|  |  |  |  |
| --- | --- | --- | --- |
| 1/17/2016 |  |  | |
| |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |  | | |  | |
| 3D-Computerspiel - DOFGII  *Ein Top-Down-Shooter* | | | |
| |  |  |  | | --- | --- | --- | | **Name** | **E-Mail** | **Matrikelnummer** | | Jennifer Kuschnig | [Kuschnigje60640@th-nuernberg.de](mailto:Kuschnigje60640@th-nuernberg.de) | 2665234 | | Tobias Velke | [VelkeTo61783@th-nuernberg.de](mailto:VelkeTo61783@th-nuernberg.de) | 2693554 | | Micha Zimmermann | [Zimmermannmi61778@th-nuernberg.de](mailto:Zimmermannmi61778@th-nuernberg.de) | 2750812 | | Samuel Seiz | [Seizsa61777@th-nuernberg.de](mailto:Seizsa61777@th-nuernberg.de) | 2710196 | | Christoph Huxhagen | [HuxhagenCh49172@th-nuernberg.de](mailto:HuxhagenCh49172@th-nuernberg.de) | 2318692 | |  | |  |
|  |  | |  |

Inhalt

[1 Einführung 2](#_Toc439847919)

[1.1 Projekthintergrund 2](#_Toc439847920)

[1.2 Projektauftrag 2](#_Toc439847921)

[1.3 Aufgabenbereich Teammitglieder 2](#_Toc439847922)

[2 Analyse 3](#_Toc439847923)

[2.1 Funktionale Anforderungen 3](#_Toc439847924)

[2.2 Nicht Funktionale Anforderungen 5](#_Toc439847925)

[3 Entwurf 5](#_Toc439847926)

[3.1 Sequenzdiagramm 5](#_Toc439847927)

[3.2 Use Case Diagramm 6](#_Toc439847928)

[4 Implementierung 10](#_Toc439847929)

[4.1 Quellcode-Konventionen 10](#_Toc439847930)

[5 Tests 11](#_Toc439847931)

[5.1 Akzeptanztest 11](#_Toc439847932)

[5.2 Balance Test 12](#_Toc439847933)

[6 Fazit 13](#_Toc439847934)

[7 Glossar 13](#_Toc439847935)

# Einführung

## Projekthintergrund­

Dieses Projekt zielt darauf ab ein vollständiges Softwareprojekt zu entwickeln. Dabei muss das Team eigenständig alle Phasen des Projektes von der ersten Konzeptionierung bis hin zu abschließenden Qualitätstest durchführen.

Hierbei sollen die Teammitglieder erste Erfahrungen mit Methoden der Softwareentwicklung sammeln, Techniken entwickeln und sich in neue Werkzeuge einarbeiten.

## Projektauftrag

Die Praktikumsaufgabe umfasst die team-basierte Konzeptionierung, Umsetzung und Dokumentation eines Softwareprojekts unter Anwendung der in der Vorlesung behandelten Methoden und Techniken, welche folgendermaßen lautet:

***Entwickeln Sie eine Computerspiel-Demo basierend auf einer (2D-)Game-Engine*.**

Die Game-Engine soll vom Team gewählt werden, wobei darauf geachtet werden muss, dass diese eine Entwicklung mit C# bzw. C++ ermöglicht. Zudem muss sich das Team selbstständige in die gewählte Engine einarbeiten.

Bei der Entwicklung sollen die gelernten Prinzipien der objektorientierten Softwaretechnik angewendet werden. Der Auftrag soll in folgenden Kernphasen der Softwaretechnik gegliedert sein:

* Analyse
* Entwurf
* Implementierung
* Tests

Als abschließendes Projektziel soll eine lauffähige Demoversion präsentiert werden.

