Python Advanced (Full-Stack)

Python Advanced (Full-Stack Python)

Want to learn the different ways a Python App can be built? We start with an overview of the language, then develop a game using Python's popular gaming libraries. Our second project is a Python GUI app to store and report on data. Our last app is a Web App, using Django.

Prerequisites / Further Training

Python Beginner

Also have a look at our Python Bootcamp

Intended Audience

The goal of this course is to bring you up to speed with Python as quickly as possible so you can build programs that work—games, data visualizations, and web applications—while developing a foundation in programming that will serve you well for the rest of your career. Python Advanced is written for Python Beginner Developers who want to learn how to apply the language practically – it is not a Python beginner's course – you should have passed the level of Python Beginner before doing this course . In this course we look at the 3 most popular ways to develop Python apps – Games, GUI-based and Web-based.

After this course you should be able to

- In the first part of this course you'll learn basic programming concepts you need to know to write Python programs.
  You'll learn about different kinds of data and the ways you can store data in lists and dictionaries within your programs.
- You'll learn to build collections of data and work

through those collections in efficient ways.
You'll learn to use while and if loops to test for certain conditions so you can run specific sections of code while those conditions are true and run other sections when they're not—a technique that greatly helps to automate processes.
You'll learn to accept input from users to make your programs interactive and to keep your programs running as long as the user is active.
You'll explore how to write functions to make parts of your program reusable, so you only have to write blocks of code that perform certain actions once, which you can then use as many times as you like.

- You'll then extend this concept to more complicated behavior with classes, making fairly simple programs respond to a variety of situations.
- You'll learn to write programs that handle common errors gracefully. After working through each of these basic concepts, you'll write a few short programs that solve some well-defined problems.
- Finally, you'll take your first step toward intermediate programming by learning how to write tests for your code so you can develop your programs further without worrying about introducing bugs.
- In the first project you'll create a Space Invaders–style shooting game called Alien Invasion, which consists of levels of increasing difficulty.
  After you've completed this project, you should be well on your way to being able to develop your own 2D games.
  The second project introduces you to data visualization. Data scientists aim to make sense of the vast amount of information available to them through a variety of visualization techniques.
- You'll work with data sets that you generate through code, data sets downloaded from online sources, and data sets your programs download automatically.
  After you've completed this project, you'll be able to

write programs that sift through large data sets and make visual representations of that stored information. In the third project you'll build a small web application that allows you to keep a journal of ideas and concepts you've learned about a specific topic. You'll be able to keep separate logs for different topics and allow others to create an account and start their own journals.
- You'll also learn how to deploy your project on Heroku so anyone can access it online from anywhere.

Course Material

Supplied

Course Contents

Day 1

- Python 2 vs Python 3
- Python on different Operating Systems
- Running Python from a Terminal
- Python Variables and Simple Data Types
- Lists
- Working with Lists
- if Statements
- Dictionaries

Day 2

- User Input and while Loops
- Functions
- Classes, Inheritance, Importing Classes
- Python Standard Library
- Files and Exceptions,
- Testing Your Code

Day 3

Project 1: Game Alien Invasion

- Planning phase
- Installation Pygame and more Packages
- Building a game
- Detecting mouse events and hiding the cursor.
- Help button to display instructions on how to play.
- Modify the speed of the game as it progresses
- Building a Spaceship That Fires Bullets
- Watching out for Aliens!
- Display information in textual and non-textual ways.
- Implement a progressive scoring system

Day 4

Project 2: Data Visualization

- Generating Data
- Installing Matplotlib
- Plotting a Line Graph
- Changing the Label Type and Graph Thickness
- Correcting the Plot
- Plotting and Styling Individual Points with scatter()
- Plotting a Series of Points with scatter()
- Calculating Data Automatically
- Removing Outlines from Data Points
- Defining Custom Colors
- Using a Colormap
- Saving Your Plots Automatically
- Random Walks
- Creating the RandomWalk() Class
- Choosing Directions
- Plotting the Random Walk
- Generating Multiple Random Walks
- Styling the Walk
- Coloring the Points
- Plotting the Starting and Ending Points
- Cleaning Up the Axes
- Adding Plot Points
- Altering the Size to Fill the Screen
- Installing Pygal

