Woche 6: Fortgeschrittene OOP

Einleitung

Herzlich willkommen zur sechsten Woche Ihrer Python-Reise! Sie haben bereits eine beeindruckende Menge an Wissen und Fähigkeiten mit Python erworben.

In dieser Woche liegt der Fokus auf einigen fortgeschrittenen Konzepten der objektorientierten Programmierung, insbesondere auf der Vererbung und deren Einsatz in Python. Die Vererbung ist ein mächtiges Werkzeug, das es ermöglicht, den Code effizienter und lesbarer zu gestalten, indem die Strukturen und Funktionen, die bereits erstellt wurden, wiederverwendet und erweitert werden.

Im Verlauf dieser Woche wird das Erstellen von Unterklassen, das Aufrufen und Überschreiben geerbter Methoden sowie die Nutzung spezieller Funktionen namens "Dunder"-Methoden erlernt. Auch das Konzept der Abstraktion durch abstrakte Klassen und Methoden wird behandelt und es wird gezeigt, wie mit Schnittstellen und Protokollen gearbeitet werden kann.

Die Themen dieser Woche können herausfordernd sein, aber denken Sie daran: Niemand ist allein auf dieser Reise. Nutzen Sie die zur Verfügung stehenden Ressourcen, stellen Sie Fragen, wenn Sie welche haben, und vor allem: Bleiben Sie neugierig und offen für neue Ideen.

Die bevorstehenden Übungen helfen dabei, das Gelernte zu festigen und in der Praxis anzuwenden. Sie sind eine großartige Gelegenheit, um die Konzepte, die erlernt werden, wirklich zu verstehen und zu verinnerlichen.

Es erwartet Sie also eine weitere aufregende Woche voller Python.

Gesamtüberblick

Hier ein Überblick über die Inhalte und Aktivitäten der aktuellen Woche:

- Selbststudium:
    - Vererbung und Erstellen von Unterklasse
    - Aufrufe von geerbten Methoden
    - Überschreiben von Methoden und Polymorphie
    - Abstrakte Klassen und Methoden
    - Schnittstellen und Protokolle
    - Dunder-Methoden
- Aufgabe:
    - Anlage einer Vererbungshierarchie
    - Überladen von Methoden
    - Einsatz von Dunder-Methoden
- Übungstag:
    - Wiederholung
    - Vertiefung: Klassenhierarchien
    - Vertiefung: Überladung
    - Ergänzung: Mehrfachvererbung
    - Ergänzung: Mixin-Klassen
    - Ergänzung: Dependency-Injection und Delegating
    - Ausblick

Inhalte und thematische Abgrenzung

Die folgende Auflistung zeigt detailliert, welche Themen Sie in der Woche behandeln und bearbeiten. Sie sind eine Voraussetzung für die folgenden Wochen und sollten gut verstanden worden sein. Wenn es Verständnisprobleme gibt, machen Sie sich Notizen und fragen Sie am Präsenztag nach, so dass wir gemeinsam zu Lösungen kommen können. Und denken Sie bitte immer daran: es gibt keine „dummen“ Fragen!

- Vererbung
    - Verstehen der Grundprinzipien der Vererbung
    - Erstellen von Unterklasse
    - Arbeiten mit geerbten Methoden
    - Überschreiben von Methoden in einer Unterklasse (Polymorphie)
- Abstrakte Klassen und Methoden
    - Erstellen von abstrakten Klassen und Methoden
    - Verstehen, wie Abstraktion in der OOP verwendet wird
- Schnittstellen und Protokolle
    - Verstehen, was Schnittstellen und Protokolle sind
    - Arbeiten mit Schnittstellen und Protokollen in Python
- Dunder-Methoden
    - Einführung in Dunder-Methoden (auch als magische Methoden bezeichnet)
    - Arbeiten mit Dunder-Methoden wie init, str, len, etc.
    - Verstehen, wie und wann Dunder-Methoden verwendet werden
- Mehrfachvererbung
    - Verstehen der Grundprinzipien der Mehrfachvererbung
    - Arbeiten mit Mehrfachvererbung in Python
- Mixin-Klassen
    - Verstehen, was Mixin-Klassen sind
    - Arbeiten mit Mixin-Klassen in Python
- Dependency-Injection und Delegating
    - Verstehen der Konzepte Dependency-Injection und Delegating
    - Anwenden von Dependency-Injection und Delegating in Python

Lernpfad

Der Lernpfad ist ein Vorschlag, in welcher Reihenfolge Sie die Inhalte der Woche angehen können. Betrachten Sie ihn gerne als eine Todo-Liste, die Sie von oben nach unten abhaken. So können Sie sicher sein, dass Sie alle wichtigen Themen bearbeitet haben und sind gut vorbereitet für die folgenden Wochen.

