<the deadly snake>

Anforderungsanalyse

|  |  |
| --- | --- |
| **Autor(en):** | Luca Aeberhard |
| **Version:** | 1.0 |
| **Erstellt am:** | 5. Dezember 2019 |
| **Letzte Änderung:** | 5. Dezember 2019 |
| **Speicherort:** | Dokument2 |

Inhalt

[1 Revisionen 3](#_Toc26444794)

[2 Einleitung 4](#_Toc26444795)

[2.1 Systemidee 4](#_Toc26444796)

[Die wichtigsten Funktionen sind: 4](#_Toc26444797)

[2.2 Management Summary 4](#_Toc26444798)

[2.3 Projektteam 4](#_Toc26444799)

[2.4 Glossar 5](#_Toc26444800)

[3 Ausgangslage (IST), Problembereiche 5](#_Toc26444801)

[3.1 Beschreibung der Ausgangslage 5](#_Toc26444802)

[3.2 Problembereiche und Schwachstellen 5](#_Toc26444803)

[4 Ziele (SOLL) 6](#_Toc26444804)

[4.1 Beschreibung der Ziele 6](#_Toc26444805)

[4.2 Produktperspektive, Nutzen 6](#_Toc26444806)

[4.3 Zielkonflikte 6](#_Toc26444807)

[4.4 Abgrenzung 6](#_Toc26444808)

[5 Anforderungsanalyse 7](#_Toc26444809)

[5.1 Identifizierung der Akteure 7](#_Toc26444810)

[5.2 Anforderungskatalog 8](#_Toc26444811)

[A: Funktionale Anforderungen 8](#_Toc26444812)

[NF: Nichtfunktionale Anforderungen 9](#_Toc26444813)

[6 Systemablaufmodelle (Aktivitäten) 10](#_Toc26444814)

[6.1 Aktivität "Spiel" 10](#_Toc26444815)

[7 Risiko-Analyse 11](#_Toc26444816)

[7.1 Risikokatalog 11](#_Toc26444817)

[8 Anhang 12](#_Toc26444818)

[8.1 Termine 12](#_Toc26444819)

# Revisionen

|  |  |  |
| --- | --- | --- |
| **Datum** | **Name** | **Kommentar** |
| 05.12.2019 | Luca Aeberhard | Dokument erstellt. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Einleitung

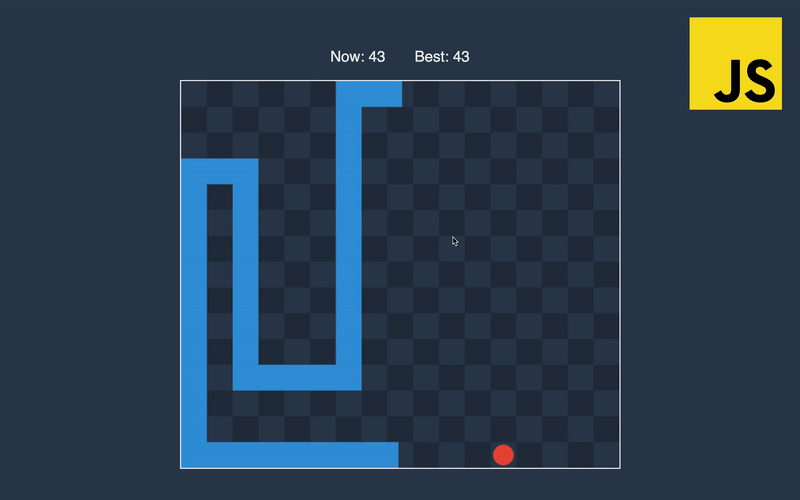
## Systemidee

### Die wichtigsten Funktionen sind:

* Funktion 1: Man kann das Spiel Snake starten.
* Funktion 2: Man kann die Schlange nach rechts, links, oben und unten bewegen.
* Funktion 3: Man kann «Äpfel» einsammeln, dann wächst die Schlange.

## Management Summary

Snake ist ein einfaches Spiel, bei dem man eine Schlange in einem Feld nach rechts, links, oben und unten bewegen. Wenn man die Äpfel einsammelt, welche auf dem Feld generiert werden, wächst die Schlange. Wenn man mit der Schlange sich selbst berührt, dann ist das Spiel zu ende. Es gibt Varianten von dem Spiel, bei dem man auch stirbt, wenn man die Wand berührt. In gewissen Varianten kann man in die Wand hineinfahren und dann kommt man auf der gegenüberliegenden Wand wieder raus.



