|  |
| --- |
| B:IB: international college |
| Platform Fighter |
| Lernaufgabe 2013 |
|  |
| **Markus Baars und Fabian Meise** |
| **25.09.2013** |

|  |
| --- |
|  |

Inhalt

[1.Pflichtenheft 1](#_Toc368034019)

[1.1Aufgabenstellung Einführung 1](#_Toc368034020)

[1.1.1 Einführung 1](#_Toc368034021)

[1](#_Toc368034022)

[1.1.2 Aufgabenstellung 2](#_Toc368034023)

[1.2 Produktumgebung 2](#_Toc368034024)

[1.2.1 Anwendungsbereiche 2](#_Toc368034025)

[1.2.2 Anwendergruppen 2](#_Toc368034026)

[1.2.3 Basismaschine 2](#_Toc368034027)

[1.2.4 Mengengerüst 2](#_Toc368034028)

[1.3 Produktmodell 3](#_Toc368034029)

[1.3.1 Anwendungsfälle der Software 4](#_Toc368034030)

[1.3.2 Ablaufanalyse 5](#_Toc368034031)

[1.4.Qualitätsanforderungen 5](#_Toc368034032)

[1.5.Benutzerschnittstelle 6](#_Toc368034033)

[1.5.1 Benutzermodell 6](#_Toc368034034)

[1.5.2 Kommunikationsstrategie 6](#_Toc368034035)

[1.5.3 Kommunikationsaufbau 7](#_Toc368034036)

[1.6.Entwicklungsumgebung 7](#_Toc368034037)

[1.7.Literaturverzheichnis 7](#_Toc368034038)

[2.Entwurfsdokumentation 8](#_Toc368034039)

[2.1 Softwarearchitektur 8](#_Toc368034040)

[2.1.1 Technische Infrastruktur 8](#_Toc368034041)

[2.1.2 Packetentwurf 8](#_Toc368034042)

[2.2 Paketbeschreibung 11](#_Toc368034043)

[2.3 Datenstrukturen 12](#_Toc368034044)

[2.4 Sonstiges 12](#_Toc368034045)

[3.Implementierungsdokumentation 13](#_Toc368034046)

[3.1 Ablauf der Implementierung 13](#_Toc368034047)

[3.1.2 Probleme 13](#_Toc368034048)

[3.2 Quellen 14](#_Toc368034049)

[3.2.1 Skriptreferenz 14](#_Toc368034050)

[3.2.2.1 Benutzte Grafiken 14](#_Toc368034051)

[4.Testdokumentation 15](#_Toc368034052)

[4.1 Testplan 15](#_Toc368034053)

[4.1.1 Teststrategien der Teststufen 15](#_Toc368034054)

[4.1.2 Testumgebung 15](#_Toc368034055)

[4.1.3 Testwerkzeuge 15](#_Toc368034056)

[4.1.4 Testdaten 15](#_Toc368034057)

[4.2 Testfälle 15](#_Toc368034058)

[4.2.1 Funktionale Tests 15](#_Toc368034059)

[4.3 Testprotokolle 16](#_Toc368034060)

[5.Benutzerdokumentation 17](#_Toc368034061)

[5.1 Über dieses Handbuch 17](#_Toc368034062)

[5.1.1 Zielgruppe 17](#_Toc368034063)

[5.1.2 Gebrauch des Handbuches 17](#_Toc368034064)

[5.2 Einleitung 17](#_Toc368034065)

[5.3 Produktbeschreibung 17](#_Toc368034066)

[5.4 Bedienungsanleitung 18](#_Toc368034067)

[5.4.1 Steuerung 18](#_Toc368034068)

[5.4.2 Benutzeroberfläche 20](#_Toc368034069)

[5.5 Beispielanwendung 22](#_Toc368034070)

[5.6 Glossar 22](#_Toc368034071)

[6.Installationshandbuch 23](#_Toc368034072)

[6.1 Inhaltsverzeichnis 23](#_Toc368034073)

[6.2 Beschreibung der Dateien 23](#_Toc368034074)

[6.3 Beschreibung der Installation 23](#_Toc368034075)

[6.4 Glossar 23](#_Toc368034076)

[7. Fazit 24](#_Toc368034077)

