ADAP CW#02

# Projektdaten

Projekt name: 3D-Printing

Projekt repository: [https://github.com/**ModischFabrications**/wahlzeit](https://github.com/ModischFabrications/wahlzeit)

This week’s tag: adap-cw02 on master

Homework diff: <https://github.com/ModischFabrications/wahlzeit/compare/adap-initial...adap-cw02>

CI: [https://travis-ci.org/**ModischFabrications**/wahlzeit](https://travis-ci.org/ModischFabrications/wahlzeit)

Docker Hub Repo: [https://hub.docker.com/r/**modischfabrications**/wahlzeit](https://hub.docker.com/r/modischfabrications/wahlzeit)

# Hausaufgaben

## Implementation

### Neue Klassen (falls vorhanden)

[…]

### Veränderte Klassen (falls vorhanden)

[src/main/java/org/wahlzeit/main/ModelMain.java](https://github.com/ModischFabrications/wahlzeit/compare/adap-initial...adap-cw02#diff-a451ea419a1988286d2c11530572b0ca)

### Erklärung

Ich habe mir zuerst das Thema “3D-Printing” ausgesucht, da ich dort eine breite Menge eigener Bilder ohne spezielle Lizenzbedingungen besitze und die Domäne interessant und leicht zu ergänzen ist. Nachteil ist eventuell die Nomenklatur, die Objekte werden selbst auch oft als “(3D) model” bezeichnet, dies kann mit den bisherigen Java-Klassen kollidieren.

Dann habe ich per GitHub einen Fork erstellt und diesen geklont. Das Readme dieser Version habe ich mit einem Prefix markiert, um Verwechslungen auszuschließen und anschließend den Basisstand mit “adap-initial” getagged.

Travis CI habe ich anschließend durch anlegen einer Verknüpfung mit meinem Repository und eintragen von DOCKERHUB\_USER & DOCKERHUB\_PW (access token) eingerichtet, der Build war problemlos möglich. Im Rahmen dieser Maßnahme wurde auch das Docker Image erfolgreich gebaut und hochgeladen.

Dann habe ich eine kleine Auswahl eigener Bilder anonymisiert, in den Ordner [src/main/resources/pictures/](https://github.com/ModischFabrications/wahlzeit/commit/d417e0fc6b957189b6c854d355547d71a4a37a77#diff-e35f463ef705d0d2f6b194efbf97e0e1) eingefügt und zusammen mit diesem Dokument committed. Ich habe private Bilder verwendet, da diese im Gegensatz zu Beispielbildern aus dem Internet frei von mir verwendbar und ohne externe Lizenz sind. Um diese Änderungen auch unter Windows verwendbar zu machen musste ich einen Hotfix einfügen, der die Pfade korrekt auflöst, hierfür habe ich die oben beschriebene Klasse angepasst.

Zuletzt habe ich alle Änderungen gepusht und den letzten Commit mit dem oben genannten Tag für die Abgabe markiert.

## Fragen / Aufgaben

### Dockerfiles

Eine “Dockerfile” wird als Instruktionsliste genutzt, um eine Anwendung mit Docker zu containerisieren, also in einer isolierten Umgebung ausführbar zu machen. Basis beider Container ist ein openjdk Image, dass eine Implementierung von Java mitbringt, für einen single-stage build unüblich ist dieses hier als builder definiert. Als nächstes werden alle relevanten Dateien kopiert, implizit gebaut und dann getestet. Zuletzt wird der zu benutzende Port markiert und der Zugangspunkt des Containers festgelegt.

Die Unterschiede von “Dockerfile” zu “simple-Dockerfile” sind vor allem die Verwendung eines multi-stage builds, also eines mehrstufigen Bauprozesses. Dieser erlaubt es, nur die gebauten Files im finalen Container mitzunehmen und alle temporären Artefakte des Bauprozesses “zurückzulassen”, wodurch die Containergröße deutlich reduziert werden kann. Der zweite Unterschied ist die Verwendung eines manuell definierten Zugangspunktes, der nicht auf die von gradlew bereitgestellten Shortcuts zurückgreift.