

## CASOS DE PRUEBA

Serenity:

Borrar una publicación:

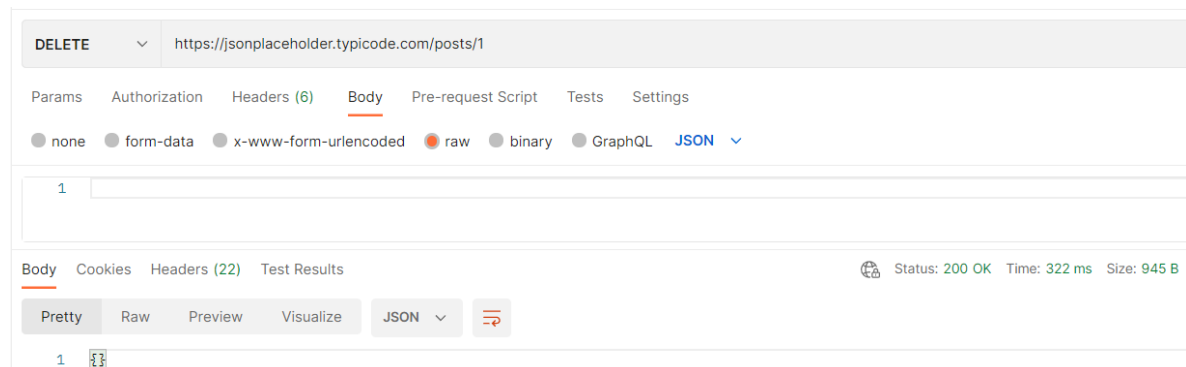
Feature:

```
Feature: Borrar una publicacion
  AS usuario de la pagina de JsonPlaceholder
  WANT borrar un post
  BECAUSE no ocupar espacio con informacion antigua

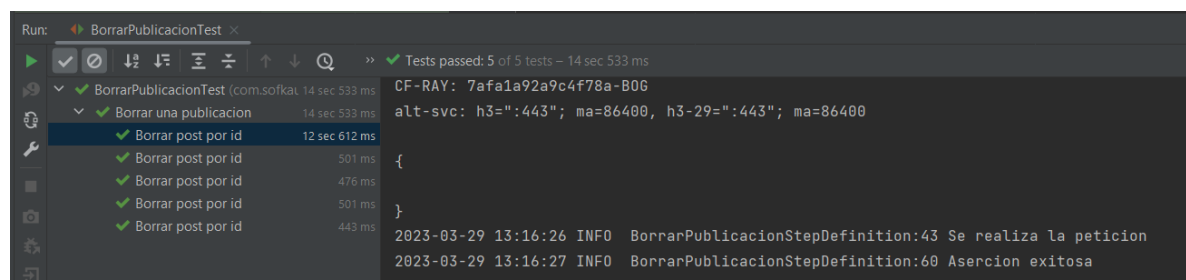
  @BorrarPosts
  Scenario Outline: Borrar post por id
    Given El usuario se encuentra en la web de JsonPlaceholder para borrar un post
    When El usuario envia una solicitud del post que se desea borrar con el <id>
    Then El usuario debe recibir un respuesta de status <code>

    Examples:
      | id      | code |
      | "5"     | 200  |
      | "6"     | 200  |
      | "10"    | 200  |
      | "abc"   | 200  |
      | "1512"  | 200  |
```

Prueba en postman:



Prueba con Serenity:



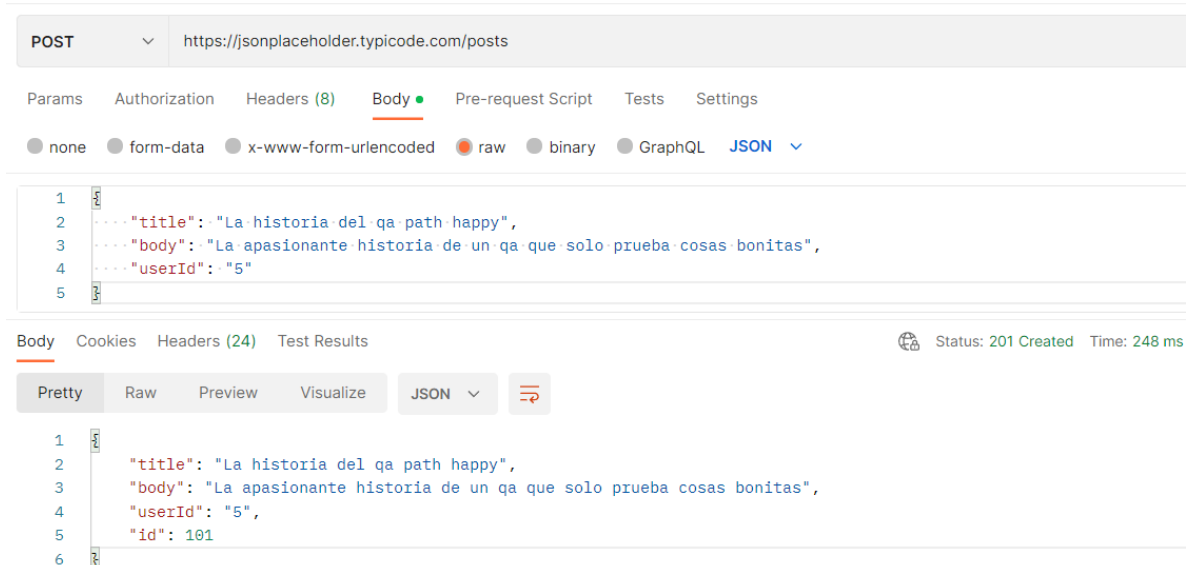
## Crear una publicación:

### Feature:

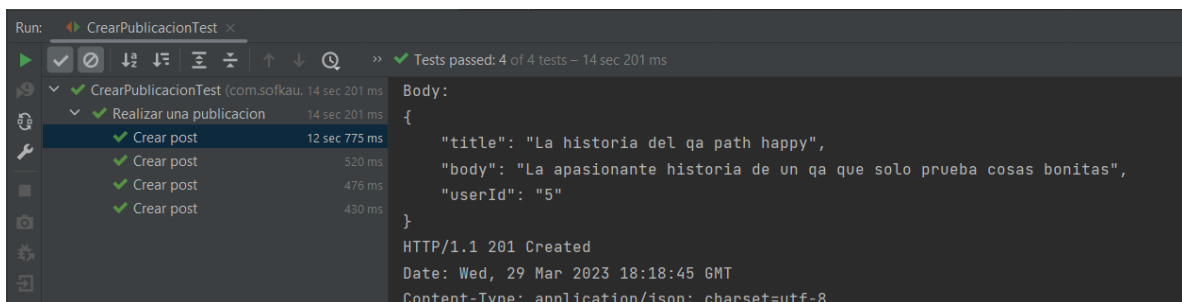
```
Feature: Realizar una publicacion
  AS usuario de la pagina de JsonPlaceholder
  WANT publicar un nuevo post
  BECAUSE crear nuevas publicaciones

  @CrearPosts
  Scenario Outline: Crear post
    Given El usuario se encuentra en la web de JsonPlaceholder para crear un post
    When El usuario envia una solicitud del post que se desea crear con <title> el <userid> y el <body>
    Then El usuario debe recibir un respuesta de status <code> y el post
    Examples:
      | title | body | userid | code |
      | "La historia del qa path happy" | "La apasionante historia de un qa que solo prueba cosas bonitas" | "5" | 201 |
      | "CUMPLE, O NO CUMPLE" | "Solo esa es la pregunta o blanco o negro" | "6" | 201 |
      | "TESTIGOS DE QA" | "Tiene un minuto de su vida para el ISTQB" | "5" | 201 |
      | "" | "" | "999" | 201 |
```

### Prueba con postman:

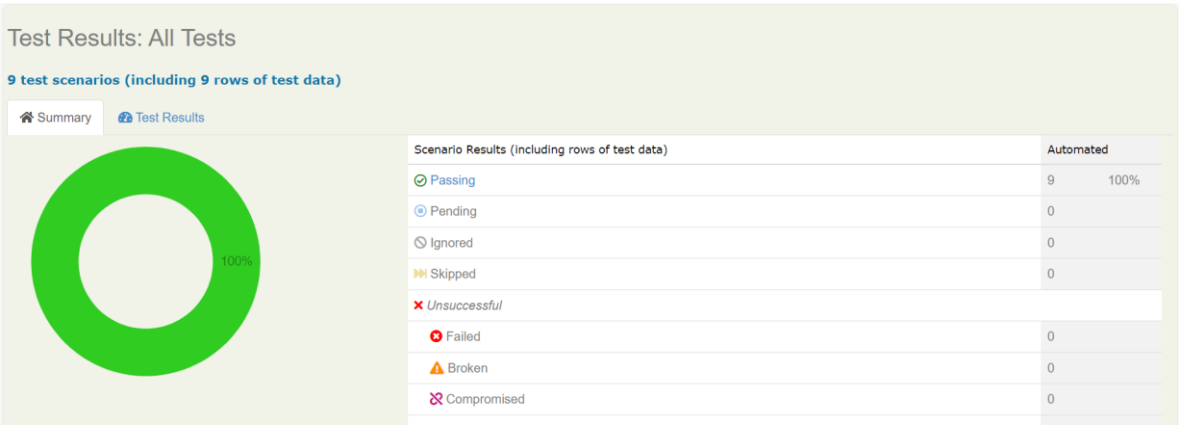


### Prueba con Serenity:



Reporte generado por Serenity:

Resumen:



Resultados de las pruebas:

9 test scenarios (including 9 rows of test data)

SummaryTest Results

Automated Tests

Show 10 entries

Filter

Feature	Scenario	Steps	Start Time	Duration	Result
Borrar una publicacion	Borrar post por id (5 examples)	3	13:16:15	13s 656ms	Passing
Realizar una publicacion	Crear post (4 examples)	3	13:18:35	12s 947ms	Passing

