JAVA SPRING BOOT

Czym jest Spring Boot?

- Spring Boot to framework oparty na Javie, który służy do tworzenia samodzielnych aplikacji, które są łatwe do zbudowania i uruchomienia. Spring Boot to rozszerzenie Springa.

Czym jest Spring Boot?

- Spring Boot to framework oparty na Javie, który służy do tworzenia samodzielnych aplikacji, które są łatwe do zbudowania i uruchomienia. Spring Boot to rozszerzenie Springa.

- Spring Boot oferuje:

Czym jest Spring Boot?

- Spring Boot to framework oparty na Javie, który służy do tworzenia samodzielnych aplikacji, które są łatwe do zbudowania i uruchomienia. Spring Boot to rozszerzenie Springa.

- Spring Boot oferuje:
  - Łatwiejsze uruchamianie aplikacji

Czym jest Spring Boot?

- Spring Boot to framework oparty na Javie, który służy do tworzenia samodzielnych aplikacji, które są łatwe do zbudowania i uruchomienia. Spring Boot to rozszerzenie Springa.

- Spring Boot oferuje:
  - Łatwiejsze uruchamianie aplikacji
  - Automatyczną konfigurację

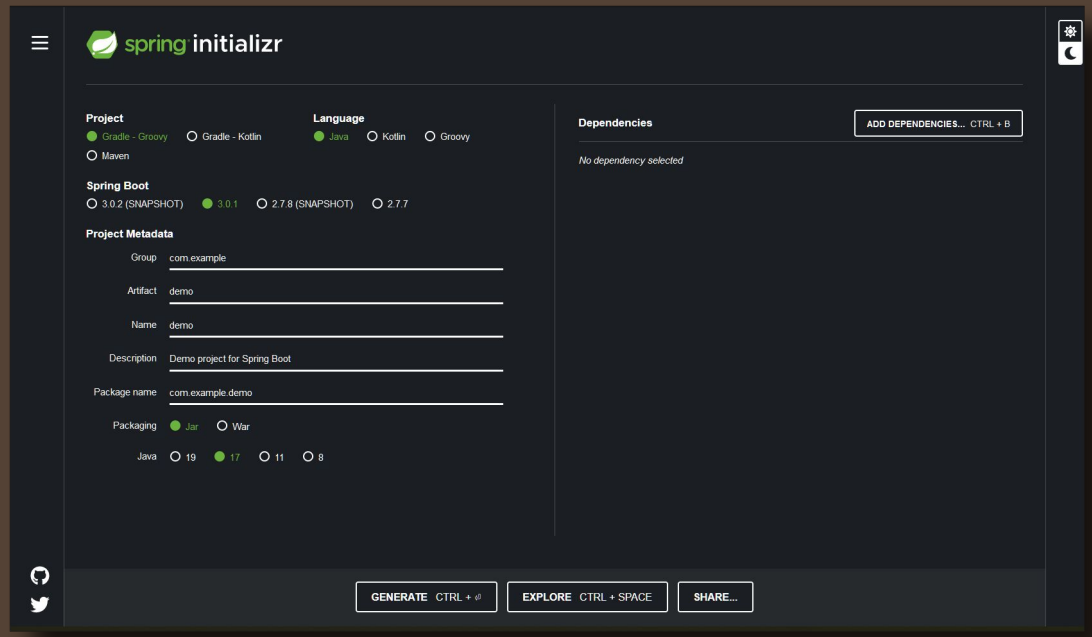
Czym jest Spring Boot?

- Spring Boot to framework oparty na Javie, który służy do tworzenia samodzielnych aplikacji, które są łatwe do zbudowania i uruchomienia. Spring Boot to rozszerzenie Springa.

- Spring Boot oferuje:
  - Łatwiejsze uruchamianie aplikacji
  - Automatyczną konfigurację
  - Szybsze tworzenie aplikacji

Czym jest Spring Boot?

- https://start.spring.io/



Jak narodziła się koncepcja Spring Boota?

Jak narodziła się koncepcja Spring Boota?

- Sam Spring ma pewne… Niedogodności:

Jak narodziła się koncepcja Spring Boota?

- Sam Spring ma pewne… Niedogodności:

  - szeroko bazuje na konfiguracji

Jak narodziła się koncepcja Spring Boota?

- Sam Spring ma pewne… Niedogodności:

  - szeroko bazuje na konfiguracji

  - kompatybilne biblioteki i ich konfiguracje

Jak narodziła się koncepcja Spring Boota?

- Sam Spring ma pewne… Niedogodności:

  - szeroko bazuje na konfiguracji

  - kompatybilne biblioteki i ich konfiguracje

Jak narodziła się koncepcja Spring Boota?

- Sam Spring ma pewne… Niedogodności:

  - szeroko bazuje na konfiguracji

  - kompatybilne biblioteki i ich konfiguracje

  - wymagany serwer

Jak narodziła się koncepcja Spring Boota?

- Sam Spring ma pewne… Niedogodności:

  - szeroko bazuje na konfiguracji

  - kompatybilne biblioteki i ich konfiguracje

  - wymagany serwer

  - boilerplate code

Jak narodziła się koncepcja Spring Boota?

