

## Mathias Jesse, Daniel Leustik

## Daniel Leitner, Friedemann Zindler, Petra Tschinderle

Projektarbeit für Software Engineering II - Gruppe A

Alpen Adria Universität Klagenfurt

Universitätsstraße 65-67, 9020 Klagenfurt am Wörthersee

# Teammitglieder

## Programmierer und Tester

Daniel Leitner

**Matrikelnummer:** 1360320

**Studium:** Angewandte Informatik

Friedemann Zindler

**Matrikelnummer:** 1361463

**Studium:** Angewandte Informatik

## Programmierer und GUI

Daniel Leustik

**Matrikelnummer:** 1460379

**Studium:** Angewandte Informatik

## Grafiker und GUI

Mathias Wolfgang Jesse

**Matrikelnummer:** 1460119

**Studium:** Informationsmanagement

## Do­ku­men­ta­rist & Testerin

Petra Tschinderle

**Matrikelnummer:** 1460117

**Studium:** Informationsmanagement

# Stundenliste

## 1 Vorbereitung

## Mathias Jesse

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 07.03.2016 | 1 Stunde | Kennenlernen, Projektbesprechung |
| 08.03.2016 | 1 Stunde | Logo |
| 11.04.2016 | 1,5 Stunde | Product Backlog, Burn Down Charts, Testplan |

## Daniel Leustik

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 07.03.2016 | 1 Stunde | Kennenlernen, Projektbesprechung |
| 10.04.2016 | 0,25 Stunden | Aktualisierung der Anforderungen aus Kundengespräch |
| 11.04.2016 | 1,5 Stunde n | Product Backlog, Burn Down Charts, Testplan |

## Daniel Leitner

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 07.03.2016 | 1 Stunde | Kennenlernen, Projektbesprechung |
| 15.03.2016 | 1 Stunde | Android Programmierung |
| 09.04.2016 | 1 Stunde | User Stories, Android Projekt erstellt |
| 11.04.2016 | 1,5 Stunden | Product Backlog, Burn Down Charts, Testplan |

## Friedemann Zindler

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 07.03.2016 | 1 Stunde | Kennenlernen, Projektbesprechung |
| 11.04.2016 | 1,5 Stunden | Product Backlog, Burn Down Charts, Testplan |

## Petra Tschinderle

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 07.03.2016 | 1 Stunde | Kennenlernen, Projektbesprechung |
| 08.03.2016 | 1 Stunde | Dokumentation: Deckblatt, Teamübersicht, Stundenliste, Fragen fürs Kundengespräch |
| 11.03.2016 | 1 Stunde | Dokumentation: Glossar: SCRUM, Android-Programmierung |
| 11.04.2016 | 1,5 Stunden | Product Backlog, Burn Down Charts, Testplan |

## 1rster Sprint

## Mathias Jesse

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 13.04.2016 | 1 Stunde | Angebot erstellen, Weekly-Meeting |
| 19.04.2016 | 2 Stunden | Erste Sprites erstellen + Hintergrund |
| 20.04.2016 | 1 Stunde | Meeting und Nachbesprechung |
| 22.04.2016 | 2 Stunden | Einzelne Spielfeldelemente isolieren und Größe anpassen |
| 24.04.2016 | 3 Stunden | Charakterauswahl-Bilder |

## Daniel Leustik

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 13.04.2016 | 1 Stunde | Angebot erstellen, Weekly-Meeting |
| 20.04.2016 | 2 Stunden | Meeting und Nachbesprechung, Setup libgdx |
| 21.04. | 1 Stunde | Setup libgdx |
| 25.04. | 5 Stunden | Startscreen erstellen |
| 26.04 | 6 Stunden | Spielboard implementieren |

## Daniel Leitner

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 13.04.2016 | 1 Stunde | Angebot erstellen, Weekly-Meeting |
| 13.04.2016 | 4 Stunden | Spielfeld Logik Prototyp |
| 20.04.2016 | 1 Stunde | Meeting und Nachbesprechung |
| 23.04.2016 | 4 Stunden | Charakterauswahl |

