PyMemory

Projektbeschreibung

Im Spiel PyMemory geht es darum, sich verschiedene Karten zu merken. Es gibt zu jedem Bild ein Paar. Das Spiel besteht aus einem quadratischen Spielfeld, die Karten sind verdeckt hinzulegen. Eine Runde besteht aus dem zufälligen Umdrehen von zwei verdeckten Karten. Falls diese das gleiche Bild besitzen, so bleiben sie aufgedeckt. Wenn alle Karten aufgedeckt geblieben sind, gilt das Spiel als beendet.

Projektanforderungen

In-Scope:

- GUI mit Spielfläche
- Anklickbare Karten
- Rundenbasiertes Spiel
- Spiellogik
- Spielstatistiken

Out-of-Scope

- Multiplayer
- Animationen
- Benutzerdefinierte Karten

Ist-Zustand

Es wurden folgende Anforderungen umgesetzt:

In-Scope:

- GUI mit Spielfläche
- Anklickbare Karten
- Rundenbasiertes Spiel
- Spiellogik
- Spielstatistiken

Out-of-Scope

- Benutzerdefinierte Karten

Das Projekt wurde objektorientiert aufgebaut.

Die Datei „stats.json“ enthält die Highscores der jeweiligen Spieler.

Es existiert ein Hauptmenu in dem der Spielername gewählt werden kann.

Außerdem kann der Hintergrund der Karten verändert werden. Es gibt 4 unterschiedliche Arten von Hintergründen:

1. Normal: Es werden vordefinierte Bilder von Umweltgegenständen als Hintergrund verwendet
2. Logos: Es werden vordefinierte Bilder von Logos von bekannten Programmiersprachen verwendet.
3. Gemischt: Die in 1. und 2. genannten Hintergründe werden gemischt und auf die Spielgröße zufällig angepasst.
4. Benutzerdefiniert: Der Benutzer kann eigene Bilder im .jpg Format im Verzeichnis „images/custom“ abspeichern. Diese Bilder müssen die Auflösung 60x60 Pixel besitzen. Es werden aus dem gegebenen Ordner zufällig so viele Bilder verwendet, wie in die angegebene Größe des Spielfelds passen.

Nicht umgesetzt wurden verschiedene Variationen von Spielgrößen; derzeit wird nur 4x4 unterstützt. Dies ist jedoch leicht erweiterbar.

Außerdem wird der Highscore von dem im Textfeld „Name“ angegebenen Spieler im Label „Highscore“ als Prozentzahl ausgegeben.

Die Klasse PlayerStatistics implementiert die Logik, um verschiedene Highscores der Spieler in einem Dictionary zu speichern. Dazu hat es die Funktionalität, diese aus einer Datei zu laden oder in einer Datei zu speichern.

Die Klasse StateManager speichert die Zustände der Karten. Falls eine Karte aufgedeckt ist, wird dies in einer Liste gespeichert. Falls beide Karten nun in dieser Liste sind, wird die Liste geleert und die Karten, falls der Hintergrund nicht gleich ist, auf dem Bildschirm wieder verdeckt.

Die Klasse Tile repräsentiert die Karte als Objekt.

In der Klasse MemoryGame ist die Spiellogik verankert, dort wird das Hauptmenu gesteuert, das Spiel gestartet und verwaltet und die Karten/Tiles vorbereitet und verwaltet. Außerdem ist die Klasse für das aufrufen der Methoden zum selektieren, aufdecken und verdecken in einem Tile Objekt verantwortlich. Außerdem ist die Klasse dafür zuständig, dass das Spiel runden-basiert abläuft.

Um eine Karte umzudrehen, muss sie angeklickt werden. Wenn zwei Karten ausgewählt wurden, wird die Menge der ausgewählten Karten zurückgesetzt und es können 2 weitere Karten ausgewählt werden. Falls die Karten die gleichen Hintergründe haben, bleiben sie umgedreht, ansonsten werden die Karten wieder umgedeckt. Wenn alle Karten aufgedeckt sind, ist das Spiel gewonnen und die Zeit, welche benötigt wurde, wird angezeigt. Falls ein neuer Highscore erreicht wurde, wird dieser auch ausgegeben.

Ausnahmebehandlung

Es werden Assertions im Programm aufgestellt um die Korrektheit des Spielfelds zu garantiert. Falls diese Assertions nicht erfüllt werden, wird das Programm einen Laufzeitfehler werfen und das Programm wird unterbrochen.

Es wurde eine Exception-Klasse „TileException“ entworfen um Spiel-interne Ausnahmen abzufangen.

Verwendete Bibliotheken

Im Projekt wurden folgende externen Bibliotheken verwendet:

- pygame
- pygame-menu
- easygui

Pygame wurde verwendet um den Aufwand zur Erstellung eines Grundgerüsts zu reduzieren. Dadurch konnten die Karten/Tiles leichter auf dem Bildschirm angezeigt werden.

PyGame-Menu wurde verwendet, um ein einfaches, kompaktes und vor allem fehlerfreies Hauptmenu aufzubauen. Dadurch konnten die Spieleinstellungen einfach eingetragen werden und pro Aktionsknopf verschiedene Aktionen zugewiesen werden.

EasyGui wurde verwendet um eine Benutzernachricht bei einem erfolgreich beendeten Spiel zu hinterlassen. Dies war zwingend notwendig, da PyGame standardmäßig keine Möglichkeit zulässt, eine Benutzerinformation in einem zusätzlichen Fenster anzuzeigen.

Zusammenfassende Betrachtung

Die In-Scope Anforderungen wurden erfüllt und es wurde ein Element der Out-of-Scope Anforderungen erfüllt. Das Projekt ist erfolgreich und zeitlich fristgerecht abgelaufen.

Installation/Betriebsanleitung/Zielplatformen

Das Spiel wurde in Python 3.8 entwickelt und getestet. Eine Abwärtskompatibilität ist nicht gewährleistet. Das Spiel wurde auf Windows 10 Pro und Fedora 32 getestet.

(Derzeit nicht möglich da pip in Docker die Abhängigkeiten nicht herunterlädt.)
Um das Projekt mit Docker zu starten:

```
docker build . --tag pymemory
docker run pymemory
```

Um das Projekt selbst zu starten:

Windows:

(Wenn pip nicht installiert ist, kann pip über get-pip.py installiert werden)

```
pip install -r requirements.txt
python PyMemory.py
```

Red Hat Distributionen (Fedora/CentOS):

```
python3 -m pip install pyperclip
dnf install python3-pygame python3-tkinter
pip install -r requirements.txt
python PyMemory.py
```