# 1.Pflichtenheft

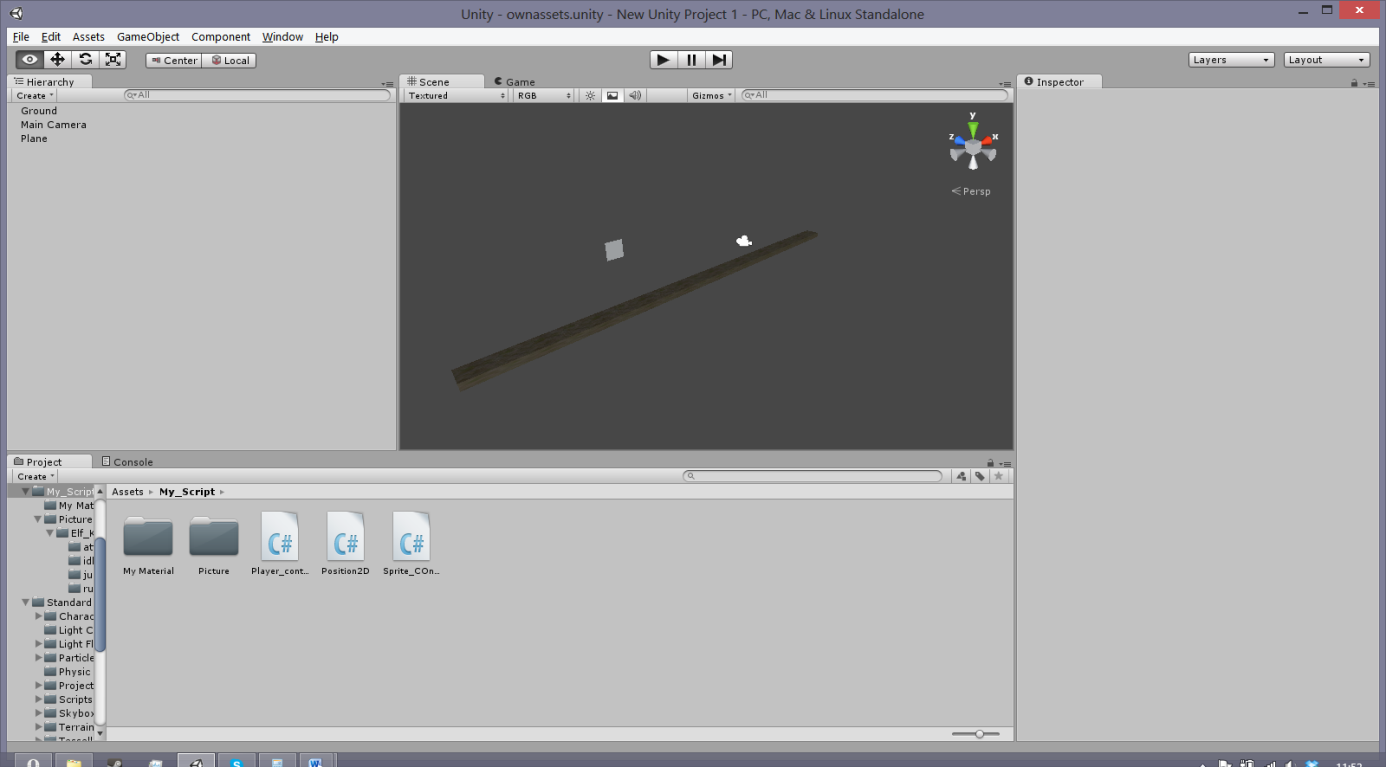
## 1.1Aufgabenstellung Einführung

### 1.1.1 Einführung

Im Rahmen der Ausbildung findet ein 2 wöchiges, frei wählbares Projekt statt, welches in Teams bearbeitet wird. Wir haben uns für ein Softwareprojekt entschieden. Dabei handelt es sich um ein Computerspiel. Um uns mit der Materie der Spieleentwicklung genauer zu bescäftigen, haben wir uns dazu entschlossen das Spiel mit Hilfe der Spiele-Engine Unity3d zu entwickeln. Die Verwendung dieser kostenlosen Spiele-Engine bietet verschiedene Vorteile. Zum einen wird uns eine Menge Arbeit abgenommen, durch eine umfrangreiche Bibliothek müssen wir nicht ganz von Anfang an beginnen. Zum anderen bekommen wir einen Einblick und eine Weiterbildung in der Spiel-Engine, welche sonst im späteren Ausbildungsverlauf behandelt wird.

Unity3d ist relativ simpel aufgebaut:

GUI



Eigenschaften von Objekten

Skripte

Spielobjekte

### 1.1.2 Aufgabenstellung

Wir wollen ein Spiel entwickeln, welches zu zweit spielbar ist. Hierzu wird das Gamedesign erstellt und die Programmierung erledigt. Unsere Aufgabe besteht darin alle Bausteine so zusammen zu setzen, dass unser Spiel entsteht. Fehlende Bausteine müssen wir selber noch erstellen. Die von uns verwendeten Grafiken werden passend bearbeitet.

