and `ereg_replace()`. However, if your program code comprises of nothing but regular expressions, then `preg_replace()` and `ereg_replace()` will prove to be much faster and efficient than the `str_replace()` function.

The use of ternary operators: Rather than use an if/else statement altogether in your PHP programming code, consider the use of a ternary operator. They are effective in terms of freeing up a lot of line space for better readability of your programs. The code looks less clustered and it is easier to scan it in the future for any unexpected errors. However, be careful that you do not have more than one ternary operator in a single statement. Very often, PHP gets confused regarding its course of action in such situations.

Use a PHP Framework: Use of a PHP Framework is seemingly impossible for every project that you create. However, the use of such frameworks as CodeIgniter, Symfony, Zend and CakePHP helps in greatly reducing the time required for developing a website. A Web framework is typically a software that bundles itself along with functionalities that are commonly needed, aiding to

speed up the development process. By letting a framework handle all of the repetitive tasks required for website programming, you can work on the development process at a much faster rate. Less coding you have to take care of, lesser will be the testing and debugging requirements thus letting you speed up with the overall process.

Make correct use of the suppression operator: The @ symbol is typically the error suppression operator. Place it in front of a PHP expression and it will simply instruct all errors in that particular statement to show themselves. This is a very handy variable if you have doubts over a value but would not want them to show up when the script is run. Many PHP programmers, though, very often use this error suppression operator in an incorrect manner. The @ operator is unusually slow and can be rather costly if you are planning to write a code keeping its resultant performance in mind. While alternate methods exist to sidestep the errors brought about by the use of the @ operator they may result in accidental side-effects and should only be used in areas from where they cannot further affect other areas of the code.

Rather than use strlen, opt for isset: If you are planning to implement a function that checks the length of a string, opt for the use of isset rather than strlen. Isset allows your function call to work as much as five times faster than the other functions. Also note that by the use of isset you can have a valid function call even if there is a non-existent variable.

While this is a small change to make, it nevertheless adds up to a leaner and quicker code, akin to all of the aforementioned tips that we have presented in this article.

PHP Programmers is a business solution service provider company specializing in website development, PHP framework programming and CMS development. Contact us to hire our team of expert and dedicated professionals who will serve your business as per your requirements.

IMPORTANCE OF CODING STANDARDS

All too frequently a programmer is left to produce code, without standards or quality assurance. It is only after a major problem that the months and in some cases, the years of programming are found to be worthless. Some control on the code being produced is essential - and this starts with a well documented Coding Standard.

No matter how busy you are, a Coding Standard will be a time saver in the long run - it is fundamental to good programming. Coding Standards and programming productivity go hand-in-hand. Adhering to a Coding Standard will lead to a significant increase in development productivity, whatever the programming language. A consistently applied set of Coding conventions will standardise the structure and coding style of an application, so a programmer can readily read and understand other programmers' code, as well as their own code.

Good Coding Standards result in precise, readable, and unambiguous source code that is consistent and as intuitive as is possible. The object of a Coding Standard is to make the program easy to read and understand without cramping the programmer's natural creativity with excessive constraints and arbitrary restrictions.

MAINTENANCE COSTS

Following a Coding Standard will increase productivity, not only to system development, but increase productivity in enhancing and maintaining systems as well. And it is also important to control maintenance costs- these can constitute most of the lifetime cost of software.

To reduce the high maintenance cost, it is essential to have code written that can be easily deciphered and enhanced by any other programmer in your Company. This is important, as usually very little software is maintained by the original programmer.

ENFORCING THE STANDARD

All the programmers should agree on the Coding Standard. Once agreement has been reached, the Standard must be enforced, as part of quality assurance. The use of the agreed coding techniques and good programming practices play an important role in software quality and performance. By consistently applying a well-defined Coding Standard, a team of programmers working on a software project is more likely to yield quality software system code that is intelligible, error free, cost effective and maintainable.

Here then are some ideas to help create your Coding Standard. Make it short and to the point:

READABLE CODE

The code must be created so that it is easy for other programmers to understand. It should not be written only with just the computer in mind. The code should be easily readable, with lots of white space.

BLOCKS OF CODE

Blocks of Code should be broken into sections that are well defined and understandable procedure chunks. It may be helpful to use a "Main Line" to split a lengthy procedure.

CODE INDENTATION

Those "If" "If" "Else" "Else" constructs are hard to follow, if the indentation of control structures is inconsistent.

VARIABLE NAMES

The variable naming convention should be well thought out, meaningful and consistently applied. The variable name should describe the content of the variable. Try not to have abbreviations, unless they are consistently applied. The standards for capitals, Camel or Pascal case need to be set.

PROCEDURE NAMES

Much as for variable naming conventions, procedure names should describe their purpose.

CODE DOCUMENTATION

No matter how well the code is written, changes or enhancements will eventually be required. Time should be spent when actually writing the code, commenting on the intent and technical aspects of the code. This will save time and effort later on.

In addition to external documentation, all code should be liberally documented with comments. Each procedure should have a heading with comments

describing the input, output and function of the procedure.

COMPLEX CONSTRUCTS

There is always the temptation to use the latest and greatest feature - this is true especially with inexperienced programmers. Complexity and the esoteric will make a program code difficult to follow, maintain and debug. Simplicity, above all, must be enforced.

KEEP THE CODING SIMPLE

If another programmer cannot understand, at a glance, what the Code is all about, then the Code is badly written. So often it becomes necessary when debugging or enhancing code, that the code be rewritten. This is a costly exercise for the company as well as being a non-productive and wasted effort.

ISOLATE COMPLEX CODE

There will be the occasion when simplicity must give way to complexity, when functionality or performance is required. In these cases (hopefully, few and far between), the complex code should be isolated in a "black box" library procedure.

REUSABLE CODE

The Coding Standard should include the gathering and documentation of

reusable Code. With reusable Code, that is Code that is not duplicated throughout an entire organisation, the programmers will build programs faster - and programs that run faster. Using less Code means greater productivity and faster development cycles. By using the same Code repeatedly by every programmer in every project, errors are identified and eliminated sooner.