## Friedemann Zindler

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 13.04.2016 | 1 Stunde | Angebot erstellen, Weekly-Meeting |
| 13.04.2016 | 4 Stunden | Spielfeld Logik Prototyp |

## Petra Tschinderle

|  |  |  |
| --- | --- | --- |
| **Datum** | **Zeitaufwand** | **Themen** |
| 13.04.2016 | 1 Stunde | Angebot erstellen, Weekly-Meeting |
| 20.04.2016 | 2 Stunden | Meeting, Installation, Einlesen in AndroidStudio |
| 26.04.2016 | 1 Stunde | BurnDownChart, Stundenliste, Fragen zum Kundengespräch |

# Fragen für das Kundengespräch

## Betriebssystem

Welche Android-Version soll das Spiel haben?

Welche Android-API-Version?

## Layout

Wie soll das Design vom Spiel aussehen? Unser Vorschlag ist…

Soll es Querformat oder Hochformat spielbar sein?

## Plattform

Auf welchen Geräten soll das Spiel anwendbar sein?

Welche Displaygröße?

Wie soll der Multiplayer aussehen? (Lokal oder Online)

Soll das Spiel über Wlan, Lan oder Bluetooth laufen?

Was passiert, wenn ein Spieler vor Spielschluss aus dem Spiel aussteigt?

Wie viele Spieler sollen möglich sein?

Welche Sprachen soll das Spiel haben?

Sollen wir die Möglichkeit bieten, Werbung zu intrigieren?

Soll das Spiel die Möglichkeit haben, den Spielstand zu speichern?

## Gesten und Sprachen

Wie soll die Sprachfunktion verwendet werden?

Soll das Spiel auch andere Funktionen haben?

Soll es Soundeffekte geben?

## Sonstiges

Wie sollen wir die Qualität des Projektes aufrechterhalten? (JUnit-Tests, Metriken)

Was soll die App alles können? Soll man nur spielen können? Oder soll es auch eine Message-Funktion haben?

Wie soll gewürfelt werden? Wie soll die Figur oder der Stein bewegt werden?

Welche Funktionen soll das Spiel beinhalten?

Wer ist die Zielgruppe?

Welches Ziel verfolgen wir?

Was soll bei Fehlern passieren? (Programmintern)

Was wäre, wenn wir das Projekt nicht fertigstellen können?

Wie definieren sie Erfolg in diesem Projekt? Oder wie würde Erfolg für sie aussehen?

Gibt es etwas, dass in diesem Gespräch noch nicht gesagt wurde, aber wichtig ist?

# Product Backlog

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Beschreibung** | **Story Points** | **Priorität** | **Notizen** | **Verantwortlichkeit** |
| 1 | Als Spieler will ich ein neues Spiel beginnen können. | 3 | 1 | App starten - Startscreen | Daniel Leustik |
| 2 | Als Spieler will ich mit anderen Spielern gemeinsam spielen können. | 20 | 3 | WLAN Kommunikation. Sessions. | Daniel Leitner |
| 3 | Als Spieler will ich meine Spielfiguren wählen können. | 3 | 3 | Welche Figur man spielt. | Friedemann Zindler |
| 4 | Als Spieler will ich würfeln und meine Spielfiguren bewegen können. | 8 | 2 | Inklusive Zugmechanismus. | Mathias Jesse/Friedemann Zindler |
| 5 | Als Spieler will ich einen Stein bewegen können, wenn ich auf dessen Feld ziehe. | 2 | 2 | Stein und seine Funktion. | Petra Tschinderle |
| 6 | Als Spieler will ich gewinnen, wenn meine Spielfigur das Zielfeld erreicht. | 2 | 3 | Gewinnerscreen - Soundeffekte | Daniel Leustik |
| 7 | Als Spieler will ich Schummeln können. | 5 | 3 | Lügen beim Würfeln. | Mathias Jesse |
| 8 | Als User sollen mir die Regeln klar ersichtlich sein. | 1 | 4 | Overlay | Petra Tschinderle |
| 9 | Als User möchte ich ein interaktives Spielbrett. | 5 | 2 | Mögliche Zugpositionen. | Friedemann Zindler |
|  | **Gesamt:** | **49** |  |  |  |

