

# Campus Team 2.2

---

Polina Luzina, Helen Laible, Nik Studenikin, Jann Pischke,  
Vadim Rodt

# Gliederung

- Achievements
- Funktionen
- Klassenstruktur
- Lessons Learned
- Nächste Schritte

# Achievements

- DataCollector Klassen als Datensammelstellen erstellt  
→ spätere Übergabe von Daten an Persons & Appointments
- Beschäftigung mit Architektur von Interfaces und Tests
- Search Klasse Funktionalität erhalten  
→ wenn String im Namen, wird es als Ergebnis ausgegeben
- Erste Tests geschrieben

# Funktionen

## geplant:

- Anbindung an Google Maps
- Verlinkung zu den anderen Services
  - Mensa
  - Appointments um Events anzuzeigen  
Orten zu finden (optional)
  - Stundenplan (Raum finden tc.)
- Empfehlungen für Bars, Restaurants, Cafes, Clubs, etc. (optional)

## vorhandene Funktionen:

- Suchfunktion auf Professoren, Namen, Raumnummern, Fakultäten, Straßennamen, Einrichtungen
- Zugriffs- und Manipulationsmethoden für einzelne Klassen (Gebäude, Räume, Campus)

# Klassenstruktur

Hashmaps Beispiele:

## - Hashmaps

- Tabellenartige Speicherung
- Unsere Grundstruktur zur Einkapselung der Daten für den zukünftigen interapplikationaren Transport

## - Interfaces

- Appointments (Informationen die Wir abgeben)
  - Informationen zu den Standort
  - Raumnummer / Gebäude / Campus
- Appointments (Informationen die Wir brauchen)
  - Informationen zu den Veranstaltungen
  - Eventname / Datum / Standort
- Persons (Informationen die Wir abgeben)
  - Raumnummer / Gebäude / Campus

```
public Map<String, Integer>
```

roomId

Room 1	1
--------	---

```
public Map<String, List<String>>
```

roomPersons

2	['Knauf', 'Wolf', ...]
---	------------------------

campusFaculties

AltonaerStr	['AI', 'GB', ...]
-------------	-------------------

# Lessons learned

- Grundlegendes **Verständnis** von Aufbau Mehrklassiger Java-Applikationen
- Komplexe Datenformate und deren (teilweise komplexe) **Java-Syntax**
- **Datenmanipulation** und **-transport** durch die gesamte Applikation
- Basics von **Projektmanagement** und Zuständigkeiten Aufteilung

# Nächste Schritte

- **Testbasiertes** programmieren (z.B. Hashmap-Tests schreiben)  
→ bestehende Funktionen testen
- Funktion zum bereitstellen von Daten für den **Service Persons** definieren
- Funktion zum bereitstellen von Daten für den **Service Appointments** definieren
- Interface zum Service Appointments erstellen
- Bestehende Applikationseinheiten **optimieren** und **erweitern**
- Fokus auf **Qualität** und nicht Quantität

Vielen Dank für Ihre Aufmerksamkeit