## Projektteam

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Kürzel** | **Funktion** | **Kontakt (Telefon, Mail...)** |
| Urs Nussbaumer | Nuu | Projektleiter | 041 371 24 28 urs.nussbaumer@ict-bz.ch |
| Luca Aeberhard | lua | Programmierer | 079 734 39 78 luca.aeberhard@gmx.net |
| Alessio Vangelisti | alv | Tester | 079 120 35 25 alessio.vangelisti04@gmail.com |

## Glossar

Die im Projekt verwendeten Fachbegriffe und Abkürzungen in alphabetischer Reihenfolge.

|  |  |
| --- | --- |
| **Begriff** | **Beschreibung** |
| SRS | Software Requirement Specification (Anforderungsspezifikation nach IEEE) |
| SQL | Structured Query Language |
| Snake | Computerspiel, bei dem man eine Schlange steuert und Äpfel einsammeln muss damit diese wächst. Wenn man die Wand oder sich selber berührt hat man verloren. |
| Score | Punktzahl welche man im Spiel erreicht hat. |
|  |  |
|  |  |

# Ausgangslage (IST), Problembereiche

In diesem Kapitel wird die Ausgangslage beschrieben und es werden die Problembereiche identifiziert, die sich in der heutigen Situation zeigen.

## Beschreibung der Ausgangslage

Ich stehe jetzt am Anfang meines Projekts.

## Problembereiche und Schwachstellen

Bis jetzt habe ich noch keine Probleme. Probleme, welche sich ergeben könnten sind z.B. Probleme mit dem Code oder mit der Ausführung des Spiels.

# Ziele (SOLL)

In diesem Kapitel werden die übergeordneten Ziele beschrieben, die mit dem zu entwickelnden System erreicht werden sollen.

## Beschreibung der Ziele

Das übergeordnete Ziel ist es ein funktionsfähiges Snake-Spiel zu programmieren.

1. Das erste Ziel ist es eine Spielwelt zu erstellen.
2. Das zweite Ziel ist, dass man die Schlange in alle Himmelsrichtungen steuern kann.
3. Das dritte Ziel ist, dass wenn man die Wand oder sich selber berührt, man stirbt.
4. Das vierte Ziel ist, dass die Schlange wächst, wenn man Äpfel einsammelt.
5. Das fünfte Ziel ist, dass man zwischen verschiedenfarbigen Schlangen zur Auswahl hat.
6. Das sechste Ziel ist, dass man mit hohen Scores neue Schlangen freischalten kann.

## Produktperspektive, Nutzen

Der Nutzen dieses Spiels ist, dass man einen Zeitvertreib hat der Spass macht.

## Zielkonflikte

Es wird wahrscheinlich keine Zielkonflikte geben, da ich alleine an diesem Projekt arbeite.

## Abgrenzung

Ich will keine App für mein Spiel programmieren.

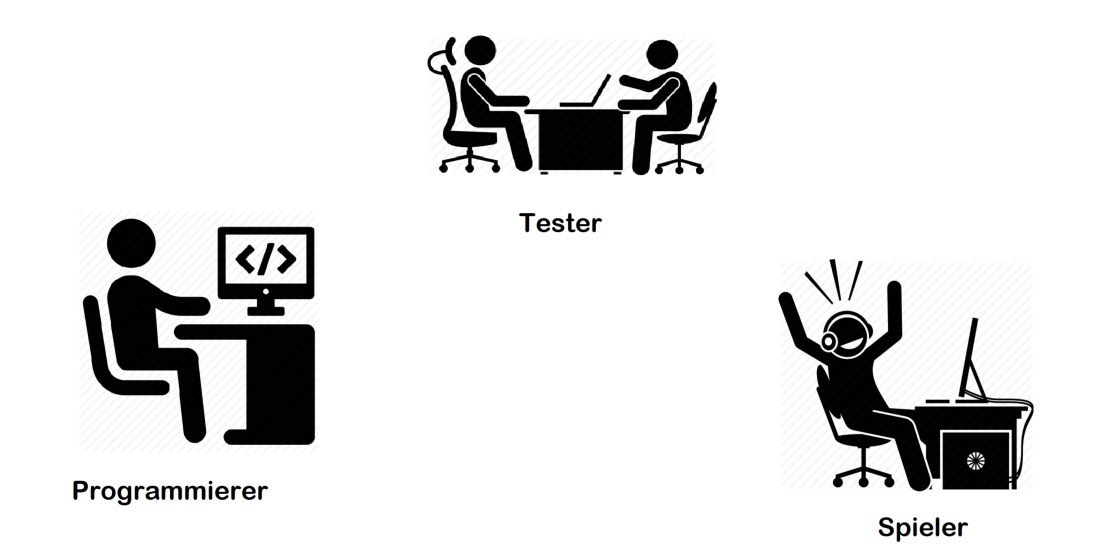
# Anforderungsanalyse

Die Anforderungen an das zu entwickelnde System definieren alle zu erfüllenden Eigenschaften oder die zu erbringende Leistung, sowie allfällige technische Vorgaben und weitere Forderungen des Kunden im Zusammenhang mit dem (den) zu erstellenden Produkt(en).

