

# Programmentwurf

**SimpsonsGame**

von

**Dominik Veith**

Abgabedatum: 04. April 2023

Bearbeitungszeitraum: 04.10.2022 - 04.04.2023

Matrikelnummer, Kurs: 3352220, TINF20B2

## **Abstract**

- *Deutsch* -

Dies ist der Beginn des Abstracts. Für die finale Bachelorarbeit musst du ein Abstract in deinem Dokument mit einbauen. So, schreibe es am besten jetzt in Deutsch und Englisch. Das Abstract ist eine kurze Zusammenfassung mit ca. 200 bis 250 Wörtern.

Versuche in das Abstract folgende Punkte aufzunehmen: Fragestellung der Arbeit, methodische Vorgehensweise oder die Hauptergebnisse deiner Arbeit.

# Inhaltsverzeichnis

<b>Abkürzungsverzeichnis</b>	<b>IV</b>
<b>Abbildungsverzeichnis</b>	<b>V</b>
<b>Tabellenverzeichnis</b>	<b>VI</b>
<b>Quellcodeverzeichnis</b>	<b>VII</b>
<b>1 Einführung</b>	<b>1</b>
1.1 Übersicht über die Applikation . . . . .	1
1.2 Start der Applikation . . . . .	1
1.3 Testen der Applikation . . . . .	1
<b>2 Clean Architecture</b>	<b>2</b>
2.1 Was ist Clean Architecture? . . . . .	2
2.2 Analyse der Dependency Rule . . . . .	2
2.3 Analyse der Schichten . . . . .	2
<b>3 SOLID</b>	<b>3</b>
3.1 Analyse Single-Responsibility-Principle (SRP) . . . . .	3
3.2 Analyse Open-Closed-Principle (OCP) . . . . .	3
3.3 Analyse Liskov-Substitution- (LSP), Interface-Segregation- (ISP), Dependency-Inversion-Principle (DIP) . . . . .	3
<b>4 Weitere Prinzipien</b>	<b>4</b>
4.1 Analyse GRASP: Geringe Kopplung . . . . .	4
4.2 Analyse GRASP: Hohe Kohäsion . . . . .	4
4.3 Don't Repeat Yourself (DRY) . . . . .	4
<b>5 Unit Tests</b>	<b>5</b>
5.1 10 Unit Tests . . . . .	5
5.2 ATRIP: Automatic . . . . .	5
5.3 ATRIP: Thorough . . . . .	5
5.4 ATRIP: Professional . . . . .	5
5.5 Code Coverage . . . . .	5
5.6 Fakes und Mocks . . . . .	5
<b>6 Domain Driven Design</b>	<b>6</b>
6.1 Entities . . . . .	6

6.2	Value Objects . . . . .	6
6.3	Aggregates . . . . .	6
<b>7</b>	<b>Refactoring</b>	<b>7</b>
7.1	Code Smells . . . . .	7
7.2	2 Refactorings . . . . .	7

# Abkürzungsverzeichnis

<b>AIR</b>	Adobe Integrated Runtime
<b>AJAX</b>	Asynchronous Javascript and XML
<b>ANSI</b>	American National Standards Institute
<b>API</b>	Application Programming Interface
<b>AR</b>	Augmented Reality
<b>BAPI</b>	Business Application Programming Interface
<b>BIOS</b>	Basic Input Output System
<b>CDMA</b>	Code Division Multiple Access
<b>HTTPS</b>	Hypertext Transfer Protocol Secure
<b>ISBN</b>	Internationale Standardbuchnummer
<b>OLAP</b>	Online Analytical Processing
<b>ORDBMS</b>	Object-Relational DataBase Management System
<b>SDK</b>	Software Development Kit
<b>SEO</b>	Search Engine Optimization
<b>SSH</b>	Secure Shell
<b>UEFI</b>	Unified Extensible Firmware Interface
<b>USB</b>	Universal Serial Bus
<b>VLAN</b>	Virtual Local Area Network
<b>WYSISWG</b>	What You See Is What You Get
<b>XSL</b>	Extensible Stylesheet Language

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Quellcodeverzeichnis



# **1 Einführung**

## **1.1 Übersicht über die Applikation**

## **1.2 Start der Applikation**

## **1.3 Testen der Applikation**

## **2 Clean Architecture**

### **2.1 Was ist Clean Architecture?**

### **2.2 Analyse der Dependency Rule**

#### **2.2.1 Positiv-Beispiel: Dependency Rule**

#### **2.2.2 Negativ-Beispiel: Dependency Rule**

### **2.3 Analyse der Schichten**

## **3 SOLID**

### **3.1 Analyse Single-Responsibility-Principle (SRP)**

#### **3.1.1 Positiv-Beispiel**

#### **3.1.2 Negativ-Beispiel**

### **3.2 Analyse Open-Closed-Principle (OCP)**

#### **3.2.1 Positiv-Beispiel**

#### **3.2.2 Negativ-Beispiel**

### **3.3 Analyse Liskov-Substitution- (LSP), Interface-Segregation- (ISP), Dependency-Inversion-Principle (DIP)**

#### **3.3.1 Positiv-Beispiel**

#### **3.3.2 Negativ-Beispiel**

## **4 Weitere Prinzipien**

### **4.1 Analyse GRASP: Geringe Kopplung**

#### **4.1.1 Positiv-Beispiel**

#### **4.1.2 Negativ-Beispiel**

### **4.2 Analyse GRASP: Hohe Kohäsion**

### **4.3 Don't Repeat Yourself (DRY)**

## **5 Unit Tests**

### **5.1 10 Unit Tests**

### **5.2 ATRIP: Automatic**

### **5.3 ATRIP: Thorough**

### **5.4 ATRIP: Professional**

### **5.5 Code Coverage**

### **5.6 Fakes und Mocks**

## **6 Domain Driven Design**

### **6.1 Entities**

### **6.2 Value Objects**

### **6.3 Aggregates**

# **7 Refactoring**

## **7.1 Code Smells**

## **7.2 2 Refactorings**