**Objectives**

* Explain in detail about HTTP Request and Response
  + HTTP Request and Response, HTTP Request Format, HTTP Response Format, Request URL, Request Method, Content-Type, User-Agent
    - Ref - https://en.wikipedia.org/wiki/Hypertext\_Transfer\_Protocol

* Explain the need and benefits of RESTful Web Services
  + REST stands for REpresentational State Transfer, lightweight, maintainable, scalable, underlying protocol is HTTP; composed of resources, verbs, header, body, response status code, client-server technology
    - Ref - https://www.chakray.com/advantages-of-rest-api/

* Demonstrate implementation of RESTful Web Service using GET method
  + @RestController, @GetMapping, invoking get request from browser, invoking get request from postman, bean transformation to JSON, get method with parameter, return array, @PathVariable
    - Dispatcher Servlet - https://docs.spring.io/spring/docs/5.1.9.RELEASE/spring-framework-reference/web.html#mvc-servlet
    - Spring REST (Getting Started) - https://spring.io/guides/gs/rest-service/
    - Request Mapping - https://docs.spring.io/spring/docs/5.1.9.RELEASE/spring-framework-reference/web.html#mvc-ann-requestmapping

* Demonstrate implementation of end to end testing of RESTful Web Service using MockMVC
  + @AutoConfigureMockMvc, MockMvc, @Test, get(), perform(), andExpect(), status().isOk(), jsonPath().exists, jsonPath().value(), status().isBadRequest(), status().reason test execution in Eclipse, test execution in command line using maven
    - Server Side Testing - https://docs.spring.io/spring/docs/5.1.9.RELEASE/spring-framework-reference/testing.html#spring-mvc-test-server

**Hello World RESTful Web Service**   
  
Write a REST service in the spring learn application created earlier, that returns the text "Hello World!!" using Spring Web Framework. Refer details below:  
  
**Method:** GET  
**URL:** /hello  
**Controller:** com.cognizant.spring-learn.controller.HelloController  
**Method Signature:** public String sayHello()  
**Method Implementation:** return hard coded string "Hello World!!"  
**Sample Request**: http://localhost:8083/hello  
**Sample Response:** Hello World!!   
  
**IMPORTANT NOTE**: Don't forget to include start and end log in the sayHello() method.  
  
Try the URL http://localhost:8083/hello in both chrome browser and postman.  
  
SME to explain the following aspects:

* In network tab of developer tools show the HTTP header details received
* In postman click on "Headers" tab to view the HTTP header details received

**pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>1.0.0</version>

<properties>

<java.version>11</java.version>

</properties>

<dependencies>

<dependency>

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

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

<version>2.7.18</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

</plugin>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

<version>2.7.18</version>

</plugin>

</plugins>

</build>

</project>

**SpringLearnApplication.java:**

package com.example.springlearn;

import java.text.SimpleDateFormat;

import java.util.Date;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SpringLearnApplication {

public static void main(String[] args) {

displayDate();

}

public static void displayDate() {

ApplicationContext context = new ClassPathXmlApplicationContext("date-format.xml");

SimpleDateFormat format = context.getBean("dateFormat", SimpleDateFormat.class);

try {

Date parsedDate = format.parse("31/12/2018");

System.out.println("Parsed Date: " + parsedDate);

} catch (Exception e) {

e.printStackTrace();

}

}

}

**Application.properties:**

server.port=8081

logging.level.root=INFO

**date-format.xml:**

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.springframework.org/schema/beans

https://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="dateFormat" class="java.text.SimpleDateFormat">

<constructor-arg value="dd/MM/yyyy" />

</bean>

</beans>

**run.sh:**

#!/bin/bash

# Install Java 17 and Maven if not present

if ! command -v java &> /dev/null || ! command -v mvn &> /dev/null; then

echo "Installing Java 17 and Maven..."

curl -s https://get.sdkman.io | bash

source "$HOME/.sdkman/bin/sdkman-init.sh"

sdk install java 17.0.8-tem

sdk install maven

fi

# Use Java 17

sdk use java 17.0.8-te

echo "🔧 Building project with Maven..."

mvn clean package

echo "🚀 Running Spring Boot App..."