HOW TO PROPERLY DEBUG YOUR CODE

As a lot of criteria would differ a newbie programmer from an experienced and skillful one, one of the most important criteria is the ability to debug their own code.

A beginner programmer would write his code snippet, leave parts "for later", without spotting the fact that they left a bunch of bugs, compile and run the code, and then see that his code doesn't work, for some weird reason. Well, starting from the compilation part to noticing that everything is screwed up is a part of programming itself, and bugs on the first compilation doesn't make you a bad programmer.

But the difference between bad/inexperienced programmers and skillful/experienced programmers starts here. Most of the "programmers" out there would think about their problem for (maximum) 5-10 minutes, then go to StackOverflow/FXP/Their senior programmer friend, and post something like that:

Hello.

My code doesn't work. Here it is:

<some code>

Please fix it to me.

If you can't spot at least 3 wrong things about this post, well, you should start thinking about a new career. This post shows a typical "I have never really tried to solve this problem, solve it for me" way of thinking. You can immediately see that:

"My code doesn't work" - Well, OK there. If it worked, you wouldn't have posted it here, right? What really doesn't work? Do you expect your friends to understand your code by themselves?

"Please fix it to me" - This shows zero will to really solve the problem. You have to understand that as long as the problem is yours, you are the one who should solve it.

And of course, I must mention that asking other people/your smarter/more experienced friend should be the LAST thing you do. First, you try and solve the problem by yourself.

So we saw what a newcomer would do. Now, how would a more experienced developer handle this annoying but frequent situation?

First, before we can solve the bug, we have to spot the bug. If an exception is raised, well, you can easily tell the location of the bug. Otherwise (referring to logical bugs):

If you work with a good IDE, like Visual Studio, Eclipse, PyCharm, etc., you can run the program step by step, check your variables' content, and spot the line/function that fails to run properly. If not, you can print checkpoints to see where the code reaches before it breaks, and you can print your variables' values to see what goes wrong. Of course, the first approach is preferred as it's easier and more convenient way of working.

After spotting where the bug occurs, you should start looking for why it does. Usually, you'll find a problematic value of one of the variables on the line of the bug (can be a never raised flag, a never incrementing counter, a null or falsy value, etc.). If the problem is not variable related, it might be an uninitialized object, unopened connection, or other things you might forget to do - import a library, reset a connection, etc.

Now, what's left is to trace the problematic variable's value, see where it's assigned wrongly, and find a way to fix it (or at least find the cause to the problematic assignment). If you don't know why a specific function does not function as you'd expect it to, what a specific class does, or what a weird error means, always remember that GOOGLE is your best friend. Googling your problem does not make you a bad programmer. It actually makes you a better one - a programmer that can find his needed information by himself.

Now, you should remember. Debugging is not a science, it's an art. It doesn't

have a clear step-by-step solution. Always start debugging "open minded" - bugs occur because of strange reasons, in unexpected places.

Also, remember that there is nothing wrong in asking for help - never feel ashamed to ask your friends/a programming forum/StackOverflow for answers. But also remember that other people will be more open to help if you show them that you at least tried to deal with your problems. Don't let them feel that they do your work.

And finally, a good programmer that uses the methods described here, Google and friends in a wise and balanced way would find and fix his problems a thousand times faster than a programmer that every bug he sees equals to another phone call his experienced friend receives.

So in one sentence: Do your best to investigate it using the tools you have, and ask others only if really needed.

REUSABLE CODE

Programmers, when given a task, will generally just sit down and start writing the code. Whether they are using an Object Oriented programming language or not, there is seldom a global view of all the previous work done by other programmers in their organization. All too frequently the same code is repeated in numerous programs and projects. Duplicated code can be created by the same programmer over time, by the programmers in a team, by different teams, and by all the past programmers that ever worked in the company. So reinventing the wheel starts, and continues, and...

Microsoft sets a good example that is seldom followed. They provide commonly used and invaluable routines for programmers. They use these routines primarily for themselves and throughout the entire range of products, in creating and standardizing their Windows Operating system. Microsoft has to be efficient if they want to survive. They try to eliminate all duplicate code in their software. But programmers in a company seldom attempt to create common code. In most companies, once a routine has been written, it would take a brave programmer to attempt to rationalize code that has been working for years.

It is self evident that for any company to be efficient, it must automate. Yet the cost of computerization can be high - highly trained technicians are expensive. And all too often the software produced is inflexible and costly to change.

With reusable code, that is code that is not duplicated throughout an entire organization, the programmer can build programs faster - and programs that run faster. Using less code means greater productivity and faster development cycles. Project handovers are easier for the programmer, with the familiar coding routines. By using the same code repeatedly by every programmer in every project, errors are quickly identified and eliminated. Reusable code also benefits the organization through greater end-user productivity. Users are more comfortable and need less training with standardized software programs.

As the Information Technology Manager of a large company (in another life!), I had a team devoted to finding duplicated code in all the programs ever written. This involved searching through all the Source code used by the company. The duplicate code was identified and converted into centralized and reusable code

for all to share. Errors were identified and eliminated - only the best variation of the code was used. Optional parameters allowed for the many variations of the functionality.

Then the redundant code was removed, and replaced by single line functions. This reduced the size and complexity whilst improving maintainability of the programs. Using only the best centralized and reusable code, controlled and documented by senior programmers, was an enormous productivity boost. Change management was simple - only one modification was need, to be immediately available to all programs. The programmers soon saw the advantage of the reusable code, and they started supplying suggestions on additional functions that could be rationalized.

The code that was commonly duplicated was:

- Database and Files - Opening and closing, reading and updating records
- Validation - Eliminating spurious characters like carriage return and line feeds.
- Formatting - Justifying text left, right and centre.
- Error handling routines - user level, recoverable errors and fatal crashes.
- Grid handling - Listing and updating database records.

