Share on your Social Media

Python Full Stack Tutorial

Published On: July 8, 2024

Python Full Stack Tutorial

Learn to utilize Python as their primary programming language to handle every aspect of web application development. Our Python full-stack tutorial helps you with the fundamental understanding of developing web apps easily and effectively.

Python Full Stack Tutorial PDF

Introduction to Python Full Stack

A Python full-stack developer is a web developer who handles all facets of web application development using Python as their primary programming language. They use Python to create both the website's front end, or user interface, and its back end, or server logic storage.

A Python Full-Stack Developer Must Have the Following Skills

- Familiarity with Core Python
- Expertise with Debugging
- Understanding of Python Frameworks, such as Django, Flask, etc.
- Understanding of machine learning, deep learning, and artificial intelligence.
- Understanding of ORMs (Object Relational Mapping)

Related Cour
at SLA

- Python Fullstack Online Training
- Python Fullstack Training in Chennai
- Python Full Stack Developer Course

Related Posts

Core Python

The Python programming language, namely the most recent version 3, is utilized in cutting-edge software industry technologies such as machine learning and web development.

Learning the language's syntax comfortably is the first and most important stage in learning anything. Python's syntax is simple and suitable for beginners.

There are two methods for running a Python program:

- Initially, we create a program in a file and execute it once.
- Second, examine each line of the code.

Example

> *print("Hello World! It's my first Python program."*

Output

> Hello World! It's my first Python program.

Python Syntax

The collection of guidelines known as Python syntax specifies the symbol combinations that, when combined, are regarded as appropriately organized Python scripts.

These guidelines ensure that Python programs are prepared and organized correctly so that the Python interpreter can comprehend and run them.

Operations:

Like most high-level programming languages, Python has calculator functionality. To view some of the outcomes, type a few of these into your Python shell.

Operation	Output
a + b	The sum of a and b.
a – b	The difference between a and b.

MongoDB Tutorial for Beginners

Published On: September 25, 2024

MongoDB Tutorial for Beginners In real-world contexts like e-commerce, finance, healthcare, journalism, and social networks,...

Mobile App Development Tutorial

Published On: September 25, 2024

Mobile App Development Tutorial Businesses looking to grow and scale can benefit greatly from mobile...

Microsoft Azure Tutorial for Beginners

Published On: September 24, 2024

Microsoft Azure Tutorial for Beginners We can use a wide range of services from Microsoft...

MEAN Stack Tutorial

a * b	The product of a and b.
a / b	The quotient of a and b.
a // b	The rounded-down quotient of a and b.
a % b	The remainder of a and b.
a ** b	A to the b power.

Python Full Stack Syllabus PDF

Python Indentation

Python indentation is the practice of starting a line of code in whitespace, such as spaces or tabs. It serves as a definition for the code blocks.

While it makes Python code easier to understand, it also makes it more challenging to fix indentation mistakes.

> **Example**

```
if 10 > 5:
  print("This is true!")
if 10 > 5:
  print("This is true!")
```

Python Variables

In Python, variables are simply named references that point to memory-based objects. Python will determine the type dynamically based on the provided value.

> **Example**

```
a = 18
type(a)
class 'int'
a = "SLA"
type(a)
class 'str'
```

Python Identifiers

for Beginners

Published On: September 24, 2024

MEAN Stack Tutorial for Beginners 'M' refers to MongoDB, 'E' for Express, 'A' for Angular,...

Identifiers are special names given to classes, variables, functions, and other entities. They serve to distinguish each entity in the program individually.

They can have underscores(_) or letters, digits, or other characters after the initial letter (a-z, A-Z). The identifier "first_name" in the example below stores a string value.

identi_sample = "SLA"

The following are the rules for Python identifiers:

- There shouldn't be any spaces or special characters in them.
- Alphabets (uppercase or lowercase), digits (0–9), and the underscore character (_) can all be used as identifiers.
- An identifier's initial letter needs to be either an underscore or an alphabet.
- Every identifier should be unique within its scope, or namespace.
- Reserved identifiers are terms in Python that have particular meanings and functionalities. These terms cannot be selected as identifiers as a result.

For example, in Python, the word "print" already has a defined meaning and cannot be used as an identifier.

Python Keywords

Python-reserved words with specific meanings are called keywords.

For example, while, if, else, and so on. They can't be put to use as markers.