## Aufgabenbereich Teammitglieder

|  |  |
| --- | --- |
| Mitglied | Aufgaben |
| Jennifer Kuschnig | Shop-Design, Weapon-Design, Level-Design |
| Micha Zimmermann | Enemy-Design, Level-Design, Art-Design |
| Samuel Seiz | Enemy-Design, Git-Hub, Level-Design |
| Tobias Velke | Game-UI, Bonus-Design, Projektmanagement |
| Christoph Huxhagen | Player-Design, Weapon-Design |

# Analyse

Die in diesem Kapitel aufgeführten Anforderungen verstehen sich nur als ein Auszug aller von uns bestimmten Anforderungen an das zu entwickelnde Produkt und spiegeln lediglich exemplarisch unsere Vorgehensweise wieder.

## Funktionale Anforderungen

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Story\_ID:** | **1** | **Story\_Name:** | Player\_Movement | |  | **Risk:** | 8 |
| **As an:** | Spieler | |  |  |  | **Priority:** | 10 |
| **I would like to** | möchte ich das Spielobjekt auf der XY-Ebene in alle Richtungen bewegen können. | | | | | **est. Work [h]:** | 6 |
|  |  |  |
| **Given:** | Das Spiel befindet sich im Play Mode | | | | |  |  |
|  |  | | | | |  |  |
|  |  | | | | |  |  |
| **So that:** | jeder Punkt der X-Z-Ebene ist durch betätigen der WASD-Tasten erreichbar | | | | |  |  |
|  | es gibt keinen Bereich der X-Z-Ebene der nicht erreichbar ist | | | | |  |  |
|  |  | | | | |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Story\_ID:** | 2 | **Story\_Name:** | Player\_Attack | |  | **Risk:** | 10 |
| **As an:** | Spieler | |  |  |  | **Priority:** | 8 |
| **I would like to** | möchte ich die gegnerischen Spielobjekte durch Schüsse zerstören können. | | | | | **est. Work [h]:** | 3 |
|  |  |  |
| **Given:** | Das Spiel befindet sich im Play Mode | | | | |  |  |
|  |  | | | | |  |  |
|  |  | | | | |  |  |
| **So that:** | Das gegnerische Spielobjekt nicht mehr existiert, | | | | |  |  |
|  | Ich als Spieler dafür Punkte bekommen | | | | |  |  |
|  | Und ggf. ein neues gegnerisches Spielobjekt erstellt wird | | | | |  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Story\_ID:** | 3 | **Story\_Name:** | Player\_Health | |  | **Risk:** | | 6 |
| **As an:** | Spieler | |  |  |  | **Priority:** | | 7 |
| **I would like to** | Möchte ich über einen vorgegebenen Vorrat an Lebenspunkten verfügen, welche durch Kollision oder durch Treffer gegnerischer Spielobjekte reduziert werden | | | | | **est. Work [h]:** | | 2 |
|  |  |  | |
| **Given:** | Das Spiel befindet sich im Play Mode | | | | |  |  | |
|  |  | | | | |  |  | |
|  |  | | | | |  |  | |
| **So that:** | Ich eine optische Anzeige meiner verbleibenden Lebenspunkte habe | | | | |  |  | |
|  | Wenn mein Lebenspunkte Vorrat gleich 0 ist, das Spiel beendet wird | | | | |  |  | |
|  | Ich meinen Vorrat an Lebenspunkte zu Beginn eines Levels oder durch spezielle Boni wieder auffüllen kann | | | | |  |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Story\_ID:** | 4 | **Story\_Name:** | Weapon\_Shop | |  | **Risk:** | 7 |
| **As an:** | Spieler | |  |  |  | **Priority:** | 7 |
| **I would like to** | möchte ich nach absolvieren eines Levels in einen Waffenshop wechseln können um dort neue Waffen für mein Spielobjekt kaufen zu können. | | | | | **est. Work [h]:** | 4 |
|  |  |  |
| **Given:** | Das Spiel befindet sich im Play Mode | | | | |  |  |
|  |  | | | | |  |  |
|  |  | | | | |  |  |
| **So that:** | Nach beenden eines Levels soll ein klar erkennbarer Wechsel in einen Shop stattfinden | | | | |  |  |
|  | In diesem Shop soll es mir abhängig der Anzahl meiner Coins möglich sein neue Waffen zu kaufen. | | | | |  |  |
|  | Nach dem Kauf soll zurück zum Spiel und somit zum Beginn eines neuen Levels gewechselt werden | | | | |  |  |

## Nicht Funktionale Anforderungen

Neben den funktionalen Anforderungen, die sich eher auf die Kernfunktionen, also um das „Was macht das Programm“ beziehen, bestehen weitere Ansprüche an die Software, um Rahmenbedingungen festzusetzen. Da nicht-funktionale Anforderungen nicht genau in ihrer Art und Weise definiert sind sollen hier einige der für das Projekt wichtigen Punkte dargestellt werden.

