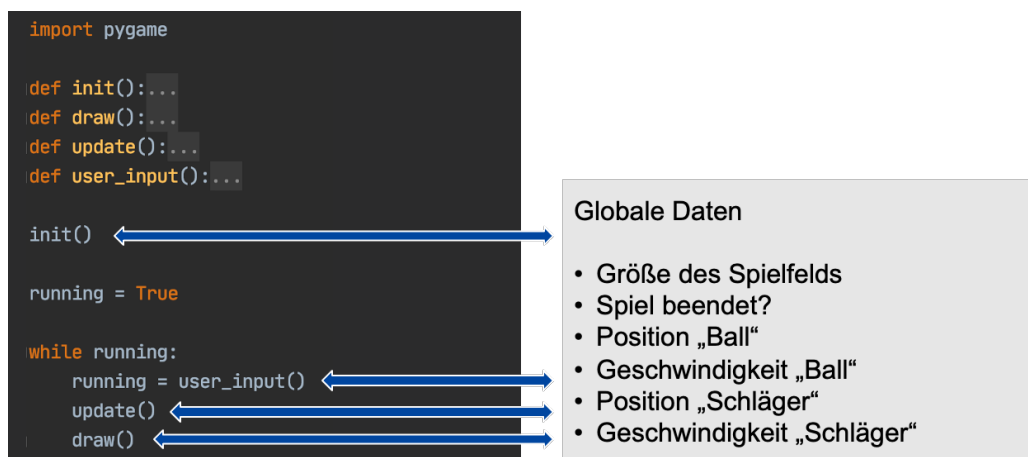


Arbeitsblatt

„Pong - objektorientiert“

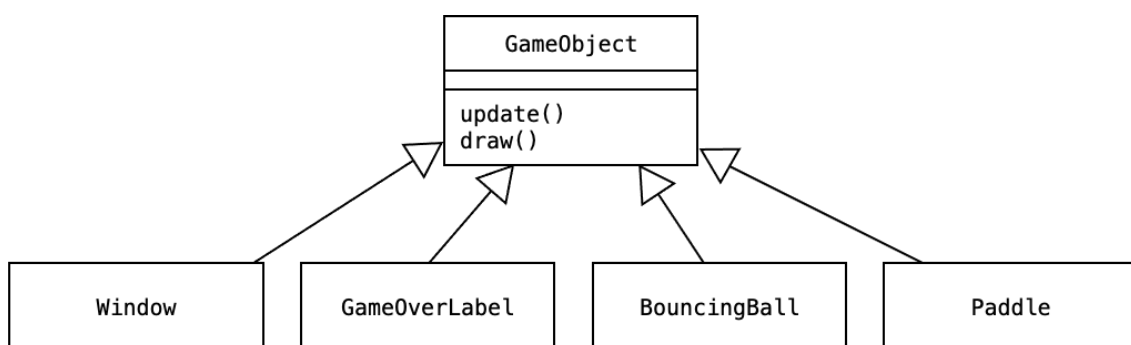
Beim klassischen Pong mit `pygame` haben wir das Problem, dass wir den gesamten Spielzustand zwischen den Funktionen `init`, `user_input`, `update` und `draw` austauschen müssen. Das kann z.B. mit globalen Variablen gemacht werden, allerdings wird das bei größeren Spielen schnell unübersichtlich.



Ein alternativer Ansatz wäre, die relevanten Spielelemente (Spielfeld, Ball, Schläger) in Form von Python-Objekten zu realisieren, die

- die eigenen Daten (Position, Geschwindigkeit, etc.) als Attribute mitführen
- mit eigenen `update`- und `draw`-Methoden das individuelle Verhalten und Aussehen des Objekts sicherstellen

Dazu wurde in einer unvollständigen Rumpf-Implementierung eine Oberklasse `GameObject` erstellt, von der alle relevanten Spielelemente abgeleitet werden sollen:



Die Unterklassen von `GameObject` sollen folgende Funktion wahrnehmen:

- Die Klasse `Window` repräsentiert das Fenster, in dem das Spiel abläuft. Attribute sind u.a. die Größe des Fensters und die Fensterüberschrift. Die `draw`-Funktion soll dafür sorgen, dass der Fensterhintergrund schwarz ist.
- Die Klasse `GameOverLabel` ist für die Darstellung des Textes "Game Over" verantwortlich. In einem Attribut wird außerdem vermerkt, ob das Spiel bereits beendet ist.
- Die Klasse `BouncingBall` soll das wandernde Ohm-Zeichen mit allen Attributen (Position, Geschwindigkeit) realisieren. In der Methode `update` muss die neue Position bestimmt und ggf. das Abprallen von den Wänden oder vom Schläger berücksichtigt werden. `Draw` ist selbstverständlich für die Darstellung des Zeichens zuständig.
- Die Klasse `Paddle` ist die Repräsentation des Schlägers mit seinen Attributen. `Update` und `draw` sind dementsprechend zu realisieren.

Instanzen dieser Klassen werden in der `main`-Funktion im Dictionary `game_objects` vorgehalten, das anschließend in der Game Loop an die bekannten Funktionen `user_input`, `update` und `draw` übergeben wird. Dort werden die spezifischen Funktionen der Spielelemente aufgerufen.

Sie erhalten im eLearning-Kurs eine rudimentäre Implementierung dieser Ideen und sind nun mit der – durchaus realitätsnahen – Herausforderung konfrontiert, diesen fremden Code zu lesen, zu verstehen und zu erweitern.

Aufgabe 1

Einige der genannten Klassen sind nur bruchstückhaft implementiert - zu erkennen am Kommentar

```
# Hier müssen Sie Code ergänzen
```

Vervollständigen Sie bitte diese Klassen, so dass das bekannte Spiel entsteht.

Aufgabe 2

Implementieren Sie als ein weiteres Spielelement einen Counter, der in der oberen linken Ecke angezeigt wird. Er beginnt mit dem Wert 0 und wird mit jedem Zusammentreffen des Schlägers und des Balls um 10 erhöht. Zeigen Sie den Wert auch noch an, wenn die "Game-Over"-Meldung dargestellt wird.