

Softwarepraktikum

Annie-Isabell Langenfeld, Frank Schüssele, Tobias Kolzer

October 22, 2026

[Büro von Mitarbeiter S., Donnerstag 1400, Wochenendstimmung]

Chef:S., wir müssen ein Computerspiel für die WiKe AG
(Wichtiger Kunde AG) entwickeln!

S.:Ein Computerspiel? Aber das haben wir noch nie gemacht!

Chef:Stimmt, aber Sie und ihr Team bekommen das hin.
Man soll sich ja nie etwas Neuem gegenüber verschließen.

[Büro von Mitarbeiter S., Donnerstag 1400, Wochenendstimmung]

Chef:S., wir müssen ein Computerspiel für die WiKe AG
(Wichtiger Kunde AG) entwickeln!

S.:Ein Computerspiel? Aber das haben wir noch nie gemacht!

Chef:Stimmt, aber Sie und ihr Team bekommen das hin.
Man soll sich ja nie etwas Neuem gegenüber verschließen.

Im Softwarepraktikum ...

- ▶ sind Sie Mitarbeiter **S.**
- ▶ sind die Dozenten die Vertreter der WiKe AG (Wichtiger Kunde AG)
- ▶ simulieren wir ein Softwareentwicklungsprojekt

Mitarbeiter S. beginnt mit der Arbeit am Projekt.
Die ersten Schritte sind ...

Mitarbeiter S. beginnt mit der Arbeit am Projekt.
Die ersten Schritte sind ...

- ▶ die Domäne analysieren und verstehen

Mitarbeiter S. beginnt mit der Arbeit am Projekt.
Die ersten Schritte sind ...

- ▶ die Domäne analysieren und verstehen
- ▶ um die Anforderungen zu erheben zu und verstehen

Mitarbeiter S. beginnt mit der Arbeit am Projekt.
Die ersten Schritte sind ...

- ▶ die Domäne analysieren und verstehen
- ▶ um die Anforderungen zu erheben zu und verstehen
- ▶ um die Umsetzung zu konzipieren

Mitarbeiter S. beginnt mit der Arbeit am Projekt.
Die ersten Schritte sind ...

- ▶ die Domäne analysieren und verstehen
- ▶ um die Anforderungen zu erheben zu und verstehen
- ▶ um die Umsetzung zu konzipieren
- ▶ um das benötigte Produkt zu implementieren.

- ▶ Organisatorisches
- ▶ Vorgehen
- ▶ Gesamttablauf
- ▶ Vorgehensmodell: Scrum
- ▶ Scrum im Sopra
- ▶ Die ersten Schritte

Organisatorisches

- ▶ Selbständiges Einarbeiten in ein Gebiet
- ▶ Arbeiten im Team
- ▶ Umgang mit Komplexität
- ▶ Anwendung softwaretechnischer Prinzipien

- ▶ Selbständiges Einarbeiten in ein Gebiet
- ▶ Arbeiten im Team
- ▶ Umgang mit Komplexität
- ▶ Anwendung softwaretechnischer Prinzipien

Wir setzen voraus, dass Sie bereits programmieren können!

- ▶ **Tutoren**

Thierry Meiers, Luis Schaffner, Lasse Liebler,
Dejan Dudukovic, Mirjana Stifter, Yannic Kooistra, Dennis Hämmerle, Arved Steuer

- ▶ **Dozenten**

Frank Schüssele, Tobias Kolzer, Annie-Isabell Langenfeld

- ▶ **Infrastruktur**

Daniel Dietsch, Tobias Kolzer

- ▶ Gruppen mit 5-7 Studenten
- ▶ 6 ECTS (180h) über 17 Wochen
 - ▶ 14 Arbeitsabschnitte (Sprints), je eine Woche
- ▶ 4 Vorlesungen
- ▶ Termine
 - ▶ 4 Abgaben
 - ▶ 3 Präsentationen (Postersession) (Do. 14-18 Uhr)
 - ▶ 1 Besprechung mit den Dozenten
 - ▶ Wöchentliches Gruppentreffen (Mo-Do)

- ▶ **Serviceübersicht**
- ▶ **Fragen und Informationen**
Gruppenmitglieder, Tutoren, Wiki, Discourse, Sprechstunde und Mattermost
- ▶ **Primärdienste**
Gitea, Git, Jenkins, Sonar, Dashboard, Mailinglisten (`sopra-crew@...`, `sopraXX@...`)
- ▶ **Sekundärdienste**
Mattermost
- ▶ **Werkzeuge**
C#, .NET 9, Monogame 3.8, Rider, (Visual Studio Community), Resharper



Mitarbeit

- ▶ Kontinuierliche Mitarbeit während der Sprints:
 - ▶ **Commits** im Git-Repository **und**
 - ▶ **Aktivität** (Tickets, Kommentare, etc.) in Gitea
 - ▶ Es darf max. 2 Sprints nicht mitgearbeitet werden
- ▶ Im Durchschnitt Aufgaben mit min. **7h geschätzter Arbeitszeit** pro Sprint erledigt
- ▶ Abgabe der Hausaufgabe (Programm)

Mitarbeit

- ▶ Kontinuierliche Mitarbeit während der Sprints:
 - ▶ **Commits** im Git-Repository **und**
 - ▶ **Aktivität** (Tickets, Kommentare, etc.) in Gitea
 - ▶ Es darf max. 2 Sprints nicht mitgearbeitet werden
- ▶ Im Durchschnitt Aufgaben mit min. **7h geschätzter Arbeitszeit** pro Sprint erledigt
- ▶ Abgabe der Hausaufgabe (Programm)

ECTS:

$$\frac{6 * 30h}{180h}$$

Termine:

$$4 * 2h = 8h$$

$$3 * 4h = 12h$$

$$\frac{14 * 2h = 28h}{48h}$$

Arbeitszeit:

$$\frac{(180h - 48)}{14}$$

9,43h

Mitarbeit

- ▶ Kontinuierliche Mitarbeit während der Sprints:
 - ▶ **Commits** im Git-Repository **und**
 - ▶ **Aktivität** (Tickets, Kommentare, etc.) in Gitea
 - ▶ Es darf max. 2 Sprints nicht mitgearbeitet werden
- ▶ Im Durchschnitt Aufgaben mit min. **7h geschätzter Arbeitszeit** pro Sprint erledigt
- ▶ Abgabe der Hausaufgabe (Programm)

Gruppentreffen

- ▶ Anwesenheitspflicht
 - ▶ Es darf max. 1x gefehlt werden

ECTS:

$$\frac{6 * 30h}{180h}$$

Termine:

$$4 * 2h = 8h$$

$$3 * 4h = 12h$$

$$\frac{14 * 2h = 28h}{48h}$$

Arbeitszeit:

$$\frac{(180h - 48)}{14}$$

9,43h

Mitarbeit

- ▶ Kontinuierliche Mitarbeit während der Sprints:
 - ▶ **Commits** im Git-Repository **und**
 - ▶ **Aktivität** (Tickets, Kommentare, etc.) in Gitea
 - ▶ Es darf max. 2 Sprints nicht mitgearbeitet werden
- ▶ Im Durchschnitt Aufgaben mit min. **7h geschätzter Arbeitszeit** pro Sprint erledigt
- ▶ Abgabe der Hausaufgabe (Programm)

Gruppentreffen

- ▶ Anwesenheitspflicht
 - ▶ Es darf max. 1x gefehlt werden

Präsentationen

- ▶ Anwesenheitspflicht

ECTS:

$$\frac{6 * 30h}{180h}$$

Termine:

$$4 * 2h = 8h$$

$$3 * 4h = 12h$$

$$\frac{14 * 2h = 28h}{48h}$$

Arbeitszeit:

$$\frac{(180h - 48h)}{14}$$
$$\mathbf{9,43h}$$

Note

Berechnet sich aus:

- ▶ 50% Endprodukt
- ▶ 50% Einzelnote

Endprodukt bewertet nach Kriterien:

F eatures
A rtefakte
U sability
S pass
T echdemo

Einzelnote

Pro Sprint bekommt jeder Studierende 5 Punkte

- ▶ Zugeteilte Arbeit **sinnvoll** erledigt?
- ▶ Note ergibt sich aus Punkten

Mitarbeit (cont.)

Ihre Mitarbeit wird ausschließlich aufgrund der erbrachten **Eigenleistung** bewertet.

Eigenarbeit erkennbar machen

- ▶ Bibliotheken und
- ▶ Copy-Paste Code referenzieren
- ▶ Einsatz von Generativer KI angeben und Eigenarbeit kenntlich machen
- ▶ (Grafik-Assets sind davon ausgenommen)

Mitarbeit (cont.)