- Sam Spring ma pewne... Niedogodności:

  - szeroko bazuje na konfiguracji

  - kompatybilne biblioteki i ich konfiguracje

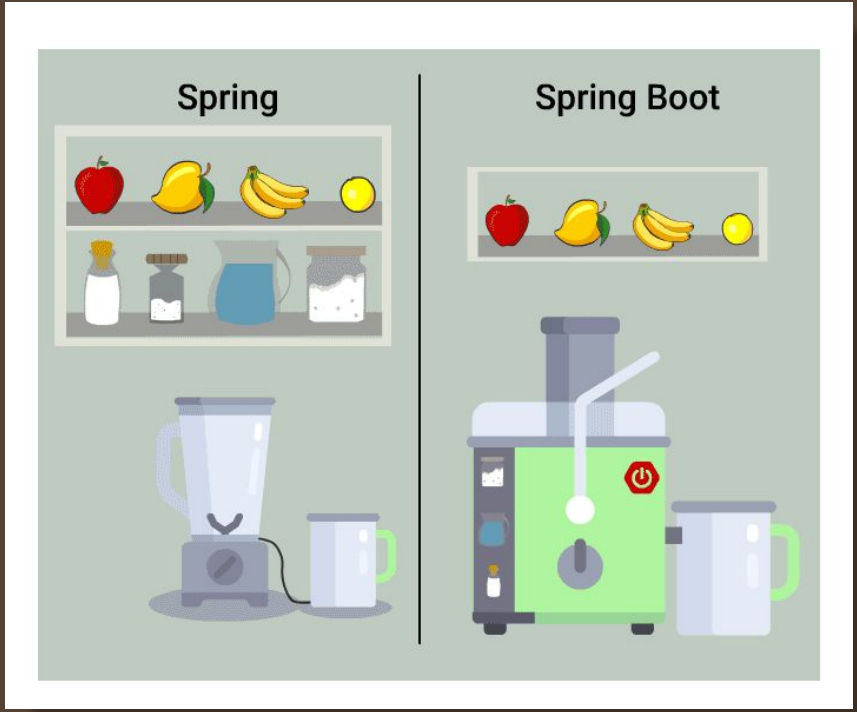
  - wymagany serwer

  - boilerplate code

```java
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

```python
print('Hello, World!')
```

Spring vs Spring Boot

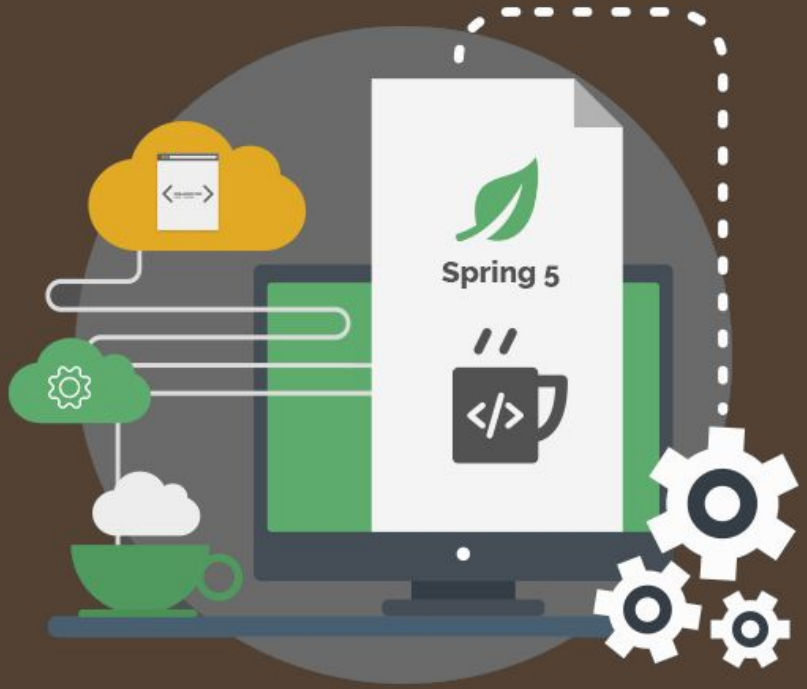


https://devrant.com

Jak narodziła się koncepcja Spring Boota?

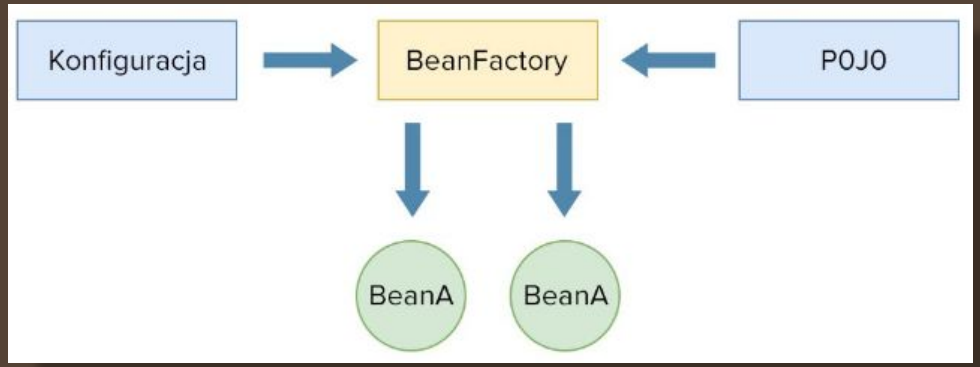
Spring	Spring Boot
Konieczność ręcznego tworzenia konfiguracji	Konfiguracje tworzone są automatycznie (ale mogą być edytowane)
Pozwala tworzyć aplikacje oparte na mikroserwisach	Pozwala tworzyć aplikacje stand-alone
Wymaga zewnętrznego serwera	Posiada wbudowany i skonfigurowany serwer (Tomcat lub Jetty)
Wymaga dużej ilości boilerplate code	Znacząco redukuje ilość potrzebnego kodu
Wymaga ręcznego zarządzania bibliotekami	Odpowiednie biblioteki są dołączane automatycznie

Techniczna strona Spring Boot



Obiekt zarządzany - Bean

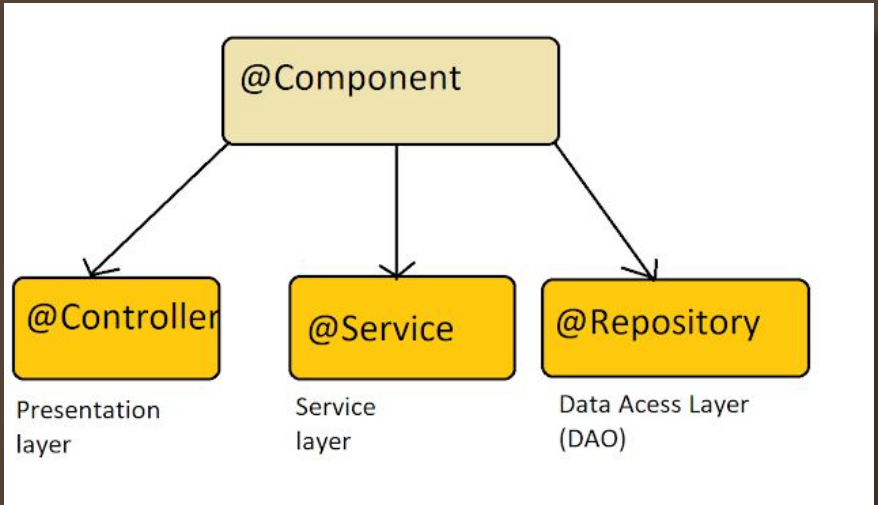
- Context and Dependency Injection
- Główne zalety
- Sposoby zdefiniowania w kodzie
  - Adnotacje
  - Instancje w metodach klasy konfiguracyjnej
  - XML
- Kiedy stosować Beany



Schemat tworzenia Beanów w Spring

Spring Boot - Adnotacje

- Najpopularniejsze adnotacje
  - @Component
  - @Repository
  - @Service
  - @Controler/@RestController



https://www.bdabek.pl/wp-content/uploads/2020/12/stereotypes.png

Mechanizm wstrzykiwania zależności

- Sposoby wstrzykiwania zależności
  - Konstruktor
  - Pole
  - Właściwości / metody