- Collecting Data
- Analysing Data
- Creating Graphs
- Downloading Data
- The CSV File Format
- Parsing the CSV File Headers
- Printing the Headers and Their Positions
- Extracting and Reading Data
- Plotting Data in a Temperature Chart
- The datetime Module
- Plotting Dates
- Plotting a Longer Timeframe
- Plotting a Second Data Series
- Shading an Area in the Chart
- Mapping Global Data Sets: JSON Format
- Downloading Lots of Data
- Extracting Relevant Data
- Converting Strings into Numerical Values
- Building a Data Map
- Plotting Numerical Data on a Map
- Plotting a Complete Data Map
- Grouping Categories within Categories
- Styling Maps in Pygal
- Lightening the Color Theme
- Using a Web API
- Git and GitHub
- Requesting Data Using an API Call
- Installing Requests
- Processing an API Response
- Working with the Response Dictionary
- Summarizing the Top Repositories
- Monitoring API Rate Limits
- Visualizing Repositories Using Pygal
- Refining Pygal Charts
- Adding Custom Tooltips
- Plotting the Data
- Adding Clickable Links to Our Graph .

- Use APIs to write self-contained programs that automatically gather the data they need and use that data to create a visualization.
- Use GitHub API to explore the most-starred Python projects on GitHub
- Explore the Hacker News API
  Use the requests package to automatically issue API calls to GitHub and process the results of that call.
- Use Pygal settings to further customize the appearance of our generated charts.

Day 5

Project 3: Web Applications

- Getting Started with ***Django***
- Writing a Spec
- Creating a Virtual Environment
- Installing virtualenv
- Activating the Virtual Environment
- Installing Django
- Creating a Project in Django
- Creating the Database
- Viewing the Project
- Defining Models
- Activating Models
- The Django Admin Site
- Defining the Entry Model
- Migrating the Entry Model
- Registering Entry with the Admin Site
- The Django Shell
- Mapping a URL
- Writing a View
- Writing a Template
- Template Inheritance
- The Topics Page
- Individual Topic Pages
- Allowing Users to Enter Data
- Adding New Topics

- Adding New Entries
- Editing Entries
- Setting Up User Accounts
- The users App
- The Login Page
- Logging Out
- The Registration Page
- Blog Accounts
- Allowing Users to Own Their Data
- Restricting Access with @login_required
- Connecting Data to Certain Users
- Restricting Topics Access to Appropriate Users
- Protecting a User’s Topics
- Protecting the edit_entry Page
- Associating New Topics with the Current User
- The django-bootstrap3 App
- Using Bootstrap to Style Learning Log
- Modifying base.html
- Styling the Home Page Using a Jumbotron
- Styling the Login Page
- Styling the new_topic Page
- Styling the Topics Page
- Styling the Entries on the Topic Page
- Other Forms
- Stylish Blog
- Deploying Learning Log
- Making a Heroku Account
- Installing the Heroku Toolbelt
- Installing Required Packages
- Creating a Packages List with a requirements.txt File
- Specifying the Python Runtime
- Modifying settings.py for Heroku
- Making a Procfile to Start Processes
- Modifying wsgi.py for Heroku
- Making a Directory for Static Files
- Using the gunicorn Server Locally
- Using Git to Track the Project’s Files

- Pushing to Heroku
- Setting Up the Database on Heroku
- Refining the Heroku Deployment
- Securing the Live Project
- Committing and Pushing Changes
- Creating Custom Error Pages
- Ongoing Development
- The SECRET_KEY Setting
- Deleting a Project on Heroku
- Competency project

Duration and pricing

- In Pricing Group A

Certificate

About **Our Certificates**

Schedule

On the calendar on this page below.
If your browser doesn't display the calendar below, please click on this link or try using Google Chrome, alternatively please enquire via our 'Contact Us' page.

Bookings

You can download the course registration form on our home page or by clicking here

Brochure

You may download a pdf copy of this page by clicking here.

Questions

Please email us

We are a member of the Python Software Foundation


python
SOFTWARE
FOUNDATION