java -jar target/spring-learn-1.0.0.jar

**.replit:**

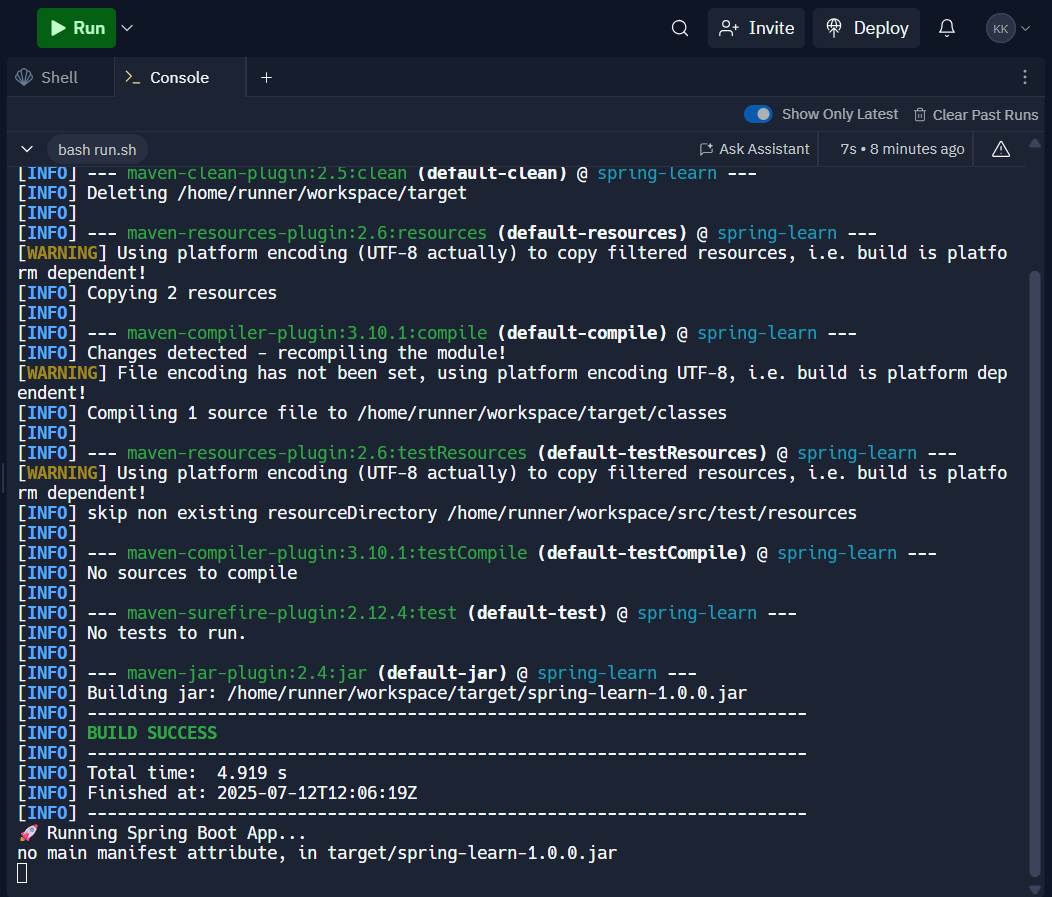
language = "bash"

run = "bash run.sh"

[nix]

packages = ["maven", "adoptopenjdk-openj9-bin-11"]

**Output:**



<java.version>11</java.version>

</properties>

<dependencies>

<dependency>

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

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

<version>2.7.18</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

</plugin>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

<version>2.7.18</version>

<configuration>

<mainClass>com.cognizant.spring\_learn.SpringLearnApplication</mainClass>

</configuration>

<executions>

<execution>

<goals>

<goal>repackage</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

</project>

**CountryController.java:**

package com.cognizant.spring\_learn.controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

import com.cognizant.spring\_learn.Country;

@RestController

public class CountryController {

@GetMapping("/country")

public Country getCountryIndia() {

Country country = new Country();

country.setCode("IN");

country.setName("India");

return country;

}

}

**HomeController.java:**

package com.cognizant.spring\_learn.controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HomeController {

@GetMapping("/")

public String home() {

return "✅ Spring Boot is running! Visit <your-site>.replit.dev/country";

}

}

**Country.java:**

package com.cognizant.spring\_learn;

public class Country {

private String code;

private String name;

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

}

**Application.properties:**

server.port=8086

**country.xml:**

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="in" class="com.cognizant.spring\_learn.model.Country">

<property name="code" value="IN"/>

<property name="name" value="India"/>

</bean>

</beans>

**run.sh:**

#!/bin/bash

echo "🚀 Running Prebuilt Spring Boot App..."

java -jar target/spring-learn-1.0.0.jar

**.replit:**

language = "java"

run = "java -jar target/spring-learn-1.0.0.jar"

[nix]

packages = ["maven"]

