Dokumentation Robin Steiger

|  |  |
| --- | --- |
| Teammitglieder | Toni Kostal  Robin Steiger  Henri Schmekies  Fabian Eichenberger |
| Projektleiter | Henri Schmekies |
| Kunde | M. Schneider |
| Beschreibung Projekt | Ein Game mit Löve2D erstellen und sich damit in die Programmiersprache von Löve2D (Lua) einarbeiten |
| Datum | 29. April 2015 |
| [🡨 Zurück zur Übersicht](../projektuebersicht.docx) |  |

Inhaltsverzeichnis

[Metainformationen 3](#_Toc417457973)

[Löve2d 4](#_Toc417457974)

[Einstieg 4](#_Toc417457975)

[Version 1 4](#_Toc417457976)

[Version 2 4](#_Toc417457977)

[Version 3 4](#_Toc417457978)

[Aufbau Löve2d 4](#_Toc417457979)

[Benutzerdefinierte Funktionen 4](#_Toc417457980)

[CheckCollision() 4](#_Toc417457981)

[CreateEnemy() 5](#_Toc417457982)

[EnemyBullets() 5](#_Toc417457983)

[PlayerBullet() 6](#_Toc417457984)

[DrawTable() 6](#_Toc417457985)

[BossAttack1() 6](#_Toc417457986)

[BossAttack2() 7](#_Toc417457987)

[Weitere Strukturen 7](#_Toc417457988)

[Bewegung 7](#_Toc417457989)

[Spiel schliessen 8](#_Toc417457990)

[Bilder laden 8](#_Toc417457991)

[Lebensbalken 8](#_Toc417457992)

[SpaceShooter Spiel 9](#_Toc417457993)

Metainformationen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Datum | Status | Version | Autor | Bemerkung |
| 14.01.2015 | In Bearbeitung | 1.0 | Robin Steiger | Dokument erstellt |
| 28.01.2015 | In Bearbeitung | 1.1 | Robin Steiger | Dokument überarbeitet |
| 22.04.2015 | In Bearbeitung | 1.2 | Robin Steiger | Dokument überarbeitet |
| 29.04.2015 | Freigegeben | 1.3 | Robin Steiger | Dokument fertiggestellt |

|  |  |
| --- | --- |
| Aktueller Status: | Freigegeben |

Löve2d

Löve2d ist ein Framework für die Programmiersprache Lua, damit können kleine 2d Spiele erstellt werden.

Einstieg

Da Löve2d auf Lua basiert habe ich mir zuerst dazu ein [Tutorial](http://nova-fusion.com/2012/08/27/lua-for-programmers-part-1/) gesucht, welches die Grundlagen erklärt. Dann habe ich mithilfe eines [Löve2d-Tutorials](http://osmstudios.com/tutorials/your-first-love2d-game-in-200-lines-part-1-of-3) meine ersten Versuche mit Löve2d gemacht. Mein Ziel war es ein Bullethell-Spiel zumachen, da mir dies relative einfach erschien.

[Version 1](../robin/Spiel_Robin_V1)

Die erste Version von meinem Spiel habe ich grundsätzlich beendet, weil ich mit Funktionen, wie Kollisionen erkennen oder Gegner spawnen, grosse Probleme hatte.

[Version 2](../robin/Spiel_Robin_V2)

In Version 2 habe ich noch mal alles überarbeitet, weil ich fast keine Funktionen sondern einfach sehr viele Male Copy&Paste verwendet habe, habe ich nach den Ferien komplett den Überblick in der Version 2 verloren, darum habe ich mich entschieden erneut von vorne zu beginnen.

[Version 3](../robin/Spiel_Robin_V3)

Die Version 3 ist gleichzeitig auch die letzte Version, in dieser Version habe ich deutlich mehr Funktionen verwendet und auch mehr Kommentare geschrieben. Diese Version habe ich auch in ein [.exe](../robin/SpaceShooter) umgewandelt, so dass sie auch auf einem Computer ohne Love2d gestartet werden kann.

Aufbau Löve2d

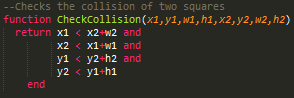
Love2d ist in drei Grundfunktionen unterteilt.

