ExperTeach NETWORKING

# Python for Advanced Learners

## Object Orientation, Tools and Best Practice

**Programming skills are proving to be an increasingly valuable skill, regardless of the application. Python is currently one of the most widely used and sought-after programming languages. If you are already familiar with the basics of Python, this Python training will take you to the next level.**

**You will start by learning the basic elements of object-oriented programming, important building blocks of the standard library and more in-depth software concepts. Software is subject to increasingly strict quality standards. Therefore, we will deal with topics such as Python code design and code analysis tools.**

**Course Contents**

- Basic concepts of object orientation
- Decorators, generators
- Important elements of standard libraries
- Troubleshooting, debugging, log files
- Package and dependency management
- Virtual Python environments
- Static code analysis
- Test automation, test libraries
- GIT integration in Visual Studio Code

E-Book The detailed digital documentation package, consisting of an e-book and PDF, is included in the price of the course.

**Target Group**

The course is aimed at anyone who already has some programming knowledge and now wants to learn more about Python and object-oriented programming and who are facing their first project tasks that they would like to implement with Python.

**Prerequisites**

You will need a solid grounding in Python in terms of data types, functions and loop constructions for this course. These can be acquired, for example, in our course Python for beginners - introduction to programming.

**Course Target**

You will gain an understanding of advanced Python topics in object orientation, as well as functional knowledge of decorators, logging, generators, and an insight into asynchronous programming. In addition, you will acquire knowledge in the design and programming of GUIs.

**This Course in the Web**

You can find the up-to-date information and options for ordering under the following link: www.experteach-training.com/go/**PYFI**

**Reservation**

On our Website, you can reserve a course seat for 7 days free of charge and in an non-committal manner. This can also be done by phone under +49 6074/4868-0.

**Guaranteed Course Dates**

To ensure reliable planning, we are continuously offering a wide range of guaranteed course dates.

**Your Tailor-Made Course!**

We can precisely customize this course to your project and the corresponding requirements.

| Training | | Prices, excl. of V.A.T. |
|---|---|---|
| Classes in Germany | 5 Days | € 2,995 |
| Classes in Austria | 5 Days | € 2,995 |
| Online Training | 5 Days | € 2,995 |

**Date/course venue** — Course language German

| | | | |
|---|---|---|---|
| 26/01-30/01/26 | Frankfurt | 03/08-07/08/26 | Frankfurt |
| 26/01-30/01/26 | Online | 03/08-07/08/26 | Online |
| 23/03-27/03/26 | Düsseldorf | 07/09-11/09/26 | Hamburg |
| 23/03-27/03/26 | Online | 07/09-11/09/26 | Online |
| 04/05-08/05/26 | Online | 12/10-16/10/26 | Düsseldorf |
| 04/05-08/05/26 | Wien | 12/10-16/10/26 | Online |
| 22/06-26/06/26 | München | 23/11-27/11/26 | Online |
| 22/06-26/06/26 | Online | 23/11-27/11/26 | Wien |

Status 11/26/2025

Python for Advanced Learners

ExperTeach

# Table of Contents

## Python for Advanced Learners – Object Orientation, Tools and Best Practice


**1 Visual Studio Code und Jupyter**
1.1 Was ist Visual Studio Code?
1.1.1 Visual Studio Code vs. Visual Studio
1.1.2 Visual Studio Code unter Windows installieren
1.1.3 Optionen während der Installation
1.1.4 Das Python-Extension Pack für VS Code
1.1.5 Sprachunterstützung von VS Code
1.1.6 Visual Studio Code updaten
1.1.7 Die Oberfläche von VS Code
1.1.8 Die Oberfläche von VS Code - Activity Bar und Sidebar
1.1.9 Das Search Tool
1.1.10 Die Git-Leiste
1.1.11 Git in Visual Studio Code nutzen
1.1.12 Ein Remoterepository in VS Code klonen
1.1.13 Ein lokales Git-Repository anlegen
1.1.14 Lokales Repository mit Remoterepository synchronisieren
1.1.15 GitLens
1.1.16 Neue Dateien in VS Code anlegen
1.1.17 Mit einzelnen Codedateien in Visual Studio Code arbeiten
1.1.18 Visual Studio Code einen Ordner hinzufügen
1.1.19 Mit Workspaces in VS Code arbeiten
1.1.20 Struktur von Workspaces
1.1.21 Auf Remote Terminals entwickeln
1.1.22 Debugging in Visual Studio Code
1.1.23 Einen Linter für Python in VS Code nutzen
1.1.24 Linter in Aktion
1.1.25 Python Code in Visual Studio Code testen
1.1.26 Ein Testframework in VSCode aktivieren
1.1.27 Tests erzeugen und durchführen
1.2 Jupyter Notebooks
1.2.1 Interaktive Code-Zellen
1.2.2 Grafische Oberflächen funktionieren
1.2.3 Exporte in andere Formate und Hilfen
1.2.4 Editieren und Ausführen von Zellen
1.2.5 Shortcuts sind im Markdown Modus gefährlich
1.2.6 Dynamische HTML Seiten einbetten
1.2.7 Variable Inspector als Debugger
1.3 Anaconda - Scientific Power Package für ML
1.3.1 Die Anaconda Foren/Learning - Dashboards
1.4 Jupyter Notebooks im VSC