```java
public class Car {

    1 usage
    private CarService carService;

    @Autowired
    public Car(CarService carService) {
        this.carService = carService;
    }
}
```

Wstrzykiwanie przez konstruktor

```java
public class Car {
    1 usage
    private CarService carService;

    @Autowired
    public void setCarService(CarService carService){
        this.carService = carService;
    }
}
```

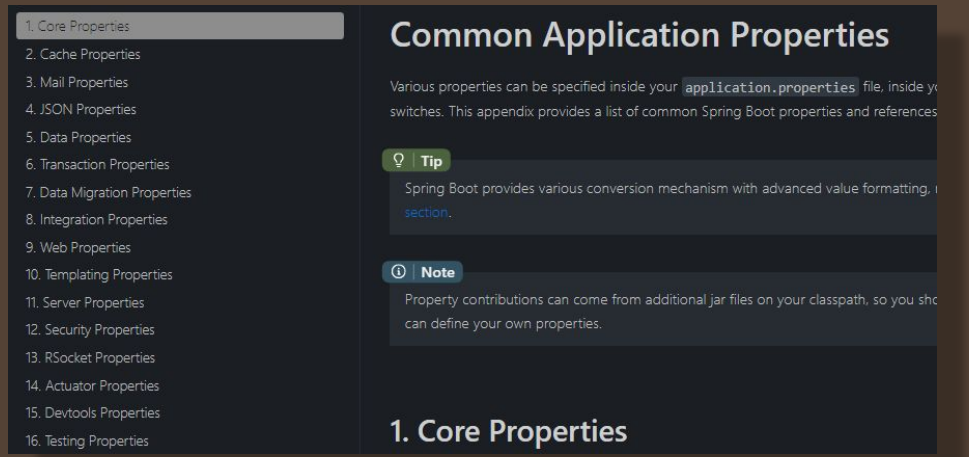
Wstrzykiwanie przez właściwości / metody

