Functional Web Services or API Test Automation using POST MAN

Authored by: Lakshmi Chandrasekhar Mamilla
Shanmukha Rontala

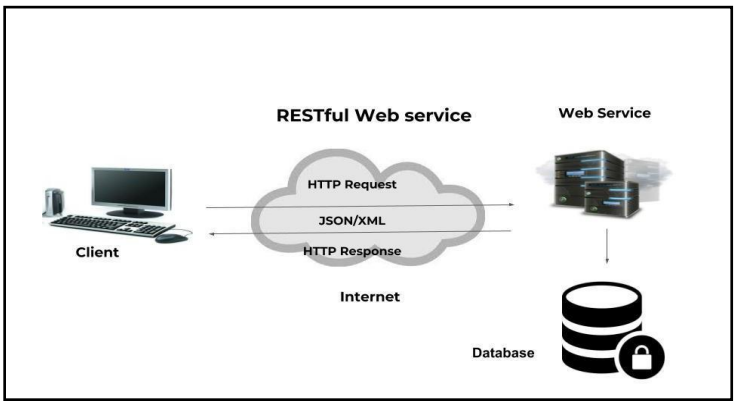
TABLE OF CONTENTS

1) Introduction

The main goal of functional Web Services or API testing is to validate the functionality of the business logic layer which occurs before the GUI integration test. Recognizing the common challenges and formulating the best test approach ensures the full test coverage and eventually reduces the number of defects rates throughout the test life cycle. This whitepaper will provide a brief introduction to Functional automation testing using Postman Tool and Newman tool for report generation.

2) What is a Web Service or API?

A Web Service is a unit of managed code that can be remotely invoked using HTTP, i.e., it can be activated using HTTP requests. So, Web Services allows you to expose the functionality of your existing code over the network. Once it is exposed to the network, other applications can use the functionality of your program. Both API and Web Service serve as a means of communication. The only difference is that a Web service facilitates interaction between two machines over a network. An API acts as an interface between two different applications so that they can communicate with each other any specified format JSON or XML.



There are two broad classes of web service:

- SOAP Services - These web services make use of the Web Service Definition Language (WDSL) and communicate using HTTP POST requests. They are essentially a serialization of RPC object calls into XML that can then be passed to the web service. The XML passed to the SOAP web services needs to match the format specified in the WSDL. SOAP web services are fully self-descripting, so most clients do not directly work with the SOAP XML language, but instead use a client-side proxy generator that creates client object representations of the web service (e.g. Java, .NET objects). The web service consumers interact with these language-specific representations of the SOAP web service.

- REST Services - A RESTful web API (also called a RESTful web service) is a web API implemented using HTTP and REST principles. Unlike SOAP-based web services, there is no "official" standard for RESTful web APIs. This is because REST is an architectural style, unlike SOAP, which is a protocol. Typically REST web services expose their operations as a series of unique "resources" which correspond to a specific URL. Each of the standard HTTP methods (POST, GET, PUT, and DELETE) then maps to the

four basic CRUD (Create, Read, Update, and Delete) operations on each resource. REST web services can use different data serialization methods (XML, JSON, RSS, etc.).

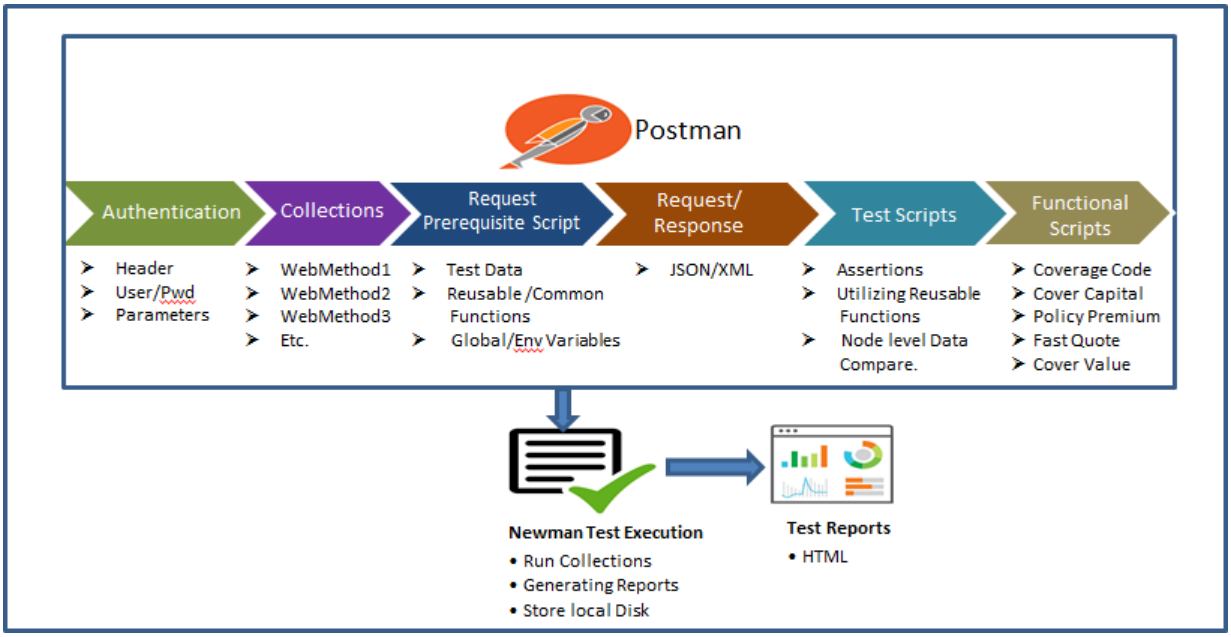
3) How important is Web Services or API testing?

