|  |
| --- |
|  |
| Schiffe versenken |
| Fachinformatiker Anwendungsentwicklung  E2FI1 2017/2018  Tobias Schacherbauer, Jürgen Huber  Abgabe: 06.07.2018 |

Inhaltsverzeichnis

[Selbständigkeitserklärung 2](#_Toc518483086)

[Quellen 2](#_Toc518483087)

[Projektidee / Projektziel 3](#_Toc518483088)

[Pflichtenheft 4](#_Toc518483089)

[Musskriterien 4](#_Toc518483090)

[Wunschkriterien 4](#_Toc518483091)

[Projektverlauf 6](#_Toc518483092)

[Implementierung 6](#_Toc518483093)

[Tests 6](#_Toc518483094)

[Klassen-Struktur / UML 7](#_Toc518483095)

[Spielablauf 8](#_Toc518483096)

[Probleme während des Projekts 9](#_Toc518483097)

[Fazit 10](#_Toc518483098)

# Selbständigkeitserklärung

Hiermit erklären wir, dass wir die vorliegende Projektarbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Die Stelle der Projektarbeit, die andere Quellen im Wortlaut oder dem Sinn nach entnommen wurden sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

Ulm, den 06.07.2018

Tobias Schacherbauer, Jürgen Huber

## Quellen

Newtonsoft Json Bibliothek: Serialisierung der Einstellungen ins JSON-Format, um sie auf der Festplatte speichern zu können, Deserialisierung der Einstellungs-Datei, um gespeicherte Einstellungen zu laden

Extended WPF Toolkit: Verwendung des „Color-Pickers“ in den Einstellungen

# Projektidee / Projektziel

# Pflichtenheft

Das Ziel des Projekts ist es, ein Programm zu erstellen, welches das Spielprinzip „Schiffe versenken“ auf einer zweidimensionalen Spielfläche darstellt. Das Spiel ist für einen Spieler ausgelegt, der gegen einen computergesteuerten Gegner (im folgenden KI-Gegner) antritt.

## Musskriterien

* Ein grafische Oberfläche mit: Schiffsarten, aktuellem Spielbrett, Trefferanzeige auf dem Spielbrett, Statuskarte mit Anzeige von Treffern und Nicht-Treffern, Zeitanzeige, verbleibende Anzahl an gegnerischen Schiffen
* Der Spieler kann vor dem Spielbeginn seine Schiffe auf dem Spielbrett platzieren
* Vor dem Spielbeginn gibt der Spieler einen Namen an
* Unterschiedliche Spielmodi
  + Standard (klassisches Schiffe versenken)
  + Survival (neue Gegner erscheinen, bis man verliert)
  + Advanced (Survival) Mode (Schiffe besitzen Spezialfähigkeiten, bspw. Torpedo; Kombinierbar mit Survival)
* Einstellungsmöglichkeiten
  + Größe des Spielfelds
  + Schiffsfarben
  + Stärke der KI-Gegner (basierend auf Algorithmus):
    - Leicht: KI-Gegner schießt jedes Mal auf ein zufälliges Feld
    - Normal: KI-Gegner schießt zunächst zufällig und bei Treffer versucht es das getroffene Schiff komplett zu zerstören
  + Anzahl der Schiffe pro Schiffstyp
  + Spielmodus
  + Rundentimer

## Wunschkriterien

* KI-Gegner Stärke „Schwer“ (Allwissender KI-Gegner)
* KI-Gegner basierend auf einem Neural Network
* Spezialfähigkeiten für Schiffe:
  + Torpedo (Schuss der eine komplette Spalte trifft)
  + Radar (deckt eine bestimmte Anzahl an Feldern auf)
  + Scattershot (trifft mehrere nebeneinander liegende Felder)
  + Rüstung (gepanzertes Feld (eines Schiffes) muss mehrmals getroffen werden)
  + Reparatur (eine bestimme Anzahl an Feldern kann nach einem Treffer wiederhergestellt werden)
  + Und mögliche weitere
* Es soll eine Auswahl an verschiedenen Munitionstypen geben. Besondere Munition soll begrenzt zur Verfügung stehen
* Nach einem Spiel werden Punkte errechnet und mit dem Spielernamen abgespeichert. Diese Highscore-Liste kann im Spielmenü abgerufen werden
* Es sollen verschiedene Kartentypen zur Verfügung stehen, die alle spezielle Eigenschaften haben, z.B. Hindernisse auf der Karte

# Projektverlauf

## Allgemeines

Mittwoch, 30.05.:

* Einrichtung eines gemeinsamen Git-Repositories
* Planung des Klassendiagramms (UML)
* Genehmigung der Projektidee

Mittwoch, 06.06.:

* Abgabe des Pflichtenhefts
* Erstellung der geplanten Klassen
* Erstentwurf eines Schiffplatzierungs-Algorithmus

Mittwoch, 13.06.:

* Erstellung einer rudimentären GUI
* Verfeinerung der Schiffsplatzierungslogik
* Erstellung erster Schnittstellen zwischen GUI und Spiellogik

Mittwoch, 20.06.:

* Entwurf eines Zielalgorithmus für KI-Gegner
* Erstellung des Einstellungsfensters
* Bugfixes

Mittwoch, 27.06.:

* Implementierung der Einstellungsspeicherung per Serialisierung
* Weitere Bugfixes in den KI-Algorithmen
* Beginn der Arbeit an der Dokumentation

Mittwoch, 04.07.:

