Elias Kupeczki

10/02/2023

Modul 335

CsBe – Unity Projekt

Inhalsverzeichnis

[Projektaufbauorganisation 3](#_Toc126848204)

[Zeitplan 3](#_Toc126848205)

[Arbeitsjournal 4](#_Toc126848206)

[Tag 1 4](#_Toc126848207)

[Tag 2 5](#_Toc126848208)

[Aufgabenstellung 6](#_Toc126848209)

[Standards 6](#_Toc126848210)

[Schutzbedarfsanalyse 6](#_Toc126848211)

[Organisation der Arbeitsergebnisse 6](#_Toc126848212)

[Testkonzept 6](#_Toc126848213)

[Testfälle 6](#_Toc126848214)

[Durchführung und Auswertung der Tests 7](#_Toc126848215)

[White Box 7](#_Toc126848216)

[Black Box 7](#_Toc126848217)

[Anwendung der Fachsprache 9](#_Toc126848218)

[Deployment-Ziele 9](#_Toc126848219)

[Software-Ergonomie 9](#_Toc126848220)

[Design 10](#_Toc126848221)

[Scene 10](#_Toc126848222)

[Camera 10](#_Toc126848223)

[Menu 10](#_Toc126848224)

[Game Over 10](#_Toc126848225)

[Border 10](#_Toc126848226)

[Player 11](#_Toc126848227)

[Controls 11](#_Toc126848228)

[Collision 11](#_Toc126848229)

[Ball 11](#_Toc126848230)

[Direction 11](#_Toc126848231)

[Collision 11](#_Toc126848232)

[Enemy 11](#_Toc126848233)

[Collision 11](#_Toc126848234)

[Benutzerfreundlichkeit 11](#_Toc126848235)

[Grafiken, Bilder, Diagramme und Tabellen 12](#_Toc126848236)

[Storyboard 13](#_Toc126848237)

[Arbeitsergebnisse 13](#_Toc126848238)

# Projektaufbauorganisation

# Zeitplan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **06-Feb** | **07-Feb** | **08-Feb** | **09-Feb** | **10-Feb** |
|  |
| **Planen** | **0.5** | **0.5** |  |  |  |
| **Docum-entation  Schreiben** |  | **2** | **1** | **1** | **1** |
| **Scene erstellen** |  | **1** |  |  |  |
| **C# Classes erstellen** |  | **2** | **2** | **2** |  |
| **Testing** |  |  | **3** |  |  |
| **GUI  erstellen** |  | **2** | **2** |  |  |
| **zusatz funktionen** |  | **1** | **1** |  |  |

# Arbeitsjournal

### Tag 1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **06-Feb** | | **07-Feb** | | **08-Feb** | | **09-Feb** | | **10-Feb** | |
|  | SOLL | IST | SOLL | IST | SOLL | IST | SOLL | IST | SOLL | IST |
| **Planen** | **0.5** | **1** | **0.5** | **0.5** |  |  |  |  |  |  |
| **Docum-entation  Schreiben** |  |  | **2** | **3** |  |  |  |  |  |  |
| **Scene erstellen** |  |  |  |  |  |  |  |  |  |  |
| **C# Classes erstellen** |  |  |  |  |  |  |  |  |  |  |
| **Testing** |  |  |  |  |  |  |  |  |  |  |
| **GUI  erstellen** |  |  |  |  |  |  |  |  |  |  |
| **zusatz funktionen** |  |  |  |  |  |  |  |  |  |  |
| **Reflektion** | **Heute habe ich an der Planung für die Erstellung eines Pong-Spiels in Unity gearbeitet. Ich begann mit der Erstellung eines Plans für das Spiel, einschließlich der Regeln, Ziele und des Gesamtdesigns. Als Nächstes sammelte ich die Assets, die ich für die Erstellung des Spiels benötigte, z. B. Sprites, Hintergründe und Sounds. Danach habe ich an der Spiellogik gearbeitet, einschließlich des Punktesystems und der Spielmechanik. Schließlich testete ich das Spiel, um sicherzustellen, dass es richtig funktionierte. Es war eine Menge Arbeit, aber ich freue mich schon darauf, das Endergebnis zu sehen!** | | | | | | | | | |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