The list of Python keywords is as follows,

False	await	else	True
import	except	pass	None
class	break	raise	in
finally	is	return	and
continue	for	lambda	try

as	def	from	nonlocal
while	assert	del	global
not	async	elif	if
or	yield		

Python Comments

In Python, comments are statements that are written within the code.

They serve to clarify how a code operates; they have no bearing on how a program is executed or what happens when it runs.

Single-Line Comment

The “#” sign comes before single-line comments. Anything on the same line that follows this symbol is regarded as a comment and not as a part of the code that is being executed.

We can observe that a single-line comment has no impact on the output.

> **Example**
>
> ```
> first_name = “SLA”
>
> last_name = “Institute”
>
> # print full name
>
> print(first_name, last_name)
> ```

Output

> ```
> SLA Institute
> ```

Multi-Line Comment

There is no special syntax for multi-line comments in Python. Though technically they are string literals, programmers frequently utilize several single-line comments, one after the other, or occasionally triple quotes (either “‘ or “””).

> **Example**
>
> ```
> # This is a Python program
> ```

```
# to explain multiline comments.

’’’

Functions to print table of

any number.

’’’

def print_table(n):

 for i in range(1,11):

   print(i*n)

print_table(4)
```

Output

```
4

8

12

16

20

24

28

32

36

40
```

[Python Full Stack Interview Questions & Answers]

Python Multiple Line Comments

Long statements are not readable or practical to write in code. Because breaking up a lengthy single-line statement into numerous lines makes it easier to understand, Python has this capability that allows us to split up long statements into various lines in various ways, such as,

Backslashes(\): The backslash (\) in Python allows you to divide a statement into multiple lines. This approach is helpful, particularly when performing mathematical calculations or working with strings.

Example

```
sentence = “This is a lengthy sentence that we would want to” \
         “divide into several lines for easier reading.”
print(sentence)
# For mathematical operations
total = 1 + 2 + 3 + \
      4 + 5 + 6 + \
      7 + 8 + 9
print(total)
```

Output

```
This is a lengthy sentence that we would want to divide into several lines
for easier reading.

45
```

Parentheses: We can divide statements across many lines inside the parenthesis for some structures, such as lists, tuples, or function parameters, without the need for backslashes.

Example

```
# Create list
numbers = [1, 2, 3,
         4, 5, 6,
         7, 8, 9]
def total(num1, num2, num3, num4):
  print(num1+num2+num3+num4)
# Function call
total(23, 34,
     22, 21)
```

Output

```
100
```

Triple Quotes for Strings

Triple quotes are an option when working with docstrings or multiline strings (single “‘ or double “””).

> **Example**
>
> ```
> text = “”” Online and offline SLA courses
>
> to improve your programming.
>
> Utilize live and virtual sessions
>
> to study and practice challenges.”””
>
> print(text)
> ```

> **Output**
>
> ```
> Online and offline SLA courses
>
> to improve your programming.
>
> Utilize live and virtual sessions
>
> to study and practice challenges.
> ```

Python Quotation

Python quotes that contain a single (‘), double (“), or triple (“‘ or“”) can be used to enclose strings. Simple strings can be defined with either single or double quotations, while multiline strings can be created with triple quotations, as we employed in the case above.

Example

> ```
> # Embedded single quote inside double.
>
> text1 = “He said, ‘I learned Python from SLA'”
>
> # Embedded double quote inside single.
>
> text2 = ‘He said, “I have created a project using Python.”‘
>
> print(text1)
>
> print(text2)
> ```

> **Output**
>
> ```
> He said, ‘I learned Python from SLA’
>
> He said, “I have created a project using Python.”
> ```

Command-Line Arguments in Python

Python scripts can receive input from the command line during runtime by using command-line arguments.

Stated differently, arguments that follow a program's name in the operating system's command line shell are known as command line arguments.

We have several options for handling this kind of argument using Python. Here are the most popular methods:

- Using sys.argv
- Using getopt module
- Using argparse module

python full stack Developer Salary

Python Sets

The Python Set is an unordered, changeable, iterable collection of data types without duplicates. Even though a set may contain a variety of components, the order in which they appear is not specified.

Using a set instead of a list has the main benefit of having a highly efficient way to determine whether a given piece is included in the set.

> **Example**

```
# Creating a Set with a List of Numbers

# (Having duplicate values)

set1 = set([1, 2, 4, 4, 3, 3, 3, 6, 5])

print("\nSet with the use of Numbers: ")

print(set1)

# Creating a Set with a mixed type of values

# (Having numbers and strings)

set1 = set([1, 2, 'Welcome', 4, 'to', 6, 'SLA'])

print("\nSet with the use of Mixed Values")

print(set1)
```

> **Output**

```
Set with the use of Numbers:
```

> {1, 2, 3, 4, 5, 6}
>
> Set with the use of Mixed Values
>
> {1, 2, 4, 6, ‘Welcome’, ‘to’}

What is Debugging in Python Full-Stack Development?