## 1.2 Produktumgebung

### 1.2.1 Anwendungsbereiche

Da es sich bei unserem Projekt um keine Nutzsoftware sondern eine Unterhaltungssoftware handelt, beschränkt sich die Nutzung auf zu Hause.

### 1.2.2 Anwendergruppen

Die Anwender der Software sind alle Computerspielinteressierte. Da es sich bei dem Spiel um ein Beat Em Up handelt, würde es in die Kategorie ab 12 Jahren freigegeben des §14 des Jugendschutzgesetzes fallen und so die Nutzergruppe einschränken. Die vorausgesetzten Kenntnisse des Benutzers sind minimal, er muss lediglich das Spiel starten können und mit dem XBOX Controller vertraut sein.

### 1.2.3 Basismaschine

Als Basismaschine des Spiels dienen ein Windows-PC und 2 XBOX Controller. Ziel ist es das unser Spiel auch auf älteren PC lauffähig ist.

### 1.2.4 Mengengerüst

Wir streben einen Speicherverbrauch von etwa 300MB an.

## 1.3 Produktmodell

MUSS-Kriterien:

Als grundlegendste Funktion sollte das Spiel startbar und beendbar sein. Das Starten und beenden findet in einem Startbildschirm statt. Jedem Spieler steht ein XBOX Controller zur Verfügung, womit der Charakter gesteuert wird. Der Spieler kann so den Charakter bewegen oder Schläge ausführen. Wenn der Spieler den Gegenspieler schlägt zieht er im so Leben ab und kann ihn so besiegen. Der Charakter kann auf Plattformen springen und sich auf der Fläche bewegen. Im Laufe der Zeit fallen Items vom Himmel, diese können durch das herüberlaufen aufgesammelt werden. Die Items heilen bzw. regenerieren Leben sowie Ausdauer. Ausdauer wird zum Blocken, Springen und ausführen von stärkeren Angriffen benötigt. Der Charakter besteht aus einer animierten Sprite, diese besitzt passende Animationen für alle Bewegungsabläufe. Damit die Charaktere nicht war los durch die Luft fliegen, muss der Boden eine Gravitation besitzen. Während einer Spielrunde läuft eine dezente Hintergrundmusik ab.

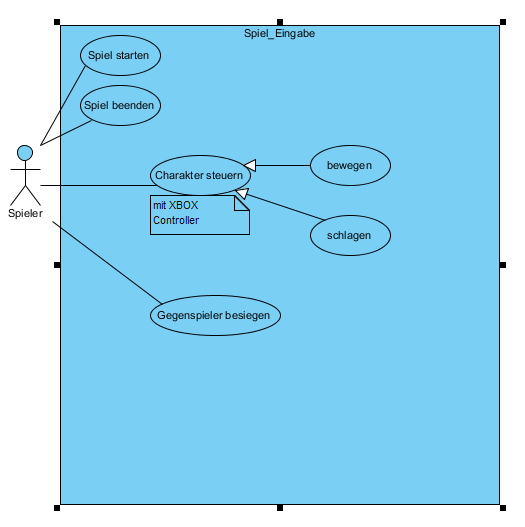
KANN-Kriterien:

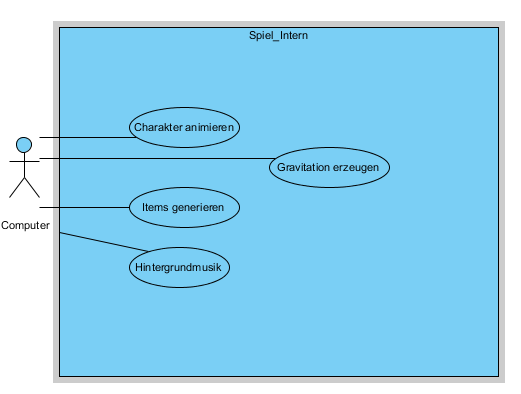
Ein echtes Nice-to-have wäre eine Fader wodurch unterschiedliche Hintergründe möglich wären.

Außerdem würden auch Charaktersounds das Spielgefühl verbessern.

Als weiteres Bewegungsfeature wäre ein Ausweichschritt ein Ergänzens Wert.

