Webservices testen

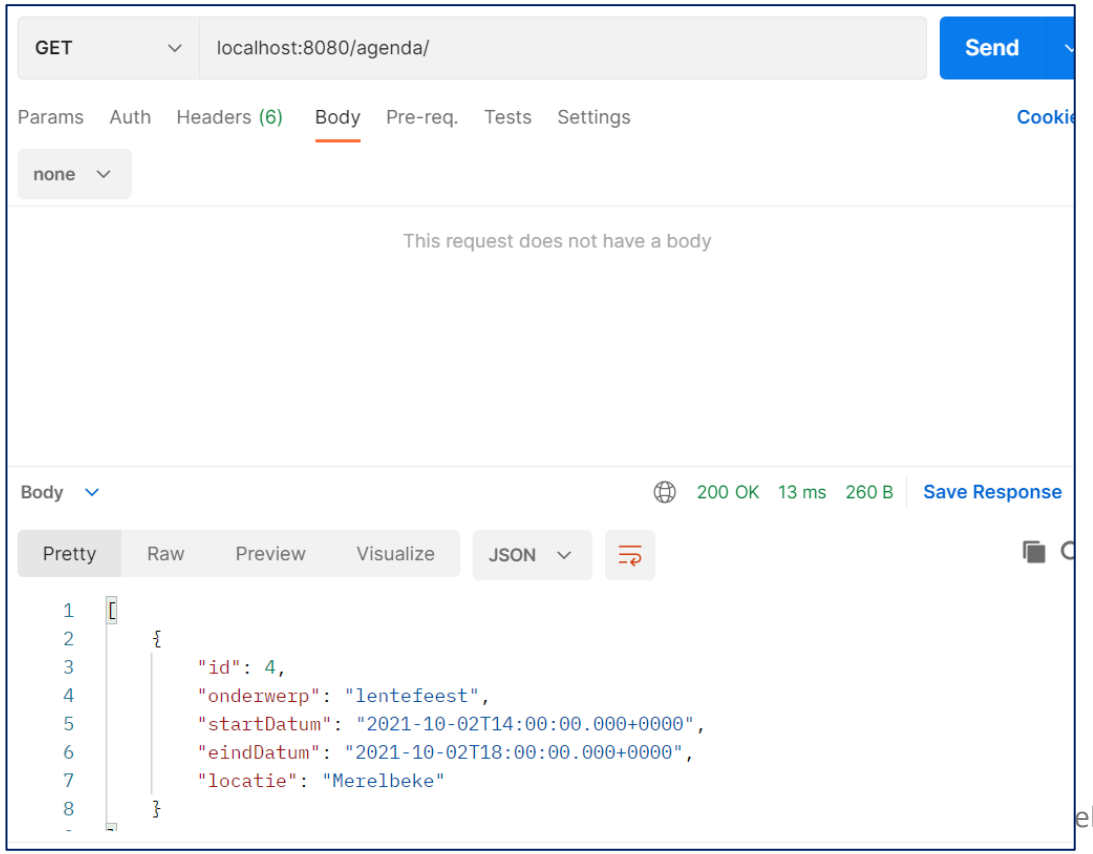
Veerle Ongenae

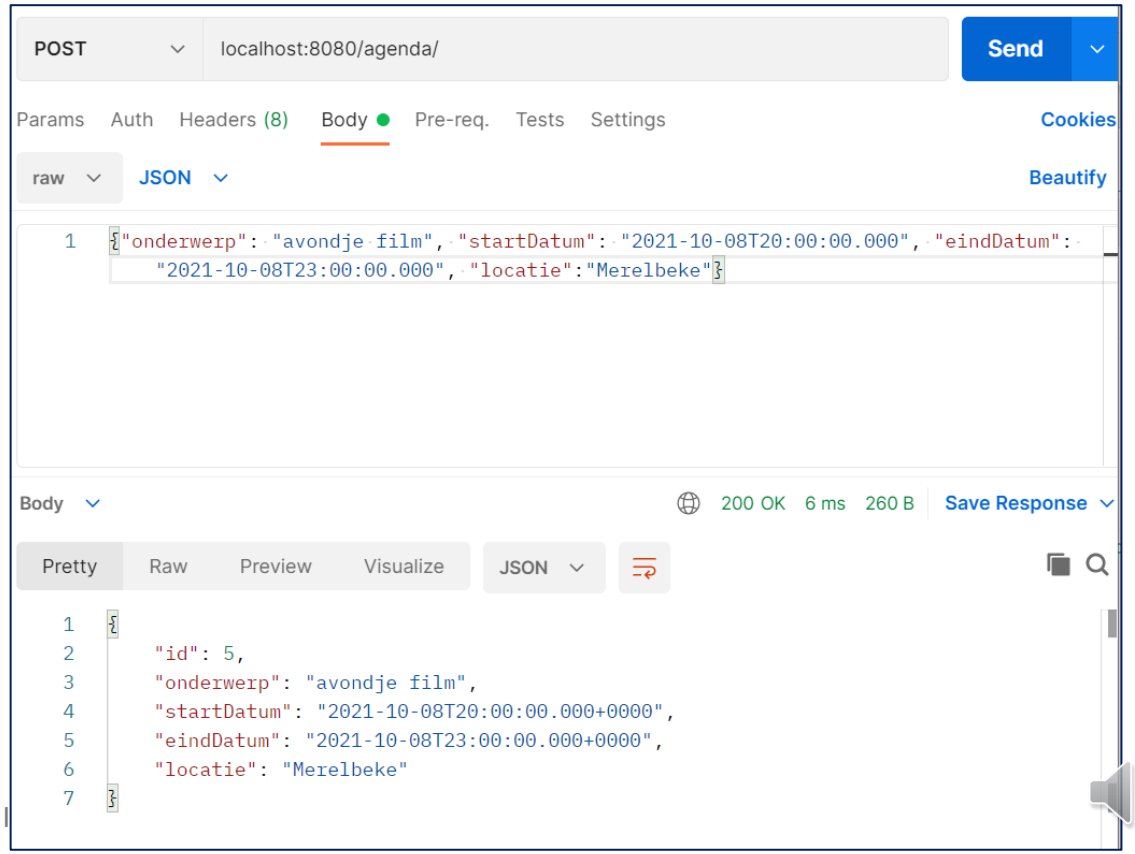
Webservices testen

- Manueel (algemeen)
- Herhaling Junit (Java)
- WebTestClient (Java – Spring)

Manueel testen webservice

- Tool
  - HTTP-berichten te sturen
  - Resultaat (body, headers, …) te bekijken
  - Bv. Postman





Webservices testen

- Manueel (algemeen)
- Herhaling Junit (Java)
- WebTestClient (Java – Spring)

Unit test

- Testen apart onderdeel (klasse, methode, …)
- Mockups (dummy’s) voor afhankelijkheden
- Vroeg tijdens ontwikkeling
- Snel uitgevoerd
  - Weinig afhankelijkheden

Fixture en fases

UNIVERSITEIT GENT

- Fixture
  - = vaste context/begintoestand voor testen
  - = baseline state
  - bv. lege databank, geformatteerde schijf, geen gebruikers, etc.

- Vier fases:
  1. **Set up:** test fixture klaarzetten
  2. **Exercise**: interageer met het te onderzoeken systeem
  3. **Verify**: controleer of verwacht resultaat bereikt is
  4. **Tear down**: test fixture afbreken naar originele toestand