* Das Spiel soll leicht zugängig für jedermann sein

Ein neuer Spieler (Anwender) soll ohne große Einweisung sofort die Steuerung verinnerlichen können. Des Weiteren sollte das Spielziel klar ersichtlich sein.

* Das Programm soll Ressourcen-Effizient arbeiten und flüssig laufen

Nicht benötigte Objekte und Instanzen werden verworfen. Beim der Programmierarbeit wird darauf geachtet Speichereffizient zu arbeiten, um das Spiel auf jedem zeitgemäßen Computer spielen zu können.

* Eine eventuelle Portierbarkeit auf andere (mobile) Systeme soll gewährleistet sein

Bei der Auswahl der Graphik-Engine soll darauf geachtet werden, dass diese kompatibel zu anderen, eventuell auch mobilen Systemen ist. Der Schritt der Portierung soll in der Zukunft möglich gemacht werden

* Es soll einen eigenen optischen Charme haben und sich von dem Genre abheben

Als typischer Genre Klassiker soll sich das Spiel optisch abheben.

* Die Teamarbeit muss gewährleistet sein und die Prozesse für jeden zu jeder Zeit ersichtlich abgebildet werden

Um effektiv im Team zu arbeiten muss die Aufgabenteilung und wieder-zusammenführung durch verschiedene Hilfsmittel sinnvoll unterstützt werden. Des Weiteren wird jede Aufgabe und Bearbeitungsstatus zentral festgehalten um den Überblick über den Status des Projektes zu bewahren.

# Entwurf

## Sequenzdiagramm



## Use Case Diagramm

Im folgendem Use Case Diagramm werden die elementaren Spielhandlungen abgebildet und jeweils kurz erläutert.



|  |  |
| --- | --- |
| 1. Move Playerobject | |
| Art | Spielen |
| Kurzbeschreibung | Der Spieler kann das Spielobjekt über die Tasten WASDQR auf der Oberfläche in alle Richtungen bewegen |
| Auslöser | Der Spieler möchte das Spielobjekt bewegen |
| Ergebnis | Das Spielobjekt bewegt sich auf die gewünschte Position |
| Akteure | Spieler |
| Eingehende Informationen | x- y- Koordinaten |
| Vorbedingungen | Spielobjekt muss vorhanden und bewegbar sein |
| Nachbedingungen | Spielobjekt befindet sich auf gewünschte Stelle |
| Essentielle Schritte | Taste W => Spielobjekt fliegt nach vorne  Taste A => Spielobjekt fliegt nach links  Taste S => Spielobjekt fliegt nach hinten  Taste D => Spielobjekt fliegt nach rechts  Taste Q => Spielobjekt dreht nach links  Taste E => Spielobjekt dreht nach rechts |
| Varianten | Tasten W+A=> Spielobjekt fliegt schräg links n. vorne  Tasten W+D=> Spielobjekt fliegt schräg rechts n. vorne  Tasten A+S=> Spielobjekt fliegt schräg links n. hinten  Tasten D+S=> Spielobjekt fliegt schräg rechts n. hinten |

|  |  |
| --- | --- |
| 1. Shot Weaopn | |
| Art | Schießen |
| Kurzbeschreibung | Der Spieler kann mit der Waffe auf Objekte schießen |
| Auslöser | Der Spieler möchte schießen können |
| Ergebnis | Der Spieler kann schießen |
| Akteure | Spieler |
| Eingehende Informationen | Waffeninformationen (z.B FireRate, Spread) |
| Vorbedingungen | Spieler besitzt eine Waffe |
| Nachbedingungen | Spieler kann mit Waffe Objekte zerstören |
| Essentielle Schritte | Mit Mausklick entsendet Spieler ein Schuss |
| Varianten | Mit STRG entsendet Spieler ein Schuss |