The importance of functional API testing is described as the following with reasons:

- Web Services or API testing can be done independently of GUI ensuring that business logic is defect free.
- Web Services or API testing can help detect application behavior problems even before GUI integration & development
- Web Service Testing is to verify that all of the Application Programming Interfaces (APIs) exposed by your application operate as expected. In some ways, they are similar to unit tests in that they test specific pieces of code rather than user interface objects.
- Less time consuming than GUI functional testing.
- Lightweight XML or JSON data transfer modes.

4) Test Approach

The test approach explained about verification and validation of Request and Response of APIs. Postman works with Collections, and Collections are a group of API calls related to each other. Postman allows users to automate test cases in JavaScript with salient features like write test suites, build requests that can contain dynamic parameters, pass data between requests, etc. For validation of WS or API, on receiving a response, Postman validates the response as described in the test scripts and pre-request scripts. This document explains how to execute the Collections, generate report using the Newman tool, and send the reports to respective people.



5) Verification and validation process

A Postman test script is essentially JavaScript code executed after the request is sent, allowing access to response using postman command **pm.response** as an object. We can add Test data, reusable functions in Prerequisite scripts, and can include assertions and utilizing of reusable functions in test Scripts for as many tests as needed, depending on how many things you want to test for. Tests are saved as part of collection requests.

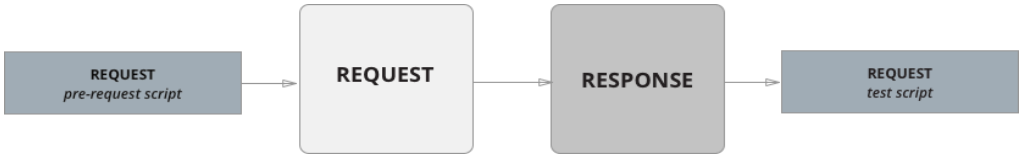
We can add test scripts to a collection, a folder, or a single request within a collection. A test script associated with a collection will run after every request in the collection. A test script associated with a folder will run after every request in the folder. This allows you to reuse commonly executed tests after every request. Collection and folder scripts can be updated in the collection or folder details respectively.

This is perfect for both back-end and front-end developers to ensure that everything is working properly with the WS or API.

Execution order of scripts

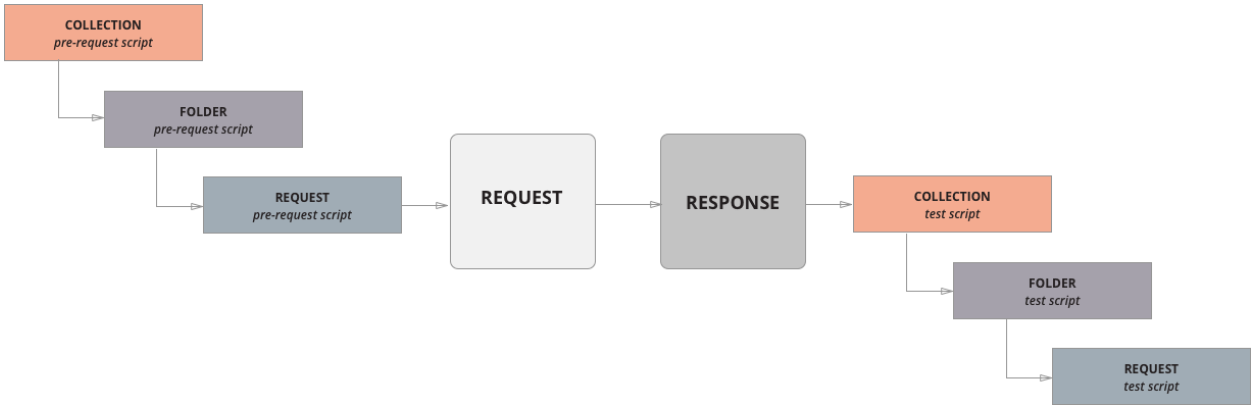
In Postman, the script execution order for a single request looks like this:

- A pre-request script associated with a request will be executed before the request is sent.
- A test script associated with a request will be executed after the request is sent.



For every request in a collection, scripts will execute in the following order:

- A pre-request script associated with a collection will before every request in the collection.
- A pre-request script associated with a folder will run before every request in the folder.
- A test script associated with a collection will run after every request in the collection.
- A test script associated with a folder will run after a request in the folder.



For every request in a collection, the scripts will always run according to the following:

Hierarchy: Collection-level script (if any), folder-level script (if any), and request-level script (if any). Note that this order of execution applies to both pre-request and test scripts.

6) About Newman Tool

I. Introduction to Newman

Newman is Postman's collection runner engine that sends API requests, receives the response, and then runs your test against the response. It is like postman's command line companion that generates specified reports. It is extensible and so can be integrated into continuous integration (CI/CD) servers and build systems.

II. Benefits of using Newman

- Makes it easy to run a collection of tests using command line
- Gives the ability to run a collection of tests written in postman right from most build tools
- Enables generating and storing report directly from the command line after Installing Newman.

III. How to install Node with Newman:-

For Windows, refer the blog given below:
http://blog.getpostman.com/2015/04/09/installing-newman-on-windows/

7) Scope for improvements

- Using external scripts to send mails to respective people with attached reports.
- Using .bat file to schedule the jobs to execute collections and send test results.
- Using Mock API Services to send Test Data to client-based collection to validate the responses.

8) Limitations

- Invoking external libraries in side postman scripts.