```java
public class Car {
    @Autowired
    private CarService carService;
}
```

Wstrzykiwanie przez pole

Konfiguracja i profilowanie

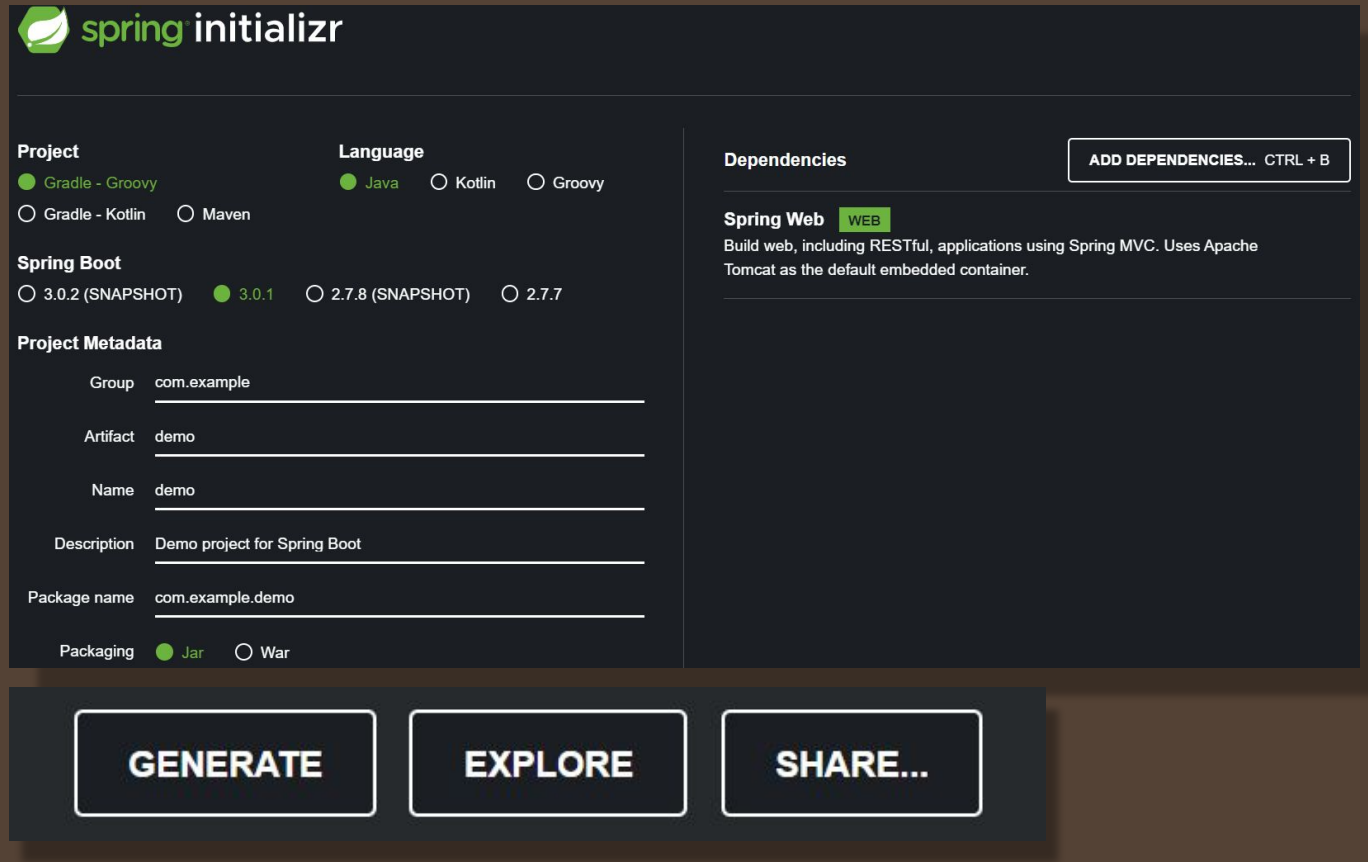
- Sposoby zarządzania konfiguracją
  - Nadpisanie istniejącej konfiguracji
  - Definiowanie własnych zmiennych
- Wczytywanie konfiguracji
  - @Value
- Profilowanie
  - application.properties
  - application-prod.properties
  - application-dev.properties



https://docs.spring.io/spring-boot/docs/current/reference/html/application-properties.htm

Przykładowy projekt “Hello World”

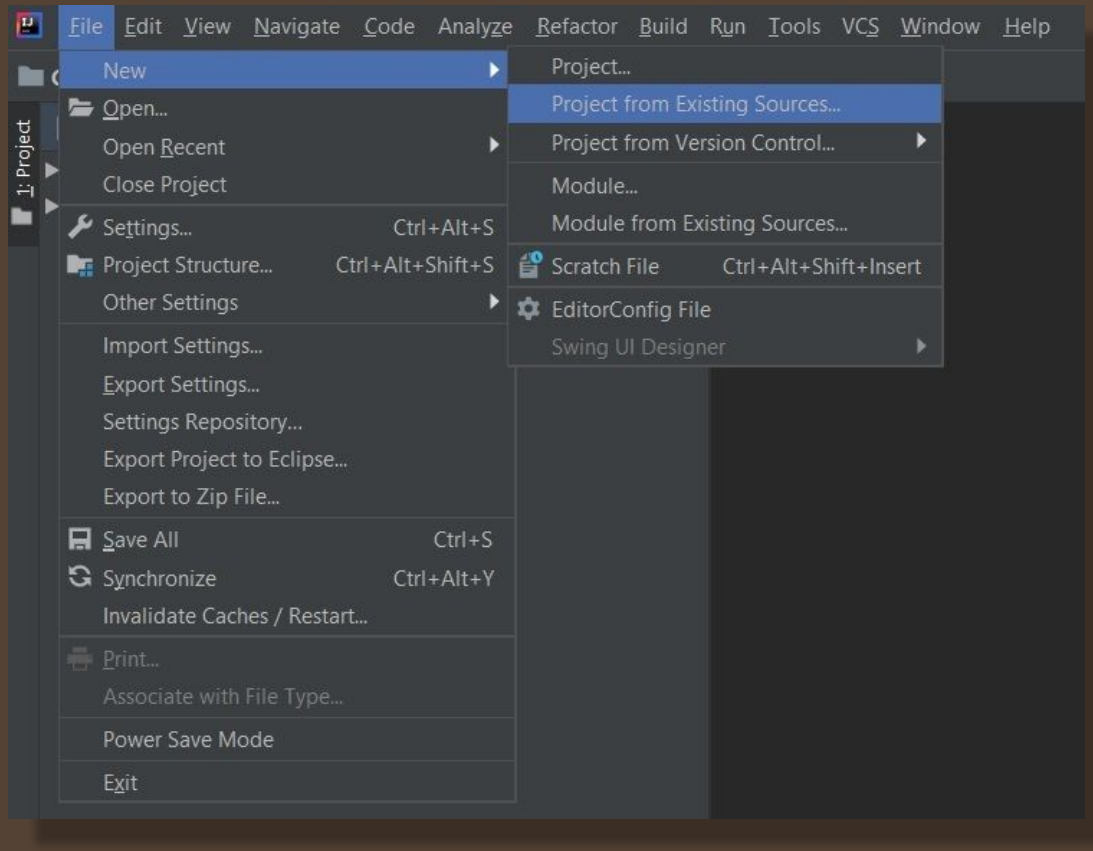
Spring Initializr



https://start.spring.io