### Tag 2

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **06-Feb** | | **07-Feb** | | **08-Feb** | | **09-Feb** | | **10-Feb** | |
|  | SOLL | IST | SOLL | IST | SOLL | IST | SOLL | IST | SOLL | IST |
| **Planen** |  |  |  |  |  |  |  |  |  |  |
| **Docum-entation** |  |  |  |  |  |  |  |  |  |  |
| **Schreiben** |
| **Scene** |  |  | **1** | **1** | **0** | **0.5** |  |  |  |  |
| **erstellen** |
| **C# Classes** |  |  | **2** | **2** | **2** | **2** |  |  |  |  |
| **erstellen** |
| **Testing** |  |  |  |  |  |  |  |  |  |  |
| **GUI** |  |  | **2** | **2** | **2** | **2** |  |  |  |  |
| **erstellen** |
| **zusatz** |  |  |  |  |  |  |  |  |  |  |
| **funktionen** |
| **Reflektion** | **Die Entwicklung von Pong in Unity war für mich eine Herausforderung, da ich zum ersten Mal C# und Unity verwendet habe. Ich war mit der Syntax und der Struktur des Codes nicht vertraut, aber er hatte ähnliche Komponenten wie Java. Ich musste lernen, wie ich mich in der Unity-Benutzeroberfläche zurechtfinde, die neu war, aber der Unreal Engine 5 ähnelte (mit der ich schon einige Erfahrung habe). Ich musste lernen, wie man Objekte erstellt, wie man Spiellogik erstellt und wie man sie mit der visuellen Schnittstelle verbindet.Anfangs fand ich es schwierig zu visualisieren, wie mein Code mit dem Geschehen auf dem Bildschirm zusammenhängt, weil ich es gewohnt war, den Ablauf meines Codes in Unreal mit den "Blueprints" visuell darzustellen. Ich musste viel recherchieren und experimentieren, um den Code und die Benutzeroberfläche zum Funktionieren zu bringen.Schließlich habe ich die Schwierigkeiten bei der Entwicklung von Pong in Unity überwunden, indem ich Schritt für Schritt vorgegangen bin und aus meinen Fehlern gelernt habe. Ich musste auch lernen, wie man den Code an die visuelle Schnittstelle bindet und wie man Fehler oder Bugs behebt, auf die ich stieß.** | | | | | | | | | |
|  |
|  |

# Aufgabenstellung

# Standards

# Schutzbedarfsanalyse

# Organisation der Arbeitsergebnisse

# Testkonzept

# 

Das Ziel des Testkonzepts ist es, bestimmte Fehler im Unity Game zu finden und zu dokumentieren zu dokumentieren und nach einem standardisierten und strukturierten Verfahren zu beheben. Der Test dieser Anwendung besteht aus funktionalen und nicht-funktionalen Testfällen.

# Testfälle

Dieses Testkonzept ist für das zu testende System, ein Pong-Spiel, das in Unity unter Verwendung von C# entwickelt wurde. Die relevanten Testfälle basieren auf der Kernmechanik des Spiels, wie der Bewegung des Balls, der Bewegung des Paddels und der Kollisionserkennung. Die erwarteten Ergebnisse für diese Testfälle sind, dass sich das Spiel wie erwartet verhält, dass sich der Ball und das Paddel bewegen und auf Benutzereingaben reagieren, und dass das Kollisionserkennungssystem genau registriert, wenn der Ball entweder das Paddel oder die Wand trifft.

Das Testkonzept prüft absichtlich nicht die Gesamtleistung des Spiels, wie z. B. die Bildrate und die Latenzzeit. Dies ist darauf zurückzuführen, dass das Testen dieser Elemente den Zugang zu spezieller Hardware erfordern würde, was nur im Rahmen von White-Box-Tests möglich sein wird.

Zu den relevanten Testmitteln für dieses Testkonzept gehören ein Computer, auf dem Unity und das Pong-Spiel laufen, sowie eine Maus, eine Tastatur und ein Touch für Benutzereingaben. Die Testmethode wird eine Kombination aus manuellen Tests sein, bei denen der Tester das Spiel spielt.

# Durchführung und Auswertung der Tests

## White Box

## 

Ich werde das Spiel mehrmals spielen, um sicherzustellen, dass die Spielmechanik und die Regeln korrekt umgesetzt werden. Ich werde das Spiel mit verschiedenen Schwierigkeitsgraden und Eingaben testen, um sicherzustellen, dass das Spiel richtig reagiert. Ich werde das Spiel auch mit verschiedenen Kombinationen von Eingaben testen, um sicherzustellen, dass der Ball richtig von den Schlägern, Wänden und anderen Objekten abprallt. Ich überprüfe den Spielstand, um sicherzustellen, dass er korrekt ist und nach jeder Runde aktualisiert wird. Ich stelle sicher, dass alle Spielregeln und Ziele korrekt befolgt werden. Ich überprüfe auch, ob alle Spielelemente korrekt und in der richtigen Reihenfolge, mit den richtigen Farben und Texturen gerendert werden. Schließlich werde ich das Spiel testen, um sicherzustellen, dass alle Menüs und Bildschirme richtig funktionieren.