Converting the error routines into reusable code took on a life of its own. By having one centralized error routine handling all problems, statistics were produced to show the error count for each project and for each project team. This was a valuable management tool, partially to judge the quality of the

programming, but mainly to determine which project needed additional resources or further analysis. When important batch jobs were run overnight, a phone call system was initiated to advise the project leader of any problem. This, perhaps more than any other measure, improved the software quality dramatically. Programmers - partners were not amused at being woken up in the early hours of the morning!

Maximizing code reuse should be a fundamental goal of the professional programmer. Achieving this aim involves convincing senior management of the benefits and cost savings of using reusable code, and that resources should be allocated to this end.

CODE IGNITER WEB DEVELOPMENT

Here is yet another treat for all those developers who use PHP as their main platform to design and develop engaging and dynamic websites. Yes, we are talking about Code Igniter web development. We know that there are a whole lot of significant open source frameworks in Codeigniter Web Development are put to use for getting rapid web application development. Code Igniter web development framework remains the most popular one in the category. This too comes in the MVC pattern which facilitates the web developers in transforming the illustrative as well as innovative concepts that the users come up with into real applications. For people who shun away any kind of complexity and would want effective and simple solutions, it is then Code Igniter for them definitely.

The main benefit that one can get with the help of Code Igniter is competent solutions in web development that render effective service to the users. It is

beneficial not only for the users but for the developers too. As there are a rich set of libraries, the developers can accomplish a wide range of web applications and development services faster. Providing an easy interface coupled with logical structure to gain access to the libraries is the main feature that Code Igniter bestows its users.

With the help of Code Igniter one can get websites that are fully functional and they come about in absolutely simple structure. Managing the web applications also becomes absolutely easy without much ado about anything and the websites offer steadfast performance too. This is one highly secure development platform that the developers can make use for a wide range of applications. The development process is absolutely transparent as there is thorough and clear documentation. One can get a whole lot of flexible applications with the help of customized features that are a part of Code Igniter framework.

This is search engine friendly and hence when it comes in with clean URLs there is definitely a prominent online presence for the website. It is strongly believed that Code Igniter offers a very fast and reliable platform for development. A simple interface yet robust range of libraries enables faster development of the web projects in comparison to the writing of programming codes is what makes this framework a special one.

The Code Igniter framework is the most preferred framework by huge business conglomerates as it is an effective and light weight platform to create applications which have to deal with huge amounts of data as well as security purposes for the applications that come out of this platform are considered to be highly secure and absolutely dependable.

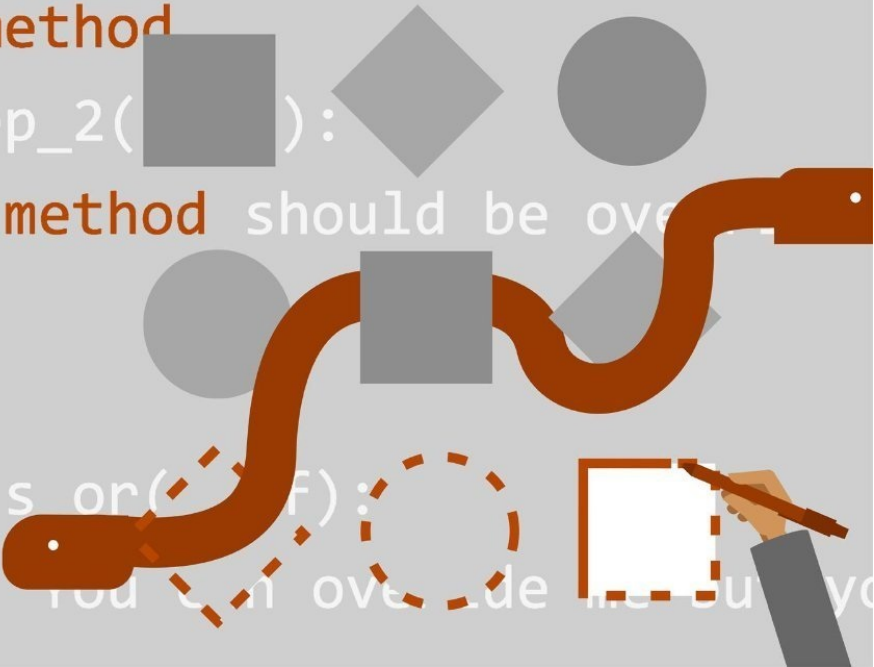
Yet another main reason why big enterprises opt for this framework is that they do not have to spend a whole lot of time in coding and structuring the coding. TechiesTown InfoTech claims to offer customized Code Igniter framework services across expansive industry verticals throughout the globe. Development, deployment, complete implementation, maintenance and support are the services that the company claims to offer its clients. If you are looking out for a simple yet reliable web development solution, then perhaps Code Igniter is the one for you.

CHAPTER 7

BASICS OF THE OBJECT ORIENTED PROGRAMMING

```
@abstractmethod
def do_step_2( ):
    #This method should be over
    pass

def do_this_or(F):
    print( "you can over ride me but yo
```



The programming style that we usually call object-oriented programming (OOP) has appeared relatively recently in the history of programming languages. This is a particular and very convenient style for many situations. It was designed to overcome the limits of structured programming based mainly on the widespread use of procedures, functions, pointers or other more or less developed data types.

Although structured programming is very practical for small software systems or reduced non-graphic applications, it should be avoided when dealing with large applications that use graphic elements where object-oriented programming is

recommended. Object orientation means organizing software resources as a collection of distinct and discrete objects, which includes both data structures and their processing functions. This organization is an extension of structured programming in which the data structures and processing functions are only loosely connected. All items have their own identity and are perfectly distinct.

Web programming is an aspect of web site development and the role of web programmer is very significant just as web designer's role in web design aspect of web site development. Programming languages have developed from machine language to low-level language and then to high-level language. The high-level language which is a language close to natural language (the language we speak) is written using certain approaches. Notable are the monolithic and structural programming approaches. With the monolithic style, you write a whole program in one single block. In structured programming approach, a program is divided into blocks of codes called modules with each module performing a specific task. BASIC, COBOL, PASCAL, C, and DBASE that ran on MS-DOS platform could be written using both approaches.