|  |  |
| --- | --- |
| 1. Destroy Enemy | |
| Art | Eliminieren |
| Kurzbeschreibung | Der Spieler kann durch schießen zukommende Gegner eliminieren. |
| Auslöser | Der Spieler möchte durch das Schießen zukommende Gegner eliminieren. |
| Ergebnis | Beim Treffen eines Gegners durch einen Schuss, wird dieser eliminiert und gegeben falls wird ein BonusItem erzeugt |
| Akteure | Spieler |
| Eingehende Informationen | Kollision zwischen Schussobjekt und Gegner |
| Vorbedingungen | Der Spieler muss eine Waffe besitzen und es müssen Gegner vorhanden sein |
| Nachbedingungen | Der Gegner wurde eliminiert und gegeben falls BonusItem generiert. |
| Essentielle Schritte | Spielobjekt muss auf Gegner gerichtet sein und es muss ein Schuss erfolgen. |
| Varianten |  |

|  |  |
| --- | --- |
| 1. Collect BonusItem | |
| Art | BonusItem sammeln |
| Kurzbeschreibung | Der Spieler kann BonusItems sammlen. |
| Auslöser | Der Spieler möchte BonusItems sammeln können |
| Ergebnis | Durch Einsammeln von BonusItems können Coins, HealthItems bzw. Enemy zufällig erzeugt werden. |
| Akteure | Spieler |
| Eingehende Informationen | Gesammelte BonusItems |
| Vorbedingungen | Ein eliminierter Gegner muss ein BonusItem hinterlassen |
| Nachbedingungen | Das Resultat des eingesammelten BonusItem kann eingesammelt bzw. bekämpft werden. |
| Essentielle Schritte | BonusItem mit Spielobjekt einsammeln |
| Varianten |  |

|  |  |
| --- | --- |
| 1. Collect Coins | |
| Art | Geld sammeln |
| Kurzbeschreibung | Der Spieler kann durch eliminieren von BonusItems Währung einsammeln |
| Auslöser | Der Spieler möchte Währung sammeln können |
| Ergebnis | Der Spieler kann durch das Treffen einer Münze seinen Währungsstand erhöhen |
| Akteure | Spieler |
| Eingehende Informationen | Gesammelte Münzen |
| Vorbedingungen | Es muss eine Münze am Spielfeld vorhanden sein |
| Nachbedingungen | Die Münze wurde eingesammelt und der Währungsstand erhöht |
| Essentielle Schritte | Münze mit einem Schuss treffen |
| Varianten | Münze mit Spielobjekt überfahren |

|  |  |
| --- | --- |
| 1. Collect HealtItem | |
| Art | HealthItem sammeln |
| Kurzbeschreibung | Der Spieler kann durch eliminieren von BonusItems seine Healthbar wieder auf 100% auffüllen. |
| Auslöser | Der Spieler möchte seine Healthbar wieder voll auffüllen |
| Ergebnis | Der Spieler kann durch das Treffen eines HealthItems seine Healthbar wieder auffüllen. |
| Akteure | Spieler |
| Eingehende Informationen | Gesammelte HealtItmes |
| Vorbedingungen | Es muss ein HealthItem am Spielfeld vorhanden sein |
| Nachbedingungen | Das HealthItem wurde eingesammelt und die Healthbar wurde aufgefüllt |
| Essentielle Schritte | HealthItem mit einem Schuss treffen |
| Varianten | HealthItem mit Spielobjekt überfahren |