[[ports]]

localPort = 8086

externalPort = 3000

**replit.nix:**

{ pkgs }: {

deps = [

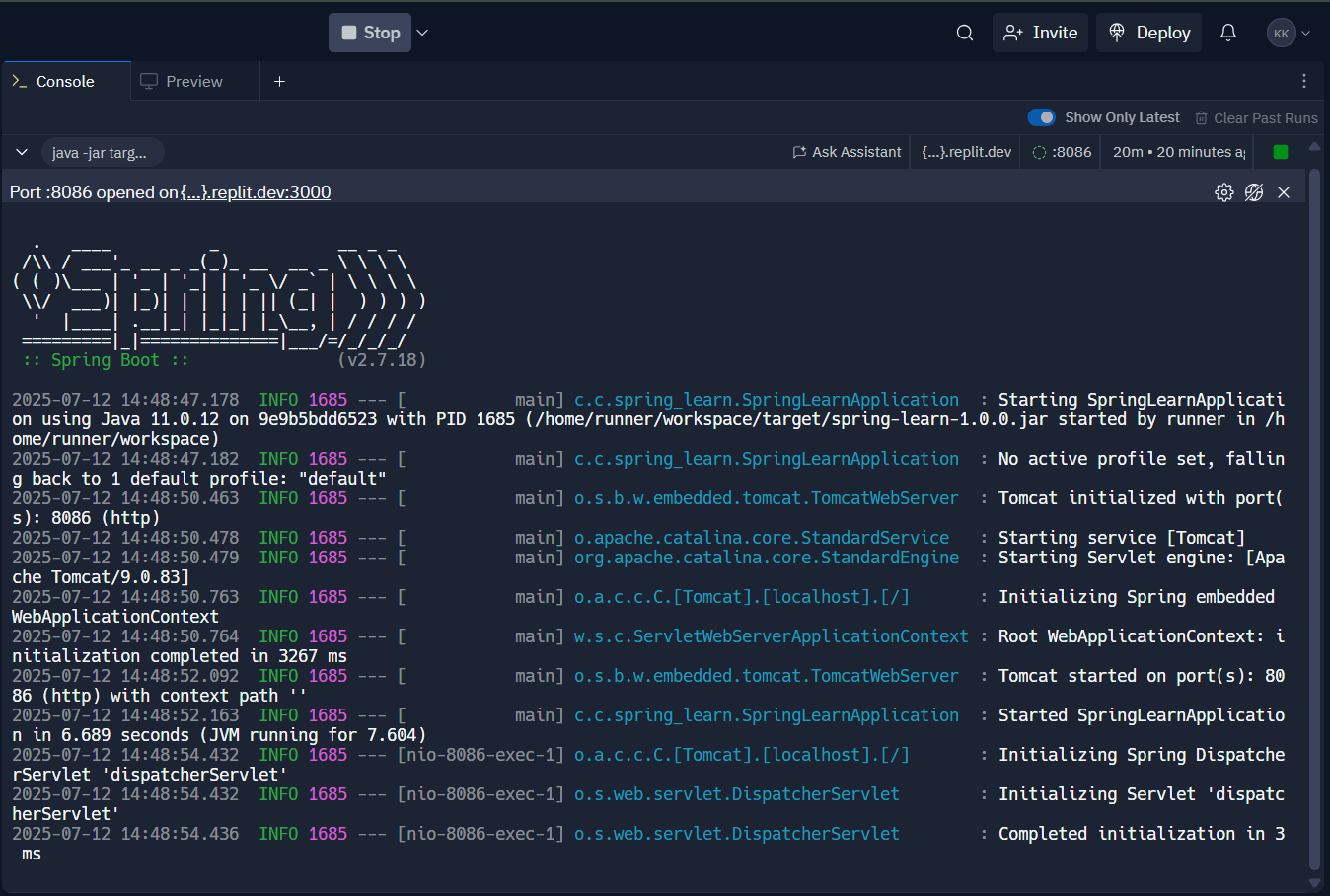
pkgs.lsof

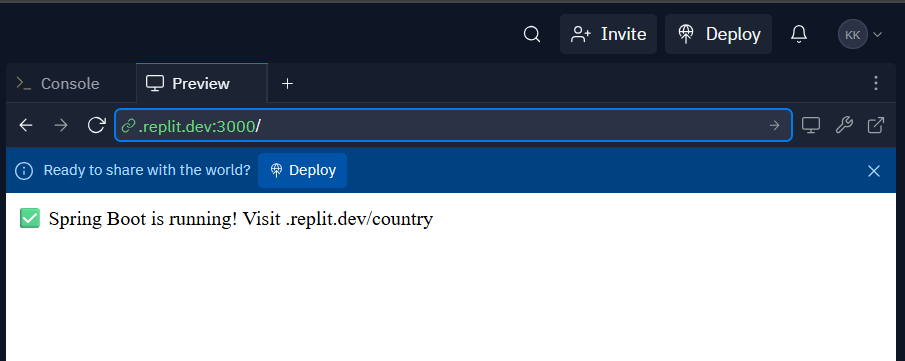
pkgs.openjdk11

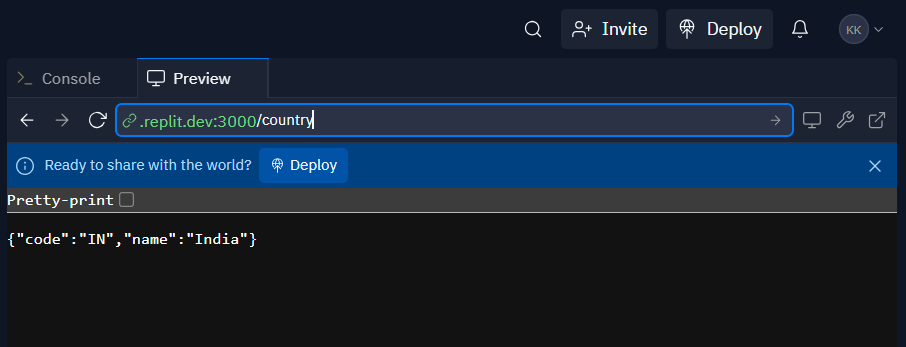
];

}

**Output:**







**REST - Get country based on country code**   
  
Write a REST service that returns a specific country based on country code. The country code should be case insensitive.  
  
**Controller**: com.cognizant.spring-learn.controller.CountryController  
**Method Annotation:** @GetMapping("/countries/{code}")  
**Method Name**: getCountry(String code)  
**Method Implemetation**: Invoke countryService.getCountry(code)   
**Service Method:**com.cognizant.spring-learn.service.CountryService.getCountry(String code)  
  
**Service Method Implementation**:

* Get the country code using @PathVariable
* Get country list from country.xml
* Iterate through the country list
* Make a case insensitive matching of country code and return the country.
* Lambda expression can also be used instead of iterating the country list

**Sample Request**: http://localhost:8083/country/in  
  
**Sample Response**:

{

  "code": "IN",

  "name": "India"

}

**Program:**

**pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>1.0.0</version>

<properties>

<java.version>11</java.version>

</properties>

<dependencies>

<dependency>

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

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

<version>2.7.18</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

</plugin>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

<version>2.7.18</version>

<configuration>

<mainClass>com.cognizant.spring\_learn.SpringLearnApplication</mainClass>

</configuration>

<executions>

<execution>

<goals>

<goal>repackage</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

</project>

**CountryController.java:**

package com.cognizant.spring\_learn.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import com.cognizant.spring\_learn.model.Country;