Following the revolution of windows operating system, it became possible to write programs using a more advanced structured programming approach than the type used on MS-DOS platform. This is the Object-Oriented Programming (OOP) approach where a program is divided into classes and each class is subdivided into functions or methods with each function providing a specific service. C++ and Java are typical examples of Object-Oriented Programming (OOP) languages which were originally developed for non-web solutions. As the preference for web applications grew more and more according to the historical development of the internet and the historical development of web, the need to improve on scripting languages continued to arise and one of the ways they

embarked on it was by making scripts Object-Oriented. Java applet and PHP (Hypertext Preprocessor) are examples of Object-Oriented Programming (OOP) languages for web solutions. PHP was originally non Object-Oriented but it has been fully upgraded to an Object-Oriented Programming language (OOP) demonstrating the 3 pillars of Object-Oriented Programming (OOP) - Encapsulation, Inheritance, and Polymorphism. Thus, it is possible to write server-side scripts in an Object-Oriented fashion.

The major motivating factor in the invention of object-oriented approach is to salvage some of the flaws encountered in the procedural approach.

Developments in software technology continues to be dynamic. New tools and techniques are announced in quick succession. This has forced the software industry and software engineers to continuously look for new approaches to software design and development, which is becoming more and more critical in view of the increasing complexity of software systems as well as the highly competitive nature of the industry. Software engineers have been trying various tools, methods, and procedures to control the process of software development in order to build high-quality software with improved productivity.

The object-oriented paradigm draws heavily on the general systems theory as a conceptual background. A system can be viewed as a collection of entities that interact together to accomplish certain objectives. Entities may represent physical objects such as equipment and people, and abstract concepts such as data files and functions. In object-oriented analysis, the entities are called the objects. As the name indicates, the object-oriented paradigm places greater emphasis on the objects that encapsulate data and procedures. They play the central role in all the stages of the software development and therefore there exists a high degree of overlap and iteration between the stages. The entire development process becomes evolutionary in nature. Any graphical

representation of the object-oriented version of the software development life cycle must therefore take into account these two aspects of overlap and iteration. The result is a "fountain model" in place of the classic "water fall model".

Object-oriented analysis (OOA) refers to the methods of specifying requirements of the software in terms of real-world objects, their behaviour, and their interactions. Object-oriented design (OOD) on the other hand, turns the software requirements into specifications for objects and derives class hierarchies from which the objects can be created. Finally, Object-oriented programming (OOP) refers to the implementation of the programs using objects, in an object-oriented programming language, such as C++.

OOA provides us with a simple, yet powerful, mechanism for identifying objects, the building blocks of the software to be developed. The analysis is basically concerned with the decomposition of a problem into its component parts and establishing a logical model to describe the system functions.

OOD is concerned with the mapping of objects in the problem space into objects in the solution space and creating an overall structure and computational models of the system. This stage normally uses the bottom-up approach to build the structure of the system and the top-down functional decomposition approach to design the class member functions that provide services. It is particularly important to construct structured hierarchies, to identify abstract classes, and to simplify the inner-object communications. Reusability of classes from the previous designs, classification of the objects into subsystems and determination of appropriate protocols are some of the considerations of the design stage.

Object-Oriented Programming (OOP) structures program into classes and functions or methods. To use a class and access the services rendered by each function, you must create an instance of the class. When an instance is created, an object is produced which is held by an object variable. It is this object that will now be used to access each function and make use of its service. The syntax of class instantiation statement for object creation varies from language to language. In PHP, you use the new keyword. For instance, if you have a class with name customer and you want to instantiate it and use the object to access function select_records() in the class, you go about it this way-

```
$cust = new customer();
```

```
$cust->select_records();
```

The first line created an instance of class customer and an object held by object variable \$cust. The second line accesses the service provided by function select_records() with the object variable \$cust. Java too uses the new keyword for object creation but the application of the keyword in C++ is different where it is used by a pointer variable during dynamic memory allocation. I mentioned earlier the three pillars of Object-Oriented Programming (OOP)-Encapsulation, Inheritance, and Polymorphism. They are the integral features of PHP.

Encapsulation is the process of hiding all the details of an object that do not contribute to its essential characteristics. This is achieved by making all instance variables of a class private so that only the member functions of the class can access its private instance variables. Inheritance is a situation in which a class derives a set of attributes and related behavior from a parent class. The parent class is called super class or base class and the inheriting class is called sub class. The member variables of the super class become member variables of the

sub class (derived class). In PHP, you use the keyword extends to implement inheritance just like Java, for example

```
class customer extends products
```

Polymorphism is an extension of inheritance. It is a situation when a sub class overrides a function in the super class. When a function or method is overridden, the name and the signature of the function in the super class are retained by the overriding function in the sub class but there is a change in the function code.

Another important feature of Object-oriented Programming (OOP) language is constructor. A constructor is a function or method bearing the same name as its class name and it is used for initialization of member variables and invoked as soon as the class is instantiated unlike other member functions that are invoked only with the use of the object variable. At this point, let us use submission of data with, for instance, fixed asset register form for further illustration. Your PHP script needs to retrieve data posted from the form, connect to database, print custom error messages and insert data into the database table. Using the Object-Oriented Programming (OOP) approach, you need 4 functions in the class-

- The constructor- to retrieve the posted data from the form.
- A function to connect to MySQL database.
- A function to insert record to the database using the INSERT SQL statement.
- A function to print custom error messages.

Because your program is in an organized form, it is easier to understand and debug. This will be highly appreciated when dealing with long and complex scripts like those incorporating basic stock broking principles. Within the limit of the structured programming capabilities of the non Object-Oriented Programming languages of BASIC, COBOL, PASCAL etc, you could organize program too by dividing it into smaller manageable modules. However, they lack the encapsulation, inheritance, and polymorphism capabilities of Object-Oriented Programming (OOP) which demonstrates a great advantage of the Object-Oriented Programming (OOP) approach.