Ihre Mitarbeit wird ausschließlich aufgrund der erbrachten **Eigenleistung** bewertet.

Eigenarbeit erkennbar machen

- ▶ Bibliotheken und
- ▶ Copy-Paste Code referenzieren
- ▶ Einsatz von Generativer KI angeben und Eigenarbeit kenntlich machen
- ▶ (Grafik-Assets sind davon ausgenommen)

Lernerfahrung

- ▶ Bibliotheken müssen verstanden werden
- ▶ Fremdcode lesen, verstehen
- ▶ KI als Codingpartner/Doku nutzen und
- ▶ Rückfragen zum Code stellen.

Vorgehen

Domäne

Unter einer (Problem)-Domäne versteht man ein bestimmtes Einsatzgebiet für Computersysteme oder Software.

- ▶ Die Domäne analysieren und verstehen
- ▶ um die Anforderungen zu erheben und verstehen
- ▶ um die Umsetzung zu konzipieren
- ▶ um das benötigte Produkt zu implementieren.

es Problemfeld oder

Domäne

Unter einer (Problem)-Domäne versteht man [...] in der Softwaretechnik ein abgrenzbares Problemfeld oder einen bestimmten Einsatzbereich für Computersysteme oder Software.

Domäne

Unter einer (Problem)-Domäne versteht man [...] in der Softwaretechnik ein abgrenzbares Problemfeld oder einen bestimmten Einsatzbereich für Computersysteme oder Software.

- ▶ **Entität**, d.h. welche Objekte gibt es in der Domäne?
- ▶ **Funktionen**, d.h. welche kontinuierlich messbaren Größen gibt es in der Domäne?
- ▶ **Ereignisse**, d.h. welche Ereignisse treten in der Domäne auf?
- ▶ **Interaktionen**, d.h. wie können Elemente der Domäne interagieren?

Domäne

Unter einer (Problem)-Domäne versteht man [...] in der Softwaretechnik ein abgrenzbares Problemfeld oder einen bestimmten Einsatzbereich für Computersysteme oder Software.

- ▶ **Entität**, d.h. welche Objekte gibt es in der Domäne?
- ▶ **Funktionen**, d.h. welche kontinuierlich messabren Größen gibt es in der Domäne?
- ▶ **Ereignisse**, d.h. welche Ereignisse treten in der Domäne auf?
- ▶ **Interaktionen**, d.h. wie können Elemente der Domäne interagieren?
- ▶ Aus Sicht verschiedener **Stakeholder**
 - ▶ Stakeholder sind in irgendeiner Weise von der Domäne betroffen
 - ▶ z.B.: Auftraggeber, Programmierer/Spieledesigner, **Freiwillige Selbstkontrolle (FSK)**, **Hardwarehersteller**

Domäne

Unter einer (Problem)-Domäne versteht man [...] in der Softwaretechnik ein abgrenzbares Problemfeld oder einen bestimmten Einsatzbereich für Computersysteme oder Software.

- ▶ **Entität**, d.h. welche Objekte gibt es in der Domäne?
- ▶ **Funktionen**, d.h. welche kontinuierlich messabren Größen gibt es in der Domäne?
- ▶ **Ereignisse**, d.h. welche Ereignisse treten in der Domäne auf?
- ▶ **Interaktionen**, d.h. wie können Elemente der Domäne interagieren?

▶ Aus Sicht

- ▶ Stakeholder
- ▶ z.B.:

Welche der folgenden Aussagen ist korrekt?

- ▶ Ein Hotel hat unendlich viele Zimmer.
- ▶ Ein Hotel hat Zimmer die man buchen kann.
- ▶ Ein Hotel wiegt ca. 2.1 g.
- ▶ Ein Hotel ist 400 km hoch.

Hersteller

Domäne

Unter einer (Problem)-Domäne versteht man [...] in der Softwaretechnik ein abgrenzbares Problemfeld oder einen bestimmten Einsatzbereich für Computersysteme oder Software.

- ▶ **Entität**, d.h. welche Objekte gibt es in der Domäne?
- ▶ **Funktionen**, d.h. welche kontinuierlich messabren Größen gibt es in der Domäne?
- ▶ **Ereignisse**, d.h. welche Ereignisse treten in der Domäne auf?
- ▶ **Interaktionen**, d.h. wie können Elemente der Domäne interagieren?

▶ Aus Sicht

- ▶ Stakeholder
- ▶ z.B.:

Welche der folgenden Aussagen ist korrekt?

- ▶ Ein Hotel hat unendlich viele Zimmer.
- ▶ Ein Hotel hat Zimmer die man buchen kann.
- ▶ Ein Hotel wiegt ca. 2.1 g.
- ▶ Ein Hotel ist 400 km hoch.

Mathematik

Architektur hersteller

(Brett-)Spieldesign

Science-Fiction

Beispiel: Brettspiel



- ▶ Entität
- ▶ Funktionen
- ▶ Ereignisse
- ▶ Interaktionen

Beispiel: Brettspiel



- ▶ Entität
Spielfigur, Würfel, Spielbrett, Spieler, Verlag,...
- ▶ Funktionen
- ▶ Ereignisse
- ▶ Interaktionen

Beispiel: Brettspiel



- ▶ **Entität**
Spielfigur, Würfel, **Spielbrett**, **Spieler**, **Verlag**,...
- ▶ **Funktionen**
Punktestand, Position, Spieldauer,
Spielermotivation, (**Flow**), ...
- ▶ **Ereignisse**
- ▶ **Interaktionen**

Beispiel: Brettspiel



- ▶ **Entität**
Spielfigur, Würfel, **Spielbrett**, **Spieler**, **Verlag**,...
- ▶ **Funktionen**
Punktestand, Position, Spieldauer,
Spielermotivation, (**Flow**), ...
- ▶ **Ereignisse**
Spielstart, Feld erreicht, Karte ziehen, ...
- ▶ **Interaktionen**

Beispiel: Brettspiel



- ▶ **Entität**
Spielfigur, Würfel, **Spielbrett**, **Spieler**, **Verlag**,...
- ▶ **Funktionen**
Punktestand, Position, Spieldauer,
Spielermotivation, (**Flow**), ...
- ▶ **Ereignisse**
Spielstart, Feld erreicht, Karte ziehen, ...
- ▶ **Interaktionen**
Spieler verrückt Spielfigur, Geld bezahlen/empfangen, Karte aufdecken, **Tisch umstoßen**, ...

Beispiel: Brettspiel



- ▶ **Entität**
Spielfigur, Würfel, **Spielbrett**, **Spieler**, **Verlag**,...
- ▶ **Funktionen**
Punktestand, Position, Spieldauer,
Spielermotivation, (**Flow**), ...
- ▶ **Ereignisse**
Spielstart, Feld erreicht, Karte ziehen, ...
- ▶ **Interaktionen**
Spieler verrückt Spielfigur, Geld bezahlen/empfangen, Karte aufdecken, **Tisch umstoßen**, ...

Was ist mit dem **Drumherum**?

Vier grundlegende Elemente eines Spiels¹

- ▶ Ästhetik
- ▶ Handlung
- ▶ Mechanik
- ▶ Technologie

¹Jesse Schell, The Art of Game Design, 2008, ISBN 978-0-12-369496-6

Vier grundlegende Elemente eines Spiels¹

- ▶ Ästhetik
 - ▶ Handlung
 - ▶ Mechanik
 - ▶ Technologie
-
- ▶ Jedes Element lässt sich weiter zerlegen
 - ▶ Ästhetik z.B.: Aussehen, Stimmung, Metaphern
 - ▶ Handlung z.B.: Akte, Spannungskurve, Ereignisse in der Handlung
 - ▶ Mechanik z.B.: Spielobjekte und Interaktion (Entitäten, Funktionen, Ereignisse, Interaktionen)
 - ▶ Technologie z.B.: Engine, Bibliotheken, Computergrafik

¹Jesse Schell, The Art of Game Design, 2008, ISBN 978-0-12-369496-6

Vier grundlegende Elemente eines Spiels¹

- ▶ Ästhetik
 - ▶ Handlung
 - ▶ Mechanik
 - ▶ Technologie
-
- ▶ Jedes Element lässt sich weiter zerlegen
 - ▶ Ästhetik z.B.: Aussehen, Stimmung, Metaphern
 - ▶ Handlung z.B.: Akte, Spannungskurve, Ereignisse in der Handlung
 - ▶ Mechanik z.B.: Spielobjekte und Interaktion (Entitäten, Funktionen, Ereignisse, Interaktionen)
 - ▶ Technologie z.B.: Engine, Bibliotheken, Computergrafik
 - ▶ Das Spiel sollte aus Sicht jedes dieser Elemente betrachtet werden.