- Test suite: (onafhankelijke) unit tests met zelfde fixture

Dummy's

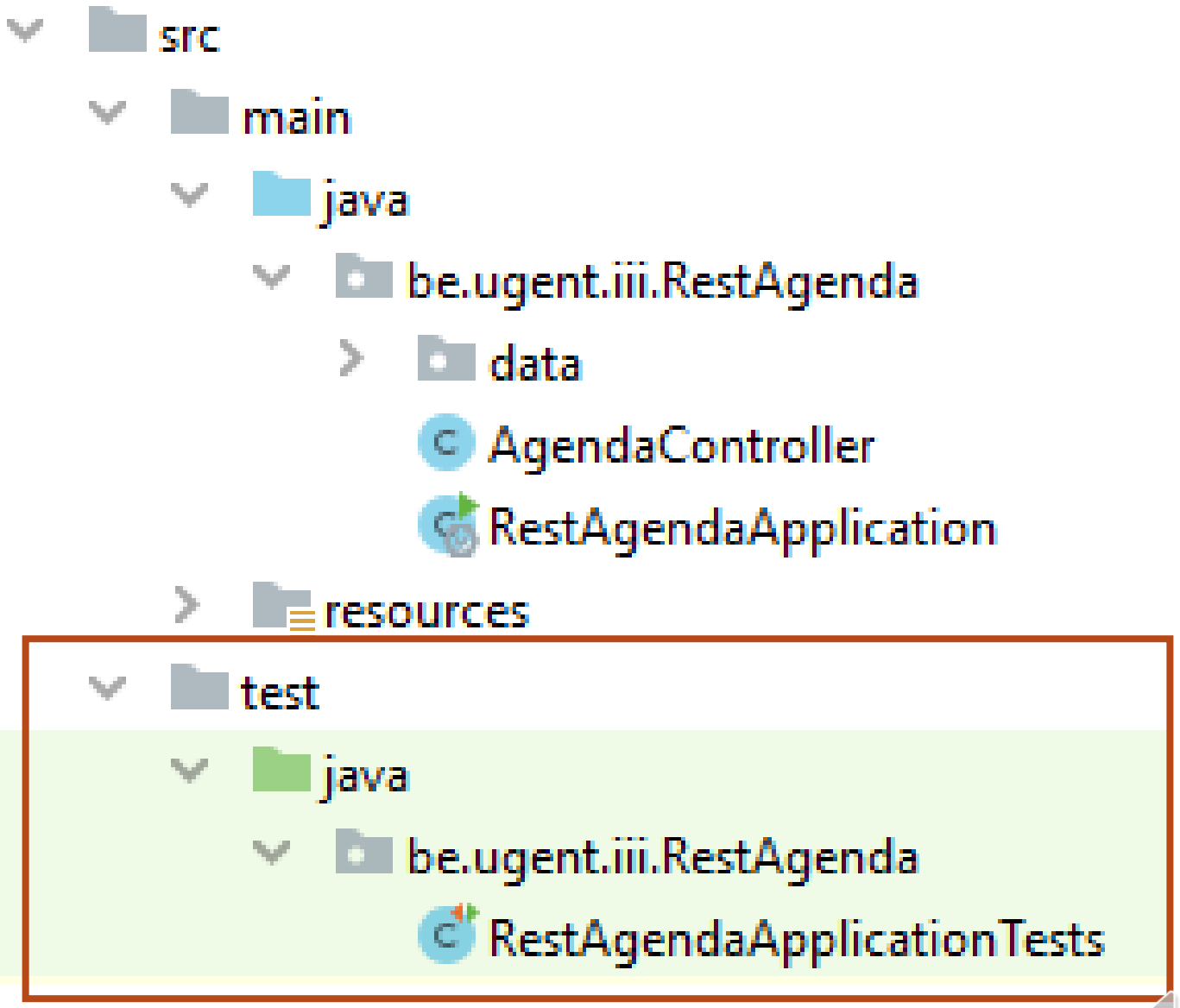
- Relatie tussen componenten
- Vervang een of meerdere componenten door dummy:
  - Zelfde "interface"
  - Eenvoudiger implementatie, werking
  - Eenvoudiger deployment
- Ideaal voor Inversion of Control
  (interfaces, abstract factory, …)

Mockup

- = Gegenereerde dummy-klasse
- Implementeert interface
- Heel eenvoudige werking
  - bv. bepaalde vaste waarden teruggeven

- Mockito, EasyMock, JMock, PowerMock

Structuur project



Maven – pom.xml

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
    <exclusions>
        <exclusion>
            <groupId>org.junit.vintage</groupId>
            <artifactId>junit-vintage-engine</artifactId>
        </exclusion>
    </exclusions>
</dependency>
```

→ Oudere versies JUnit

Test schrijven

```
package be.ugent.iii.RestAgenda;

import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
class RestAgendaApplicationTests {

    @Test
    void contextLoads() {
    }

}
```

Test-annotatie uit Junit 5

Volledige context ter beschikking

Methode met testcode

Testen schrijven - assertions

UNIVERSITEIT GENT

- Controle resultaat

```
class MyFirstJUnitJupiterTests {

    private final Calculator calculator = new Calculator();

    @Test
    void addition() {
        assertEquals(2, calculator.add(1, 1));
    }

}
```

Controle resultaat

Assertions

UNIVERSITEIT GENT

- https://junit.org/junit5/docs/5.0.1/api/org/junit/jupiter/api/Assertions.html

static void	**assertArrayEquals**(double[] expected, double[] actual, String message) *Asserts* that expected and actual double arrays are equal.
static void	**assertEquals**(double expected, double actual, String message) *Asserts* that expected and actual are equal.
static void	**assertFalse**(boolean condition, String message) *Asserts* that the supplied condition is not true.
static void	**assertNotEquals**(Object unexpected, Object actual, String message) *Asserts* that expected and actual are not equal.
static void	**assertNotNull**(Object actual, String message) *Asserts* that actual is not null.

Nog meer assertions met AssertJ

UNIVERSITEIT GENT