## Black Box

Dazu wird das Spiel manuell gespielt, um sicherzustellen, dass es korrekt funktioniert, die visuellen Komponenten des Spiels getestet, um zu gewährleisten, dass alle Grafiken korrekt angezeigt werden, die Benutzeroberfläche getestet, um sicherzustellen, dass alle Schaltflächen und Menüs korrekt funktionieren, die Leistung des Spiels auf verschiedenen Geräten getestet und die Funktionen zum Speichern und Laden des Spiels getestet, um sicherzustellen, dass sie korrekt funktionieren.

Test 7: Überprüfen der Spielschwierigkeit

Beschreibung: Überprüfen Sie, ob der Schwierigkeitsgrad des Spiels mit zunehmender Punktzahl steigt.

Schritte: Starten Sie das Spiel, spielen Sie, bis der Punktestand steigt, und beobachten Sie den Schwierigkeitsgrad des Spiels.

Erwartetes Ergebnis: Der Schwierigkeitsgrad des Spiels sollte mit zunehmender Punktzahl steigen.

Test 8: Überprüfen der Spielsteuerung

Beschreibung: Überprüfen Sie, ob die Spielsteuerung korrekt funktioniert.

Schritte: Starten Sie das Spiel, drücken Sie die Spielsteuerung und beobachten Sie die Ergebnisse

Erwartetes Ergebnis: Die Spielsteuerung sollte korrekt funktionieren.

Test 9: Prüfen Sie das Punktelimit

Beschreibung: Überprüfen Sie, ob das Punktelimit erreicht ist, wenn das Spiel endet.

Schritte: Starten Sie das Spiel und spielen Sie, bis das Punktelimit erreicht ist.

Erwartetes Ergebnis: Das Spiel sollte enden, wenn das Punktelimit erreicht ist.

Test 10: Überprüfen der Spielpausenfunktionalität

Beschreibung: Prüfen, ob das Spiel pausiert werden kann

Schritte: Starten Sie das Spiel, drücken Sie die Pausentaste und beobachten Sie das Ergebnis

Erwartetes Ergebnis: Das Spiel sollte pausiert werden, wenn die Pausentaste gedrückt wird.

|  |  |
| --- | --- |
| Testfall: 01 | Bezug: Überprüfen der Ballbewegung |
| Ablauf | Starten Sie das Spiel und beobachten Sie die Bewegung des Balls. |
| Erwartetes Ergebnis | Der Ball sollte sich im Spiel bewegen. |

|  |  |
| --- | --- |
| Testfall: 02 | Bezug: Überprüfen der Punkteerhöhung |
| Ablauf | Starten Sie das Spiel, beobachten Sie den Punktezähler |
| Erwartetes Ergebnis | Die Punkte sollten sich erhöhen, wenn der Ball die Paddles trifft. |

|  |  |
| --- | --- |
| Testfall: 03 | Bezug: Überprüfen der Wandkollision |
| Ablauf | Starten Sie das Spiel, beobachten Sie die Ballkollision |
| Erwartetes Ergebnis | Der Ball sollte von den Wänden abprallen. |

|  |  |
| --- | --- |
| Testfall: 04 | Bezug: Überprüfen der Soundeffekte |
| Ablauf | Starten Sie das Spiel, hören Sie auf Soundeffekte |
| Erwartetes Ergebnis | Die Soundeffekte sollten abgespielt werden, wenn der Ball die Schläger und Wände trifft. |

|  |  |
| --- | --- |
| Testfall: 05 | Bezug: Überprüfen der Ballgeschwindigkeit |
| Ablauf | Starten Sie das Spiel und beobachten Sie die Geschwindigkeit des Balls. |
| Erwartetes Ergebnis | Die Geschwindigkeit des Balls sollte sich erhöhen, wenn er auf die Paddles trifft. |

|  |  |
| --- | --- |
| Testfall: 06 | Bezug: Überprüfen Sie das Zurücksetzen des Spielstands |
| Ablauf | Starten Sie das Spiel, starten Sie das Spiel neu und beobachten Sie den Spielstand. |
| Erwartetes Ergebnis | Der Spielstand sollte zurückgesetzt werden, wenn das Spiel neu gestartet wird. |

|  |  |
| --- | --- |
| Testfall: 07 | Bezug: |
| Ablauf |  |
| Erwartetes Ergebnis |  |

# Anwendung der Fachsprache

# Deployment-Ziele

Android only exclusive

# Software-Ergonomie

# Design

General Design

The Pong game will only be produced for Mobile so that I don’t need to deal with tablets and their different sizes

The Mobile app will be vertical only as it this will limit the chance of problems occurring from rotation and also having Pong being top down would not work as well on a landscape screen as the distance between the players will be limited compared to the height of a vertical screen.

