C:\Users\Sangeeta-Laptop\AppData\Local\Microsoft\Windows\INetCache\Content.Word\icons8-database-52.png

Data Factory

Date: 20th May 2020

Author: Abhishek Prajapati

# Data Factory

**Github:** www.github.com/frostyaxe

**Team:** Frostyaxe

**Email:** prajapatiabhishek1996@gmail.com

## Abstract

Data Factory is a utility written in java that allows the user to retrieve the data from any application that can handle rest API request. The focus is on ease of use, reusability and maintainability.

**Copyright © 2020 Frostyaxe. This work is licensed under the Apache License 2.0.**

# Introduction

Data Factory in selenium allows the user to retrieve the data from the application that could handle the rest API request. It is created specifically for the cucumber projects where we need create to create data tables in the feature files.

For example, in this book, we are going to see how spring boot application will act as a data provider. With the help of spring boot application, we will read the JSON data and based on the request we will process the JSON data, if required and then we will send this data to the Selenium data class.

What is selenium Data Class?

Selenium Data Class is the class that stores the data retrieved from the spring boot application (or any other application that handles the rest API request). If you want to process the retrieved data further, then you can do that too in the selenium data class. We will talk more about it later.

Why do we need this?

You do need to keep the test data in the project itself. In any server you can place the test data and then with the help of spring boot application you can process the data and then you send the processed data back to the user in the form of JSON. In cucumber, we create data tables to provide required data to the step definitions from the feature file. What happens when you have large amount of data? Maintaining such kind of table will be hectic. So, in order to resolve this issue, we are going to use data factory in the cucumber project and we will see how it resolves our day to day issues too.

# Data Provider

If you do not have much knowledge about spring boot then you can learn it from the below URL.  
  
<https://www.javatpoint.com/spring-boot-tutorial>

Spring boot application always starts with the main method. SpringApplication class provides a convenient way to bootstrap a Spring application. Check the sample script below.

@SpringBootApplication

public class Application

{

public static void main(String[] args)

{

SpringApplication.run(Application.class, args);

}

}

}

After that you need to create the service classes. Service class file has normal Java function that reads the data and performs the desired operations on it. For example, the data present in the excel sheet can be read with the help of Apache POI in Java. Hence in the service class you can use apache POI and read the data from excel sheet and then you can convert it into the JSON String.

Example:

public class Service

{

public String getData()

{

// Parser the JSON file content here

// Process the JSON file content

return content; // return the JSON content in string

}

}

Once the service file is created, we can now create the rest controller class. Rest Controller class maps the rest API path with the service method. So when you send a rest API request, based on the path it calls the service method and the data that service method returns it sends that data back to the client that has sent the rest API request.

This is how controller file looks like

@RestController // Spring boot rest controller annotation

public class Controller

{

@RequestMapping("/get") // Mapping the rest API path

public String getData()

{

return new Service().getData(); // Calling the getData method of service

}

}

In this way, you can create the basic data provider and with the help of it you can retrieve the data in the data class.

# Data Class

In order to create data class, you need to work with the two annotations: FactoryConfig and SearchData.

**FactoryConfig:** Thisannotation accepts the host name or IP and port number on which the spring boot application is running. This annotation is used with class.

*Example:*

@FactoryConfig(host = "localhost", port = 8081)

**SearchData:** This annotation accepts the rest API path to retrieve the data from particular service. This annotation is used with variables.

*Example:*

@SearchData(path = “get”)

You will also have to call the initData method as shown below.

DataFactory.initData(this); // this refers to current data class

Sample Data Class:

@FactoryConfig( host = "localhost", port = 8081 )

public class HotelDataClass

{

@SearchData( path = "hotel/goingTo" )

private String goingTo;

public HotelDataClass()

{

DataFactory.initData(this);

}

public String getGoingTo()

{

return goingTo;

}

}