|  |  |
| --- | --- |
| 1. Collect Points | |
| Art | Highscore |
| Kurzbeschreibung | Der Spieler kann über das eliminieren von Gegner Punkte sammeln |
| Auslöser | Der Spieler möchte Punkte sammeln können |
| Ergebnis | Der Spieler kann durch das Eliminieren von Gegnern seinen Punktstand erhöhen |
| Akteure | Spieler |
| Eingehende Informationen | Gesammelte Punkte |
| Vorbedingungen | Es müssen Gegner zum Eliminieren vorhanden sein |
| Nachbedingungen | Gegner verschwinden nach treffen eines Schusses und Punktestand erhöht sich |
| Essentielle Schritte | Siehe Punkt 2. Mit Waffe schießen und 3. Gegner eliminieren |
| Varianten | Siehe Punkt 2. Mit Waffe schießen und 3. Gegner eliminieren |

|  |  |
| --- | --- |
| 1. Buy Weapon | |
| Art | Waffen kaufen |
| Kurzbeschreibung | Der Spieler kann anhand genügend gesammelter Münzen sich eine neue Waffe kaufen |
| Auslöser | Der Spieler möchte im Spielverlauf eine neue Waffe erwerben können |
| Ergebnis | Der Spieler hat eine neue Waffe erworben |
| Akteure | Spieler |
| Eingehende Informationen | Gesammelte Währung, Gesammelte Punkte |
| Vorbedingungen | Der Spieler muss genügend Punkte besitzen um in den Waffenshop wechseln zu können und zudem braucht er genügend Münzen um eine Waffe kaufen zu können |
| Nachbedingungen | Münzenstand ausreichend => Spieler hat eine neue Waffe |
| Essentielle Schritte | 1. Genügend Punkte sammeln 2. Genügende Münzen besitzen für Kauf |
| Varianten | - |

|  |  |
| --- | --- |
| 1. Changed Weapon | |
| Art | Waffenwechsel |
| Kurzbeschreibung | Der Spieler kann während des Spielverlaufs seine Waffe wechseln |
| Auslöser | Der Spieler möchte während des Spiels seine Waffe wechseln können |
| Ergebnis | Der Spieler wechselt seine Waffe |
| Akteure | Spieler |
| Eingehende Informationen | Gesammelte Währung, Gesammelte Punkte |
| Vorbedingungen | Der Spieler muss genügend Punkte besitzen um in den Waffenshop wechseln zu können und zudem braucht er genügend Münzen um eine Waffe kaufen zu können |
| Nachbedingungen | Waffe gekauft => neue Waffe im nächsten Level |
| Essentielle Schritte | 1. Waffe kaufen 2. Waffenshop verlassen |
| Varianten | - |

# Implementierung

## Quellcode-Konventionen

|  |  |  |
| --- | --- | --- |
| Konvention | Beschreibung | Positiv Beispiel |
| Tabs | Tabs als Spaces = 4 (VisaulStudio Standard) | public void Update( )  {  Shot();  } |
| Positionierung geschweifte Klammern | * Immer auf Anfang der Codezeile * Linke und rechte Klammer (d. h. Codeblockanfang und -ende) immer "übereinander" | if(someCondition)  {  doSometing( );  } |
| Leere Zeilen | am Code-Block-Anfang vermeiden | while (someCondition)  {  ...  } |
| Einbindung von Pakten | Nur die relevanten Pakete einbinden (mittels using) | Using System; |
| Benennung von Bezeichnern | Alle Bezeichner für Namensraum, Klassen, Variablen für Menschen lesbar |  |
| Öffentliche Variablen und Methoden | Diese Variablen und Methoden müssen großgeschrieben werden | Public void Update( ) |
| Nicht öffentliche Variablen und Methoden | Diese Variablen und Methoden müssen kleingeschrieben werden | Private int level = 0; |
| Kommentare | Methoden sollen kommentiert werden | //….  Public void Update( ) |