```
Project: Maven

Language: Java

Spring Boot: 2.2.8

Packaging: JAR

Java: 8

Dependencies: Spring
              Web
```

Otworzenie projektu w IDE



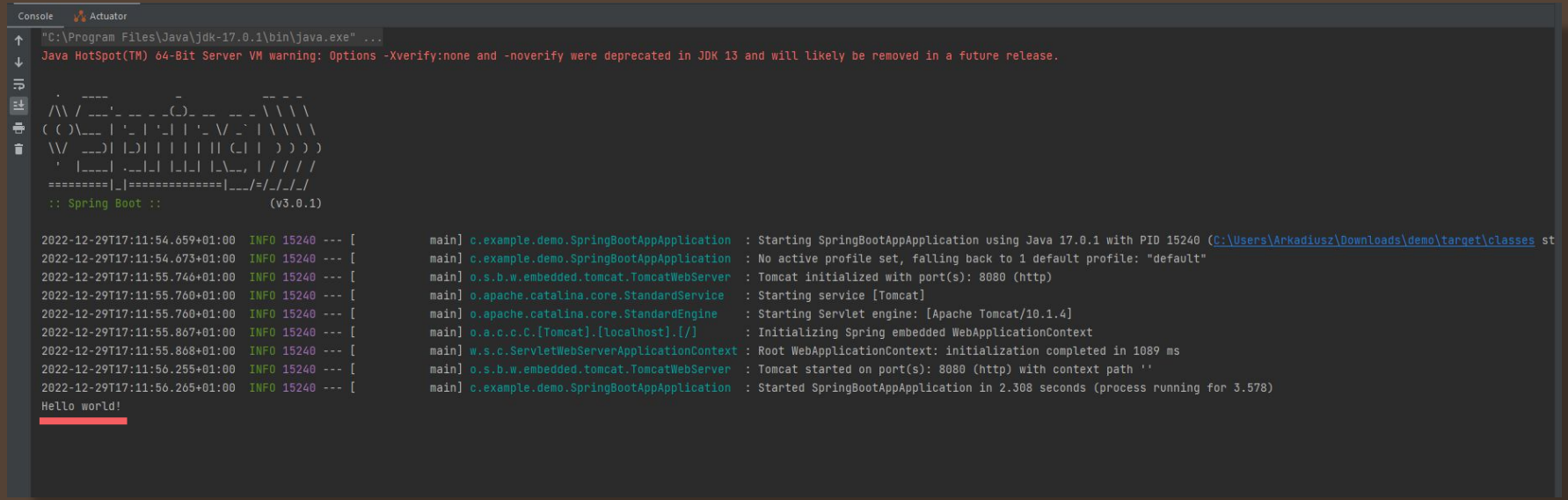
Wypakuj pobrany plik ZIP.
W IntelliJ wybierz File > New > Project from Existing Sources…

Dodanie pliku z głównymi funkcjami

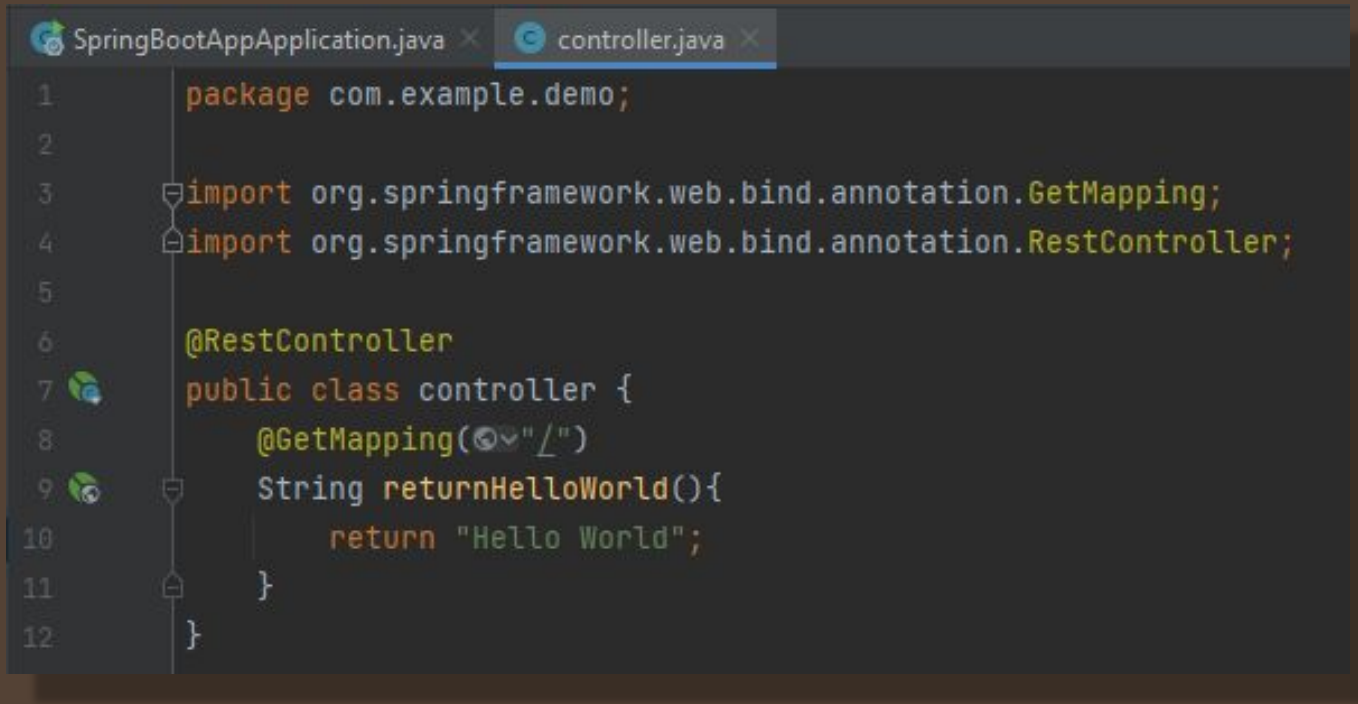
```java
@SpringBootApplication