```java
assertThat(frodo)
  .isNotEqualTo(sauron)
  .isIn(fellowshipOfTheRing);

assertThat(frodo.getName())
  .startsWith("Fro")
  .endsWith("do")
  .isEqualToIgnoringCase("frodo");

assertThat(fellowshipOfTheRing)
  .hasSize(9)
  .contains(frodo, sam)
  .doesNotContain(sauron);
```

- https://www.baeldung.com/introduction-to-assertj
  - Meer mogelijkheden
  - Verschillende testen op één object

Voor en na een test

```java
@BeforeAll
static void setup() {
    log.info("@BeforeAll - executes once before all test methods in this class");
}

@BeforeEach
void init() {
    log.info("@BeforeEach - executes before each test method in this class");
}
```

```java
@AfterEach
void tearDown() {
    log.info("@AfterEach - executed after each test method.");
}

@AfterAll
static void done() {
    log.info("@AfterAll - executed after all test methods.");
}
```

bron: https://www.baeldung.com/junit-5

Webservices testen

- Manueel (algemeen)
- Herhaling Junit (Java)
- WebTestClient (Java – Spring)

REST-webservice testen: Client nodig

- Client nodig om HTTP-berichten te sturen

```
@ExtendWith(SpringExtension.class)
@SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)
@AutoConfigureWebTestClient

class RestAgendaApplicationTests {

    @Autowired
    private WebTestClient webClient;
```

Junit 5 combineren met Spring, niet nodig bij recente Springversies

willekeurige poort gebruiken voor te testen REST-service (conflicten vermijden)

Injecteren WebTestClient

HTTP-antwoord testen met WebTestClient

UNIVERSITEIT GENT

- Juiste inhoud? Juiste headers? Juiste statuscode?

```
@Test
public void test2GetAllGithubRepositories() {
    webTestClient.get().uri("/api/repos")
            .accept(MediaType.APPLICATION_JSON_UTF8)
            .exchange()
            .expectStatus().isOk()
            .expectHeader().contentType(MediaType.APPLICATION_JSON_UTF8)
            .expectBodyList(GithubRepo.class);
```

HTTP-methode + URI instellen

Headers instellen

Bericht versturen

Controle statuscode

Controle headers

Controle inhoud body

WebTestClient – test API

UNIVERSITEIT GENT

- Headers aanvraag instellen https://www.javadoc.io/static/org.springframework/spring-test/6.0.12/org/springframework/test/web/reactive/server/WebTestClient.RequestHeadersSpec.html
- HTTP-status controleren https://www.javadoc.io/static/org.springframework/spring-test/6.0.12/org/springframework/test/web/reactive/server/StatusAssertions.html
- HTTP-headers controleren https://www.javadoc.io/static/org.springframework/spring-test/6.0.12/org/springframework/test/web/reactive/server/HeaderAssertions.html
- HTTP-body controleren https://www.javadoc.io/static/org.springframework/spring-test/6.0.12/org/springframework/test/web/reactive/server/WebTestClient.ResponseSpec.html
- Inhoud body opvragen (één object) https://www.javadoc.io/static/org.springframework/spring-test/6.0.12/org/springframework/test/web/reactive/server/WebTestClient.BodySpec.html
- Inhoud body opvragen (lijst) https://www.javadoc.io/static/org.springframework/spring-test/6.0.12/org/springframework/test/web/reactive/server/WebTestClient.ListBodySpec.html

Profielen gebruiken op te testen

UNIVERSITEIT GENT

- Andere beans (componenten, services, …) gebruiken afhankelijk van een profiel (test, ontwikkeling, productie, …)
- Werkwijze
  - Bean annoteren met een profiel
  - Profiel toekennen aan klasse met programma, test, …

Bean - @Profile

```
@Component
@Profile("dev")
public class DummyDao implements Dao {…}
```

Bean hoort bij een welbepaald profiel

```
@Component
@Profile("!dev")
public class RealDao implements Dao {…}
```

Bean hoort bij alle profiellen behalve het opgegeven profiel

- Afhankelijk van profiel zal het Springframework een ander type object injecteren
- Geen @Profile → standaardprofiel (*default*)

Profiel instellen

UNIVERSITEIT GENT

- Bij testen via @ActiveProfiles

```
@SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)
@AutoConfigureWebTestClient
@ActiveProfiles("test")
public class TestsClass {…}
```

- Uitvoeren toepassing
  - In configuratie, omgevingsvariabele, maven-profiel, commandolijn-optie, …

Webservices testen

- Manueel (algemeen)
- Herhaling Junit (Java)
- WebTestClient (Java – Spring)