Debugging is a crucial ability for increasing an application’s efficiency by streamlining its logic and optimizing performance, in addition to correcting errors. A Python full-stack developer must invest time and practice to become proficient in debugging because it’s an advanced skill.

Popular Debugging Techniques

Typical debugging techniques include the following:

- Printing the status and variable values at specific points while the program runs.
- Modifying a program’s state to change its behavior. This is referred to as changing the program’s “path”.
- Stepping through a program’s execution line-by-line.
- Breakdowns and Trace Elements
- Put an end to the show at specific points
- By looking at a program’s output in a debugger window.

Popular Debugging Tools for Python Full-Stack Development

There are numerous debugging tools available, some of which are stand-alone programs and others of which are integrated into IDEs like PyCharm.

Web-PDB: A web-based pdb user interface called Web-PDB makes it simpler to comprehend what’s occurring when you examine your code while it’s running.

Pdb: The debugger most developers use to try to debug their programs is called pdb, and it is a part of the Python standard library.

wdb lets you debug Python code that is executed from a web browser by utilizing WebSockets.

Pyflame: A profiling tool called Pyflame (source code) creates flame graphs when running Python programs.

Objgraph: Graphviz is used by objgraph (source code) to illustrate the relationships between Python objects that are executing in an application.

Python Frameworks for Python Full-Stack Development

Some of the popular frameworks in Python that are used for full-stack web development are as follows:

Django: Running on a web server, Django is an open-source, free web framework based on Python. It adheres to the architectural model-template-view paradigm.

Flask: Python-based Flask is a microweb framework. Flask is a helpful tool for developing small, straightforward applications.

CherryPy: CherryPy is a Python programming language object-oriented web application framework. Additionally, CherryPy lets programmers write web apps in the same way as they would any other Python object-oriented program.

Web2Py: The Python programming language is used to create the open-source Web2py web application framework. Web developers can use Web2py to program dynamic web content.

Machine Learning for Python Full-Stack Development

Machine learning engineering and full-stack development are both in great demand and pay well. The particular skills needed for each field can, however, differ greatly.

- Full-stack development can be the ideal career choice for you if you have a strong interest in both front-end and back-end development and enjoy working with a variety of technologies.

- However, machine learning engineering might be a better fit if you have a solid foundation in mathematics and statistics and are especially interested in leveraging that knowledge to construct intelligent systems.

Following are the reasons to choose machine learning skills for Python full-stack development:

High Demand: As more businesses realize how machine learning can be used to better their goods and services, there is a sharp rise in the need for machine learning engineers.

A fascinating and cutting-edge topic, machine learning offers engineers the chance to work on cutting-edge technology and tackle challenging issues. The discipline is continually evolving.

Job Flexibility: Depending on their interests and abilities, machine learning engineers can operate in a variety of capacities as researchers, data scientists, or software engineers.

Career Advancement: Data science managers and chief machine learning officers are two leadership roles that machine learning engineers can potentially reach.

Positive Impact: By creating intelligent systems that can enhance healthcare, finance, transportation, and many other industries, machine learning engineers have the potential to have a major positive impact on society.

Versatility and Flexibility: Python full-stack developers can work on any part of a web application, from the back-end server and database to the front-end user interface.

Possibilities for entrepreneurship: Python full-stack developers can utilize their expertise to launch their websites, goods, or services.

Deep Learning for Python Full-Stack Development

Keras is an incredibly potent and user-friendly Python library for creating and assessing deep learning models.

It encapsulates the effective numerical computation libraries TensorFlow and Theano.

The key benefit of this is that it's a simple and enjoyable method to get started with neural networks.

Keras: The most amazing machine learning library, Keras, is utilized in Python projects to make neural network expression simpler. It also offers the greatest tools for processing datasets, visualizing graphs, and compiling models.

TensorFlow: TensorFlow uses N-dimensional matrices to represent data, and it is speed-optimized by applying XLA algorithms to perform rapid linear algebra operations. Specialized features of TensorFlow, such as Responsive Construct, provide simple visualization with SciKit or NumPy.

Scikit: In addition to training method implementations like logistic regressions, it features a cross-validation feature that allows the use of several metrics.