* Arbeit an der Dokumentation

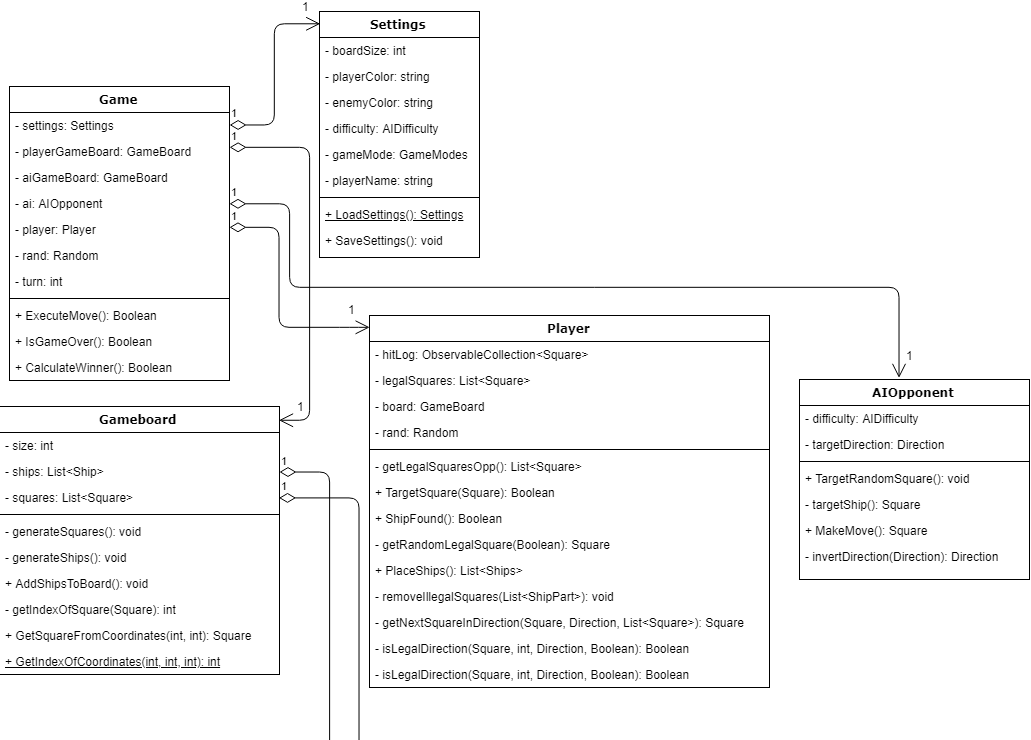
## Implementierung

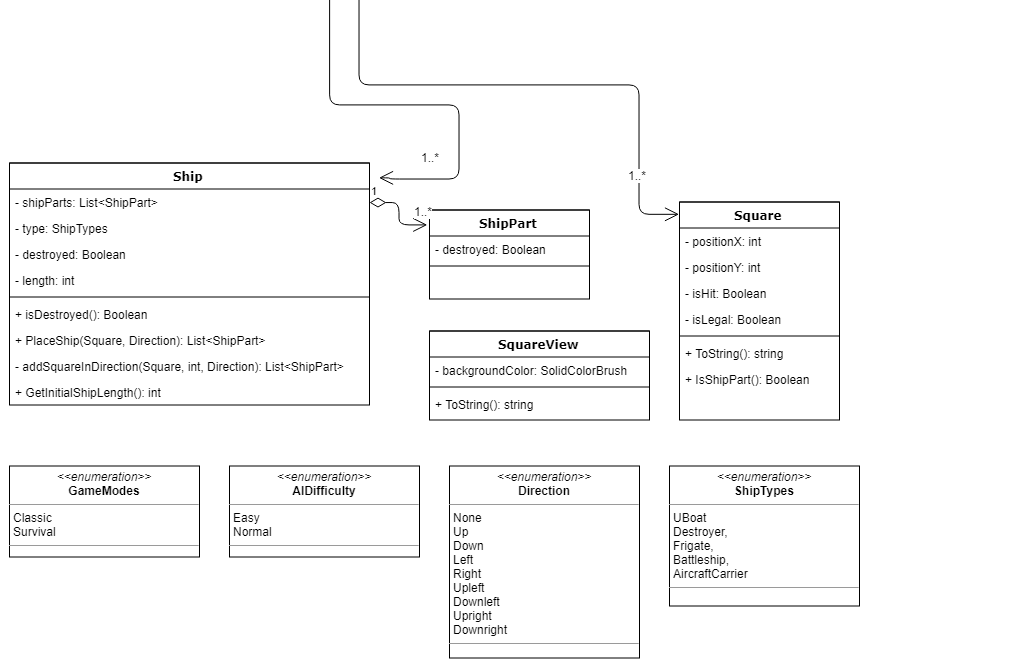
Der erste Schritt bei der Implementierung unseres Projekts war, die vorher im UML-Diagramm geplanten Klassen zu erstellen und zu füllen. Im nächsten Schritt planten wir das Design der GUI und die benötigten Schnittstellen zur Spiellogik, für eine möglichst einfache Wartbarkeit und die bestmögliche Erweiterbarkeit des Programms. Anschließend entwarfen wir Algorithmen, welche das Spielfeld mit Schiffen füllen und die KI-Gegner befähigt, gefundene Schiffe zielgenau zu zerstören.

## Tests

Die gleichzeitige Entwicklung von GUI und Spiellogik ermöglichte es uns, unsere Funktionen und Algorithmen kontinuierlich zu testen, anzupassen und eventuelle Bugs zeitnah zu beheben.

# Klassen-Struktur / UML





# Spielablauf

# Probleme während des Projekts

# Fazit