Python für Fortgeschrittene

Python sicherer, effizienter und gezielter einsetzen

01 INTRO

Wer die Grundlagen von Python beherrscht, kann Programme schreiben. Im nächsten Schritt geht es darum, die Sprache selbst besser zu verstehen und ihre Möglichkeiten gezielter zu nutzen.

Der Kurs vertieft zentrale Python-Konzepte und zeigt Techniken, mit denen sich Programme kompakter, flexibler und robuster entwickeln lassen. Dabei geht es auch um typische Python-Idiome und darum, vorhandenen Code besser zu verstehen und weiterzuentwickeln.

DAUER

3 Tage

FORMAT

Präsenz · Live Online · Inhouse

VORKENNTNISSE

Sichere Python-Grundkenntnisse

02 TRAININGSPROGRAMM

Inhalte

Python-Datenmodell

- Objekte, Identität und Typen
- veränderliche und unveränderliche Objekte
- Referenzen und Kopien
- Gleichheit und Identität
- Truth Values
- wichtige Special Methods

Fortgeschrittene Datenstrukturen

- List-, Set- und Dictionary-Comprehensions
- verschachtelte Datenstrukturen
- collections
- defaultdict, Counter, deque
- Sortieren mit key
- Packing und Unpacking

Funktionen als Objekte

- Funktionen als First-Class Objects
- variable Argumentlisten
- Keyword-only Arguments
- Closures
- Lambdas sinnvoll einsetzen
- Funktionen als Parameter und Rückgabewerte

Iteratoren und Generatoren

- Iterator-Protokoll
- eigene Iteratoren
- Generatorfunktionen
- Generator Expressions
- yield und yield from
- speichereffiziente Verarbeitung von Datenströmen

Decorators und Context Manager

- Funktionen dekorieren
- Decorators mit Parametern
- functools.wraps
- Context-Manager-Protokoll
- eigene Context Manager
- contextlib

Module, Packages und Imports

- Python-Importmechanismus
- absolute und relative Imports
- Packages strukturieren
- __name__ und __main__
- Sichtbarkeit und öffentliche Schnittstellen
- Abhängigkeiten zwischen Modulen

Fehlerbehandlung und robuste Programme

- Exception-Hierarchie
- eigene Exceptions
- Exception Chaining
- else und finally
- Ressourcen sicher verwalten
- Fehler behandeln oder weiterreichen?

Typisierung und moderne Python-Werkzeuge

- Type Hints vertiefen
- Collections und generische Typen
- Optional, Union und moderne Syntax
- Protocol und strukturelle Typisierung im Überblick
- statische Typprüfung
- Type Hints als Dokumentation und Entwicklungswerkzeug

Python-Werkzeugkasten

- itertools
- functools
- operator
- pathlib
- ausgewählte Funktionen der Standardbibliothek
- wann Standardbibliothek statt eigener Lösung?

Performance verstehen

- messen statt vermuten
- grundlegendes Profiling
- Zeit- und Speicherverhalten
- Generatoren vs. materialisierte Collections
- typische Performance-Fallen
- wann Optimierung überhaupt sinnvoll ist

03 DURCHFÜHRUNG

Rahmen und Organisation

Dauer	3 Tage
Durchführung	Präsenz oder Live Online, vorzugsweise als Inhouse-Schulung
Schwerpunkt	Inhouse-Schulungen für Unternehmen und Teams
Sprache	Deutsch, englische Fachbegriffe und Dokumentation
Unterlagen	Kursunterlagen, Beispiele und Übungsaufgaben
Technik	Eigener Rechner mit aktueller Python-Version und Entwicklungsumgebung

> **Inhouse-Schulungen**
> Die Schwerpunkte können an vorhandene Kenntnisse, eingesetzte Technologien und konkrete Anforderungen des Teams angepasst werden.

04 ZIELGRUPPE

Für wen ist der Kurs gedacht?

Der Kurs richtet sich an Entwicklerinnen und Entwickler, die bereits mit Python arbeiten und über die grundlegenden Sprachkenntnisse hinausgehen möchten.

Er eignet sich insbesondere für Teilnehmer, die vorhandenen Python-Code besser verstehen, die Möglichkeiten der Sprache gezielter einsetzen oder ihre bisher überwiegend pragmatisch erworbenen Python-Kenntnisse systematisieren möchten.

Die Teilnehmer sollten mit folgenden Themen sicher umgehen können:

- Variablen und grundlegende Datentypen
- Listen, Dictionaries, Sets und Tupel
- Bedingungen und Schleifen
- Funktionen
- Module
- grundlegende Fehlerbehandlung
- einfache Klassen und Objekte

05 LERNZIELE

Was Sie nach dem Kurs können

Nach dem Kurs können die Teilnehmer:

- fortgeschrittene Sprachkonzepte von Python gezielt einsetzen
- Iteratoren und Generatoren für effiziente Datenverarbeitung verwenden
- Funktionen, Closures und Decorators sicherer einsetzen
- Context Manager verstehen und selbst implementieren
- komplexere Datenstrukturen idiomatisch verarbeiten
- Packages und Module sinnvoll strukturieren
- Type Hints gezielt zur Verbesserung von Code und Schnittstellen einsetzen
- Fehlerbehandlung bewusster gestalten
- wichtige Werkzeuge der Python-Standardbibliothek nutzen
- Performance-Probleme erkennen und systematisch untersuchen

06 FAQ

Häufige Fragen

Ist PY-002 eine Fortsetzung von PY-001?

Inhaltlich ja, organisatorisch nicht zwingend. Wer die Voraussetzungen bereits erfüllt, kann direkt mit PY-002 einsteigen.

Wie viel Programmiererfahrung ist erforderlich?

Die grundlegende Python-Syntax sollte sicher beherrscht werden. Allgemeine Entwicklungserfahrung ist hilfreich, wichtiger ist jedoch praktische Erfahrung mit Python.

Wird objektorientierte Programmierung behandelt?

Python-spezifische Aspekte des Objektmodells werden behandelt. Für eine systematische Vertiefung von OOP ist PY-003 „Objektorientierte Programmierung mit Python“ vorgesehen.

Geht es auch um Testing, FastAPI oder Datenbanken?

Nur soweit diese Themen zur Erläuterung eines Python-Konzepts sinnvoll sind. Sie werden in eigenen Kursen ausführlich behandelt.

Kann der Kurs an unser Team angepasst werden?

Ja. Themen können vertieft, gekürzt oder mit passenden Inhalten aus anderen Python-Kursen kombiniert werden.

UC IT Service · Ulrich Cuber · kontakt@uc-it.de
https://www.uc-it.de/coaching/python-fortgeschritten/