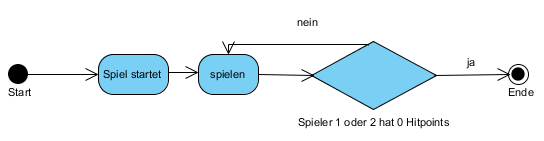
### 1.3.1 Anwendungsfälle der Software



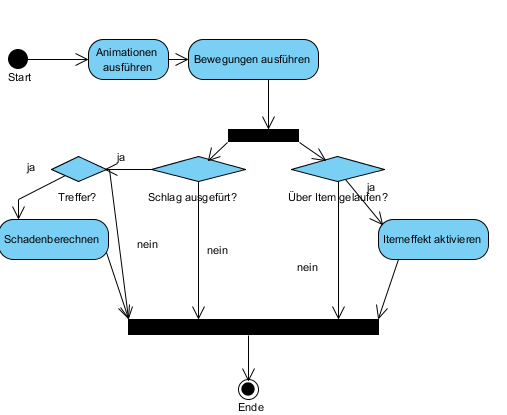


### 1.3.2 Ablaufanalyse

Aktivität: Spielrunde



Aktivität: Spielen



## 1.4.Qualitätsanforderungen

Wir wollen ein Spiel entwickeln, welches klar strukturiert ist. Dabei soll die Bedienung intuitiv sein, so dass der Anwender nicht überfordert ist und sofort weiß wie er das Programm steuert. Darüber hinaus sollte das Spiel ohne Installation auskommen, dadurch werden Fehler bei der möglichen Installation vermieden und zum anderen müssen keine Rechte eingeholt werden um das Programm zu installieren. Das Spiel muss so in einer kompakten Datei bleiben und eine Veränderung von Dateien ist nicht möglich, um eventuelle Kopierfehler zu vermeiden.

## 1.5.Benutzerschnittstelle

### 1.5.1 Benutzermodell

[](http://wiki.unity3d.com/index.php/File:X360Controller2.png)

[Bild] von http://wiki.unity3d.com/index.php/Xbox360Controller

A = joystick button 0 = Springen

B = joystick button 1 = Schlagen

X = joystick button 2 = Schlagen2

Y = joystick button 3 = Nichts

LB = joystick button 4 = Schild

RB = joystick button 5 = Schild

Back = joystick joystick button 6 = Nichts

Start = joystick button 7 = Pause

Left Analogue Press = joystick button 8 = Bewegen

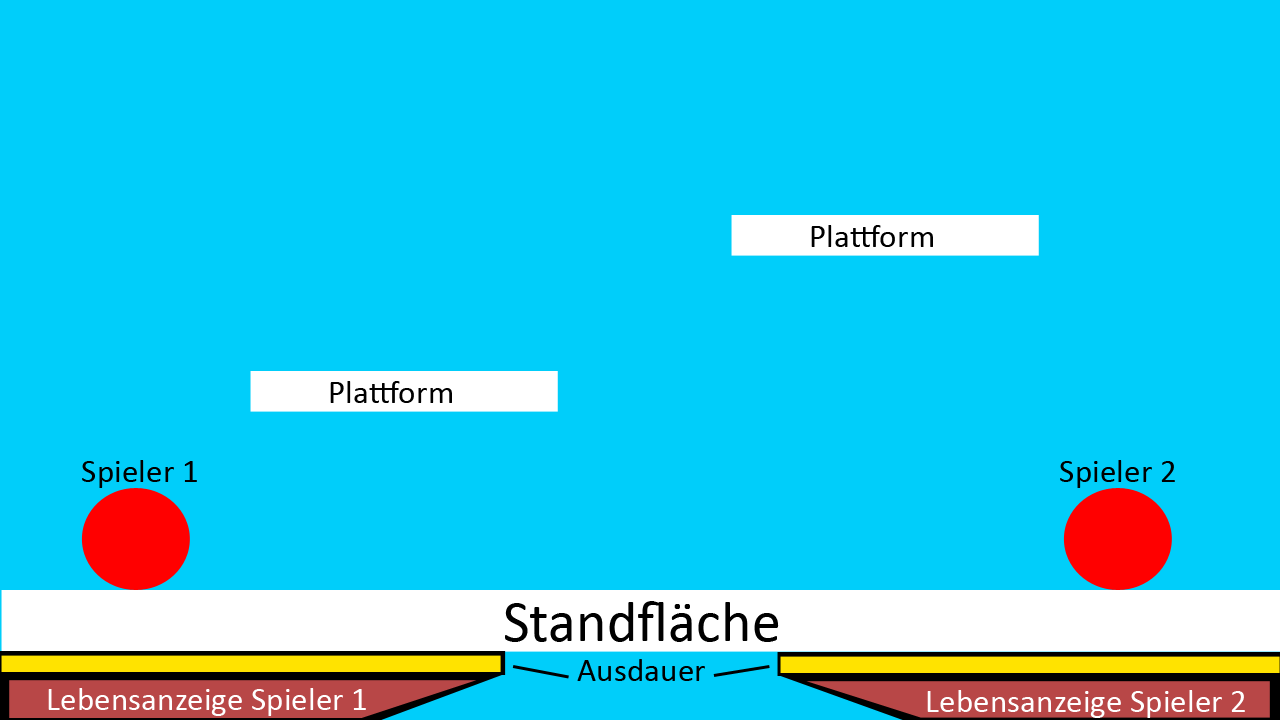
Right Analogue Press = joystick button 9 = Nichts