## Identifizierung der Akteure

Ein Akteur bezeichnet eine Rolle, die mit dem System interagiert.

* **Spieler:** Der Spieler kann das Spiel starten und spielen.
* **Entwickler:** Der Entwickler hat das Spiel programmiert.
* **Tester:** Der Tester darf das Spiel testen.



## Anforderungskatalog

Der Anforderungskatalog ist eine priorisierte Liste, die alles enthält, was im zu entwickelnden Produkt enthalten sein soll. Es wird unterschieden zwischen funktionalen und nicht-funktionalen Anforderungen.

Jede Anforderung ist in Alltagssprache, in der Form einer "User-Story", formuliert, mit einer ID zur späteren Identifizierung versehen sowie mittels 3 verschiedener Kategorien priorisiert: 1 = hohe Prorität, 2 = mittlere Priorität, 3 = keine Priorität. Diese drei Prioritäten repräsentieren die Verpflichtungen "must", "should" und "nice to have".

Im Verlauf der Zeit können neue Anforderungen hinzukommen und/oder bestehende Anforderungen können wegfallen. Anforderungen, die wegfallen, sind im Dokument zu belassen und als ~~gestrichen~~ zu markieren.

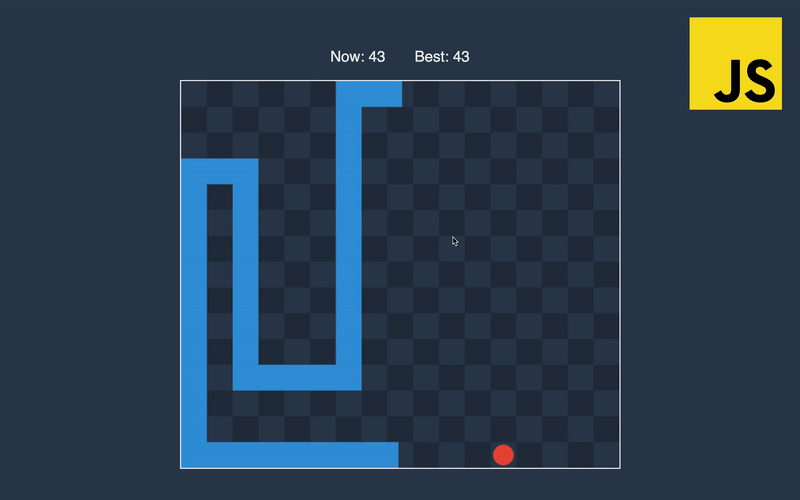
Hinter einer Anforderung kann ihn eckigen Klammern in der Form [Z#] eine Zusatzinformation hinterlegt werden und mit [F#] bzw. [F#, F#, …] können eine oder mehrere offene Fragen referenziert werden.

### A: Funktionale Anforderungen

|  |  |  |
| --- | --- | --- |
| **ID** | **Anforderung** | **Prio** |
| A001 | Als Benutzer möchte ich eine Spielwelt mit einer Schlange sehen. [Z1] | 1 |
| A002 | Als Benutzer möchte ich meine Schlange nach links, recht, oben und unten bewegen | 1 |
| A003 | Als Benutzer möchte ich mit meiner Schlange «Äpfel» einsammeln können und dann wächst meine Schlange. | 2 |
| A004 | Als Benutzer möchte ich einen schönen Hintergrund sehen (Bild). | 2 |
| A005 | Als Benutzer möchte ich zwischen verschiedenfarbigen Schlangen aussuchen können. | 3 |
| A006 | Als Benutzer möchte ich verschiedenfarbige Schlangen mit hohen Scores freischalten. [Z2] | 3 |
|  |  |  |
|  |  |  |

#### Offene Fragen

#### Zusatzinformationen

[Z1] Die Spielwelt sollte am Anfang etwa so aussehen:  
  


[Z2] Mit Score ist die Punktzahl welche man in dem Spiel erreicht hat gemeint.