1. Einführung in die Vererbung

- Entdecken Sie, was Vererbung ist und warum sie in der objektorientierten Programmierung so wertvoll ist.
- Betrachten Sie einfache Beispiele für Vererbung in Python und versuchen Sie, diese zu erweitern oder zu modifizieren.

2. Erstellen einer Unterklasse und Verstehen der Hierarchie

- Lernen Sie, wie man in Python eine Unterklasse erstellt.
- Verstehen Sie die Beziehung zwischen einer Elternklasse und einer Unterklasse und die Bedeutung der Vererbungshierarchie.

3. Vererbte Methoden

- Erfahren Sie, wie man Methoden aus der Elternklasse in der Unterklasse aufruft und verwendet.
- Üben Sie das Aufrufen von geerbten Methoden mit verschiedenen Arten von Unterklasse-Objekten.

4. Überschreiben von Methoden und Polymorphie

- Lernen Sie, wie und wann Methoden überschrieben werden sollten, um die Funktionalität einer geerbten Methode zu ändern.
- Entdecken Sie das Konzept der Polymorphie und wie es sich auf das Überschreiben von Methoden auswirkt.

5. Einführung in abstrakte Klassen und Methoden

- Erfahren Sie, was abstrakte Klassen und Methoden sind
- Lernen Sie, wie sie in Python implementiert werden.
- Erkunden Sie die Vorteile der Verwendung von Abstraktion in Ihrer Programmierarbeit.

6. Arbeiten mit Schnittstellen und Protokollen

- Lernen Sie, was Schnittstellen und Protokolle sind
- Erfahren Sie wie man diese Konzepte auf Python übertragen kann und wie sie in Python verwendet werden.
- Üben Sie das Erstellen und Implementieren von Schnittstellen und Protokollen in Ihren eigenen Klassen.

7. Verstehen und Anwenden von Dunder-Methoden

- Einführung in die sogenannten "Dunder"-Methoden in Python.
- Erkunden Sie, wie diese Methoden das Verhalten Ihrer Klassen und Objekte anpassen können.

8. Übungsaufgaben

- Entwerfen und Implementieren Sie Klassenhierarchien
- Implementieren Sie Dunder-Methoden in Ihren Klassen

9. Selbstbewertung

- Multiple-Choice-Bewertung absolvieren Wichtige Konzepte und Fähigkeiten überprüfen
- Schwierige Themen gegebenenfalls erneut durcharbeiten

10. Online-Schulungstag

Programmieraufgaben

Die folgenden Programmieraufgaben sollen Ihnen eine Anregung geben. Haben Sie eigene Ideen und Themen, die Sie ausprobieren wollen, dann sollten Sie diesen nachgehen. Wichtig ist vor allem, dass Sie „Dinge ausprobieren“. Und auch, dass Sie Fehler machen, sowohl syntaktische als auch semantische. Versuchen Sie diese Fehler zu finden und aufzulösen, dann gerade aus den Fehlern lernen Sie am Ende am meisten.

1. Erstellen einer Vererbungshierarchie:

    Sie sind dazu aufgefordert, eine einfache Vererbungshierarchie zu erstellen. Beginnen Sie mit einer Basisklasse Fahrzeug und erstellen Sie Unterklasse Auto und Motorrad. Fügen Sie den Klassen geeignete Attribute und Methoden hinzu.

2. Überschreiben von Methoden:

    Unter Verwendung der Fahrzeugklassenhierarchie aus der vorherigen Aufgabe, überschreiben Sie eine Methode in einer der Unterklassen. Sie könnten beispielsweise die Methode starten() in der Klasse Auto überschreiben, um eine spezielle Nachricht auszugeben, wenn das Auto gestartet wird.

3. Arbeiten mit Dunder-Methoden:

    Erstellen Sie eine Klasse Buch mit Attributen wie titel und autor. Überschreiben Sie die Dunder-Methode __str__() in dieser Klasse, um eine formatierte Zeichenkette zurückzugeben, die die Buchinformationen enthält, wenn eine Instanz der Klasse in eine Zeichenkette umgewandelt wird.

4. Verwendung von abstrakten Klassen und Methoden:

    Erstellen Sie eine abstrakte Klasse Form mit einer abstrakten Methode berechneFlaeche(). Erstellen Sie dann zwei Unterklasse Kreis und Rechteck, die diese Methode implementieren.

5. Schnittstellen und Protokolle:

    Definieren Sie ein Protokoll Laufbar mit den Methoden starten() und stoppen(). Implementieren Sie dieses Protokoll in den Klassen Auto und Motorrad aus der ersten Aufgabe.

Abschluss-Quiz

Das Quiz soll Ihnen einen ersten Hinweis auf Ihren Lernfortschritt geben. Nach unserer Einschätzung sollten Sie diese Fragen alle beantworten können, wenn Sie den Stoff der Woche durchgearbeitet und verstanden haben. Natürlich gibt es noch sehr viel mehr mögliche Fragen, dazu wollen wir auf die Literatur und das Internet verweisen. Geben Sie gerne einmal „python quizzes“ bei Google ein.