### 1.5.2 Kommunikationsstrategie

Der Nutzer tätigt seine Eingaben über Tastatur und XBOX Controller. Dazu wird die entsprechende Taste gedrückt, um so zu spielen oder das Spiel zu pausieren.

### 1.5.3 Kommunikationsaufbau

Startbildschirm



GUI

## 1.6.Entwicklungsumgebung

Wie bereits erwähnt ist unsere Entwicklungsumgebung Unity3d. Um Skripte, welche in C# geschrieben werden, anzufertigen benutzen wir das mitgelieferte MonoDevelolp. Zur Bearbeitung der Grafiken wird Gimp und Photoshop benötigt. Alle Grafiken und Ablaufpläne werden mit Hilfe von Visual Paradigm angefertigt. Zur Datenverwaltung benutzen wir Dropbox.

## 1.7.Literaturverzheichnis

Carsten. (kein Datum). *hummelwalker.de*. Abgerufen am 21. September 2013 von hummelwalker: www.hummelwalker.de

*docs.unity3d.com*. (kein Datum). Abgerufen am 21. 09 2013 von Unity3d-Website: http://docs.unity3d.com/Documentation/Manual/index.html

*opengameart.org*. (kein Datum). Abgerufen am 24. September 2013 von Opengameart: www.opengameart.org

Paul. (kein Datum). *spritedatabase.de*. Abgerufen am 24. September 2013 von Spritedatabase.

# 2.Entwurfsdokumentation

## 2.1 Softwarearchitektur

### 2.1.1 Technische Infrastruktur

In Unity3d gibt es keine Möglichkeit ein Klasse wie in C#-Sharp zu erstellen. Es gibt zwar die Möglichkeit seine Skripte an andere zu Vererben, doch die ist nicht mit C#Klassen zu vergleichen. Nur Unity3d gibt selber die Klassen vor mit denen gearbeitet wird. Eigene Funktionen werden als gekapseltes Script hinzugefügt und selber geschrieben.

### 2.1.2 Packetentwurf

Dies sind nur erste Überlegungen!

Position2D: Dieses Skript sorgt dafür, dass der Charakter nicht von der Bildfläche herunter fällt.

Feinentwurf: Z-Achse muss in jedem Frame korrigiert werden

HealthController: Hierdurch werden das Leben und die Ausdauer des Charakters geregelt.

Feinentwurf: Gewisses Leben als Zahl

Ausdauer besitzt Autoregenration

Schadensberechnung / Ausdauerverlust

SpriteController: Das Skript soll die Animation des jeweiligen Bewegungsinputs anpassen.

Feinentwurf: Animation für Gehen Links/Rechts

Animation für Stehen Links/Rechts

Animation für Springen Links/Rechts

Animation für Schlagen Links/Rechts

Animation für Treten Links/Rechts

Animation für Sterben Links/Rechts

PlayerController: Er nimmt und verarbeitet die Eingabe des Benutzers.

Feinentwurf: Hier sollten Inputs in Animationen umgewandelt werden eine Verknüpfung wird benötigt

SendDamageCollider: Dieses Script behandelt die Schadensberechnung.

Feinentwurf: Welche Attacke?

Trigger für Treffer

PausenMenue: Hier wird das Pausenmenü dargestellt.

Feinentwurf: wird über Start-Taste des XBOX Controllers aufgerufen

Bild „freezen“

Pausenmenü

StartMenue: Hier wird das Startmenü dargestellt. (Dieses Skript sorgt auch für den Szenenwechsel in das Spiel)

Feinentwurf: Titel wird dargestellt

Spielstarten-Button

Spielbeenden-Button

Item Generator: Er erzeugt in zufälligen abständen an zufällig Position Items die auf den Boden fallen.