**Priorität** 1-5. Schulnotensystem

|  |  |
| --- | --- |
| **ID** | **Constraint:** |
| 11 | Als User möchte ich das Spiel auf allen gängigen Geräten spielen können. |
| 12 | Als User möchte ich das Spiel ohne größere Verzögerungen spielen können. |
| 13 | Als User will ich mich im Spiel zurecht finden. |
| 14 | Als Spieler möchte ich einen gesicherten Spielablauf. |
| 15 | Als Spieler will ich eine gute Oberfläche um übersichtlich spielen zu können. |

|  |  |
| --- | --- |
| **Gesamtkosten**: |  |
| 50 Story Points \* 10h/Point \* 120€/h | € 60.000 |

# Testplan

Mit diesen Kriterien, Metriken und Methoden wollen wir die Qualität unseres Produktes sichern bzw. messen und steigern.

**Kriterien**

* Dokumentieren

**Metriken**

* LinesOfCode
* Anzahl der verwendeten Grafiken
* Weighted methods per class
* CBO and RFC

**Methoden**

* J-Unit Tests
* Plug-Ins

# Burn Down Chart

## Planung:

|  |  |  |
| --- | --- | --- |
| **Sprint** | **Woche** | **Points:** |
|  |  | 49 |
| 1 | 13.04.2016 | 43 |
|  | 20.04.2016 | 37 |
|  | 27.04.2016 | 31 |
| 2 | 04.05.2016 | 25 |
|  | 11.05.2016 | 19 |
|  | 18.05.2016 | 13 |
| 3 | 25.05.2016 | 7 |
|  | 01.06.2016 | 1 |
|  | 13.06.2016 | 0 |



# 1 Sprint

## 2 Woche

|  |  |  |
| --- | --- | --- |
| **Sprint** | **Woche** | **Points:** |
|  |  | 49 |
| 1 | 13.04.2016 | 49 |
|  | 20.04.2016 | 43 |
|  | 27.04.2016 | 37 |
| 2 | 04.05.2016 | 31 |
|  | 11.05.2016 | 25 |
|  | 18.05.2016 | 19 |
| 3 | 25.05.2016 | 13 |
|  | 01.06.2016 | 7 |
|  | 13.06.2016 | 1 |

**Erledigt**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Beschreibung** | **Story Points** | **Priorität** | **Notizen** | **Verantwortlichkeit** |
| 1 | Als Spieler will ich ein neues Spiel beginnen können. | 3 | 1 | App starten - Startscreen | Daniel Leustik |
| 3 | Als Spieler will ich meine Spielfiguren wählen können. | 3 | 3 | Welche Figur man spielt. | Friedemann Zindler |
|  |  | **6** |  |  |  |



# Android-Programmierung

## Grundlagen

Android Apps werden in Java geschrieben. Der Code wird in ein Android Package (APK) compiliert. Das APK File wird von Android-Geräten benötigt, um die App zu installieren.

* Android-Versionen
  + Da es verschiedene Versionen von Android gibt, ist es wichtig die App auf diese hinreichend zu entwickeln. Hierbei wird bei der Erstellung die minimale SDK Version definiert und die Target SDK Version. Die minimale Version beschreibt welches Betriebssystem die Endgeräte mindestens haben müssen, wohingegen die Target Version aussagt für welches System es programmiert wurde und somit optimiert ist.
* Emulator
  + Es wäre viel zu aufwendig für jeden Test die Applikation neu auf ein Endgerät zu exportieren, insofern werden von den meisten Programmierumgebung Emulatoren angeboten, welche die Testung am PC möglich machen.
* Rechtesystem
  + Jede App läuft in ihrer eigenen Welt. Alle Ressourcen bekommen vom System Rechte (Permissions), sodass nur berechtigte Apps auf jeweilige Ressourcen zugreifen können. Will also eine App auf fremde Ressourcen zugreifen, so muss sie um Erlaubnis fragen, um die Permissions zu erhalten. Eine solche Permission könnte ein SMS-Dienst sein.