¹Jesse Schell, The Art of Game Design, 2008, ISBN 978-0-12-369496-6

Anforderungen²

Anforderungen sind eine Aussage, die erwartet wird.

- ▶ Die Domäne analysieren und verstehen
- ▶ um die Anforderungen zu erheben und zu verstehen
- ▶ um die Umsetzung zu konzipieren
- ▶ um das benötigte Produkt zu implementieren.

Eigenschaften

²Helmut Balzert, Lehrbuch der Softwaretechnik - Basiskonzepte und Requirementsengineering, 2009, ISBN 978-3-8274-1705-3

Anforderungen²

Anforderungen sind eine Aussage darüber, was man von einem (Software)-System als Eigenschaften erwartet.

²Helmut Balzert, Lehrbuch der Softwaretechnik - Basiskonzepte und Requirementsengineering, 2009, ISBN 978-3-8274-1705-3

Anforderungen²

Anforderungen sind eine Aussage darüber, was man von einem (Software)-System als Eigenschaften erwartet.

Gewöhnlich werden 3 Arten von Anforderungen unterschieden:

- ▶ **Funktionale Anforderungen** legen eine vom Softwaresystem oder einer seiner Komponenten bereitzustellende Funktionen [...] fest.
- ▶ **Qualitätsanforderungen** beschreiben zusätzliche Eigenschaften, die diese Funktionen haben sollen.
- ▶ **Rahmenbedingungen** legen organisatorische und/oder technische Restriktionen für das Softwaresystem und/oder den Entwicklungsprozess fest.

²Helmut Balzert, Lehrbuch der Softwaretechnik - Basiskonzepte und Requirementsengineering, 2009, ISBN 978-3-8274-1705-3

Anforderungen

- ▶ 2D- oder 3D-Grafik (kein ASCII)
- ▶ Soundeffekte und Musik
- ▶ mindestens einen menschlich Spieler
- ▶ Echtzeit
- ▶ Eigenes Menü (komplett mit der Maus steuerbar)
- ▶ Pausefunktion
- ▶ Speichern und Laden

Chef: Die WiKe AG hat die folgenden Anforderungen geschickt.
Das Spiel soll ...

Chef: Die WiKe AG hat die folgenden Anforderungen geschickt.
Das Spiel soll ...

- ▶ 2D- oder 3D-Grafik (kein ASCII)
- ▶ Soundeffekte und Musik
- ▶ mindestens einen menschlich Spieler
- ▶ Echtzeit
- ▶ Eigenes Menü (komplett mit der Maus steuerbar)
- ▶ Pausefunktion
- ▶ Speichern und Laden
- ▶ Spielobjekte
 - ▶ Kontrollierbare Spielobjekte,
 - ▶ Auswählbare Spielobjekte,
 - ▶ Nicht-kontrollierbare Spielobjekte,
 - ▶ Kollidierende Spielobjekte und
 - ▶ Kontrollierbare, kollidierende und bewegliche Spielobjekte (im Folgenden aktive 'Spielobjekte' genannt).

Chef: Die WiKe AG hat die folgenden Anforderungen geschickt.
Das Spiel soll ...

- ▶ 2D- oder 3D-Grafik (kein ASCII)
- ▶ Soundeffekte und Musik
- ▶ mindestens einen menschlich Spieler
- ▶ Echtzeit
- ▶ Eigenes Menü (komplett mit der Maus steuerbar)
- ▶ Pausefunktion
- ▶ Speichern und Laden
- ▶ Spielobjekte
 - ▶ Kontrollierbare Spielobjekte,
 - ▶ Auswählbare Spielobjekte,
 - ▶ Nicht-kontrollierbare Spielobjekte,
 - ▶ Kollidierende Spielobjekte und
 - ▶ Kontrollierbare, kollidierende und bewegliche Spielobjekte (im Folgenden aktive 'Spielobjekte' genannt).
- ▶ mindestens 5 Statistiken
- ▶ Achievements
- ▶ Das Spiel muss eine Techdemo bereitstellen. Die Techdemo muss von jedem im Spiel vorhandenen Spielobjekt ein Exemplar enthalten, es müssen mindestens 1000 aktive Spielobjekte vorhanden sein. Die Techdemo muss ebenfalls performant laufen.
- ▶ Allen aktiven Spielobjekten muss es möglich sein, von jedem beliebigen Punkt in der Welt zu jedem anderen begehbaren Punkt zu gelangen, ohne sich gegenseitig übermäßig zu behindern, festzustecken usw.

Alternative 1

- ▶ Anzahl Spielobjekte:
 - ▶ Kategorie a: 5
 - ▶ Kategorie b: 5
 - ▶ Kategorie c: 5
 - ▶ Kategorie d: 3
 - ▶ Kategorie e: 3
- ▶ mindestens 10 verschiedene Aktionen (inkl. Laufen, Fähigkeiten usw.) enthalten.
- ▶ zweiten Spieler
- ▶ Spielfiguren müssen indirekt gesteuert werden.
- ▶ Spielobjekte müssen sich intelligent verhalten.

Alternative 2

- ▶ Anzahl Spielobjekte:
 - ▶ Kategorie a: 5
 - ▶ Kategorie b: 5
 - ▶ Kategorie c: 5
 - ▶ Kategorie d: 3
 - ▶ Kategorie e: 3
- ▶ mindestens 10 Aktionen
- ▶ mehr als zwei menschliche Spieler
- ▶ Spielobjekte müssen sich intelligent verhalten.

S.: Wir sollten uns bald entscheiden, davon hängt das weitere Design ab!

Alternative 3

- ▶ Anzahl Spielobjekte:
 - ▶ Kategorie a: 5
 - ▶ Kategorie b: 5
 - ▶ Kategorie c: 5
 - ▶ Kategorie d: 3
 - ▶ Kategorie e: 2
- ▶ mindestens 10 Aktionen
- ▶ zweiter Spieler
- ▶ realitätsnahe Physik:
 - ▶ Schwerkraft
 - ▶ zerstörbare Weltobjekte
 - ▶ Objekte die geworfen werden können

Qualitätsanforderungen

- ▶ Entwickeln Sie ein **gutes** Produkt
- ▶ Qualität der Grafik ist nicht relevant muss aber in sich stimmig sein
- ▶ Akustische Effekte sollen in sich stimmig sein

Chef: Und finden Sie heraus, was das alles bedeuten soll.

Rahmenbedingungen

- ▶ C# und/oder F# mit .NET 9
- ▶ MonoGame 3.8
- ▶ Lauffähig auf Windows 10/11 (x64) und Linux
- ▶ Visual Studio Community/Rider
- ▶ Die in den Git-branches release und master liegende Version muss immer kompilierbar und lauffähig sein.
- ▶ Keine Warnings oder Errors vom Compiler (wöchentlich), keine Buildfehler.

Anforderungen · Beispiele



Anforderungen · Gegenbeispiele und Grenzfälle



Game Design Document

Beschreibung der **wesentlichen** Merkmale des Spiels

- ▶ Die **Domäne** analysieren und verstehen
- ▶ um die **Anforderungen** zu erheben und zu verstehen
- ▶ um die **Umsetzung** zu konzipieren
- ▶ um das benötigte Produkt zu implementieren.

Game Design Document (GDD)

Beschreibung der **wesentlichen Merkmale** des Spiels.

Game Design Document (GDD)

Beschreibung der **wesentlichen Merkmale** des Spiels.