Feinentwurf: 2 unterschiedliche Zufallszahlen

1 für X-Koordinate

2 entscheidet ob Objekt erscheint

Es kann sein das noch nicht alle Klassen vorhanden sind daher handelt es sich hierbei nicht um eine vollständige Liste!

Über die selbst angefertigten Skripte werden Unity3d Objekte benutzt, unter anderem Plane und Cubes. Wichtige Assets( Unity3d eigene Skripte) sind für uns die Charaktersteuerung und der Collider.

Feinentwurf der benutzten Unity3d-Objekte und Zugriffe auf die Bibliothek:

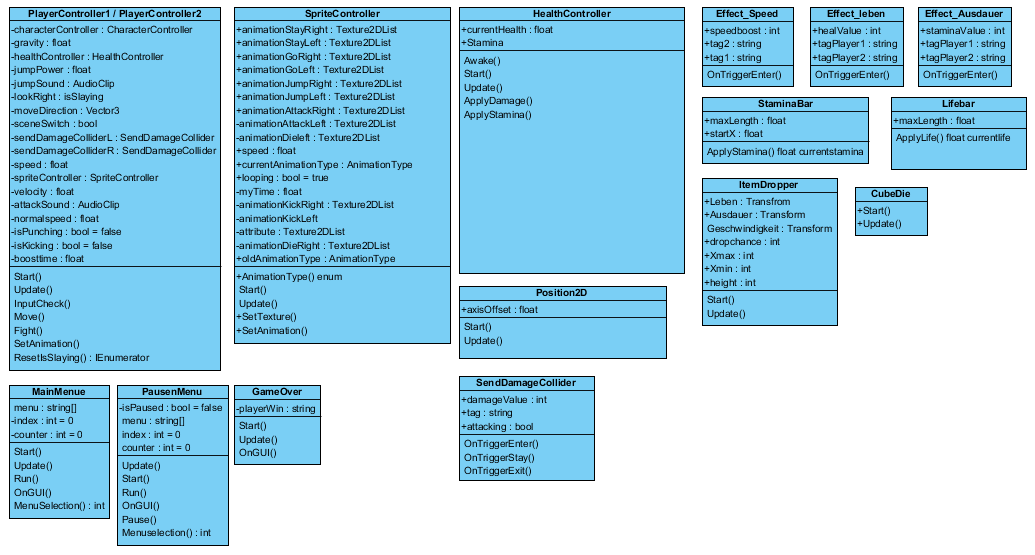
Objekte:

* Cubes, sie werden für die fallenden Items benutzt auch der Boden wird von einem extrudierten Cube dargestellt. Des Weiteren werden sie als WorldCollider benötigt.
* Plane, sie wird als Darstellungsoberfläche von Hintergrund und Charakter benötigt.
* GUI, alle Grafischen Bedienungselemente werden auf der GUI dargestellt.
* Prefabs, sind Objektorientierten Klassen ähnlich, sie haben alle Items hinterlegt.
* Lichter, zur Ausleuchtung der Bilder.

Skripte:

* Rigid Body, der Rigid Body wird mit dem Charakter Controller zur Darstellung der Physik benötigt.

### 2.2 Paketbeschreibung



## 2.3 Datenstrukturen

Alle Skripte haben öffentliche Variablen, dadurch wird eine nachhaltige Benutzung möglich. Falls ein Entwickler an dem Spiel schrauben will und so es seinen Wünschen entsprechend verändern will, braucht er nur die Unity3d Datei und kann die Werte verändern. Dies ermöglicht eine einfache Datenhandhabung, denn auch ein Laie braucht so die Struktur von Unity3d nicht verstehen.

## 2.4 Sonstiges

Zwar sollte zuerst ein Blocken möglich sein doch, dies erschien uns im Nachhinein überflüssig und würde sonst den Spielfluss verschlechtern.

# 3.Implementierungsdokumentation

## 3.1 Ablauf der Implementierung

Tag1: Grafiken und Sounds wurden ausgewählt. Das Pflichtenheft wurde angefertigt.

Tag2: Grafiken wurden zurecht gearbeitet erste Skripten wurden angelegt. Charakter Controller sowie Animation Controller

Tag3: Der erste Hintergrund Entwurf wurde angefertigt das Grundspiel war fertig der eine Spieler konnte den anderen Schlagen.

Tag4: Die Lebensverwaltung der Charaktere wurde angefertigt .Anschließend wurde noch ein Pausen und Startmenü angefertigt. Die Item Steuerung wurde eingefügt. Das Spiel ist fertig. (Pre-Alpha)

Tag5: Ein neuer Hintergrund wurde eingerichtet. Das Spiel wurde weiter aufpoliert.