HOW TO DEVELOP ADVANCED OBJECT-ORIENTED PROGRAM IN PHP

A server-side scripting language, PHP is used in the development of web pages. In recent years, this language has become largely popular because of its simplicity. The code can be combined with HTML scripts and once the PHP code gets executed, the web server sends forth the resultant content in the form of images or HTML documents that can be interpreted easily by the browser. The importance and capabilities of Object Oriented Programming or OOP is not unknown to the PHP programmers and is a technique widely used in all modern programming languages.

POPULAR OOP CONCEPTS IMPLEMENTED IN PHP

Let us now take a look at some of the commonly used concepts of Object Oriented Programming that find an implementation in the popular web

development scripting language that is PHP.

INHERITANCE

Inheritance is one of the most important OOP concepts that finds use in most of the commonly available programming languages. It is based on the principle of child and parent classes. PHP allows inheritance to be achieved by the extension of a class. When this is done, the child class inherits the properties of the parent class and also allows the addition of a number of properties.

PUBLIC, PRIVATE, AND PROTECTED

Just like in any object-oriented language, PHP includes the concepts of various access modifiers, namely Public, Private and Protected.

Public- Public access allows the code to be visible and can be modified by a code from any place else.

Private- Private access makes the code visible but can be modified only from within the user class.

Protected- With the protected access modifier, the code stays visible but can be modified from either the same class or any of the corresponding child class.

OVERLOADING

The overloading feature enables two methods to be used with the same name but

with different parameters. Let us consider the following snippet of code:

```
class Accountant extends Employee
```

```
{
```

```
public function processMonthlySalary ($eCode)
```

```
{
```

```
//code that processes salary
```

```
}
```

```
public function processMonthlySalary ($eCode, $variablePayFlag,  
$variablePercentage)
```

```
{
```

```
if ($variablePayFlag)
```

```
echo "Please process the salary with variable pay";
```

```
else
```

```
echo "Please process the salary without variable pay";
```

```
}
```

```
}
```

In the code shown above, there are two methods bearing the same name ('processMonthlySalary'). When an organisation is taken into account, not every employee will have the variable pay component included in their salary. Thus the accountant will have to process the salary as per the two different approaches mentioned in the code- with variable pay and without variable pay. When the salary is processed using variable pay, the accountant has to mention the amount involved in the variable pay component.

Mutators and Accessors

Commonly referred to as setters and getters, the Mutators and Accessors find wide application in all programming languages. Getters or accessors are functions used to return or get the value of variables at the class level. Setters or mutators are functions used for setting or modifying the value of any class level variable. These class types are used for transferring data in a collective manner from one layer to another.

Static

The use of the static keyword for methods as well as variables is one of the most common practices involved in object-oriented programming. Static indicates that the method or variable is available for every instance of the class. Static variables or methods are used without an instance of a class being created. In fact, static variables cannot be accessed via the instance of any class. When a static method is being created, care should be taken to ensure that the \$this variable does not get featured within a static method.

Abstract class

Abstract class is one feature of PHP-based object oriented programming that is used very frequently. Abstract classes are those that cannot be instantiated, rather they are inherited. A class inheriting an abstract class can be itself be another abstract class. Creation of an abstract class in PHP is possible simply by use of the 'abstract' keyword.

Interface

As far as object-oriented programming is concerned, the interface serves as a collection of definitions of a certain set of methods in a class. When an interface is implemented, the class is forced to follow the methods defined in the same. PHP defines interface just as in any other language by first defining it using the "interface" keyword. For implementing, the very same keyword needs to be mentioned in the implementing class. Let us take a look at a sample code for

better understanding.

```
interface myIface
```

```
{
```

```
public function myFunction ($name);
```

```
}
```

```
class myImpl implements myIface
```

```
{
```

```
public function myFunction ($name)
```

```
{
```

```
//function body
```

```
}
```

```
}
```

As far as the PHP programmers are concerned, this language has emerged as one of the most popular options as a website and web-based application development language. It has undergone numerous enhancements for supporting various aspects of programming. Of all these, understanding and implementing the object-oriented features ranks as the most important.

PRINCIPLES OF OBJECT ORIENTED PROGRAMMING

Object Oriented Programming (or OOP) is actually classified by three main principles.

- Encapsulation
- Inheritance
- Polymorphism

These appear to be frightening terms but are actually fairly easy principles to grasp. In order to figure out how to program with java, you'll need to understand these principles. So let's consider our first main concept of OOP, encapsulation. Encapsulation just means we want to limit the access that some other pieces of code have to this particular object. So, to illustrate, if you have a Person object, and this Person object has a first and last name as attributes. In the event another chunk of code attempts to modify your Person object's first name to be say "Frank3", you could take note of what the first name is trying to be set to, and

remove any digits so that we are simply left with "Frank". Without encapsulation, we will not have the ability to prevent "silly programmers" from modifying the values of our variables to something which wouldn't seem sensible, or worse, break the application. Seem sensible?

The second concept of OOP, and an essential principle if you wish to learn how to program with Java, is Inheritance. This specific concept refers to a super class (or parent class) and a sub-class (or child class) and the simple fact that a child class acquires each of the attributes of its parent. You can think of it in terms of a real world circumstance, like a real parent and child. A child will probably inherit certain traits from his or her parents, like say, eye colour or hair colour. Allow us to imagine yet another example in terms of programming, say we have super class "Vehicle" and sub-classes "Car" and "Motorcycle". A "Vehicle" possesses tires, therefore through inheritance so would a "Car" and a "Motorcycle", however a "Car" has doors, and a "Motorcycle" does not. So it wouldn't be accurate to state that a "Vehicle" has doors, as that declaration would be inaccurate. So you can see how we could determine all the aspects that are similar regarding a "Car" and a "Motorcycle" and thus identify them inside of the "Vehicle" super class.