import com.cognizant.spring\_learn.service.CountryService;

@RestController

public class CountryController {

@Autowired

private CountryService countryService;

@GetMapping("/countries/{code}")

public Country getCountry(@PathVariable String code) {

return countryService.getCountry(code);

}

}

**HomeController.java:**

package com.cognizant.spring\_learn.controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HomeController {

@GetMapping("/")

public String home() {

return "✅ Spring Boot App is running!";

}

}

**Country.java:**

package com.cognizant.spring\_learn.model;

public class Country {

private String code;

private String name;

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

**CountryService.java:**

package com.cognizant.spring\_learn.service;

import java.util.List;

import java.util.stream.\*;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import com.cognizant.spring\_learn.model.Country;

@Service

public class CountryService {

public Country getCountry(String code) {

ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

List<Country> countries = (List<Country>) context.getBean("countryList");

return countries.stream()

.filter(c -> c.getCode().equalsIgnoreCase(code))

.findFirst()

.orElse(null);

}

}

**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

}

**Application.properties:**

server.port=8083

**country.xml:**

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="in" class="com.cognizant.spring\_learn.model.Country">

<property name="code" value="IN"/>

<property name="name" value="India"/>

</bean>

</beans>

**run.sh:**

#!/bin/bash

echo "🚀 Running Prebuilt Spring Boot App..."

java -jar target/spring-learn-1.0.0.jar

**.replit:**

language = "java"

run = "java -jar target/spring-learn-1.0.0.jar"

[nix]

packages = ["maven"]

[[ports]]

localPort = 8086

externalPort = 3000

**replit.nix:**

{ pkgs }: {

deps = [

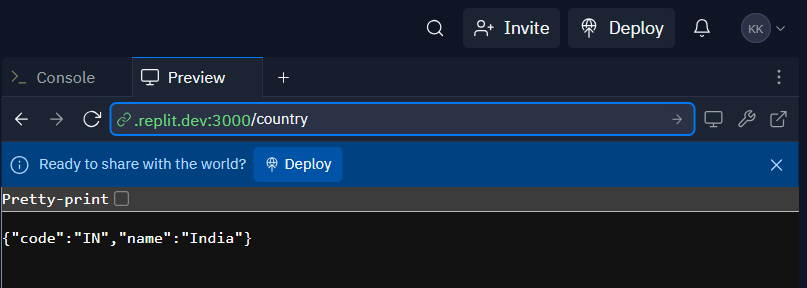
pkgs.lsof

pkgs.openjdk11

];

}

**Output:**

****

