



PORTFOLIOPRÜFUNG ALTERNATIV 3 SCHIFFE VERSENKEN LIVE-DEMONSTRATION

Amadeo Granillo
Juan Carlos Moreno
Abdul Karim Kanbar

WARUM HABEN WIR DIESE PROJEKTALTERNATIV GEWÄHLT?

- Kombination aus Programmiersprachen- und Spielauswahl
- Thema von Interesse für die Zukunft
- Grafische Darstellung lernen

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4			X							
5						X	X			
6		X						X		X
7				X						X

Foto aus: <https://gesellschaftsspiele.spielen.de/alle-brettspiele/schiffe-versenken1/>

WARUM PYTHON, UND NICHT BASH ODER C?

- Python ist leicht erlernbar
- Der Python-Code gilt als besonders gut lesbar.
- Python benötigt keine Blockklammern wie z.B. die geschweiften Klammern in Sprache C
- Python benötigt kein Semikolon am Zeilenende, wenn nur eine Anweisung in der Zeile steht
- Python eignet sich für fast alle Anwendungsprobleme
- Python ist eine Skriptsprache und kann in Anwendersoftware eingebunden werden.
- Python wird ständig weiterentwickelt



PLAYER VS CPU

- Initialisierung durch Boardsgröße
- Hauptmenü zur Auswahl des Spielmodus
- Positionierung der Schiffe von Spieler 1 (CPU macht das automatisch selbst mit random Bibliothek)
- **Spiel zwischen Spieler und CPU**
- Der Gewinner des Spiels wird angezeigt

-Verfahren für Spielverfahren vs CPU:

```
def start_cpu_spiel(gewinnen):
    gewinnen.clear()

    spieler1 = Spieler("Spieler 1")
    spieler2 = Spieler("CPU")
    spieler2.ist_cpu = True #Definieren, dass der gegnerische Spieler die CPU ist

    spieler1.schiffe_platzieren(gewinnen) #Platzieren von Schiffe auf dem Board
    gewinnen.clear()
    h, w = gewinnen.getmaxyx()
    msg = spieler1.name + "! Ihre Schiffe wurden platziert. Drücken Sie eine beliebige Taste, um fortzufahren."
    gewinnen.addstr(h//2, w//2 - len(msg)//2, msg) #Positionierung der Nachricht im Terminal
    gewinnen.refresh()

    gewinnen.getch()
    for type in SCHIFF_TYPEN:
        spieler2.platz_ship_random(type) #Da es gegen den PC geht, werden die Schiffe zufällig platziert
    h, w = gewinnen.getmaxyx()
    gewinnen.clear()
    msg = spieler2.name + " ihre Schiffe platziert hat. Drücken Sie eine beliebige Taste, um fortzufahren."
    gewinnen.addstr(h//2, w//2 - len(msg)//2, msg) #Positionierung der Nachricht im Terminal

    gewinnen.refresh()

    gewinnen.getch()
```

- Wenn der Mehrspielermodus im Hauptmenü ausgewählt wird, wird die Funktion „start_freund_spiel“ angerufen und das Programm erwartet Input von zwei Benutzern.
- Mit der Klasse „Spieler“ können die verschiedenen Benutzer unterschieden werden.
- Die Positionierung der Schiffe von Spieler 2 findet nach der Positionierung der Schiffe von Spieler 1 statt.
- Alle Prozesse werden in einer Schleife wiederholt.
- Eine Meldung wird nach jeder Runde gezeigt, so dass jeder Spieler nur sein eigenes Spielfeld sehen kann.

PLAYER VS PLAYER

```
def start_freund_spiel(gewinnen):
    gewinnen.clear()
    # Initialisieren von Spielern. Hier können Spielernamen als Benutzereingabe genommen werden
    spieler1 = Spieler("Spieler 1")
    spieler2 = Spieler("Spieler 2")

    #Schiffe für Spieler 1 platzieren
    spieler1.schiffe_platzieren(gewinnen)
    gewinnen.clear()
    h, w = gewinnen.getmaxyx()
    msg = spieler1.name + " ! Ihre Schiffe wurden platziert. Drücken Sie eine beliebige Taste,
    gewinnen.addstr(h//2, w//2 - len(msg)//2, msg)
    gewinnen.refresh()

    gewinnen.getch()
    |
    # Schiffe für Spieler 2 platzieren
    spieler2.schiffe_platzieren(gewinnen)
    gewinnen.clear()
    h, w = gewinnen.getmaxyx()
    msg = spieler2.name + " ! Ihre Schiffe sind platziert worden. Drücken Sie eine beliebige Ta
    gewinnen.addstr(h//2, w//2 - len(msg)//2, msg)
    gewinnen.refresh()
    gewinnen.getch()

    gameloop(spieler1, spieler2, gewinnen) # Starten der Spielschleife
```

DANKE FÜR IHRE
AUFMERKSAMKEIT