- ▶ Lastenheft
 - ▶ Dokumentiert die Anforderungen und Rahmenbedingungen eines Produktes
 - ▶ Aus Sicht des **Anwenders** oder **Kunden**
 - ▶ Beinhaltet keine Details über die technische Umsetzung
- ▶ Hilft bei Aufwandsabschätzung und Organisation
- ▶ Dokumentation wie die Anforderungen umgesetzt werden
Ästhetik, Handlung, Mechanik und Technologie
- ▶ Details: GDD-Vorlesung, Wiki

Ablauf

Woche	Organi- sation	Ent- wurf	MS 01	MS 02	MS 03	MS 04	MS 05	Was?	Wann und Wo?
0								- Vorlesung "Organisation und Prozess" (Einführungsveranstaltung) - Gruppeneinteilung abwarten	- Einführungsveranstaltung: 22.10., 14:00 - 16:00, Geb. 82, HS 00-006 - Fragebogen: 22.10. bis 23:59 - Gruppen ab 24.10. in Gitea
1								- Vorlesung "Game Design Document (GDD)" Abgabe Hausaufgabe	- Vorlesung: 29.10., 14:00 - 16:00, Geb. 82, HS 00-006 - Anschließend Fragestunde, Computerpool Abgabe: 31.10 bis 23:59
2								- Vorlesung "Grundlagen Softwarearchitektur" - Wiederkehrende Aufgaben: Product Owner Abgabe GDD	- Vorlesung: 05.11., 14:00-16:00, Geb. 82, HS 00-006 - Anschließend Fragestunde, Computerpool Abgabe: 07.11. bis 23:59



- ▶ Ziel: **Funktionierende** und **kompetente** Gruppen zusammenstellen
- ▶ Fragebogen: Name, Emails, Fähigkeiten
- ▶ Wir teilen Sie in Gruppen ein
- ▶ Einteilung vor Sonntag
- ▶ **Sonntag** Terminumfrage für Gruppentreffen beantworten

- ▶ Ziel: **Funktionierende** und **kompetente** Gruppen zusammenstellen
- ▶ Fragebogen: Name, Emails, Fähigkeiten
- ▶ Wir teilen Sie in Gruppen ein
- ▶ Einteilung vor Sonntag
- ▶ **Sonntag** Terminumfrage für Gruppentreffen beantworten

Bis heute Abend, 23:59

- ▶ Ziel: **Werkzeuge** und **Vorgehen** kennenlernen und **Probleme** frühzeitig aufdecken
- ▶ Beschreibung auf dem Wiki
- ▶ Ungefähr:
 - ▶ Werkzeuge installieren und testen
 - ▶ Dienste testen
 - ▶ Git und Gitea kennenlernen
 - ▶ Texte zu Clean Code lesen
 - ▶ Ein MonoGame Programm schreiben
- ▶ Die Hausaufgabe ist der erste Sprint d.h. es können 5 Punkte erreicht werden



- ▶ Ziel: **Werkzeuge** und **Vorgehen** kennenlernen und **Probleme** frühzeitig aufdecken
- ▶ Beschreibung auf dem Wiki
- ▶ Ungefähr:
 - ▶ Werkzeuge installieren und testen
 - ▶ Dienste testen
 - ▶ Git und Gitea kennenlernen
 - ▶ Texte zu Clean Code lesen
 - ▶ Ein MonoGame Programm schreiben
- ▶ Die Hausaufgabe ist der erste Sprint d.h. es können 5 Punkte erreicht werden

Chef: Aber was ist eigentlich ein "gutes Produkt"?



Meilenstein

Ein **Meilenstein** ist ein überprüfbares Ziel, dass zu einem Termin erreicht werden soll

- ▶ Meilensteine teilen den Projektverlauf in konkrete Schritte ein
 - ▶ Je eine **Etappen mit überprüfbaren Zwischenzielen**.
 - ▶ Erleichtert damit die **Projektplanung** und
 - ▶ Kontrolle des **Projektfortschritts**

Meilenstein

Ein **Meilenstein** ist ein überprüfbares Ziel, dass zu einem Termin erreicht werden soll

- ▶ Meilensteine teilen den Projektverlauf in konkrete Schritte ein
 - ▶ Je eine **Etappen mit überprüfbaren Zwischenzielen**.
 - ▶ Erleichtert damit die **Projektplanung** und
 - ▶ Kontrolle des **Projektfortschritts**
- ▶ Entwicklung gegliedert in **5 Meilensteine**
 - ▶ Referenzablauf ermöglicht das Erkennen von Problemen
 - ▶ **Müssen** entsprechend des Spielkonzepts modifiziert werden
- ▶ Vermeiden Sie **Regressionen**

Meilenstein

Ein **Meilenstein** ist ein überprüfbares Ziel, dass zu einem Termin erreicht werden soll

- ▶ Meilensteine teilen den Projektverlauf in konkrete Schritte ein
 - ▶ Je eine **Etappen mit überprüfbaren Zwischenzielen**.
 - ▶ Erleichtert damit die **Projektplanung** und
 - ▶ Kontrolle des **Projektfortschritts**
- ▶ Entwicklung gegliedert in **5 Meilensteine**
 - ▶ Referenzablauf ermöglicht das Erkennen von Problemen
 - ▶ **Müssen** entsprechend des Spielkonzepts modifiziert
- ▶ Vermeiden Sie **Regressionen**

Meilenstein #1

(Woche 4)

- Spielobjekt in der Welt bewegbar
- Interaktive Kamera
- Ein Level laden
- Soundausgabe

Meilenstein

Ein **Meilenstein** ist ein überprüfbares Ziel, dass zu einem Termin erreicht werden soll

- ▶ Meilensteine teilen den Projektverlauf in konkrete Schritte ein
 - ▶ Je eine **Etappen mit überprüfbaren Zwischenzielen**.
 - ▶ Erleichtert damit die **Projektplanung** und
 - ▶ Kontrolle des **Projektfortschritts**
 - ▶ Entwicklung gegliedert in **5 Meilensteine**
 - ▶ Referenzablauf ermöglicht das Erkennen von Problemen
 - ▶ **Müssen** entsprechend des Spielkonzepts modifiziert
 - ▶ Vermeiden Sie **Regressionen**
- Meilenstein #1 (Woche 4)**

 - Spielobjekt in der Welt bewegbar
 - Interaktive Kamera
- Meilenstein #5 (Woche 15)**

 - Spiel ist fehlerfrei
 - Spielmechanik ist ausbalanciert

Scrum als Vorgehensmodell

Scrum

- ▶ Gehört zu den **agilen Methoden**
- ▶ Entwicklung in Scrum ist **iterativ** und **inkrementell**
- ▶ Entwicklung wird in **Sprints** aufgeteilt (1 bis 4 Wochen)
- ▶ Es existiert eine **auslieferbare** Software am Ende jedes Sprints
- ▶ Scrum definiert sich über **Rollen**, **Events** und **Artefakte**

Developer

- ▶ Aufgabe: Programmieren, Design, technische Lösung, Projektablauf
- ▶ Verantwortung: Produktqualität, Zeit

Product Owner

- ▶ Aufgabe: Kundengespräche, Vermarktung, Zielvorgabe, Anforderungserhebung
- ▶ Verantwortung: Produktmerkmale, Budget, Markterfolg

Scrum Master

- ▶ Aufgabe: Organisation, Mediation, Prozesskontrolle
- ▶ Verantwortung: Compliance, Prozess

Product Backlog

- ▶ Liste von **Anforderungen** mit abgeleiteten **User Stories**
- ▶ Nach **Wichtigkeit** für den Projekterfolg geordnet

#2: **Anforderung:**

Das Spiel soll sich an Usabilitygrundsätze halten.

Product Backlog

- ▶ Liste von **Anforderungen** mit abgeleiteten **User Stories**
- ▶ Nach **Wichtigkeit** für den Projekterfolg geordnet

#2: **Anforderung:**

Das Spiel soll sich an Usabilitygrundsätze halten.

User Stories

- ▶ **Kompakte Beschreibung** einer Funktion aus Nutzersicht
- ▶ **Aufwandsabschätzung** mit Story Points
- ▶ Als <Rolle> möchte ich <Funktion>, um <Nutzen>.
 - ▶ **Wer** oder **was** wird diese Funktion nutzen?
 - ▶ **Was** ist die Funktion?
 - ▶ **Welchen** Nutzen hat diese Funktion?
- ▶ Manchmal Akzeptanzkriterium
 - ▶ **Was** kann man (neues) tun wenn es fertig ist?

#47: **Speichern**

Als **Spieler** möchte ich **den aktuellen Spielstand zu einem beliebigen Zeitpunkt abspeichern**, um **später weiterzuspielen zu können**.

Points:8

Sprint Backlog

- ▶ Liste an Aufgaben für den aktuellen Sprint
- ▶ Liste von **User Stories** mit abgeleiteten **Tasks**
- ▶ Wird für jeden Sprint neu erstellt

#47: **Speichern**

Als **Spieler** möchte ich den aktuellen Spielstand zu einem beliebigen Zeitpunkt auf der Festplatte **speichern**, um **später weiterzuspielen**.

Points: 8

Task

- ▶ Teilaufgabe einer User Story
- ▶ Innerhalb eines Sprints erfüllbar
- ▶ Aufwandsabschätzung in Personenstunden

Sprint Backlog

- ▶ Liste an Aufgaben für den aktuellen Sprint
- ▶ Liste von **User Stories** mit abgeleiteten **Tasks**
- ▶ Wird für jeden Sprint neu erstellt

#47: **Speichern**

Als **Spieler** möchte ich den aktuellen Spielstand zu einem beliebigen Zeitpunkt auf der Festplatte **speichern**, um **später weiterzuspielen**.