Tag6: Grafiken wurden überarbeitet die „IngameGUI“ wurde verbessert.

Tag7: Verbesserungen des Spielflusses wurden durchgeführt. ( Collider angepasst Werte von Variablen optimiert)

Tag8: Skripte wurden optimiert.

Tag9: Letzte Fehler wurden behoben, es wurde auf Wunsch eine zusätzliche Tastatur Steuerung eingefügt.

Tag10: minimale Ergänzung des Hauptmenüs

Tag11: Letzte Überprüfungen

Tag12: Abgabe des Projekts

### 3.1.2 Probleme

Wir mussten das Steuerungsskript noch einmal überarbeiten, da sonst die Steuerung zu sensibel war und schon auf kleinste Erschütterungen reagiert hat.

Des Weiteren wollte wir ein Geräusch für Schlagen und treten einfügen, dies hat leider nicht geklappt.

## 3.2 Quellen

Alle fremden Ideen Grafiken etc. pp werden hier aufgeführt.

### 3.2.1 Skriptreferenz

Carsten. (kein Datum). *hummelwalker.de*. Abgerufen am 21. September 2013 von hummelwalker: www.hummelwalker.de

*docs.unity3d.com*. (kein Datum). Abgerufen am 21. 09 2013 von Unity3d-Website: http://docs.unity3d.com/Documentation/Manual/index.html

*opengameart.org*. (kein Datum). Abgerufen am 24. September 2013 von Opengameart: www.opengameart.org

Paul. (kein Datum). *spritedatabase.de*. Abgerufen am 24. September 2013 von Spritedatabase.

### 3.2.2.1 Benutzte Grafiken

Hintergrund: http://spritedatabase.net/file/3831

Items: http://opengameart.org/sites/default/files/life\_0.png

http://opengameart.org/sites/default/files/icon-power.png

Charakter: http://spritedatabase.net/game/1632

Hintergrundmusik: https://www.assetstore.unity3d.com/#/content/867

Charakter-Sounds: http://opengameart.org/content/oldschool-win-and-die-jump-and-run-sounds

**3.3 Skripte**

Alle Skripte sowie das Unityprojekt an sich finden sie auf der CD/(Datei).

# 4.Testdokumentation

## 4.1 Testplan

### 4.1.1 Teststrategien der Teststufen

Es werden alle Unityszenen einzeln getestet. Dabei werden Grenzfälle ausprobiert.

### 4.1.2 Testumgebung

Somit ist unsere Testumgebung ist somit der Unity Editor er liefert uns ein Rückmeldung bei Skriptfehlern und in ihm werden alle Fehler korrigiert.

### 4.1.3 Testwerkzeuge

Die Tests werden an unseren Laptops durchgeführt.

### 4.1.4 Testdaten

Es wurde jeden Tag ein Test des Spiels durchgeführt. Dabei kam es zu keinen Fehlern. Am 22.09.2013 wurden diverse Optimierungen des Spiels durchgeführt. So wurde der Collider des Charakters verbessert. Des Weiteren wurden die Objekte etwas besser platziert.

Zusätzlich wird das Unityprojekt per „Build and Run“ in allen Grafikstufen getestet. Dabei zeigte sich dass das Spiel sehr anspruchslos an die Technische Hardware ist.

## 4.2 Testfälle

### 4.2.1 Funktionale Tests

Alle Tests waren wie bereits erwähnt erfolgreich.

## 4.3 Testprotokolle

## 

Testprotokoll : 1

Test: Fängt das Spiel an zu Ruckeln.

Testbedingungen: Die Wahrscheinlichkeit, dass ein Item vom Himmel fällt liegt bei 100%.

Test Ziel: Das Spiel soll an die Grenzen kommen.

Testergebnis: Das Spiel fängt nicht an zu Ruckeln. Der einzige Unterschied zu einer normalen „Drop“-Wahrscheinlichkeit ist, dass die Charakter Animation nicht mehr ganz so flüssig abgespielt werden kann.

Testprotokoll : 2

Test: Grenzwerte der Variablen austesten.

Testbedingungen: Alle Variablen werden mit extrem unnatürlichen ( hohe, sowie niedrige) gefüllt.

Test Ziel: Mögliche Fehler die erst bei diesen Werten eventuell auftreten werden so aufgedeckt.

Testergebnis: Es entstanden natürlich Fehler und das Spiel wurde unspielbar. So fallen die Items nicht mehr auf den Boden oder die Charaktersteuerung wird überempfindlich.