1. Was ist der Zweck von Vererbung in der objektorientierten Programmierung?

a) Es ermöglicht die Wiederverwendung von Code und die Modellierung von realen
b) Beziehungen.
c) Es hilft beim Debugging von Programmen.
d) Es ist eine Methode, um Daten in einer Datenbank zu speichern.
e) Es beschleunigt die Ausführung von Programmen.

2. Was ist eine abstrakte Methode in Python?

a) Eine Methode, die in einer abstrakten Klasse definiert und implementiert ist.
b) Eine Methode, die nur in der Theorie existiert und nicht in der Praxis implementiert
c) werden kann.
d) Eine Methode, die in einer abstrakten Klasse definiert, aber nicht implementiert ist.
e) Eine Methode, die in jeder Klasse definiert ist.

3. Was ist ein Protokoll in Python?

a) Eine Vereinbarung, die die Regeln für die Kommunikation zwischen Objekten
b) definiert.
c) Eine Möglichkeit, die Netzwerkkommunikation zu kontrollieren.
d) Eine Methode zur Fehlerbehebung in Python-Programmen.
e) Ein spezieller Typ von Python-Funktion.

4. Was bedeutet es, eine Methode zu überschreiben?

a) Die Methode in einer Unterklasse mit dem gleichen Namen, aber mit einer anderen
b) Implementierung zu definieren.
c) Die Methode in der gleichen Klasse mit einem anderen Namen zu definieren.
d) Den Rückgabewert einer Methode zu ändern.
e) Die Parameter einer Methode zu ändern.

5. Was sind Dunder-Methoden in Python?

a) Methoden, die nur in Unterklasse vorhanden sind.
b) Methoden, die zwei Unterstriche vor und nach dem Methodennamen haben und
c) spezielle Eigenschaften definieren.3. Methoden, die während der Laufzeit des Programms erstellt werden.
d) Methoden, die nur in abstrakten Klassen vorhanden sind.

6. Was ist der Hauptvorteil von Mehrfachvererbung in Python?

a) Es ermöglicht die gleichzeitige Vererbung von Eigenschaften und Methoden aus mehreren Klassen.
b) Es reduziert die Anzahl der benötigten Klassen in einem Programm.
c) Es macht den Code einfacher zu verstehen.

a) Es verbessert die Leistung des Programms.

7. Was ist eine Mixin-Klasse in Python?

a) Eine Klasse, die Methoden von mehreren anderen Klassen erbt.
b) Eine Klasse, die dazu dient, eine bestimmte Funktionalität zu vermitteln, die für verschiedene Klassen nützlich sein kann.
c) Eine Klasse, die ausschließlich abstrakte Methoden enthält.
d) Eine Klasse, die nur statische Methoden enthält.

8. Was ist Dependency Injection in Python?

a) Ein Designmuster, das die Kopplung von Code verringert und die Testbarkeit verbessert.
b) Eine Methode zum Hinzufügen neuer Abhängigkeiten zu einem bestehenden PythonProjekt.
c) Eine Technik zum Überladen von Methoden.
d) Ein Prozess zur Verbesserung der Leistung von Python-Programmen.

9. Was bedeutet es, eine Methode in Python zu überladen?

a) Eine Methode mit demselben Namen, aber mit unterschiedlichen Parametertypen oder -anzahl zu definieren.
b) Eine Methode in einer Unterklasse mit dem gleichen Namen, aber mit einer anderen Implementierung zu definieren.
c) Eine Methode in der gleichen Klasse mit einem anderen Namen zu definieren.
d) Den Rückgabewert einer Methode zu ändern.

10. Was ist das Hauptziel beim Erstellen einer Vererbungshierarchie in Python?

a) Die Wiederverwendung von Code zu maximieren und eine strukturierte und lesbare Codestruktur zu schaffen.
b) Die Anzahl der erstellten Klassen zu minimieren.
c) Die Leistung des Programms zu verbessern.
d) Die Notwendigkeit von Funktionen und Prozeduren zu eliminieren.

Ressourcen

Hier nun die Verweise auf Lernquellen, die uns für diese Woche und ihre Inhalte geeignet erscheinen. Je nachdem, welcher Lerntyp Sie sind, wählen Sie sich ihre bevorzugte Quelle, es ist nicht zwingend notwendig alle durchgearbeitet zu haben. Allerdings sollten die Inhalte des Lernpfads angesprochen und erstanden worden sein.

- **Buch**: Python Crash Course:

  Klassen: Dieses Thema wird in "Kapitel 9: Klassen" behandelt.

- Video:

  Codecademy: Learn Python 3 – Ein umfassender Kurs mit 27h Dauer

- Lab:

  Python for Developers – Grundlegendes Python inklusive Datenstrukturen