Points: 8

Task

- ▶ Teilaufgabe einer User Story
- ▶ Innerhalb eines Sprints erfüllbar
- ▶ Aufwandsabschätzung in Personenstunden

#47-1 **Savegame Ordner**

Das Spiel legt Savegame-Dateien im entsprechenden Verzeichnis an.

2h

Sprint Backlog

- ▶ Liste an Aufgaben für den aktuellen Sprint
- ▶ Liste von **User Stories** mit abgeleiteten **Tasks**
- ▶ Wird für jeden Sprint neu erstellt

#47: **Speichern**

Als **Spieler** möchte ich den aktuellen Spielstand zu einem beliebigen Zeitpunkt auf der Festplatte **speichern**, um **später weiterzuspielen**.

Points:8

Task

- ▶ Teilaufgabe einer User Story
- ▶ Innerhalb eines Sprints erfüllbar
- ▶ Aufwandsabschätzung in Personenstunden

#47-1

Das Spiel
Savegame
entspre
Verzei

#47-2 **Json Serialiser**

Eigenschaften von
GameObject als JSON
serialisieren.

8h

Sprint Backlog

- ▶ Liste an Aufgaben für den aktuellen Sprint
- ▶ Liste von **User Stories** mit abgeleiteten **Tasks**
- ▶ Wird für jeden Sprint neu erstellt

#47: **Speichern**

Als **Spieler** möchte ich den aktuellen Spielstand zu einem beliebigen Zeitpunkt auf der Festplatte **speichern**, um **später weiterzuspielen**.

Points: 8

Task

- ▶ Teilaufgabe einer User Story
- ▶ Innerhalb eines Sprints erfüllbar
- ▶ Aufwandsabschätzung in Personenstunden

#47-1

Das Sp

#47-2 **Json Serialiser**

#47-5 **Speichern Stresstest**

Speichern von mindestens
> 1500 Objekten auf Windows
und Linux testen.

3h

8h

Product Increment

- ▶ Neue Produktversion am Ende des Sprints
- ▶ Ist **direkt** an den Kunden **auslieferbar**

Definition of Done

- ▶ Definiert wann ein Item als **fertig** angesehen wird
- ▶ Wird vom Team erstellt
- ▶ Darf sich im Laufe des Projekts ändern

Product Increment

- ▶ Neue Produktversion am Ende des Sprints
- ▶ Ist **direkt** an den Kunden **auslieferbar**

Definition of Done

- ▶ Definiert wann ein Item als **fertig** angesehen wird
- ▶ Wird vom Team erstellt
- ▶ Darf sich im Laufe des Projekts ändern

Definition of Done Beispiele

- Team einverstanden (incl. PO)
- Auf *main* eingchecked
- Keine neuen Bugs
- APIs sind dokumentiert
- Tests erfolgreich
- Codestyle eingehalten
- Keine *//TODOs*

Sprint Planning

- ▶ Vor jedem Sprint
- ▶ Länge abhängig von Sprintlänge (ca. 2h je Woche)
- ▶ **Was** wird im nächsten Sprint getan?
 - ▶ **Product Owner** präsentiert die wichtigsten Items aus dem Product Backlog
... am Ende des nächsten Sprints sollte die Kernmechanik vollständig spielbar sein,
damit wir ausprobieren können ob sie Spaß macht.
- ▶ **Wie** wird es umgesetzt?
 - ▶ **Developer** wählen ihre Aufgaben beginnend beim wichtigsten Item
 - ▶ **Developer** zerlegen ausgewählte User Stories in Tasks
 - ▶ **Developer** erstellen neuen Softwareentwurf

Daily Scrum

- ▶ **Tägliches** Treffen
- ▶ Maximal 15 Minuten
- ▶ **Developer** beantworten nacheinander
 - ▶ Was habe ich seit dem letzten Treffen getan?
..., habe Spielfigur jumper angelegt, ...
 - ▶ Was plane ich bis zum nächsten Treffen zu tun?
..., werde jumper heute an die Physik anbinden, ...
 - ▶ Welche Probleme hatte ich und wo benötige ich Hilfe?
..., verstehe aber die Schnittstelle zum RigidBodyCollider nicht.
- ▶ Detailfragen und Lösungssuche gehören **nicht** zum Daily Scrum
 - ▶ Gesondertes Treffen für Detailfragen und Lösungssuche vereinbaren

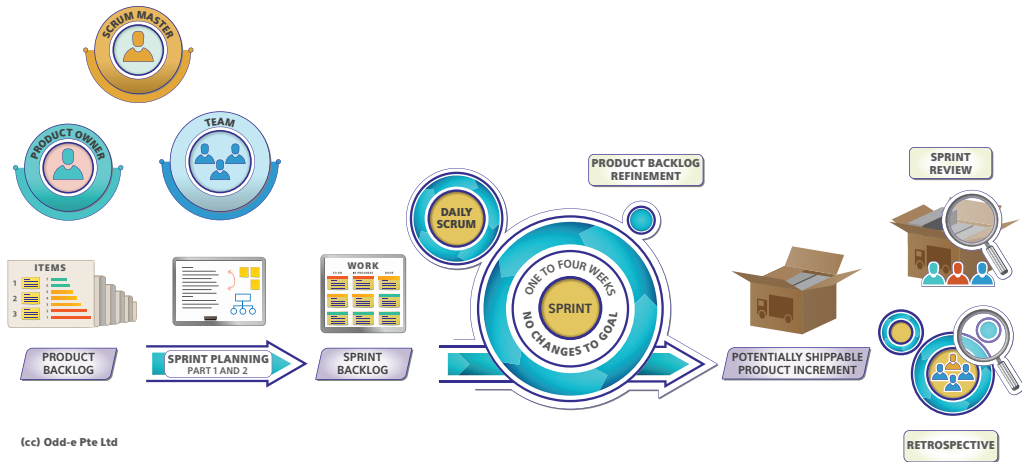
Sprint Review

- ▶ Treffen am Ende des Sprints
- ▶ Länge abhängig von Sprintlänge (ca. 1h je Woche)
- ▶ **Developer** präsentieren das **Product Increment**
... hatte den Task "Spielfigur bewegen", und guckt, jetzt kann ich hier rumrennen und springen. ...
- ▶ **Product Owner** bestimmt welche Tasks und User Stories fertig sind
- ▶ **Product Owner** erklärt, wie gut die Aufwandsabschätzung war

Sprint Retrospective

- ▶ Treffen des Teams **nach dem Sprint Review** und **vor dem Sprint Planning**
- ▶ Länge abhängig von Sprintlänge (ca. 45min je Woche)
- ▶ Mit Ziel, den Prozess zu verbessern
 - ▶ Wie lief es im letzten Sprint hinsichtlich Personen, Beziehungen, Prozessen, Werkzeugen?
 - ▶ Muss die **Definition of Done** angepasst werden?
 - ▶ Wie kann die Arbeitsweise des Teams verbessert werden?
- ▶ Unterstützt durch **Scrum Master**

SCRUM



Scrum im Sopra

- ▶ **Dozenten** sind Kundenvertreter
- ▶ **Tutoren** sind Scrum Master
- ▶ **Sie** sind Developer
- ▶ **Sie** können in Ihrer Gruppe Product Owner werden

- ▶ **Dozenten** sind Kundenvertreter
- ▶ **Tutoren** sind Scrum Master
- ▶ **Sie** sind Developer
- ▶ **Sie** können in Ihrer Gruppe Product Owner werden

Dozenten sind aber vorwiegend Dozenten, denen sie **Fragen stellen können** und die Ihnen helfen (z.B.: Bei Treffen, in Mattermost, oder als Mention in einem Gitea-Ticket)

- ▶ **Dozenten** sind Kundenvertreter
- ▶ **Tutoren** sind Scrum Master
- ▶ **Sie** sind Developer
- ▶ **Sie** können in Ihrer Gruppe Product Owner werden

Dozenten sind aber vorwiegend Dozenten, denen sie **Fragen stellen können** und die Ihnen helfen (z.B.: Bei Treffen, in Mattermost, oder als Mention in einem Gitea-Ticket)

Tutoren fungieren zusätzlich als **Kundenversther** und **Berater**

Product Backlog: List aller im Projekt noch zu erledigenden Items

Product Backlog Item

- ▶ Titel
- ▶ Sinnvolle Beschreibung
- ▶ Wichtigkeit
low, mid, high
- ▶ (Optional) Labels nach Funktion
bug, question, ...
- ▶ (Optional) Zeitschätzung
est: 0h, 1h, 2h, ..., ∞
- ▶ (Optional) Akzeptanzkriterium
Ist fertig wenn...

