Blog Spring Boot Migration Java 8, JUnit4 nach Java11 JUnit5

Das Rad dreht sich immer schneller mit Spring Boot und Java. Fast täglich könnten wir unsere Anwendung auf neue Versionen und Libraries migrieren. Eine gröbere Umstellung ist der Wechsel von Java 8 nach Java 11 und JUnit4 nach JUnit5. Wir zeigen anhand eines Beispiels die minimale Migration.

Maven Projekt als Vorlage

Laden Sie das Maven Projekt vorlage.zip herunter. Starten Sie Eclipse oder eine andere IDE und importieren Sie das Vorlagen Projekt (JumpStart):In Eclipse wird das Projekt geladen mit Java 8 und JUnit 4. Testen Sie das Projekt über das Terminal via -> mvn clean install -U Der Maven Build sollte fehlerfrei funktionieren:

Java Migration 8 nach 11

Wir wechseln nun die Java Version von 8 auf 11:Das Projekt basiert nun auf Java 11, testen Sie den Maven BuildDas Projekt funktioniert immer noch ohne Fehler.

Spring Stack Upgrade (pom.xml)

Im Maven POM File migrieren wir den Spring Stack von Version 2.1.8 nach 2.3.3:<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.3.3.RELEASE</version>
  <relativePath /> <!-- lookup parent from repository -->
</parent>Der Maven Build sollte immer noch funktionieren.

Java 11 (pom.xml)

Wir definieren im Maven pom.xml die Java 11 Version:<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>>

<java.version>11</java.version>
</properties>Der Maven Build sollte immer noch funktionieren.

JUnit 5

Die Migration nach JUnit 5 erfolgt via das spring-boot-starter-test. Passen Sie die Dependancy wie folgt an:<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>Die Unit Tests ergeben einen Kompiliationsfehler.Öffnen Sie die Test Datei JumpstartApplicationJpaTests.java und entfernen Sie die fehlerhaften import Statements.Entfernen Sie die @RunWith(SpringRunner.class) Annotationen.Importieren Sie die Annotation @Test und die assert Anweisungen neu mit den Jupiter Klassen:package ch.std.jumpstart;
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertNotNull;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.context.SpringBootTest.WebEnvironment;
import

org.springframework.boot.test.web.client.TestRestTemplate;
import org.springframework.boot.web.server.LocalServerPort;

import ch.std.jumpstart.dto.CityDTO;

@SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)
public class JumpstartApplicationJpaTests {
 ...
}Öffnen Sie die Test Datei JumpstartApplicationTests.java und korrigieren Sie die Fehler wie folgt:package ch.std.jumpstart;

import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;
import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.status;

import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.extension.ExtendWith;
import org.skyscreamer.jsonassert.JSONAssert;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.WebMvcTest;
import org.springframework.http.MediaType;
import org.springframework.test.context.ActiveProfiles;
import org.springframework.test.context.junit.jupiter.SpringExtension;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.MvcResult;

import ch.std.jumpstart.rest.CityAutoCompleteController;

@ExtendWith(SpringExtension.class)
@WebMvcTest(CityAutoCompleteController.class)
@ActiveProfiles("test")
public class JumpstartApplicationTests {
  @Autowired
  private MockMvc mvc;

  @Test
  public void contextLoads() {
  }

  @Test
  public void testCityAutoCompleteController() throws Exception {
    MvcResult mvcResult = mvc.perform(get("/rest/auto/cities?value=Bern").contentType(MediaType.APPLICATION_JSON))
      .andExpect(status().isOk()).andReturn();
    String expected = "[Bern]";
    String actual = mvcResult.getResponse().getContentAsString();
    JSONAssert.assertEquals(expected, actual, false);
  }
}Der Maven Build sollte wieder funktionieren und die Migration ist abgeschlossen.

Weitere Infos

Die JUnit 5 Libraries werden durch die Spring Boot Libraries referenziert. Referenzieren Sie die JUnit Libraries nie direkt via Eclipse.Die migrierte Anwendung finden Sie hier migriert.zip

Feedback

War dieser Blog für Sie wertvoll. Wir danken für jede Anregung und Feedback

Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

https://www.simtech-ag.ch/blog/springboot/migration11