Anhand dieser Tests konnten wir unsere Werte optimieren und so einen besseren Spielfluss gewährleisten.

# 5.Benutzerdokumentation

## 5.1 Über dieses Handbuch

### 5.1.1 Zielgruppe

Die Zielgruppe sind alle Nutzer ,die das Spiel spielen wollen.

### 5.1.2 Gebrauch des Handbuches

Das Handbuch wird eventuell gebraucht falls die Bedienung des Spiels unklar ist.

## 5.2 Einleitung

Hier finden Sie alles nötige um das Spiel erfolgreich benutzen zu können.

## 5.3 Produktbeschreibung

Unser Spiel Platform Fighter dient ausschließlich der Unterhaltung. Alle Rechte an fremden Grafiken gehören nicht uns.

## 5.4 Bedienungsanleitung

### 5.4.1 Steuerung

[](http://wiki.unity3d.com/index.php/File:X360Controller2.png)

[Bild] von http://wiki.unity3d.com/index.php/Xbox360Controller

A = joystick button 0 = Springen

B = joystick button 1 = Schlagen

X = joystick button 2 = Treten

Y = joystick button 3 = Nichts

LB = joystick button 4 = Nichts

RB = joystick button 5 = Nichts

Back = joystick joystick button 6 = Nichts

Start = joystick button 7 = Pause

Left Analogue Press = joystick button 8 = Bewegen

Right Analogue Press = joystick button 9 = Nichts

Falls Sie nur über einen oder keinen XBOX Controller verfügen können Sie das Spiel auch über Tastatur steuern.

 [Bild]http://klickdichschlau.at/contentfiles/edv/hardware/Tastatur\_de.png

Dazu hat Spieler eins die Möglichkeit den Charakter wie folgt zu Steuern:

A: Nach Links bewegen

D: Nach Rechts bewegen

E: Treten

Q: Schlagen

Leertaste: Springen

Der 2 Spieler steuert seinen Charakter mit dem NUM-Block:

1: Nach Links bewegen

3: Nach Rechts bewegen

6: Treten

4: Schlagen

0: Springen

### 5.4.2 Benutzeroberfläche

Startbildschirm

Sie können hier eine Spielrunde starten oder das Spiel beenden.



Benutzeroberfläche im Spiel

Plattformen



Ausdauer zum Schlagen, Treten und springen (regeneriert sich automatisch)

Leben des Charakters

Items die während des Spiels vom Himmel fallen:

 Die Batterie erhöht kurzfristig die Geschwindigkeit des Charakter.

C:\Users\Fabian\AppData\Local\Temp\Rar$DRa0.051\leben.png Das rote Herz gibt einen Teil des maximalen Lebens wieder.

C:\Users\Fabian\AppData\Local\Temp\Rar$DRa0.244\ausdauer.pngdas blaue Herz gibt einen Teil der maximalen Ausdauer wieder.

## 5.5 Beispielanwendung

Ablauf einer Runde:

Beide Spieler bekämpfen sich. Die Runde dauert so lange bis einer der beiden Spieler kein Leben mehr hat. Durch die Items kann sich ein Vorteil gegenüber dem anderen Spieler verschafft werden. Nach Rundenende erscheint ein schwarzer Bildschirm auf dem „GAME OVER“ steht, um dann in das Hauptmenü zurückzukehren drücken Sie A.

## 5.6 Glossar

Items: sind Gegenstände die meistens einen Nutzen für den Spieler haben.

# 6.Installationshandbuch

## 6.1 Inhaltsverzeichnis

## 6.2 Beschreibung der Dateien

Zum einen beinhaltet die Zip-Datei die Spiel.exe zum anderen einen Ordner, der alle Daten beinhaltet. Dieser Ordner darf nicht verändert werden, sonst kann ein das Spiel eventuell nicht gespielt werden und somit können wir keine Fehlerfreiheit gewährleisten.

## 6.3 Beschreibung der Installation

Die Installation unseres Spiels ist Simpel man benötigt nur die Zip Datei, die alles beinhaltet.

## 6.4 Glossar

Zip: Ist ein Dateiformat, wodurch sich Dateien komprimieren lassen.

# 7. Fazit

Während der Lernaufgabe haben wir eine Menge für unser späteres Berufsleben gelernt. Es beginnt mit bei der Arbeitsteilung und Zeiteinteilung und hört bei den fachbezogenen Kenntnissen auf. Wir könnten für uns neue Kenntnisse im Bereich GameDevelopment, sowie in der Entwicklungsumgebung Unity3d mitnehmen.