- ☐  **Productowner Task (Week 4)** est 2 organisation priority high
#72 vor 2 Minuten von vincent geöffnet
- ☐  **Als Entwickler möchte ich GLS-Shader laden können, um transparente/morphende Texturen im Spiel darstellen zu können.** est ∞
kind user story priority low
#71 vor 2 Minuten von vincent geöffnet
- ☐  **Spielernamen mit äüöß crashen das Spiel** bug priority high
#70 vor 4 Minuten von vincent geöffnet
- ☐  **Code zum texturen laden refactoren** est 2 priority low
#69 vor 5 Minuten von vincent geöffnet
- ☐  **Bogenschützen Schussfähigkeit implementieren** est 5 priority mid
#68 vor 5 Minuten von vincent geöffnet
- ☐  **Spielzustand serialisierbar machen** est 3 priority high
#67 vor 6 Minuten von vincent geöffnet

Product Backlog: List aller im Projekt noch zu erledigenden Items

Product Backlog Item

- ▶ Titel
- ▶ **Sinnvolle Beschreibung**
- ▶ Wichtigkeit
low, mid, high
- ▶ (Optional) Labels nach Funktion
bug, question, ...
- ▶ (Optional) Zeitschätzung
est: 0h, 1h, 2h, ..., ∞
- ▶ (Optional) Akzeptanzkriterium
Ist fertig wenn...

- ☐  **Productowner Task (Week 4)** est 2 organisation priority high
#72 vor 2 Minuten von vincent geöffnet
- ☐  **Als Entwickler möchte ich GLS-Shader laden können, um transparente/morphende Texturen im Spiel darstellen zu können.** est ∞
kind user story priority low
#71 vor 2 Minuten von vincent geöffnet
- ☐  **Spielernamen mit äüöß crashen das Spiel** bug priority high
#70 vor 4 Minuten von vincent geöffnet
- ☐  **Code zum texturen laden refactoren** est 2 priority low
#69 vor 5 Minuten von vincent geöffnet
- ☐  **Bogens schützen Schussfähigkeit implementieren** est 5 priority mid
#68 vor 5 Minuten von vincent geöffnet
- ☐  **Spielzustand serialisierbar machen** est 3 priority high
#67 vor 6 Minuten von vincent geöffnet

Product Backlog: List aller im Projekt noch zu erledigenden Items

Product Backlog Item

- ▶ Titel
- ▶ Sinnvolle Beschreibung
- ▶ Wichtigkeit
low, mid, high
- ▶ (Optional) Labels nach Funktion
bug, question, ...
- ▶ (Optional) Zeitschätzung
est: 0h, 1h, 2h, ..., ∞
- ▶ (Optional) Akzeptanzkriterium
Ist fertig wenn...

- ☐  **Productowner Task (Week 4)** est 2 organisation priority high
#72 vor 2 Minuten von vincent geöffnet
- ☐  **Als Entwickler möchte ich GLS-Shader laden können, um transparente/morphende Texturen im Spiel darstellen zu können.** est ∞
kind user story priority low
#71 vor 2 Minuten von vincent geöffnet
- ☐  **Spielernamen mit äüöß crashen das Spiel** bug priority high
#70 vor 4 Minuten von vincent geöffnet
- ☐  **Code zum texturen laden refactoren** est 2 priority low
#69 vor 5 Minuten von vincent geöffnet
- ☐  **Bogenschi...**
#68 vor 5 Minuten von vincent geöffnet
- ☐  **Spielzust...**
#67 vor 6 Minuten von vincent geöffnet

Annahmen:

- ▶ Das Game Design Document ersetzt die **User Stories**.
- ▶ D.h. es können direkt **Items** abgeleitet werden (in Task-Größe).

Sprint Backlog: List aller im Sprint zu erledigenden Items

Sprint Backlog Item

Wie Backlog Item, zusätzlich:

- ▶ Ist dem **Sprint** zugeordnet (in Gitea: "Meilenstein")
- ▶ **Developer** zugeordnet ($n \geq 1$)
- ▶ Zeitschätzung (Gesamtarbeitszeit)
est: 0h, 1h, 2h, ..., 8h

Sprint #5

[Öffnen](#)[Meilenstein bearbeiten](#)[Neues Issue](#)

Am Ende dieses Sprints gibt es ein minimal spielbares Spiel um die Kernmechanik auszuprobieren (Kamera bewegt sich, Tzlatko der Zwerg kann Dinge zerhauen)

Kein Fälligkeitsdatum 44% abgeschlossen

5 Offen

4 Geschlossen

Label

Autor

Zuständig

Typ

Sortieren

Productowner Task (Week 4) est 2 organisation priority high

#72 vor 8 Minuten von vincent geöffnet

Spielernamen mit äüöß crashen das Spiel bug priority high

#70 vor 10 Minuten von vincent geöffnet

Code zum texturen laden refactoren est 2 priority low

#69 vor 11 Minuten von vincent geöffnet

Bogenschützen Schussfähigkeit implementieren est 5 priority mid

#68 vor 12 Minuten von vincent geöffnet

Spielzustand serialisierbar machen est 3 priority high

#67 vor 12 Minuten von vincent geöffnet

Gruppentreffen

Zweistündiges Treffen mit dem Tutor als Moderator und Scrum Master

- | | |
|-------------------------|-------------|
| 1. Sprint Review | (ca. 30min) |
| 2. Sprint Retrospective | (ca. 15min) |
| 3. Sprint Planning | (ca. 45min) |

Gruppentreffen

Zweistündiges Treffen mit dem Tutor als Moderator und Scrum Master

- | | |
|-------------------------|-------------|
| 1. Sprint Review | (ca. 30min) |
| 2. Sprint Retrospective | (ca. 15min) |
| 3. Sprint Planning | (ca. 45min) |

Sprint Review

- ▶ Developer führen Product Increment vor
- ▶ Gruppe entscheidet was gemäß DoD (nicht) erledigt ist
 - ▶ Nicht erledigte Items zurück ins Product Backlog verschieben
- ▶ Wie gut waren die Zeitschätzungen?
- ▶ Ist der nächste Meilenstein erreicht?

Gruppentreffen

Zweistündiges Treffen mit dem Tutor als Moderator und Scrum Master

1. Sprint Review
2. Sprint Retrospective
3. Sprint Planning

(ca. 30min)

Sprint Review

- ▶ Developer führen Product Increment vor
- ▶ Gruppe entscheidet was gemäß DoD
 - ▶ Nicht erledigte Items zurück ins Product Backlog
- ▶ Wie gut waren die Zeitschätzungen?
- ▶ Ist der nächste Meilenstein erreicht?

Sopra DoD

- ▶ Das Item ist in Gitea geschlossen.
- ▶ Im Item sind die geschätzte und die tatsächliche Arbeitszeit eingetragen.
- ▶ Alle für das Item relevanten Dateien sind im aktuellen Stand des remote release Branch integriert.
- ▶ Der Tutor hat die Fertigstellung des Items im Sprint Review anhand des aktuellen Standes des remote release Branch bestätigt.

Gruppentreffen

Zweistündiges Treffen mit dem Tutor als Moderator und Scrum Master

- | | |
|-------------------------|-------------|
| 1. Sprint Review | (ca. 30min) |
| 2. Sprint Retrospective | (ca. 15min) |
| 3. Sprint Planning | (ca. 45min) |

Sprint Retrospective

- ▶ Was lief in diesem Sprint gut oder schlecht?
 - ▶ Probleme mit oder Lob für Gruppenmitglieder, Prozess, Zeiteinteilung oder Tools
 - ▶ Was lief gut? Was lief schlecht? Was muss sich ändern? Was soll so bleiben?
- ▶ Bei Bedarf DoD anpassen
- ▶ Entschlüsse schriftlich festhalten (am Besten in `readme.md`)

Gruppentreffen

Zweistündiges Treffen mit dem Tutor als Moderator und Scrum Master

- | | |
|-------------------------|-------------|
| 1. Sprint Review | (ca. 30min) |
| 2. Sprint Retrospective | (ca. 15min) |
| 3. Sprint Planning | (ca. 45min) |

Sprint Planning

- ▶ Verfügbare Arbeitszeit für diesen Sprint festlegen
- ▶ Product Owner stellt das Ziel des nächsten Sprints vor
- ▶ Product Backlog durchgehen, angefangen bei den wichtigsten Items:
 - ▶ Gemeinsam gewünschte Funktionalität besprechen und Aufwand schätzen
 - ▶ Ins Sprint Backlog verschieben
 - ▶ Wiederholen bis verfügbare Arbeitszeit aufgebraucht
- ▶ Items im Sprint Backlog an die Gruppenmitglieder verteilen

Items werden aus dem GDD

- ▶ Das GDD übernimmt im User Stories
 - ▶ Kapitel: Zusammenfassung (Objekte, Aktionen, Ablauf, ...)
- ▶ Die Domäne analysieren und verstehen
 - ▶ um die Anforderungen zu erheben und zu verstehen
 - ▶ um die Umsetzung zu konzipieren
 - ▶ um das benötigte Produkt zu implementieren.