## Begriffe

* Activity
  + Activities beinhalten das User Interface und sind für die Kommunikation zwischen App und Endnutzer verantwortlich (=Fenster). Jede App besitzt ein eigenes Activity.
* Layouts
  + Ein definiertes Aussehen für eine bestimmte Gruppe von Geräten.
* Permission
  + Rechte um auf Bereiche außerhalb der Applikation zuzugreifen.
* Sandbox
  + Eigenes Verzeichnis der App, auf welches nur die App zugreifen kann. Will die App auf Ressourcen außerhalb der Sandbox zugreifen, so muss sie nach den Permissions fragen.
* Services
  + Prinzipiell ist es möglich eine Applikation im Hintergrund des Endgerätes laufen zu lassen. Im Hintergrund bedeutet, dass der Benutzer nicht aktiv darauf zugreifen muss, damit der Prozess weiterläuft. Dies kann für verschiedenste Zwecke eingesetzt werden.
* Content Provider
  + Der Content Provider verwaltet den Zugriff auf die Daten der Applikation. Er kapselt die Daten und stellt Mechanismen für die Datensicherheit bereit. Er repräsentiert die Standard Schnittstelle für andere Prozesse, um auf diese Daten zugreifen zu können.
* Intent
  + Ein Intent ist ein Nachrichten-Objekt, welches man benötigt, um Aktionen einer anderen App-Komponente anzufordern.

## Konzepte

* Layouts
  + Um die App auf möglichst vielen Geräten mit unterschiedlichen Dimensionen bestmöglich darzustellen, müssen Layouts angelegt werden. Ein Layout ist nichts anderes als eine Ansicht auf das Programm. Somit kann für Tablets die Ansicht z.B. etwas weiträumiger ausfallen, wohingegen die Ansicht am Handy Seiten besitzt um die Inhalte aufzuteilen.
* Bilder
  + In Android sollten Bilder prinzipiell in mindestens vier Größenordnungen gespeichert werden: xhdpi, hdpi, mdpi und ldpi. Dies ist notwendig um die bestmögliche Darstellung auf allen Endgeräten zu gewähren. Dafür gibt es für jede Größe vorgesehene Ordner, in welche die Bilder gespeichert werden sollten. Zudem sollten diese Bilder Vektorgrafiken sein um eine schnellstmögliche Anpassung durchzuführen.
* Sprachen
  + Eine Erleichterung die Android bietet ist es, dass Texte prinzipiell nie 1:1 in Layouts gespeichert werden. Durch diese Tatsache ist es möglich durch einfaches Setzen einer Variablen die Sprache zu ändern. Textteile werden nun aus eigens dafür vorgesehenen XML-Files gelesen und eingesetzt.

**Quelle**: <http://developer.android.com/>

# Glossar

## SCRUM

Scrum ist ein Vorgehensmodell des Projekt- und Produktmanagements und ist insbesondere ein Vorgehensmodell zur agilen Softwareentwicklung.

Scrum besteht nur aus wenigen Regeln. Diese Regeln definieren fünf Aktivitäten, drei Artefakte und drei Rollen, die den Kern von Scrum ausmachen. Die Regeln sind im Agile Atlas oder Scrum Guide definiert. Das Scrum-Framework muss durch Techniken für die Umsetzung der Aktivitäten, Artefakte und Rollen konkretisiert werden, um Scrum tatsächlich umsetzen zu können.