Basic design and when not using really basic stuff I will be getting it from the Unity Asset Market. This is because they will provide high quality assets for free. Unity Market assets for the walls and Player this is cause I am not skilled at design.

Diagram

Description automatically generated

## Scene

### Camera

### Menu

### Game Over

### Border

## Player

if (Input.GetKey(KeyCode.A)){

transform.Translate(Vector3.left \* Time.deltaTime \* 10);

        }



### Controls

The user is given the ability to move left and right. This is done using (example right) to make the player go left and is then changed to go right by swapping “left” with “right” and (KeyCode.A) to (KeyCode.D). this is all done in the Update() Function so that every frame (60 times every second) the check to see if the User is pressing that key will be checked.

### Collision

if (collision.gameObject.tag == "ball")

        {

            rigidbody.AddForce(new Vector2(0, 1000));

        }



## Ball

### Direction

### Collision

## Enemy

### Collision

# Benutzerfreundlichkeit

I used the Classic Menu and Game Design so that the user is familiar with my game if they have played any games before hand. Such as in the menu “Play Game” Followed underneath by “Exit”. Within the Game Score will be kept in the the corner with a label and a upto date score of the user. If they haven’t it will still be clear to understand.

# Grafiken, Bilder, Diagramme und Tabellen

Es werden an vernünftigen Stellen Grafiken, Bilder, Diagramme  
oder Tabellen eingesetzt, um die Inhalte im IPA-Bericht besser  
darzustellen und den Text verständlicher zu machen;  
2. Die Wahl der Darstellungen ist durchgehend geeignet;  
3. Die Darstellung ist kontrastreich und optisch gut lesbar (als  
Referenz dient der Ausdruck auf Format A4);  
4. Die Darstellungen sind inhaltlich verständlich;  
5. Die Darstellungen sind aussagekräftig;  
6. Die Darstellungen sind im Text oder in einer Legende erklärt;  
7. Die Darstellungen passen zum Kontext.

A screenshot of a computer

Description automatically generated with medium confidenceText

Description automatically generated

# Storyboard

Diagram

Description automatically generated with medium confidence

# Arbeitsergebnisse

Die Arbeitsergebnisse (schliesst die Dokumente ein) sind ihrem  
Entwicklungsstand angemessen versioniert.  
2. Auf die Versionen der Arbeitsergebnisse kann jederzeit  
zugegriffen werden.  
3. Es besteht eine tägliche Sicherung der Arbeitsergebnisse.  
4. Die Wiederherstellung der gesicherten Arbeitsergebnisse ist  
sichergestellt.  
5. Punkte 1-4 sind im IPA-Bericht beschrieben und nachvollziehbar

Works Cited

Ansimuz. “Sunny Land | 2D Characters | Unity Asset Store.” *Assetstore.unity.com*, 10 Nov. 2022, assetstore.unity.com/packages/2d/characters/sunny-land-103349. Accessed 8 Feb. 2023.

Manuel Sollberger, ”Canvas Helper” *CanvasHelper.cs,* *Project 335\Assets\Scripts\UI\CanvasHelper.cs, 10 Feb. 2023*

Manuel Sollberger, ”Safe Area”, SafeArea*.cs,* *Project 335\Assets\Scripts\UI\* *SafeArea.cs,10 Feb. 2023*

Pathway. “Night Mountain Lake [SEAMLESS] | 2D Environments | Unity Asset Store.” *Assetstore.unity.com*, 3 2019, assetstore.unity.com/packages/2d/environments/night-mountain-lake-seamless-127703. Accessed 8 Feb. 2023.

Worlds, Tiny. “Free Pixel Font - Thaleah | 2D Fonts | Unity Asset Store.” *Assetstore.unity.com*, 5 Apr. 2019, assetstore.unity.com/packages/2d/fonts/free-pixel-font-thaleah-140059. Accessed 8 Feb. 2023.

‌