Items werden aus dem **GDD** abgeleitet

- ▶ Das GDD übernimmt im Sopra die Rolle von **User Stories**
- ▶ Kapitel: Zusammenfassung, Spiellogik (Objekte, Aktionen, Ablauf, ...)

```
[...]. Der Spieler kämpft  
sich durch einen Irrgarten  
aus zufällig generierten  
Räumen mit über die Zeit  
stärker werdenden Fallen,  
Gegnern und Schätzen. Eine  
der Spielfigur folgende  
Todeswand (die Dunkelheit)  
setzt den Spieler dabei  
unter Druck. [...]
```

Items werden aus dem **GDD** abgeleitet

- ▶ Das GDD übernimmt im Sopra die Rolle von **User Stories**
- ▶ Kapitel: Zusammenfassung, Spiellogik (Objekte, Aktionen, Ablauf, ...)

#23 **Objekt Dunkelheit erstellen**

Asset für die Dunkelheit (unendlich breite Wand mit Schatteneffekten) erstellen und im Code verfügbar machen.

?

[...]. Der Spieler kämpft sich durch einen Irrgarten aus zufällig generierten Räumen mit über die Zeit stärker werdenden Fallen, Gegnern und Schätzen. Eine der Spielfigur folgende Todeswand (die Dunkelheit) setzt den Spieler dabei unter Druck. [...]

Items werden aus dem **GDD** abgeleitet

- ▶ Das GDD übernimmt im Sopra die Rolle von **User Stories**
- ▶ Kapitel: Zusammenfassung, Spiellogik (Objekte, Aktionen, Ablauf, ...)

#23 **Objekt Dunkelheit erstellen**

Asset für die Dunkelheit (unendlich breite Wand mit Schatteneffekten) erstellen und im Code verfügbar machen.

?

#24 **Verhalten Dunkelheit implementieren**

Verhaltenskomponente für die Dunkelheit implementieren: verfolgt den Spieler mit zunehmender Geschwindigkeit.

?

[...]. Der Spieler kämpft sich durch einen Irrgarten aus zufällig generierten Räumen mit über die Zeit stärker werdenden Fallen, Gegnern und Schätzen. Eine der Spielfigur folgende Todeswand (die Dunkelheit) setzt den Spieler dabei unter Druck. [...]

Items werden aus dem **GDD** abgeleitet

- ▶ Das GDD übernimmt im Sopra die Rolle von **User Stories**
- ▶ Kapitel: Zusammenfassung, Spiellogik (Objekte, Aktionen, Ablauf, ...)

#23 **Objekt Dunkelheit erstellen**

Asset für die Dunkelheit (unendlich breite Wand mit Schatteneffekten) erstellen und im Code verfügbar machen.

?

#24 **Verhalten Dunkelheit implementieren**

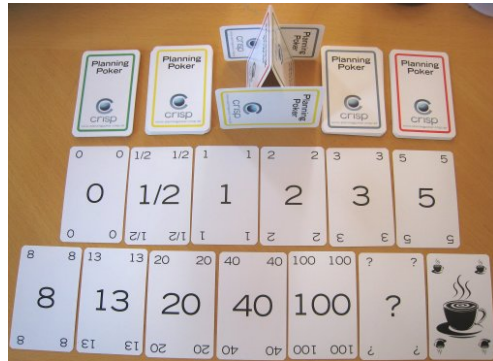
Verhaltenskomponente für die Dunkelheit implementieren: verfolgt den Spieler mit zunehmender Geschwindigkeit.

?

[...]. Der Spieler kämpft sich durch einen Irrgarten aus zufällig generierten Räumen mit über die Zeit stärker werdenden Fallen, Gegnern und Schätzen. Eine der Spielfigur folgende Todeswand (die Dunkelheit) setzt den Spieler dabei unter Druck. [...]

Mit Planning Poker

1. Item besprechen,
jeder soll versteht worum es geht
2. **Verdeckt** wählt jeder einen Wert aus
(Story Points/Stunden)
3. Gleichzeitig aufdecken
4. Wer den höchsten bzw. niedrigsten Wert
aufdeckt, erklärt **warum**
5. Wiederholen bis alle den gleichen Wert zeigen



Mit Planning Poker

1. Item besprechen, jeder soll versteht worum es geht
2. **Verdeckt** wählt jeder einen Wert aus (Story Points/Stunden)
3. Gleichzeitig aufdecken
4. Wer den höchsten bzw. niedrigsten Wert aufdeckt, erklärt **warum**
5. Wiederholen bis alle den gleichen Wert zeigen



#64 Verhalten Dunkelheit implementieren

Verhaltenskomponente für die Dunkelheit implementieren: verfolgt den Spieler mit zunehmender Geschwindigkeit.

Mit Planning Poker

1. Item besprechen,
jeder soll versteht worum es geht
2. **Verdeckt** wählt jeder einen Wert aus
(Story Points/Stunden)
3. Gleichzeitig aufdecken
4. Wer den höchsten bzw. niedrigsten Wert
aufdeckt, erklärt **warum**
5. Wiederholen bis alle den gleichen Wert zeigen



#64 Verhalten Dunkelheit implementieren

Verhaltenskomponente für die Dunkelheit implementieren:
verfolgt den Spieler mit zunehmender Geschwindigkeit. ...
wie ein Anhänger an einem langen Gummiband

5h

#46: **Feuerball** hinzufügen

Die Fähigkeit Feuerball (siehe GDD) als Komponente für Einheiten implementieren.

est:8

#46: **Feuerball** hinzufügen

Die Fähigkeit Feuerball (siehe GDD) als Komponente für Einheiten implementieren.

Akzeptanzkriterien:

- ▶ Es gibt eine (debug) Einheit, die die Fähigkeit Feuerball besitzt.
- ▶ Der Feuerball fliegt graphisch sichtbar auf ein selektiertes Ziel zu (man kann auseichen).
- ▶ Das Ziel bekommt entsprechend der Fähigkeit Feuerball Schaden.

est:8

#46: **Feuerball hinzufügen**

Die Fähigkeit Feuerball (siehe GDD) als Komponente für Einheiten implementieren.

Akzeptanzkriterien:

- ▶ Es gibt eine (debug) Einheit, die die Fähigkeit Feuerball besitzt.
- ▶ Der Feuerball fliegt graphisch sichtbar auf ein selektiertes Ziel zu (man kann auseichen).
- ▶ Das Ziel bekommt entsprechend der Fähigkeit Feuerball Schaden.

est:8

#42: **Räumliche Datenstruktur implementieren**

Effiziente Verwaltung von *IPosition* Objekten.

est:16

#46: Feuerball hinzufügen

Die Fähigkeit Feuerball (siehe GDD) als Komponente für Einheiten implementieren.

Akzeptanzkriterien:

- ▶ Es gibt eine (debug) Einheit, die die Fähigkeit Feuerball besitzt.
- ▶ Der Feuerball fliegt graphisch sichtbar auf ein selektiertes Ziel zu (man kann auseichen).
- ▶ Das Ziel bekommt entsprechend der Fähigkeit Feuerball Schaden.

est:8

#42: Räumliche Datenstruktur implementieren

Effiziente Verwaltung von *IPosition* Objekten.

- ▶ Einfügen eines Objekts per `.Add(IPosition o)`
- ▶ Finden eines Objekts per `.CheckAt(Vector2 v)`
- ▶ Abfragen von `.InFrontOf(Vector2 v)`
- ▶ Updatet wenn sich Objekt bewegt.
- ▶ Performant für Tausend Objekte
Minimum !!!111!!!

est:16

Product Owner (ab Woche 2)

- ▶ Pflege des Product Backlog
 - ▶ Items an Entwicklungsstand **anpassen**, und nach **Wichtigkeit ordnen**
 - ▶ Weitere Items aus GDD extrahieren
- ▶ Gruppentreffen vorbereiten:
 - ▶ Was ist nach **DoD** fertig?
 - ▶ Wie waren die Aufwandsabschätzungen?
 - ▶ Product Increment vorbereiten

2× Qualitätssicherung (ab Woche 3)

- ▶ Code auf Clean Code Richtlinien prüfen
- ▶ Architektur überprüfen
- ▶ Schnittstellen definieren
- ▶ In Ticket dokumentieren
 - ▶ Kurzvorstellung in Sprintreview

- ▶ Alle Recurring Tasks **müssen** verteilt werden
- ▶ Ticket mit Zeitschätzung (max. 2h) je Aufgabe und Sprint

Die ersten Schritte

- ▶ Anforderungen

Bekannte Spiele? Einschränkungen? Naheliegende Mechaniken? Begriffe unklar?