## Scrum Team

* **Product Owner**
  + Der Product Owner ist eine Person die für die Eigenschaften und den wirtschaftlichen Erfolg des Produkts verantwortlich ist Er erstellt, priorisiert und erläutert die zu entwickelnden Produkteigenschaften, und er urteilt darüber, welche Eigenschaften am Ende eines Sprints fertiggestellt wurden.
* **Entwicklungsteam**
  + Das Entwicklungsteam ist für die Lieferung der Produktfunktionalitäten in der vom Product Owner gewünschten Reihenfolge verantwortlich. Zudem trägt es die Verantwortung für die Einhaltung der vereinbarten Qualitätsstandards. Das Entwicklungsteam organisiert sich selbst. Es lässt sich von niemandem, auch nicht vom Scrum Master, vorschreiben, wie es Backlogeinträge umzusetzen hat. Ein Entwicklungsteam sollte in der Lage sein, das Ziel eines jeweiligen Sprints ohne größere äußere Abhängigkeiten zu erreichen.
* **Scrum Master**
  + Der Scrum Master ist dafür verantwortlich, dass Scrum gelingt. Dazu arbeitet er mit dem Entwicklungsteam zusammen, gehört aber selbst meist nicht dazu. Er führt die Scrum-Regeln ein und überprüft deren Einhaltung, er moderiert die Treffen und kümmert sich um die Behebung von Störungen und Hindernissen. Ein Scrum Master ist gegenüber dem Entwicklungsteam eine dienende Führungskraft. Er gibt einzelnen Team-Mitgliedern keine Arbeitsanweisungen. Weder beurteilt er sie, noch belangt er sie disziplinarisch.

## [Artefakte](https://de.wikipedia.org/wiki/Scrum#Artefakte)

* [**Product Backlog**](https://de.wikipedia.org/wiki/Scrum#Product_Backlog)
  + Das Product Backlog ist eine geordnete Auflistung der Anforderungen an das Produkt. Das Product Backlog ist dynamisch und wird ständig weiterentwickelt. Alle Arbeit, die das Entwicklungsteam erledigt, muss seinen Ursprung im Product Backlog haben. Der Product Owner ist für die Pflege des Product Backlogs verantwortlich. Er verantwortet die Reihenfolge bzw. Priorisierung der Einträge
  + Die Anforderungen im Product Backlog sollten nicht technisch, sondern fachlich und anwenderorientiert sein. Eine Möglichkeit, um diese Sichtweise zu unterstützen, ist die Formulierung der Produkteigenschaften als User Stories.
    - **USER STORIES:**
      * Jedes Backlog Item ist eine User Story
        + Wert für den Kunden / Product Owner
        + Keine technischen Details (Kundensprache)
        + Iterative Weiterentwicklung
        + Gute Planungsgröße
        + Weg vom Schreiben hin zum Sprechen.
* [**Sprint Backlog**](https://de.wikipedia.org/wiki/Scrum#Sprint_Backlog)
  + Das Sprint Backlog ist der aktuelle Plan der für einen Sprint zu erledigenden Aufgaben. Es umfasst die Product Backlog-Einträge, die für den Sprint ausgewählt wurden, und die dafür nötigen Aufgaben. Das Sprint Backlog wird laufend nach der Erledigung einer (Teil-)Aufgabe von den Team-Mitgliedern aktualisiert. Dies dient zur Übersicht des aktuellen Bearbeitungsstands.
* [**Product Increment**](https://de.wikipedia.org/wiki/Scrum#Product_Increment)
  + Das Inkrement ist die Summe aller Product-Backlog-Einträge, die während des aktuellen und allen vorangegangenen Sprints fertiggestellt wurden. Am Ende eines Sprints muss das neue Inkrement in einem nutzbaren Zustand sein und der Definition of Done entsprechen.