The 3rd concept of OOP is Polymorphism. This specific concept appears to be one of the most frightening, but I'm able to explain it in simple terms. Polymorphism means that an object (i.e. Animal) can take on several forms while your program is operating. Let's imagine you have designed an Animal class and defined the method "Speak". You then asked three of your buddies to develop kinds of animals and have them implement the "Speak" method. You won't know what sort of animals your friends create, or how their Animals will speak, unless you actually hear those animals speak. This is very comparable to

how Java addresses this issue. It's called dynamic method binding, which simply means, Java won't understand how the actual Animal speaks until runtime. So maybe your friends have created a Dog, Cat and Snake. Here are three varieties of Animals, and they each one speaks distinctly. Whenever Java asks the Dog to speak, it says "woof". Anytime Java asks the Cat to speak, it says "meow". Whenever Java requests the snake to speak, it hisses. There's the beauty of polymorphism, all we did was to define an Animal interface with a Speak method, and we can make a bunch of kinds of animals which speak in their own specialized way.

Common Misconception About Object-Oriented Programming

Scholars may quibble about the fine points of object orientation; however, one thing is for certain: merely having private data and public functions does not constitute a proper object-oriented design. Rather, proper object orientation entails much more.

One of the most basic elements is information hiding. This means that objects should only present the information that needs to be seen; that is, it should present a coherent and well-selected interface of functions-one that does not betray the data contents and internal workings of the class. In other words, the manner in which the functions are implemented remains hidden from the user, allowing the developer to alter the implementation as needed.

When a programmer declares that his code is object-oriented by virtue of having private data and public functions, he is placing the cart before the proverbial horse. Using private data and public functions is merely a means of achieving information hiding; it is not a goal in itself. For example, consider a design in which every single data member has corresponding "get" and "set" accessors

(e.g. a data member "x" would have matching "getx()" and "setx()" functions). In this example, information is poorly hidden, since the choice of functions (indeed, their very names!) betrays the manner in which the data has been implemented.

Inheritance is another key element; that is, specific classes are to be derived from more general ones. Inheritance is a means of implementing abstraction; that is, it allows the user to identify objects with common characteristics, and which should therefore use common code (or at the very least, common interfaces). This is part and parcel of thinking in terms of objects, as opposed to thinking primarily in terms of functions and procedures.

Yet another key characteristic is polymorphism, which allows a descendant object to override its parent's member functions. With polymorphism, a descendent object does not have to respond to a message exactly as its ancestor does; rather, it can have its own implementation. Note that the descendant objects do not have to override these functions; rather, they should simply be allowed to do so, as needed. There is much more that can be said about the nature of object orientation; indeed, scholars often contend over its precise definition and its principal ideas. Whatever the case though, the point remains: merely keeping private data and a set of public functions does not constitute an object-oriented design-not in any meaningful sense of the term.

OBJECT-ORIENTED DEVELOPMENT METHODS

Development methods for traditional programming languages, such as C emerged in the 1980s. The results were not always as good as hoped for-many Computer-Aided Software Engineering(CASE) systems were little more than report generators that extracted designs after the implementation was complete-

but the methods included good ideas that were occasionally used effectively in the construction of large systems.

The first object-oriented language is generally acknowledged to be Simula-67 developed in 1967. This language never had significant following, although it greatly influenced the developers of several of the later object-oriented languages. This movement became active with the widespread availability of Smalltalk in the early 1980s, followed by other object-oriented languages, such as Objective C, C++, and Eiffel. The actual usage of object-oriented languages was limited at first, but object orientation attracted a lot of attention.

There were some early attempts to unify concepts among methods. In 1996, the Object Management Group (OMG) issued a request for proposals for a standard approach to object-oriented modeling. The emergence of UML appears to be attractive to the general computing public because it consolidates the experiences of many authors.

UML was developed in an effort to simplify and consolidate the large number of object-oriented development methods that had emerged. The UML combines the commonly accepted concepts from many object-oriented methods, selecting a clear definition for each concept, as well as a notation and terminology. The UML is seamless from requirements to deployment. The same set of concepts and notation can be used in different stages of development. It is not necessary to translate from one stage to another. This seamlessness is critical for iterative, incremental development.

The UML is intended to model most application domains, including those

involving systems that are large, complex, real-time or computation intensive. There may be specialized areas in which a special-purpose language is more useful, but UML is intended to be as good as or better than any other general-purpose modeling language for most application areas.

The UML is intended to be usable for systems implemented in various implementation languages and platforms, including programming languages, databases, 4GLs, and so on. The UML is a modeling language, not a description of a detailed development process. It is intended to be usable as the modeling language underlying most existing or new development processes, just as a general-purpose programming language can be used in many styles of programming. It is particularly intended to support the iterative, incremental style of development.

PYTHON TECHNOLOGY USED FOR APPLICATION ORIENTED FIELDS

Python is a dynamic and object-oriented programming language, widely used for web application development. 90% of people prefer Python over other technology because of its simplicity, reliability and easy interfacing. It offers both powerful scripting and fast application development process across a vast range of fields. As the basis of several open-source platforms, Python supports with tools that help to build applications with excellent security and performance levels. Python follows procedural and object-oriented coding paradigms and hence, the varied applications written in Python come out with clean and readable code, making them easy to maintain.

USES OF PYTHON TECHNOLOGY FOR APPLICATION

DEVELOPMENT

Python is an open source programming language, which is widely used in a number of application domains. It can perform on almost all operating systems like Windows, Linux, UNIX, OS/2, Mac, and Amiga. The dedicated Python Development team has written several applications based on python programming language. Python being a fun and dynamic language, it has been used by a number of companies such as Google, Yahoo and IBM. It is also used widely to write custom tools and scripts for special applications.

Python is extensively used in Web applications development such as Django, Pylons, Games Applications like Eve Online, Image Applications, Science and Education Applications, Software Development, Network Programming, Mobile applications, Audio/Video Applications etc.