* Function Love.load() diese wird jeweils beim Starten ausgeführt
* Function Love.update() diese wiederholt sich immer wieder, als wäre sie eine Schlaufe
* Function Love.draw() diese wiederholt ebenfalls immer wieder, wird aber verwendet um Objekte im Fenster zu zeichnen. Z.B den Hintergrund, den Spieler und die Gegner

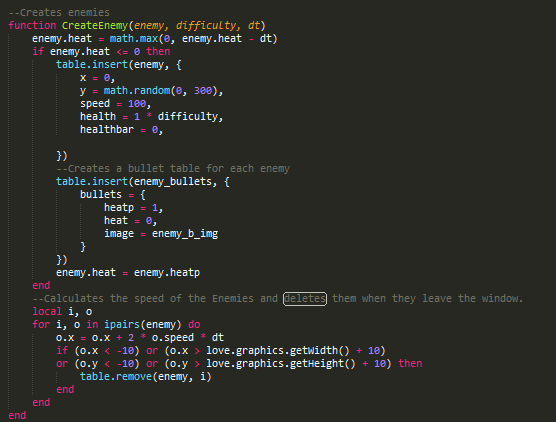
Benutzerdefinierte Funktionen

Ich habe selbst mehrere Funktionen erstellt.

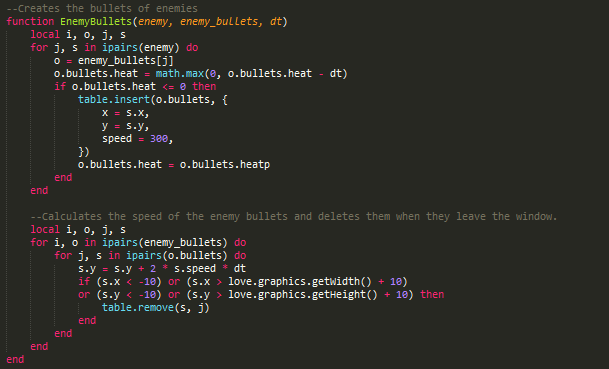
CheckCollision()

  
Die Funktion CheckCollision habe ich nicht selber erstellt, sondern vom [Love2d-Wiki](https://www.love2d.org/wiki/BoundingBox.lua). Die Funktion erkennt wenn zwei Objekte sich überschneiden und kann verwendet werden um eine Kollision zu erstellen.

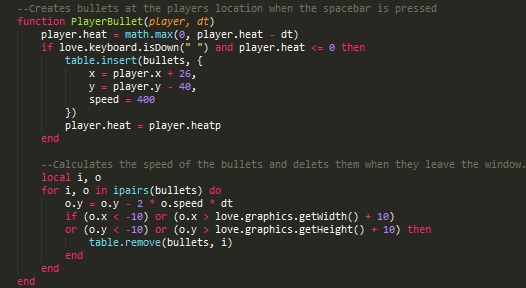
CreateEnemy()

  
Die Funktion CreateEnemy fügt Gegner in eine Tabelle ein und berechnet wie sie sich über den Bildschirm bewegen, ebenfalls löscht sie Gegner die sich ausserhalb des Bildschirmes befinden. Zu dem erstellt es für jeden Gegner eine Tabelle, welche später die Schüsse beinhaltet.

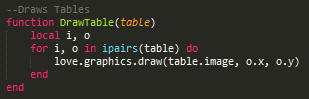
EnemyBullets()

  
Die Funktion EnemyBullets fügt für jeden Gegner Schüsse in eine Tabelle ein, hier werden ebenfalls die Flugbahnen der Schüsse berechnet, sowie das Löschen der Schüsse beim Verlassen des Bildschirmes.

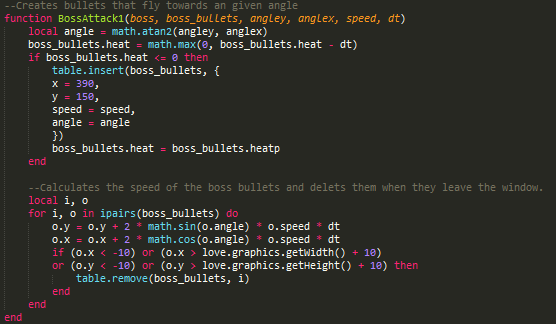
PlayerBullet()

  
Die Funktion PlayerBullet fügt immer wenn die Space-Taste gedrückt ist, Schüsse des Spielers in eine Tabelle ein, die Flugbahnen dieser wird danach berechnet, auch hier werden die Schüsse beim Verlassen des Bildschirmes gelöscht.

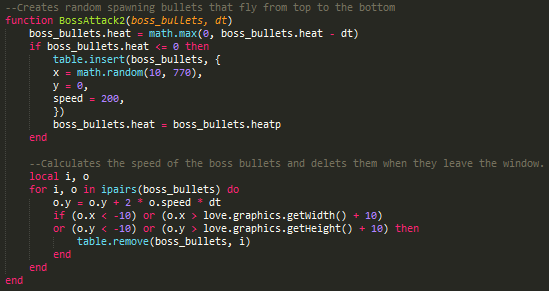
DrawTable()

  
Die Funktion DrawTable zeichnet den Inhalt einer Tabelle.

BossAttack1()

  
Die Funktion BossAttack1 fügt die Schüsse des Bosses in eine Tabelle ein. Auch hier werden wieder die Flugbahnen berechnet, hier wird einfach noch der Winkel angegeben, in welchen die Schüsse sich bewegen sollen. Verlassen die Schüsse den Bildschirm werden sie gelöscht.