// Main class
// Implementing CommandLineRunner interface
public class SpringBootAppApplication implements CommandLineRunner
{
    // Method 1
    // run() method for springBootApplication to execute
    @Override
    public void run(String args[]) throws Exception
    {
        // Print statement when method is called
        System.out.println("Hello world!");
    }

    // Method 2
    // Main driver method
    public static void main(String[] args)
    {
        // Calling run() method to execute
        // SpringBootApplication by

        // invoking run() inside main() method
        SpringApplication.run(SpringBootAppApplication.class, args);
    }
}
```

Plik src > main > java > com.example.demo > SpringBootAppApplication.java

Test uruchomienia

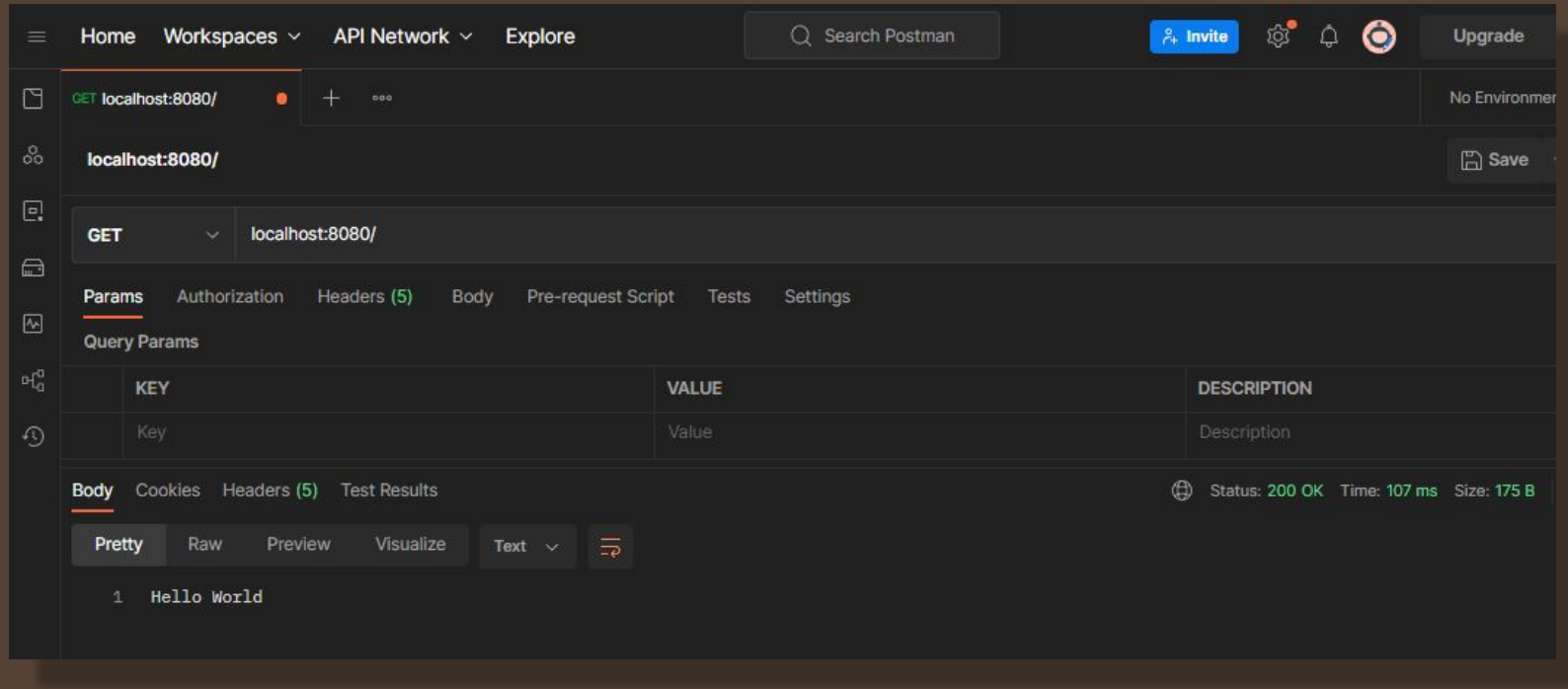


Hello World w REST API



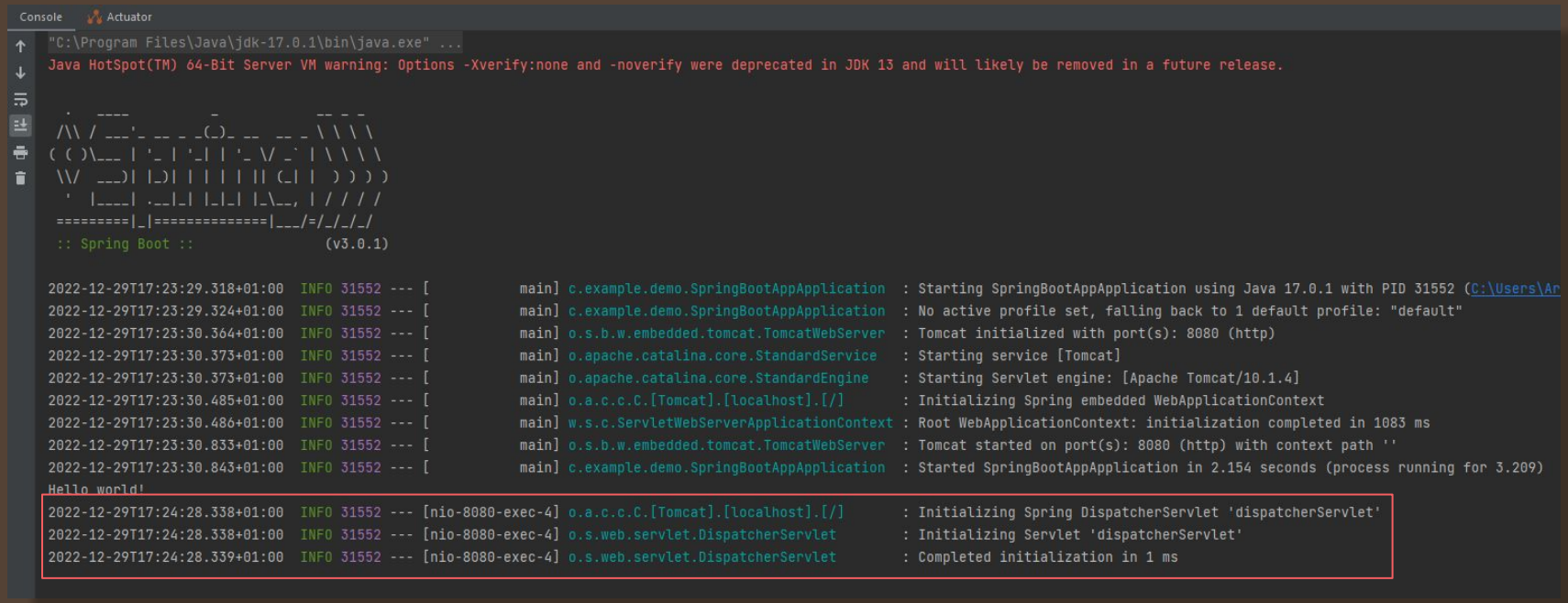
Plik src > main > java > com.example.demo > controller.java

Zapytanie

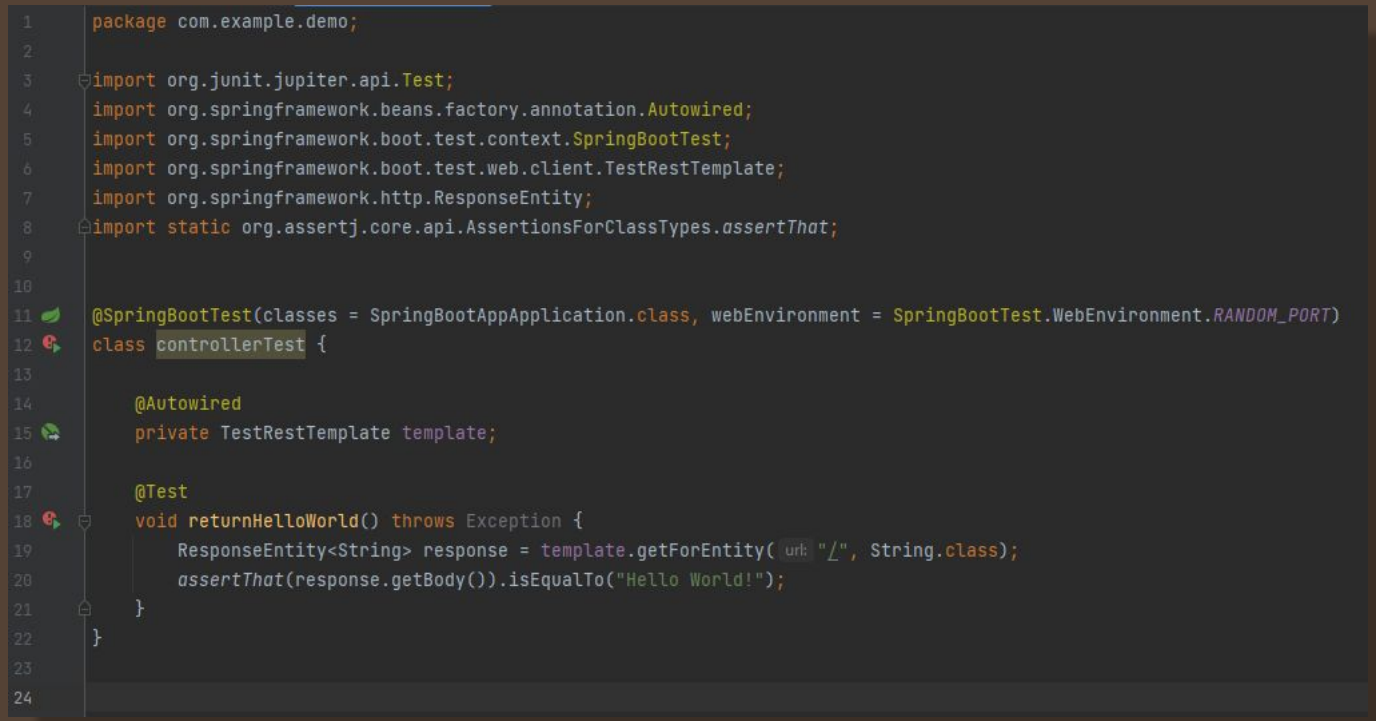


Zapytanie GET do localhost:8080 i odpowiedź aplikacji

Zapytanie


Console
Actuator
"C:\Program Files\Java\jdk-17.0.1\bin\java.exe" ...
