

**Praxisprojekt im Rahmen von  
Advanced Software Engineering**

**Technische Dokumentation zum Programmentwurf**

**Weeping Snake**

Duale Hochschule Baden-Württemberg Karlsruhe  
Fakultät Technik

Studiengang Informatik – Informatik

von

Silas Mario Schnurr

19.05.2021



**Kurs:** TINF18B5

**Dozent:**  Herr Maurice Müller

**Erklärung zur Eigenleistung**

Gemäß § 5 (3) der „Studien- und Prüfungsordnung DHBW Technik“ vom 22. September 2011.

Ich habe die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

----------------------------------- --------------------------

Ort Datum Unterschrift

**Inhaltsverzeichnis**

[Abkürzungsverzeichnis / Glossar IV](#_Toc72307016)

[Abbildungs- und Tabellenverzeichnis V](#_Toc72307017)

[1 Projektbeschreibung 1](#_Toc72307018)

[1.1 Funktionsumfang 1](#_Toc72307019)

[1.2 Code-Struktur 1](#_Toc72307020)

[2 Domain Driven Design 2](#_Toc72307021)

[2.1 Ubiquitous Language 2](#_Toc72307022)

[2.2 Analyse ?? 2](#_Toc72307023)

[3 Programming Principles 3](#_Toc72307024)

[3.1 SOLID 3](#_Toc72307025)

[3.2 GRASP 3](#_Toc72307026)

[3.3 DRY 3](#_Toc72307027)

[4 Entwurfsmuster 4](#_Toc72307028)

[5 Clean Architecture 5](#_Toc72307029)

[6 Legacy Code 6](#_Toc72307030)

[7 Refactoring 7](#_Toc72307031)

[8 Unit Tests 8](#_Toc72307032)

[8.1 Konzept 8](#_Toc72307033)

[8.2 ATRIP-Regeln 8](#_Toc72307034)

[8.3 Code Coverage 8](#_Toc72307035)

[9 Zusatz: API-Design 9](#_Toc72307036)

[9.1 API 9](#_Toc72307037)

[9.2 Analyse 9](#_Toc72307038)

# Abkürzungsverzeichnis / Glossar

**API** Application Programming Interface, Programmierschnittstelle

# Abbildungs- und Tabellenverzeichnis

**Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.**

**Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.**

# Projektbeschreibung

Das ist die schriftliche Dokumentation. Es wird die Objektorientierte Mainstream-Programmiersprache C# mit dem .Net 5.0 Framework verwendet.

enthält Ergebnisse der Analysen und Begründungen der Entscheidungen • Code-Struktur als UML-Diagramme notiert – Detailgrad an die Aussagen anpassen

## Funktionsumfang

Der klar definierte sinnvolle Nutzen dieser Programmbibliothek ist…

* Spielen, gegner rammen (5 runden lang) gibt punkte
* Ältere spuren sind hindernisse / existieren nicht

## Code-Struktur

2k Zeilen Code > 20 Klassen

Code-Struktur ist als UML-Diagramm notiert – Detailgrad an die Aussagen anpassen

|  |  |  |
| --- | --- | --- |
| **Projekt** | **Funktion** | **Zeilen von Quellcode** |
| WeepingSnake.Game |  | 2200 |
| WeepingSnake.Game.Tests |  | 600 |
| WeepingSnake.WebService |  | 200 |
| WeepingSnake.ConsoleClient |  | 200 |
| WeepingSnake.ConsoleClient.Tests |  | 200 |
|  |  |  |

# Domain Driven Design

## Ubiquitous Language

* Analyse der Ubiquitous Language mit >= 5 Beispielen

## Analyse ??

* Analyse und Begründung von
  + Repositories
  + Aggregates
  + Entities
  + Value Objects

# Programming Principles

## SOLID

Analyse und Begründung

## GRASP

Analyse und Begründung

(insbes. Kopplung und Kohäsion)

## DRY

Analyse und Begründung

# Entwurfsmuster

* >= 1 Entwurfsmuster einsetzen und begründen
* UML (vorher/)nachher

# Clean Architecture

* >= 1 Umsetzung einer Klasse der Adapterschicht
* Begründung

# Legacy Code

* >= 2 Abhängigkeiten mit Techniken aus der Vorlesung brechen
* ausgewählte Stellen und Techniken begründen

# Refactoring

* Code Smells identifizieren
* >= 2 Refactorings anwenden und begründen
* für 2 Stelle vorher/nacher UML

# Unit Tests

## Konzept

* >= 10 Unit Tests
* Einsatz von Fake-/MockObjekten (ohne Frameworks wie z.B. Mockito)
* Analyse und Begründung für
  + Umfang der Tests
  + Einsatz der Fake/MockObjekte

## ATRIP-Regeln

## Code Coverage

# Zusatz: API-Design

Freiwillig bei b5

## API

* Bereitstellung einer sinnvollen API
  + Programmiersprachenebene oder http
  + >= 10 Methoden / HTTP Endpunkte

## Analyse

* Analyse der API anhand der Qualitätsmerkmale und Begründung für das Des