DHBW
Duale Hochschule
Baden-Württemberg
Karlsruhe

Advanced Software Engineering

Programmentwurf

für die Prüfung zum

Bachelor of Science

des Studienganges Angewandte Informatik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

von

Dominik Klysch

Abgabedatum TODO

Matrikelnummer 1010579
Kurs TINF18B4

Inhaltsverzeichnis

Kapitel 1

Domain Driven Design

1.1 Analyse der Ubiquitous Language

Wort	Bedeutung
Test	Test ist sehr toll
User	User spielen auf dem Server

1.2 Analyse und Begründung der verwendeten Muster

1.2.1 Value Objects

1.2.2 Entities

1.2.3 Aggregates

1.2.4 Repositories

Factories haben nur einen einzigen Zweck: das Erzeugen von Objekten Factories sind ein allgemein nützliches Konzept, unabhängig von DDD Wenn die Logik für das Erzeugen einer Entity, eines Aggregates oder eines VO komplex wird, kann dies den eigentlichen Zweck des Objekts verschleiern (Verletzung des Single-Responsibility-Prinzips) Factories helfen, indem sie dem Objekt die Verantwortung für seine Konstruktion abnehmen; dadurch kann sich das Objekt auf sein Verhalten konzentrieren Der Begriff „Factory“ wird in OOP mehrdeutig verwendet. Es bezeichnet sowohl: Das allgemeine Konzept einer Factory: irgendein Objekt oder irgendeine Methode zur Erzeugung andere Objekte als Konstruktor-Ersatz spezielle Erzeugungs-Muster Factory Method Abstract Factory Im DDD meint man mit Factory normalerweise das „allgemeine Konzept“ Allgemein ist eine Factory irgendein Objekt/ irgendeine Methode als Konstruktor-Ersatz (siehe unten)

1.2.5 Domain Services

Kapitel 2

Clean Architecture

2.1 Schichtarchitektur planen und begründen

Kapitel 3

Programming Principles

3.1 Analyse und Begründung für SOLID

3.2 Analyse und Begründung für GRASP

3.3 Analyse und Begründung für DRY

Kapitel 4

Entwurfsmuster

4.1 Einsatz begründen, UML vorher/nachher

Kapitel 5

Refactoring

5.1 Code Smells identifizieren

5.2 Refactorings begründen