FEATURES OF PYTHON

Python can be easily interfaced with C/ObjC/Java/Fortran. The key features of Python are its natural expression of procedural code, sound introspection capabilities, very precise, readable syntax, instinctive object orientation, dynamic data types, extensions and modules easily written in C, C++, extensive standard libraries and full modularity, exception-based error handling and embeddable within applications as a scripting interface. Also, Python supports the Internet Communications Engine (ICE) and several other integration technologies.

PYTHON DEVELOPMENT SERVICES

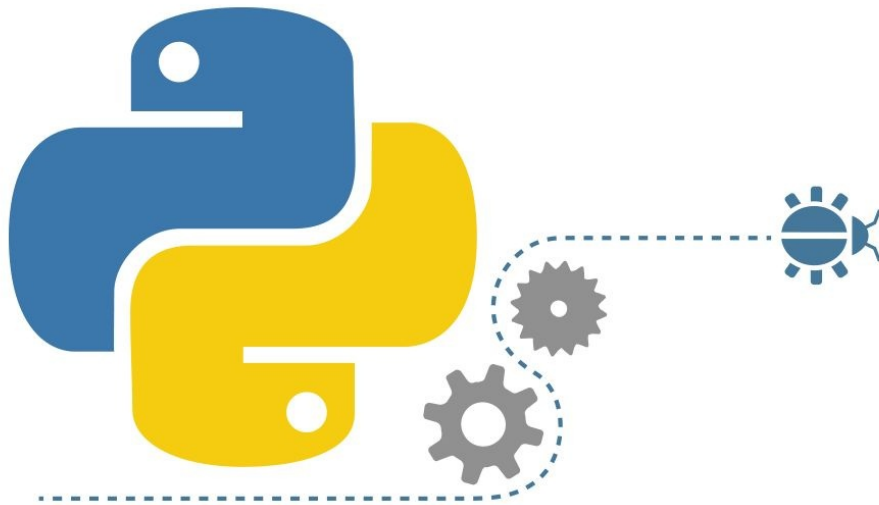
As a dynamic, general purpose programming language, Python is finding extensive usage by Python Development Services providers across the world for developing a wide range of software applications. It allows developers to express concept with less and readable code. It enable the developers to integrate Python with varied other well-known programming languages and tools flawlessly.

Frequently the Python developers have the necessity to use diverse frameworks and tools to create high-end quality software applications within a short period of time. With the support of the resources offered by the varied Python frameworks, Python developers build sophisticated applications with minimal time and effort. Also, Python developers have an option to select from a number of Python frameworks such as Kivy, Qt, PyGUI, WxPython, Django, Flask, Pyramid etc depending on the nature and requirements of individual application building projects.

Python is a popular choice for use as a scripting language for many software development processes. Similar to many other interpretative languages, Python offers more flexibility than compiled languages, and it can be efficiently used to integrate disparate systems together. Certainly, Python is a versatile programming language with several applications that are useful in diverse fields.

CHAPTER 8

REAL WORLD EXAMPLES OF PYTHON



Let's discuss python applications:

WEB AND INTERNET DEVELOPMENT

Python lets you develop a web application without too much trouble. It has libraries for internet protocols like HTML and XML, JSON, e-mail processing, FTP, IMAP, and easy-to-use socket interface. Yet, the package index has more libraries:

- Requests – An HTTP client library

- BeautifulSoup – An HTML parser
- Feedparser – For parsing RSS/Atom feeds
- Paramiko – For implementing the SSH2 protocol
- Twisted Python – For asynchronous network programming

We also have a gamut of frameworks available. Some of these are- Django, Pyramid. We also get microframeworks like flask and bottle.

We can also write CGI scripts, and we get advanced content management systems like Plone and Django CMS.

APPLICATIONS OF PYTHON PROGRAMMING IN DESKTOP GUI

Most binary distributions of Python ship with Tk, a standard GUI library. It lets you draft a user interface for an application. Apart from that, some toolkits are available:

- wxWidgets
- Kivy – for writing multitouch applications
- Qt via PyQt or PySide

And then we have some platform-specific toolkits:

- GTK+
- Microsoft Foundation Classes through the win32 extensions

- Delphi

SCIENCE AND NUMERIC APPLICATIONS

This is one of the very common applications of python programming. With its power, it comes as no surprise that python finds its place in the scientific community. For this, we have:

- SciPy – A collection of packages for mathematics, science, and engineering.
- Pandas- A data-analysis and -modeling library
- IPython – A powerful shell for easy editing and recording of work sessions. It also supports visualizations and parallel computing.
- Software Carpentry Course – It teaches basic skills for scientific computing and running bootcamps. It also provides open-access teaching materials.
- Also, NumPy lets us deal with complex numerical calculations.

SOFTWARE DEVELOPMENT APPLICATION

Software developers make use of python as a support language. They use it for build-control and management, testing, and for a lot of other things:

- SCons – for build-control
- Buildbot, Apache Gump – for automated and continuous compilation and testing

- Roundup, Trac – for project management and bug-tracking.
- Roster of Integrated Development Environments

PYTHON APPLICATIONS IN EDUCATION

Thanks to its simplicity, brevity, and large community, Python makes for a great introductory programming language. Applications of python programming in education has huge scope as it is a great language to teach in schools or even learn on your own.

PYTHON APPLICATIONS IN BUSINESS

Python is also a great choice to develop ERP and e-commerce systems:

- Tryton – A three-tier, high-level general-purpose application platform.
 - Odoo – A management software with a range of business applications.
- With that, it's an all-rounder and forms a complete suite of enterprise-management applications in-effect.

DATABASE ACCESS

This is one of the hottest Python Applications.

With Python, you have:

- Custom and ODBC interfaces to MySQL, Oracle, PostgreSQL, MS SQL

Server, and others. These are freely available for download.

- Object databases like Durus and ZODB
- Standard Database API

NETWORK PROGRAMMING

With all those possibilities, how would Python slack in network programming? It does provide support for lower-level network programming:

- Twisted Python – A framework for asynchronous network programming. We mentioned it in section 2.
- An easy-to-use socket interface

AMES AND 3D GRAPHICS

Safe to say, this one is the most interesting. When people hear someone say they're learning Python, the first thing they get asked is – 'So, did you make a game yet?'