# Tests

## Akzeptanztest

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test\_ID | Test\_Name | Test\_Preconditions | Test\_Scenario | Test\_Result |
| 1 | Player\_Movement | Spiel befindet sich im Spiel Modus, Spieler ist aktiv | Spieler mit WASD- Tasten über das Spielfeld bewegen | Der Spieler konnte zu jeder Position auf dem Spielfeld und darüber hinaus bewegt werden. Der Spieler war zu keinem Zeitpunkt manövrierunfähig. |
| 2 | Player\_Attack | Spiel befindet sich im Play Modus, Gegner und Spieler sind aktiv | Mit dem Spieler soll auf einen Gegner gezielt und drücken der STRG-Taste beschossen werden. Wurde der Gegner getroffen so soll er zerstört werden und der aktuelle Punktestand erhöht werden | Sobald ein Gegner getroffen wurde, wird es zerstört. Des Weiteren wird der aktuelle Punktestand um Eins erhöht.  Dieses Verhalten konnte bei allen verschiedenen Gegnertypen nachvollzogen werden. |
| 3 | Player\_Health | Spiel befindet sich im Play Modus, Gegner und Spieler sind aktiv, Lebenspunkte des Spieler sind >0 | Die verschieden Gegner können dem Spieler unterschiedlich Schaden zufügen um dies zu testen soll jeder Gegnertypus dem Spieler Schaden zufügen und die Auswirkung überprüft werden. | Enemy 1 erzielt 50 Schadenpunkte durch Kollision mit dem Spieler,  Enemy 3 erzielt 40 Schadenpunkte durch Schüsse auf den Spieler,  Enemy 3 erzielt 10 Schadenspunkte durch Schüsse auf den Spieler.  Sobald der |
| 4 | Weapon-Shop | Spiel befindet sich im Play-Modus,  nach zerstören eines Gegner ist die benötigte Punktzahl zum beenden des Levels erreicht | Nach beenden eines Levels soll automatisch in den WeaponShop gewechselt werde.  Dort sollen unterschiedliche Waffen zum Kauf angeboten werden, jede dieses Waffen hat einen unterschiedlichen Preis, vor dem Kauf soll überprüft werden ob der Spieler genügen Coins besitzt.  Der Spieler soll den Shop über einen Button verlassen und somit das nächste Level starten können | Nach zerstören des letzten Gegners wechselt das Spiel automatisch in den Shop  Dort kann man aus (bisher) drei unterschiedlichen Waffen wählen.  Wurde eine Waffe gewählt ohne genügend Coins zu besitzen wird ein Dialog mit entsprechender Meldung angezeigt.  Wird eine Waffe gekauft und der Shop verlassen wechselt das Spiel wieder in den LevelModus. Die vorherigen Punkte sowie die aktuelle Anzahl an Coins werden in der GUI angezeigt |
| 5 | Enemy\_Shooting | Spiel befindet sich im Play Modus, Gegner und Spieler sind aktiv | Gegner die auf den Spieler schießen, tun dies in einen für den Gegnertyp festgelegten Abstand, mit einem zufällig gewählten Startzeitpunkt nach erstellen des Gegners. Um dies zu testen sollen mehrere Level an einem Stück gespielt werden und die Feuerrate jedes Gegnertypus überprüft werden. | Enemy 1 und Enemy 3, feuern beide alle 3 Sekunden, der Startzeitpunkt ist für jede Instanz zufällig innerhalb von 5 Sekunden nach Erstellung des Gegners. |

## Balance Test

# Fazit

# Glossar

|  |  |
| --- | --- |
| Term | Explanation |
| Spieler | Damit ist das GameObject gemeint welches durch einen menschlichen Benutzer gesteuert werden kann |
| Gegner | Damit sind GameObejects gemeint welche durch das Spiel erstellt werden und autonom agieren, |
| GameObjects | Damit sind Objekte der Engine Unity gemeint welche mit verschiedensten Eigenschaften und Aufgaben innerhalb der Engine implementiert wurden. Sie repräsentieren unter anderem den Spieler, die Gegner, die einzelnen Schüsse oder aber auch die Bonusitems |
| Waffen | Als Waffe wird hier ein abstraktes Konzept einer Waffe bezeichnet. Die Waffen im Spiel unterscheiden sich in ihren Eigenschaften wie zum Beispiel der Häufigkeit mit welcher sie abgefeuert werden können oder der Geschwindigkeit der einzelnen Schüsse oder dem Preis der Waffe |
|  |  |
|  |  |
|  |  |