Agile testing for web API with Postman

Presented by Anthony Todisco
Agenda

• API and Agile
• API testing
• Rest-assured
• Postman
• Our journey with Postman
• Key takeaway
• Q&A
API and Agile

- Growth in the number of API
- Importance of time to market
API and Agile

- Agile methodology to produce quick results
- But UI testing is currently:
  - too slow
  - Not adapted to Agile
API and tests

• Test automation is the key
• Multiple possibilities for test automation

Which one to choose?
Rest-assured test

- First choice for most people
- Java Domain Specific Language (DSL) for test
- Support response parsing (XML and JSON)
- BDD Syntax:
  - Easy reading
  - Maintainable

```java
RestAssured.baseURI = "http://localhost:8080/myservice"

given()
  .param("x", "y")
  .header("z", "p")
when()
  .get("something")
then()
  .statusCode(200)
  .body("x.y", equalTo("z"));
```

- Source code and tests in the same project
Rest-assured test

- No report → need another unit test framework
- What if your code is not Java?
- If your QAs are not developers?
Postman

- A UI tool for API testing
- Good for:
  - Exploratory tests
  - Functional and automated tests
- Easy to use
- Support response parsing (XML and JSON)
- Define the concept of collection for a group of requests
Postman

- CLI (Newman) that integrates greatly with Jenkins
- CLI, XML and HTML reports
- Support SOAP call
- Basic Javascript testing with boolean

```javascript
var response = pm.response.json();

tests["Body contains headers"] = response.headers !== undefined;
tests["Header contains host"] = response.headers.host !== undefined;
```
Postman

- Headers
  - Request Headers
    - GET https://echo.getpostman.com/headers [200 OK, 851B, 182ms]
      - Body contains headers
      - Header contains host
      - Header contains test parameter sent as part of request header
  - Response Headers
      - response_headers [200 OK, 593B, 197ms]
      - Body contains Content-Type
      - response headers have key sent as part of request

- Request Methods
  - GET Request
    - GET https://echo.getpostman.com/get?test=123 [200 OK, 8488, 181ms]
      - Body contains headers
      - Body contains args
      - Body contains url
      - Args key contains argument passed as url parameter
  - POST Request
    - POST https://echo.getpostman.com/post [404 Not Found, 4148, 182ms]
Newman report for Postman tests

Well done!
No test failed on the current run using the environment.

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>Executed</th>
<th>Failed</th>
<th>Success Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterations</td>
<td></td>
<td>1</td>
<td>0</td>
<td>100.00</td>
</tr>
<tr>
<td>Test folders</td>
<td></td>
<td>0</td>
<td>0</td>
<td>100.00</td>
</tr>
<tr>
<td>Test cases</td>
<td></td>
<td>4</td>
<td>0</td>
<td>100.00</td>
</tr>
<tr>
<td>Test steps</td>
<td></td>
<td>13</td>
<td>0</td>
<td>100.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total run duration</td>
<td>1877ms</td>
</tr>
<tr>
<td>Total data received</td>
<td>7948</td>
</tr>
<tr>
<td>Average time per request</td>
<td>227ms</td>
</tr>
<tr>
<td>Total Assertions Failures</td>
<td>0</td>
</tr>
</tbody>
</table>

Requests

- Headers
  - Request Headers
  - Response Headers
- Request Methods
  - GET Request
  - POST request
Our journey with Postman: Split

488000 lines

1830 lines
500 lines
2100 lines
Less than 4000 lines
Our journey with Postman: Combine them
Our journey with Postman: BDD
Our journey with Postman: BDD

```javascript
var response = pm.response.json();

tests["Body contains headers"] = response.headers !== undefined;
tests["Header contains host"] = response.headers.host !== undefined;

var response = pm.response.json();

describe('A call to Postman echo service', () =>{
    describe('when calling get headers', () =>{
        it('should have headers property in body', () => {
            
        });
        it('should have host property in headers', () => {
            
        });
    });
});
```
Our journey with Postman: Replication

- Saved re-usable functions into Postman environment

510 lines on every request

5 lines on every collection
Our journey with Postman: Replication

- Use the collection structure at your advantage
Our journey with Postman: Feedbacks

- **80%** Reduction of file size (after splitting)
- **30%** Time reduction on develop/review per US
- **Easiness** Using BDD and re-used function
- **Better communication** Between developers, QAs and PO with BDD
Key takeaway

• Simple yet powerful tool
• Use small collections to share work
• Integration of Newman with Jenkins
• Avoid duplicate code by using environment
• Use Collection structure to your advantage
Thank you
Any questions?