PyGame, PyKyra are two frameworks for game-development with Python. Apart from these, we also get a variety of 3D-rendering libraries.

If you're one of those game-developers, you can check out PyWeek, a semi-annual game programming contest.

OTHER PYTHON APPLICATIONS

These are some of the major Python Applications. Apart from what we just discussed, it still finds use in more places:

- Console-based Applications
- Audio – or Video- based Applications
- Applications for Images
- Enterprise Applications
- 3D CAD Applications
- Computer Vision (Facilities like face-detection and color-detection)
- Machine Learning
- Robotics
- Web Scraping (Harvesting data from websites)
- Scripting
- Artificial Intelligence
- Data Analysis (The Hottest of Python Applications)

This was all about the Python Applications Tutorial. If you like this tutorial on applications of Python programming comment below.

Why should you Learn python? Refer this link to get your answer.

Ready to install Python? Refer this link [Python Installation](#).

Python is everywhere and now that we know python Applications. We can do with it, we feel more powerful than ever. If there's a unique project you've made in the Python language.

PYTHON FRAMEWORKS AND THEIR REAL WORLD USAGE

Why should you go for Python?

Python is a language which has gained a stupendous popularity in the industry within a very less span of time after the magical touch from Google. With the unladen swallow project, the speed of the technology has increased around two to three times. As a result, it is often preferred for large scale software application that needs high power and speed from the core of the language. Well in 21st century we can't ignore the importance of web domain and fortunately Python is highly flexible in developing enterprise standard web solutions. The web applications demanding more speed and power can be achieved with Python. All types of data driven web applications can be developed in Python with maximum power and potential. In fact, Python is extremely popular among the web developers due to its unmatched quality and performance along with few highly efficient Python frameworks. Let us discuss top Python frameworks of the industry.

Django framework:

This is probably the most prominent Python web framework of the industry. Django is extremely powerful and was developed by Jason Sole and Jason Mc Laugilin. It was implemented for the first time in a job portal to ensure its efficiency. Later on, it was released for everyone and received an over whelming response in the industry. This is the largest Python based web framework of the industry and hence comes with extra ordinary power and features for the Python developers. There is a huge support community for this framework who are

working 24x7 to provide support to others. It has the most powerful admin interface for the top level control of any large scale web application. Practically this framework is extremely useful in developing online forums, portals and other social networking sites. It is also extremely popular in developing quick web solutions with maximum efficiency and minimum effort.

Flask:

Flask is a micro framework developed in Python having a strange background. It was the result of an April fool surprise from an intelligent Python developer of India. Mr. Pradeep Gowda challenged his colleague to develop a single file micro framework in Python and he succeeded in the same and gifted it to his friend as the April fool surprise. Well, this is not as powerful as that of Django but it can be used for moderate size of Python web application development projects. It is best suitable for beginners who want to learn Python and start coding in less time span. It creates an insatiable thirst among the Python developers to experiment with the framework. It is often used in small web development that too within low budget and limited time frame. Hence it has received a good word of mouth response from the industry since its inception.

Pyramid:

Pyramid is yet another awesome Python development framework having extraordinary flexibility to serve a wide range of web applications. It is also powered by git hub and hence there is no fear for support. It is the combination of pylons 1.0 and repose.bfg. It is growing in a much faster pace in the community due to its flexible nature. Practically it is helpful for developing enterprise standard API Python projects and creating Python based CMS or KMS.

WHY PYTHON HAS BECOME AN INDUSTRY FAVORITE AMONG PROGRAMMERS

With the world stepping towards a new age of technology development, it isn't hard to imagine a future that will be full of screens. And if so be the case then, demand for people with strong programming skills will definitely rise with more number of people required to develop and support the applications. Python Training is always a good idea for those wishes to be a part of this constantly developing industry. Python language is not only easy to grasp, but emphasizes less on syntax which is why a few mistakes here and there doesn't give as much trouble as some other languages does.

What Makes Python a Preferred Choice Among Programmers?

Python happens to be an easy programming language which offers its support to various application types starting from education to scientific computing to web development. Tech giants like Google along with Instagram have also made use of Python and its popularity continues to rise. Discussed below are some of the advantages offered by Python:

First Steps in the World of Programming

Aspiring programmers can use Python to enter the programming world. Like several other programming languages such as Ruby, Perl, JavaScript, C#, C++, etc. Python is also object oriented based programming language. People who have thorough knowledge of Python can easily adapt to other environments. It is

always recommended to acquire working knowledge so as to become aware of the methodologies that are used across different applications.

Simple and Easy to Understand and Code

Many people will agree to the fact that, learning and understanding a programming language isn't that exciting as compared to a tense baseball game. But, Python on the other hand was specifically developed keeping in mind newcomers. Even to the eye of a layman, it will seem meaningful and easy to understand. Curly brackets and tiring variable declarations are not part of this programming language thus, making it a lot easier to learn language.

Getting Innovative

Python has helped in bringing real world and computing a lot close with it Raspberry Pi. This inexpensive, card-sized microcomputer helps tech enthusiasts to build various DIY stuffs like video gaming consoles, remote controlled cars and robots. Python happens to be the programming language that powers this microcomputer. Aspirants can select from different DIY projects available online and enhance their skills and motivations by completing such projects.

Python also Supports Web Development

With its huge capabilities, Python is also a favorite among web developers to build various types of web applications. The web application framework, Django has been developed using Python and serves as the foundation for popular websites like 'The Guardian', 'The NY Times', 'Pinterest' and more.

Python provides aspiring programmers a solid foundation based on which they can branch out to different fields. Python programming training ensures that students are able to use this highly potential programming language to the best of its capabilities in an exciting and fun way. Those who are keen to make a great career as software programmers are definite to find Python live up to their expectations.

© Copyright 2019 by **WILLIAM GRAY**

All rights reserved

If You have a few moments, I would appreciate a review on Amazon, if You found your new book useful in any way.

Enjoy !