- ▶ Bekanntes verändern und kombinieren

Mechanisch? Setting und Handlung?

- ▶ Sie sind nicht allein!

- ▶ Was ist Ihnen wichtig?
- ▶ Was wollen Sie gar nicht?
- ▶ Reden Sie mit ihrer Gruppe
- ▶ Fragen Sie nach

- ▶ Anforderungen

Bekannte Spiele? Einschränkungen? Naheliegende Mechaniken? Begriffe unklar?

- ▶ Bekanntes verändern und kombinieren

Mechanisch? Setting und Handlung?

- ▶ Sie sind nicht allein!

- ▶ Was ist Ihnen wichtig?
- ▶ Was wollen Sie gar nicht?
- ▶ Reden Sie mit ihrer Gruppe
- ▶ Fragen Sie nach

Ihre Idee . . .

- ▶ muss von der Gruppe verstanden werden.
- ▶ kann verworfen werden.
- ▶ wird sich (stark) verändern.
- ▶ wird zusammen weiterentwickelt.

- ▶ Erzählen sie sich gegenseitig einen **exemplarischen Spielablauf**
 - ▶ Von Anfang an (Drücke auf *game.exe*)
 - ▶ Bis Gewonnen und Verloren
 - ▶ Was passiert wenn?
- ▶ Welche Spielobjekte und Aktionen (Entitäten, Funktionen, Ereignisse, Interaktionen) gibt es?
- ▶ Was ist Ihnen unklar (in der Domäne und der Spielidee)?
- ▶ Lücken im Ablauf Spielablauf füllen

- ▶ Erzählen sie sich gegenseitig einen **exemplarischen Spielablauf**
 - ▶ Von Anfang an (Drücke auf *game.exe*)
 - ▶ Bis Gewonnen und Verloren
 - ▶ Was passiert wenn?
- ▶ Welche Spielobjekte und Aktionen (Entitäten, Funktionen, Ereignisse, Interaktionen) gibt es?
- ▶ Was ist Ihnen unklar (in der Domäne und der Spielidee)?
- ▶ Lücken im Ablauf Spielablauf füllen

Der Spieler kämpft als Krieger in einem Verlies.

- ▶ Erzählen sie sich gegenseitig einen **exemplarischen Spielablauf**
 - ▶ Von Anfang an (Drücke auf *game.exe*)
 - ▶ Bis Gewonnen und Verloren
 - ▶ Was passiert wenn?
- ▶ Welche Spielobjekte und Aktionen (Entitäten, Funktionen, Ereignisse, Interaktionen) gibt es?
- ▶ Was ist Ihnen unklar (in der Domäne und der Spielidee)?
- ▶ Lücken im Ablauf Spielablauf füllen

Der Spieler kämpft als Krieger in einem Verlies.

- Objekte: Verlies und Krieger
- Aktion: Kämpfen
- Hat er ein Schwert? Zauber? Was ist ein Verlies? Gibt es nur Krieger?

- ▶ Erzählen sie sich gegenseitig einen **exemplarischen Spielablauf**
 - ▶ Von Anfang an (Drücke auf *game.exe*)
 - ▶ Bis Gewonnen und Verloren
 - ▶ Was passiert wenn?
- ▶ Welche Spielobjekte und Aktionen (Entitäten, Funktionen, Ereignisse, Interaktionen) gibt es?
- ▶ Was ist Ihnen unklar (in der Domäne und der Spielidee)?
- ▶ Lücken im Ablauf Spielablauf füllen

Der Spieler kämpft als Krieger in einem Verlies.

- Objekte: Verlies und Krieger
- Aktion: Kämpfen
- Hat er ein Schwert? Zauber? Was ist ein Verlies? Gibt es nur Krieger?

Die Dunkelheit verfolgt den Spieler und setzt ihn unter Druck.

- ▶ Erzählen sie sich gegenseitig einen **exemplarischen Spielablauf**
 - ▶ Von Anfang an (Drücke auf *game.exe*)
 - ▶ Bis Gewonnen und Verloren
 - ▶ Was passiert wenn?
- ▶ Welche Spielobjekte und Aktionen (Entitäten, Funktionen, Ereignisse, Interaktionen) gibt es?
- ▶ Was ist Ihnen unklar (in der Domäne und der Spielidee)?
- ▶ Lücken im Ablauf Spielablauf füllen

Der Spieler kämpft als Krieger in einem Verlies.

- Objekte: Verlies und Krieger
- Aktion: Kämpfen
- Hat er ein Schwert? Zauber? Was ist ein Verlies? Gibt es nur Krieger?

Die Dunkelheit verfolgt den Spieler und setzt ihn unter Druck.

- Objekt: Dunkelheit (?)
- Woher weiß der Spieler wie weit die Dunkelheit entfernt ist?
- Müssen dann alle Gänge in die gleiche Richtung gehen?

- ▶ Erzählen sie sich gegenseitig einen **exemplarischen Spielablauf**
 - ▶ Von Anfang an (Drücke auf *game.exe*)
 - ▶ Bis Gewonnen und Verloren
 - ▶ Was passiert wenn?
- ▶ Welche Spielobjekte und Aktionen (Entitäten, Funktionen, Ereignisse, Interaktionen) gibt es?
- ▶ Was ist Ihnen unklar (in der Domäne und der Spielidee)?
- ▶ Lücken im Ablauf Spielablauf füllen

Der Spieler kämpft als Krieger in einem Verlies.

- Objekte: Verlies und Krieger
- Aktion: Kämpfen
- Hat er ein Schwert? Zauber? Was ist ein Verlies? Gibt es nur Krieger?

Die Dunkelheit verfolgt den Spieler und setzt ihn unter Druck.

- Objekt: Dunkelheit (?)
- Woher weiß der Spieler wie weit die Dunkelheit entfernt ist?
- Müssen dann alle Gänge in die gleiche Richtung gehen?

Spielkonzept

- ▶ Zusammenfassung
- ▶ Zentrale Spielmechanik

Benutzeroberfläche

- ▶ Spielerinterface

Spiellogik

- ▶ Aktionen und Optionen
- ▶ Spielobjekte
- ▶ Spielstruktur

Ab sofort

- ▶ Fragebogen bis spätestens **heute Abend 23:59** ausfüllen.
- ▶ Auf Gruppeneinteilung warten (vor Sonntag)
- ▶ Mit der Hausaufgabe anfangen

Danach

- ▶ Termin für Gruppentreffen (Mo-Do) finden (Doodles ausfüllen, bis Sonntag 23:59)
- ▶ Hausaufgaben fertig machen
- ▶ Spielidee entwickeln

Ab sofort

- ▶ Fragebogen bis spätestens **heute Abend 23:59** ausfüllen
- ▶ Auf Gruppeneinteilung warten (vor Sonntag)
- ▶ Mit der Hausaufgabe anfangen

Überlegen Sie **vorher**, ob Sie:

- ▶ dieses Semester genug Zeit haben
- ▶ zu allen Pflichtterminen anwesend sein können
- ▶ zugriff auf einen zum Entwickeln geeigneten Rechner haben

Danach

- ▶ Termin für Gruppentreffen (Mo-Do) finden (Doodles aus)
- ▶ Hausaufgaben fertig machen
- ▶ Spielidee entwickeln

Ab sofort

- ▶ Fragebogen bis spätestens **heute Abend 23:59** ausfüllen
- ▶ Auf Gruppeneinteilung warten (vor Sonntag)
- ▶ Mit der Hausaufgabe anfangen

Überlegen Sie **vorher**, ob Sie:

- ▶ dieses Semester genug Zeit haben
- ▶ zu allen Pflichtterminen anwesend sein können
- ▶ zugriff auf einen zum Entwickeln geeigneten Rechner haben

Danach

- ▶ Termin für Gruppentreffen (Mo-Do) finden (Doodles aus)
- ▶ Hausaufgaben fertig machen
- ▶ Spielidee entwickeln

Stellen Sie sicher, dass Ihr **Postfach** nicht voll ist und Emails von der Uni ankommen (Spamfilter)!