### NF: Nichtfunktionale Anforderungen

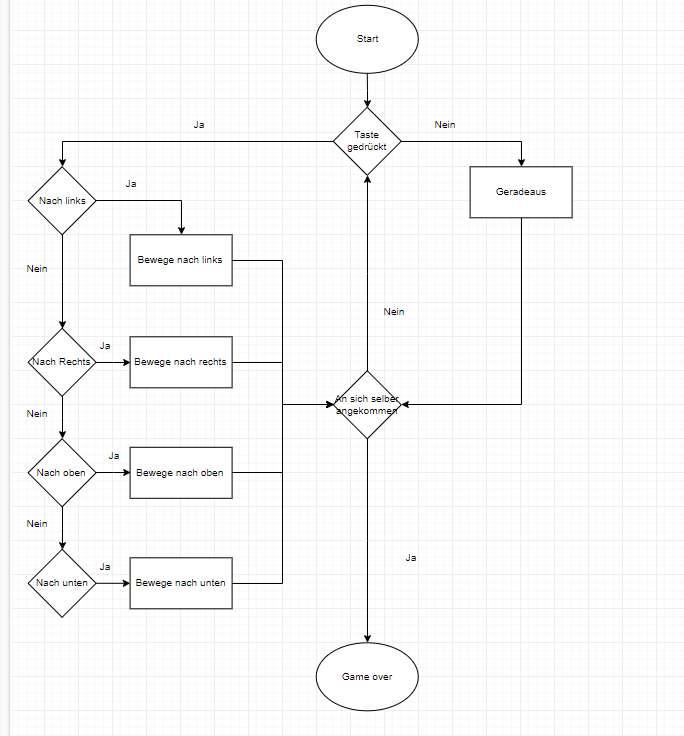
|  |  |  |
| --- | --- | --- |
| **ID** | **Anforderung** | **Prio** |
| NF001 | Die Zeitplanung muss regelmässig geführt werden. | 1 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Systemablaufmodelle (Aktivitäten)

Dieses Kapitel zeigt die wichtigsten und/oder komplexesten funktionalen Anforderungen in ihrem Ablauf. Dazu werden die einzelnen Aktivitätsschritte detailliert analysiert. Für die Darstellung der einzelnen Aktivitätsschritte werden Aktivitätsdiagramme nach UML verwendet.

## Aktivität "Spiel"

Das folgende Flussdiagramm zeigt den Ablauf des Spiels.



# Risiko-Analyse

Ziel der Risiko-Analyse ist es, die Kontrolle zu haben über die wesentlichen Projektrisiken während des Projektverlaufs sowie die Identifizierung der grössten respektive der schwerwiegendsten Risiken, welche dann bei der Implementation als erstes angegangen/getestet werden (Greatest Risk First).

## Risikokatalog

Legende:

* **w** = Wahrscheinlichkeit des Auftretens: 1 = null, 2 = klein, 4 = eher klein 6=mittel 8=hoch, 10 = sehr hoch, >50%
* **b** = Bedeutung der Auswirkungen: 1 keine, 3 minim, 6 signifikant, 8 schwer, 10 fatal, Abbruch
* **p** = Prävention, Behebung, Vermeidung: 1 gratis, 3 leicht, 4 gut 90%, 6 mittel, 8 schlecht,10 unvermeidbar

Das grösste Risiko steht am Anfang des Katalogs, das geringste am Ende.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr: Risiko** | R.1: Das Spiel funktioniert am Ende nur lokal. | **w \* b \* p** | 4 \* 6 \* 4 (96) |
| Ursache | Das Spiel wird nur lokal gespeichert. | | |
| Folgen | Man kann das Spiel nicht über die Seite des BLJ spielen. ei | | |
| Massnahmen | Spiel verlinken oder in Datei speichern. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr: Risiko** | R.2: Es funktionieren nicht alle Tastaturbefehle im Spiel | **w \* b \* p** | 6 \* 10\* 6 (360) |
| Ursache | Der Programmcode ist nicht richtig geschrieben. | | |
| Folgen | Man kann das Spiel nicht richtig spielen. | | |
| Massnahmen | Der Code muss so geschrieben sein, dass alle Funktionen funktionieren. | | |

# Anhang

## Termine

|  |  |  |
| --- | --- | --- |
| **Datum** | **Ziel, Beschreibung, Meilenstein** | **Verantwortlich** |
| 13.12.2016 | Fertigstellung Iteration 1, Prototyp dem Kunden zeigen (Sitzung) |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |