Maier, Sindel, Zeitler
October 21, 2024

Python Recap

Introduction

The purpose of this exercise is to refresh your knowledge of Python and object oriented programming. We will implement a simple game using python classes and functions and inheritance.

Der Zweck dieser Übung ist es, Ihr Wissen über Python und objektorientierte Programmierung aufzufrischen. Wir werden ein einfaches Spiel implementieren, das Python-Klassen, Funktionen und Vererbung verwendet.

First, let's set up the environment. We recommend you to use the Anaconda distribution for both Windows and Ubuntu. As IDE you can use your favorite one, e.g. PyCharm, Spyder, or Visual Studio Code.

Zuerst richten wir die Umgebung ein. Wir empfehlen Ihnen, die Anaconda-Distribution sowohl für Windows als auch für Ubuntu zu verwenden. Als IDE können Sie Ihre Lieblingsanwendung verwenden, z.B. PyCharm, Spyder oder Visual Studio Code.

Here is an example how to set up an Anaconda environment with the required packages:
Hier ist ein Beispiel, wie Sie eine Anaconda-Umgebung mit den erforderlichen Paketen einrichten können:

```
conda create -n pygame pip python=3.8
conda activate pygame
conda install -c anaconda spyder==5.4.3
pip install pygame
```

Optional: Install Spyder in the Anaconda environment:
Optional: Installieren Sie Spyder in der Anaconda-Umgebung:

```
conda install -c anaconda spyder==5.4.3
```

To run the code on command line:
Zum Ausführen des Codes in der Befehlszeile:

```
python game_exercise.py
```

If you run the code in an IDE, select "execute in an external system terminal".

Wenn Sie den Code in einer IDE ausführen, wählen Sie "In einem externen Systemterminal ausführen".

You will see a blue rectangle which you can move with the right and left arrow keys. However, here in this game, the challenge is missing. In the following tasks, you will add "traffic" objects to the screen, such that the "player" has to try to avoid a collision with them, otherwise it's game over.
Sie werden ein blaues Rechteck sehen, das Sie mit den Pfeiltasten rechts und links bewegen können. Allerdings fehlt hier in diesem Spiel die Herausforderung. In den folgenden Aufgaben werden Sie "Verkehrs"-Objekte auf dem Bildschirm hinzufügen, sodass der "Spieler" versuchen muss, eine Kollision mit ihnen zu vermeiden, sonst ist das Spiel vorbei.

1 First task

Your task will be to implement a "traffic" class that simulates oncoming traffic, which the player has to evade. Have a look at the "player" class. The "traffic" class should have a similar structure.
Ihre Aufgabe besteht darin, eine "Verkehr" Klasse zu implementieren, die den entgegenkommenden Verkehr simuliert, den der Spieler vermeiden muss. Schauen Sie sich die "Spieler" Klasse an. Die "Verkehr" Klasse sollte eine ähnliche Struktur haben.

Task:

- The "traffic" class should provide the following methods: a constructor **__init__()**, a method **move()** which moves the traffic item from the top of the screen to the bottom
  *Die "Verkehr" Klasse sollte die folgenden Methoden bereitstellen: einen Konstruktor **__init__()**, eine Methode **move()**, die das Verkehrsobjekt vom oberen Rand des Bildschirms zum unteren Rand bewegt*

- In the constructor initialize the **width** with 60, **height** with 100, **velocity** with 20 and the **color** with the color red
  *Initialisieren Sie im Konstruktor die **Breite** mit 60, die **Höhe** mit 100, die **Geschwindigkeit** mit 20 und die **Farbe** mit der Farbe rot*

- Reset the traffic object to the top once it has reached the button and speed up its movement in a set increment
  Setzen Sie das Verkehrsobjekt wieder an den oberen Rand, sobald es den unteren Rand erreicht hat, und erhöhen Sie seine Bewegungsgeschwindigkeit in einer festgelegten Inkrement

2 Second task

Just one traffic object is little fun as well. Let's create multiple ones (at least two).
Nur ein Verkehrsobjekt macht wenig Spaß. Erstellen Sie mehrere (mindestens zwei).

Task:

- Rewrite your code so that you have one **parent class** "traffic" as well as two **child classes** "car" and "truck"
  *Schreiben Sie Ihren Code so um, dass Sie eine **Elternklasse** "Verkehr" sowie zwei **Unterklassen** "Auto" und "Lastwagen" haben*
- The children **inherit** the previously defined **move()** function from the parent
  *Die Unterklassen **erben** die zuvor definierte **move()** Funktion von der Elternklasse*
- Initialize each child with different width, height, velocity, and color
  Initialisieren Sie jedes Kind mit unterschiedlicher Breite, Höhe, Geschwindigkeit und Farbe
- Adjust the **run()** function, so that one car and one truck object is created and used as oncoming traffic
  *Passen Sie die **run()** Funktion an, sodass ein Auto- und ein Lastwagenobjekt erstellt und als Gegenverkehr verwendet werden*
- Optional: Add more objects, such as "bus", "motorcycle", or "bike"
  Optional: Fügen Sie weitere Objekte hinzu, wie z.B. "Bus", "Motorrad" oder "Fahrrad"

3 Some useful links

- Anaconda: Anaconda cheat sheet
- Python tutorials: Python, Kaggle
- Pygame: Official Pygame documentation, Pygame Tutorial