NumPy: It is very interactive, simple to use, open-source, and makes complicated mathematical implementations easier. NumPy helps describe binary raw streams, sound waves, and the use of N-dimensional array implementations, among other things.

Theano: Multidimensional array processing is done with a library called Theano, which is built on a computational framework. Its efficient symbolic differentiation for numerous inputs, transparent GPU usage, and close connection with NumPy are all features.

Artificial Intelligence for Python Full-Stack Development

We've reached an interesting intersection in the digital landscape where the adaptability of Full-Stack development and the strength of artificial intelligence collide.

It’s a place where modernizing tradition improves efficiency and automation and fosters creativity.

What Benefits of AI for Full Stack Development?

Automating the routine: Artificial intelligence (AI) can handle tedious chores, such as composing functions or setting up boilerplate code.

Streamlined testing: AI makes it easier for engineers to find and fix errors.

UI/UX improvements: AI technologies can make customized design suggestions based on information about user behavior, improving the user experience.

Intelligent Data Management: By anticipating usage trends and streamlining queries, artificial intelligence (AI) may take charge of database management and improve the efficiency of web services.

Python Full Stack Training

ORMs (Object Relational Mapping) for Python Full-Stack Development

Object Relational Maps (ORMs) offer a high-level abstraction over relational databases, enabling developers to create, read, update, and delete data and schemas in their databases using Python code rather than SQL.

To retrieve every row in the USERS table where the zip_code column is 94107, for instance, a developer working without an ORM might write the following SQL statement:

*SELECT * FROM USERS WHERE zip_code=94107;*

The equivalent Python code for the Django ORM query would be as follows:

#assign to everyone in the 94107 zip code.

users = Users.objects.filter(zip_code=94107)

Relational database access can be achieved without the need for Python ORM packages.

A different package known as a database connector, like psycopg for PostgreSQL or MySQL-python for MySQL, is usually responsible for providing low-level access.

Python ORM Impementation

Many ORM implementations have been created in Python, such as

SQLAlchemy: The Python SQL toolkit and Object Relational Mapper known as SQLAlchemy provides application developers with all of SQL's versatility and capability.

Django ORM: Known as "the Django ORM" or "Django's ORM," the object-relational mapping module that comes with the Django web framework is integrated inside it.

Peewee ORM: An ORM system built in Python called Peewee aims to be "simpler, smaller, and more hackable" than SQLAlchemy.

PonyORM: Another Python ORM that is free source and covered by the Apache 2.0 license is called Pony ORM.

SQLObjectORM: Since before 2003, SQLObject has been the subject of continuous open-source development for more than 14 years.

Schema Migrations

In technical terms, schema migrations are not included in ORMs. An example of this would be when you need to add a new column to an existing table in your database.

However, using Python ORMs on web application projects commonly goes hand in hand with libraries to conduct schema migrations, since ORMs generally lead to a hands-off approach to the database.

Conclusion

We hope this Python Full-Stack Tutorial helps you understand the fundamentals of web development with Python programming. Join us for the best **Python Full-Stack training in Chennai** and start a promising career in full-stack development.

Share on your Social Media

Navigation

- About Us
- Blog Posts
- Careers
- Contact
- Placement Training
- Corporate Training
- Hire With Us
- Job Seekers
- SLA's Recently Placed Students
- Reviews
- Sitemap

Important Links

- Disclaimer
- Privacy Policy
- Terms and Conditions

Softlogic Academy

Softlogic Systems

KK Nagar [Corporate Office]

No.10, PT Rajan Salai, K.K. Nagar, Chennai – 600 078.
Landmark: Karnataka Bank Building
Phone: +91 86818 84318
Email: enquiry@softlogicsys.in
Map: Google Maps Link

OMR

No. E1-A10, RTS Food Street
92, Rajiv Gandhi Salai (OMR),
Navalur, Chennai - 600 130.
Landmark: Adj. to AGS Cinemas
Phone: +91 89256 88858
Email: info@softlogicsys.in
Map: Google Maps Link

Courses

Social Media Links

- Python
- Software Testing
- Full Stack Developer
- Java
- Power BI
- Clinical SAS
- Data Science
- Embedded
- Cloud Computing
- Hardware and Networking
- VBA Macros
- Mobile App Development
- DevOps

Review Sources

- Google
- Trustpilot
- Glassdoor
- Mouthshut
- Sulekha
- Justdial
- Ambitionbox
- Indeed
- Software Suggest
- Sitejabber