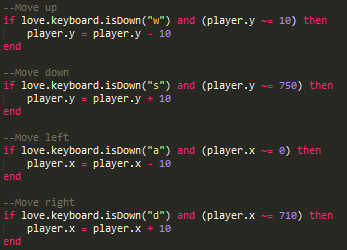
BossAttack2()

  
Die Funktion BossAttack2 fügt die Schüsse des Bosses in eine Tabelle ein. Der Punk auf der X-Achse ist hier zufällig generiert. Die Flugbahnen der Schüsse werden auch hier in der Funktion selbst berechnet. Verlassen die Schüsse den Bildschirm werden sie gelöscht.

Weitere Strukturen

Hier werde ich einige weitere Strukturen aufzeigen, wie z.B. Bewegung des Spielers oder Beenden des Spieles.

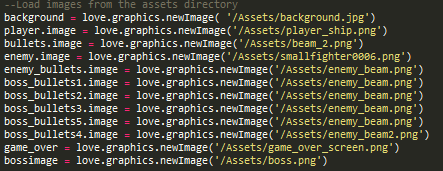
Bewegung

  
Der Spieler kann sich mittels WASD in alle Richtungen bewegen.

Spiel schliessen

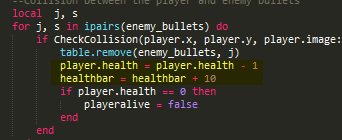
  
Mittels der Escape-Taste kann man das Spiel ganz einfach schliessen, dabei wird nichts gespeichert.

Bilder laden

  
Um Bilder verwenden zu können, müssen diese zuerst vorgeladen werden.

Lebensbalken

Alle meine Gegner und der Spieler besitzen einen Lebensbalken. Dieser funktioniert wie folgt.

  
Ausschnitt aus der CheckCollision Funktion.

Jedes Mal wenn z.B. der Spieler getroffen wird, wird einer Variablen ein Betrag, hier 10, hinzugefügt. Dieser wird dann später beim Zeichnen des Lebensbalkens wieder abgezogen.

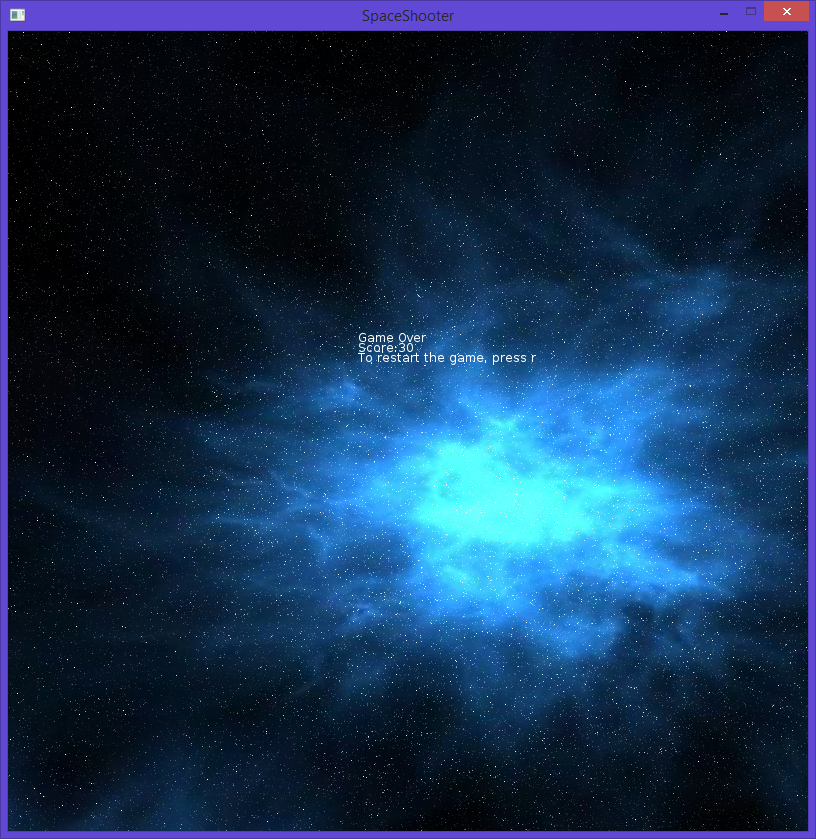


SpaceShooter Spiel

Das Spiel wird mittels WASD gesteuert, mit der Leertaste können Laser abgefeuert werden.

Hier einige Screenshots vom Endprodukt.



  
Game Over Screen, hier wird der erreichte Score ausgegeben. Man kann das Spiel neustarten in dem man „r“ drückt.



: Boss Kampf