**2 Skriptaufrufe**
2.1 Die Kommandozeilen-Optionen
2.1.1 Module aufrufen python -m
2.1.2 pdb: Der Python Debugger
2.1.3 Das timeit-Modul zur Zeitmessung
2.1.4 Mit JSON-Dateien arbeiten: json.tool
2.1.5 Weitere Beispiele: compileall und tkinter
2.2 Argumente
2.2.1 Die Liste sys.argv
2.2.2 Argumente parsen mit dem argparse-Modul
2.3 Die Shebang Line
2.4 Module und Pakete
2.4.1 Top-level code environment
2.4.2 Verwendung der Variable __name__
2.4.3 Wichtige Dateien in Paketen

**3 Objektorientierte Programmierung**
3.1 Grundprinzipien der OOP
3.1.1 Module und Systeme
3.1.2 Strukturelle Elemente objektorientierter Software
3.1.3 Beziehungen zwischen Objekten
3.1.4 Design Pattern (Entwurfsmuster)
3.2 Objektorientierung und Python
3.2.1 Basisbeispiel
3.2.2 Sichtbarkeiten
3.2.3 Destruktor
3.2.4 Statische Elemente
3.2.5 Mehrfachvererbung

**4 Dekoratoren**
4.1 Einführung
4.2 Benötigte Konzepte
4.2.1 Funktions-Referenzen
4.2.2 Funktionen in Funktionen
4.2.3 Funktionen als Übergabeparameter
4.2.4 Funktionen als Rückgabewert
4.3 Einfache Dekoratoren
4.3.1 Die Wrapper-Funktion
4.3.2 Eine Funktion dekorieren
4.4 Anwendungsfälle von Dekoratoren
4.4.1 Argumentüberprüfung
4.4.2 Funktionsaufrufe zählen
4.4.3 Aufruf einer Funktion zeigen (Logging)
4.5 Dekoratoren mit Übergabeparametern
4.6 Klassen als Dekoratoren
4.6.1 Eine Klasse als Dekorator benutzen
4.6.2 Dekorator-Klassen mit Übergabeparametern
4.7 Typische Dekoratoren und das Modul functools
4.7.1 Functools Dekoratoren

**5 Generatoren und Asynchronität in Python**
5.1 Generatoren in Python 3
5.1.1 Eigene range() programmieren
5.1.2 Generator zip()
5.1.3 range() mit richtiger Argumentreihenfolge
5.2 Generator-Funktionen
5.3 Cooperative multitasking and Coroutines
5.4 Asynchrone Code-Ausführung

**6 Unittest, Doctest**
6.1 Unittest
6.1.1 unittest - Die Klasse TestCase
6.1.2 assert Methoden
6.1.3 setUp und tearDown Methoden
6.1.4 Testen von Fehlermeldungen
6.1.5 Überspringen und erwartete Fehlschläge
6.2 doctest - Docstrings für Tests nutzen
6.2.1 Einfache Verwendung von doctest
6.2.2 Die Ausgabe von doctest
6.2.3 Testen der Methoden einer Klasse
6.2.4 Tests in einer Textdatei
6.3 Test Driven Development (TDD) – Tests zuerst
6.4 Static Code Analysis
6.4.1 Linting mit pylint
6.4.2 Type Checks mit mypy

**7 Eigene Exceptions und Logging**
7.1 Exceptions in Python
7.1.1 Hierarchie der Builtin Exceptions (Ausschnitt)
7.1.2 Eigene Exceptions definieren
7.1.3 Exception Chaining
7.2 Logging
7.2.1 Modul Logging
7.2.2 Modul Loguru

**8 Virtuelle Umgebungen**
8.1 Abhängigkeiten von Modulen
8.2 Das Konzept einer virtuellen Umgebung
8.3 Das Tool venv

**9 Eigene Packages und der PyPI**
9.1 Angeleitete Übung zur Einführung von Packages
9.1.1 Relative Importe - Schritt 2
9.1.2 Eigenes Paket mit __init__.py - Schritt 3
9.2 Eigene Packages auf PyPI veröffentlichen
9.2.1 setup.py
9.2.2 Erstellung der Distribution
9.2.3 Auf PyPI veröffentlichen
9.2.4 Wichtige Zusatzinformationen

**10 Code Distribution**
10.1 Distribution mit Python Setuptools
10.2 Das Distributions Problem
10.3 Distribution mit PyInstaller
10.3.1 Vorbereitung und Build-Erzeugnisse
10.3.2 Funktionsweise und One-File
10.3.3 Kommandozeilenargumente für PyInstaller
10.3.4 PyInstaller mit Python-Code ausführen

**11 GUI mit TKinter und PyQt5**
11.1 GUI mit Tkinter
11.2 Die Alles-In-Einem-File-Variante
11.3 PyGubu – Ein Wysiwyg - Editor für Tkinter
11.4 GUI mit PyQt5
11.4.1 PyQt5, PySide2 und Lizenzierung
11.4.2 Erster Start von PyQt5-Designer
11.4.3 Die Oberfläche von Qt-Designer
11.4.4 Layouts in Qt Designer
11.4.5 PyQt5-Designs in Python-Code laden
11.4.6 Buttons und Methoden verknüpfen



**ExperTeach Benelux B.V.**
Ceresstraat 1· 4811 CA Breda· Phone: +49 6074 4868-0 · Fax: +49 6074 4868-109 · info@experteach.de · www.experteach-training.com