## [Aktivitäten](https://de.wikipedia.org/wiki/Scrum#Aktivit.C3.A4ten)

* [**Sprint Planning**](https://de.wikipedia.org/wiki/Scrum#Sprint_Planning)
  + Im Sprint Planning werden zwei Fragen beantwortet:
    - **Was kann im kommenden Sprint entwickelt werden?**
      * Das gesamte Scrum Team arbeitet im ersten Teil der Planung daran, ein gemeinsames Verständnis für die im Sprint zu erledigende Arbeit zu entwickeln. Dabei werden die Eigenschaften und die Akzeptanzkriterien besprochen, beispielsweise die Gebrauchstauglichkeit. Außerdem einigt sich der Product Owner mit dem Entwicklungsteam auf die Kriterien, die am Ende des Sprints darüber entscheiden, ob die neue Funktionalität fertig ist oder nicht
    - **Wie wird die Arbeit im kommenden Sprint erledigt?**
      * Im zweiten Teil der Sprint Planung plant das Entwicklungsteam im Detail, welche Aufgaben (Tasks) zum Erreichen des Sprintziels und zur Lieferung der prognostizierten Product-Backlog-Einträge notwendig sind. Diese Planung macht das Entwicklungsteam, wobei der Product Owner für Fragen in Reichweite sein sollte.
* [**Daily Scrum**](https://de.wikipedia.org/wiki/Scrum#Daily_Scrum)
  + Zu Beginn eines jeden Arbeitstages trifft sich das Entwicklerteam zu einem max. 15-minütigen Daily Scrum, bei dem Scrum Master und Product Owner häufig anwesend, jedoch nicht aktiv beteiligt sind, falls sie nicht selbst Backlogelemente bearbeiten. Zweck des Daily Scrum ist der Informationsaustausch. Im Daily Scrum werden keine Probleme gelöst – vielmehr geht es darum, sich einen Überblick über den aktuellen Stand der Arbeit zu verschaffen.
  + **Im Daily Scrum werden drei Fragen beantwortet**:
    - Was hast du getan?
    - Was wirst du morgen tun?
    - (Was ist dazwischengekommen?)
* [**Sprint Review**](https://de.wikipedia.org/wiki/Scrum#Sprint_Review)
  + Das Sprint Review steht am Ende des Sprints. Hier überprüft das Scrum Team das Inkrement, um das Product Backlog bei Bedarf anzupassen. Das Entwicklungsteam präsentiert seine Ergebnisse und es wird überprüft, ob das zu Beginn gesteckte Ziel erreicht wurde. Das Scrum Team und die Stakeholder besprechen die Ergebnisse und was als Nächstes zu tun ist. Im Sprint Review ist die Beteiligung von Kunden und Anwendern wichtig, da diese die fertige Funktionalität des Inkrements benutzen und validieren können. Hieraus ergibt sich wichtiges Feedback für die weitere Produktgestaltung.
* [**Sprint Retrospektive**](https://de.wikipedia.org/wiki/Scrum#Sprint_Retrospektive)
  + Die Sprint Retrospektive steht am Ende eines Sprints. Hierbei überprüft das Scrum Team seine bisherige Arbeitsweise, um sie in Zukunft effizienter und effektiver zu machen. Der Scrum Master unterstützt das Scrum Team darin, gute Praktiken und Verbesserungen zu finden, die im nächsten Sprint umgesetzt werden. Die Retrospektive ist eine gemeinsame Aktivität des Scrum Teams. Das Team soll seine Arbeitsweise offen und ehrlich überprüfen können. Dazu müssen Kritik und unangenehme Wahrheiten offen geäußert werden können.
* [**Product Backlog Refinement**](https://de.wikipedia.org/wiki/Scrum#Product_Backlog_Refinement)
  + Das Product Backlog Refinement ist ein fortlaufender Prozess, bei dem der Product Owner und das Entwicklungsteam gemeinsam das Product Backlog weiterentwickeln. Hierzu gehören:
    - Ordnen der Einträge
    - Löschen von Einträgen, die nicht mehr wichtig sind
    - Hinzufügen von neuen Einträgen
    - Detaillieren von Einträgen
    - Zusammenfassen von Einträgen
    - Schätzen von Einträgen
    - Planung von Releases

**Quelle**: https://de.wikipedia.org/wiki/Scrum#Product\_Owner