Java HotSpot(TM) 64-Bit Server VM warning: Options -Xverify:none and -noverify were deprecated in JDK 13 and will likely be removed in a future release.
:: Spring Boot :: (v3.0.1)
2022-12-29T17:23:29.318+01:00 INFO 31552 --- [main] c.example.demo.SpringBootAppApplication : Starting SpringBootAppApplication using Java 17.0.1 with PID 31552 (C:\Users\Ar
2022-12-29T17:23:29.324+01:00 INFO 31552 --- [main] c.example.demo.SpringBootAppApplication : No active profile set, falling back to 1 default profile: "default"
2022-12-29T17:23:30.364+01:00 INFO 31552 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2022-12-29T17:23:30.373+01:00 INFO 31552 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2022-12-29T17:23:30.373+01:00 INFO 31552 --- [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.4]
2022-12-29T17:23:30.485+01:00 INFO 31552 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2022-12-29T17:23:30.486+01:00 INFO 31552 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1083 ms
2022-12-29T17:23:30.833+01:00 INFO 31552 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2022-12-29T17:23:30.843+01:00 INFO 31552 --- [main] c.example.demo.SpringBootAppApplication : Started SpringBootAppApplication in 2.154 seconds (process running for 3.209)
Hello world!
2022-12-29T17:24:28.338+01:00 INFO 31552 --- [nio-8080-exec-4] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2022-12-29T17:24:28.338+01:00 INFO 31552 --- [nio-8080-exec-4] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2022-12-29T17:24:28.339+01:00 INFO 31552 --- [nio-8080-exec-4] o.s.web.servlet.DispatcherServlet : Completed initialization in 1 ms

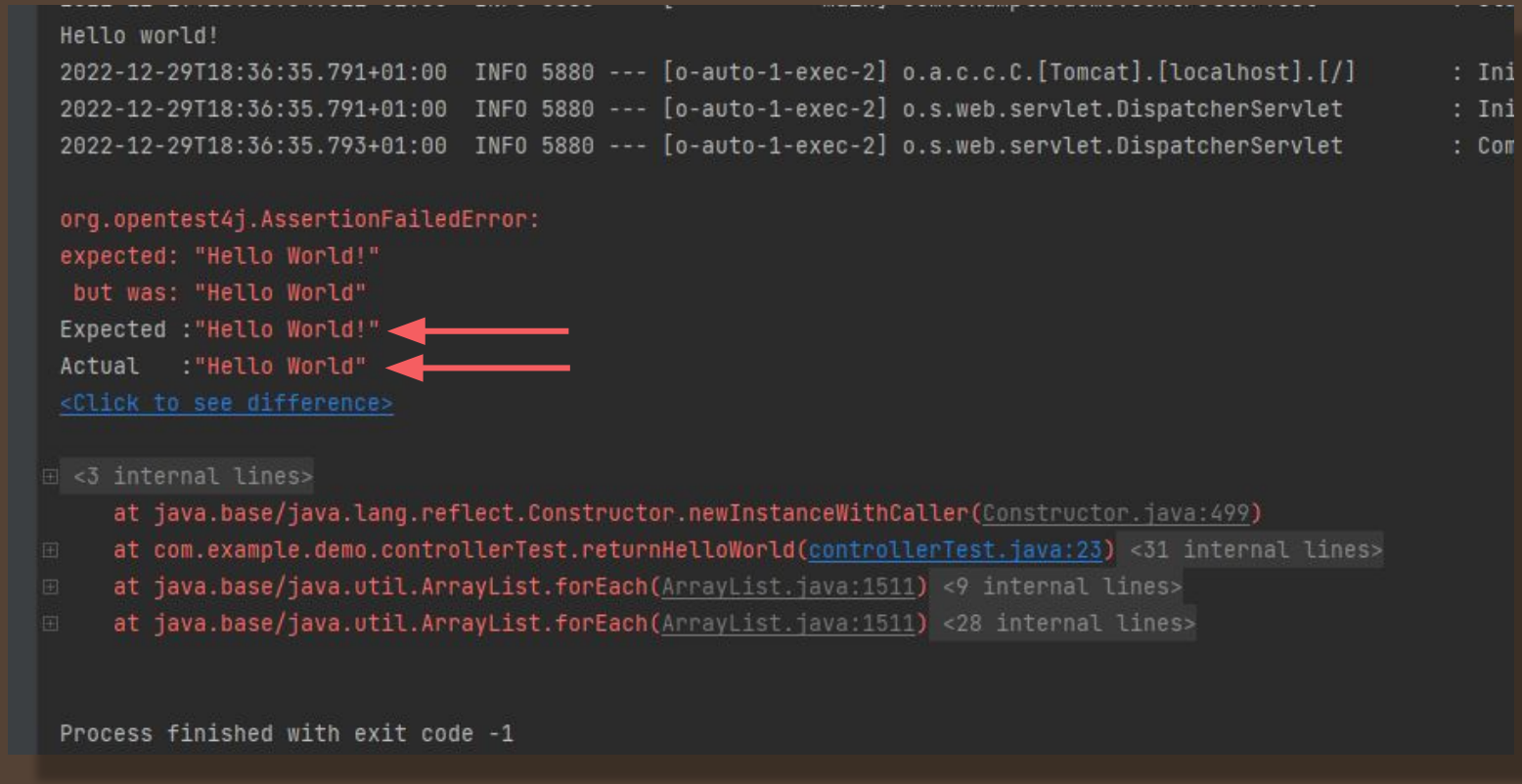
Tworzenie testów



```java
package com.example.demo;

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.web.client.TestRestTemplate;
import org.springframework.http.ResponseEntity;
import static org.assertj.core.api.AssertionsForClassTypes.assertThat;


@SpringBootTest(classes = SpringBootAppApplication.class, webEnvironment = SpringBootTest.WebEnvironment.RANDOM_PORT)
class controllerTest {

    @Autowired
    private TestRestTemplate template;

    @Test
    void returnHelloWorld() throws Exception {
        ResponseEntity<String> response = template.getForEntity( url: "/", String.class);
        assertThat(response.getBody()).isEqualTo("Hello World!");
    }
}
```

Test > java > com.example.demo > controllerTest.java

Wynik przeprowadzonego testu



Koniec

Finis coronat opus