Resultados prueba Borrar post por id:

Scenario Outline

Given El usuario se encuentra en la web de JsonPlaceholder para borrar un post  
When El usuario envia una solicitud del post que se desea borrar con el <id>  
Then El usuario debe recibir un respuesta de status <code>

Examples:

#	Id	Code
1	"5"	200
2	"6"	200
3	"10"	200
4	"abc"	200
5	"1512"	200

Steps	Outcome	⌚
Example: {id="5", code=200}	SUCCESS	11.36s
Given El usuario se encuentra en la web de JsonPlaceholder para borrar un post	SUCCESS	0.17s
When El usuario envia una solicitud del post que se desea borrar con el "5"	SUCCESS	10.45s
Then El usuario debe recibir un respuesta de status 200	SUCCESS	0.52s
Example: {id="6", code=200}	SUCCESS	0.47s
Given El usuario se encuentra en la web de JsonPlaceholder para borrar un post	SUCCESS	0.02s
When El usuario envia una solicitud del post que se desea borrar con el "6"	SUCCESS	0.41s
Then El usuario debe recibir un respuesta de status 200	SUCCESS	0.02s

## Resultados prueba Crear Post:

Scenario Outline

Given El usuario se encuentra en la web de JsonPlaceholder para crear un post  
When El usuario envia una solicitud del post que se desea crear con <title> el <userid> y el <body>  
Then El usuario debe recibir un respuesta de status <code> y el post

Examples:

#	Title	Body	Userid	Code
1	"La historia del qa path happy"	"La apasionante historia de un qa que solo prueba cosas bonitas"	"5"	201
2	"CUMPLE, O NO CUMPLE"	"Solo esa es la pregunta o blanco o negro"	"6"	201
3	"TESTIGOS DE QA"	"Tiene un minuto de su vida para el ISTQB"	"5"	201
4	" "	" "	"999"	201

Steps	Outcome	⌚
<div><div><div>+</div><div>Example: {title="La historia del qa path happy", body="La apasionante historia de un qa que solo prueba cosas bonitas", userid="5", code=201}</div></div></div>	SUCCESS	10.99s
<div><div><div>✓</div><div>Given El usuario se encuentra en la web de JsonPlaceholder para crear un post</div></div></div>	SUCCESS	0.18s
<div><div><div>+</div><div>When El usuario envia una solicitud del post que se desea crear con "La historia del qa path happy" el "5" y el "La apasionante historia de un qa que solo prueba cosas bonitas"</div></div></div>	SUCCESS	10.23s
<div><div><div>+</div><div>Then El usuario debe recibir un respuesta de status 201 y el post</div></div></div>	SUCCESS	0.36s
<div><div><div>+</div><div>Example: {title="CUMPLE, O NO CUMPLE", body="Solo esa es la pregunta o blanco o negro", userid="6", code=201}</div></div></div>	SUCCESS	0.5s
<div><div><div>✓</div><div>Given El usuario se encuentra en la web de JsonPlaceholder para crear un post</div></div></div>	SUCCESS	0.01s
<div><div><div>+</div><div>When El usuario envia una solicitud del post que se desea crear con "CUMPLE, O NO CUMPLE" el "6" y el "Solo esa es la pregunta o blanco o negro"</div></div></div>	SUCCESS	0.44s
<div><div><div>+</div><div>Then El usuario debe recibir un respuesta de status 201 y el post</div></div></div>	SUCCESS	0.02s

## Karate Framework:

### Obtener Anime:

#### Feature:

```
Feature: Obtener anime
  AS usuario de la web de anime
  WANT buscar nombres y caracteristicas de animes
  BECAUSE ver mucho anime

Scenario Outline: Obtener anime por id
  Given url 'https://api.jikan.moe/v4/anime/' + <id>
  When method GET
  Then status 200
  And match response.data != null
  And match response.data.mal_id == parseInt(id)
  And match response.data.url != null
  Examples:
    | id |
    | 1  |
    | 5  |
    | 16 |
    | 20 |

Scenario: Obtener anime por id inexistente
  Given url 'https://api.jikan.moe/v4/anime/2'
  When method GET
  Then status 404
  And match response.message == "Resource does not exist"
```

## Pruebas con Postman:

GET

https://api.jikan.moe/v4/anime/5

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

BodyCookiesHeaders (14)Test Results

Status: 200 OKTime: 849 msSize: 4.78 KB

PrettyRawPreviewVisualizeJSON

```
1  {
2    "data": {
3      "mal_id": 5,
4      "url": "https://myanimelist.net/anime/5/Cowboy_Bebop__Tengoku_no_Tobira",
5      "images": {
6        "jpg": {
7          "image_url": "https://cdn.myanimelist.net/images/anime/1439/93480.jpg",
8          "small_image_url": "https://cdn.myanimelist.net/images/anime/1439/93480t.jpg",
9          "large_image_url": "https://cdn.myanimelist.net/images/anime/1439/93480l.jpg"
6        }
7      }
8    }
9  }
```

https://api.jikan.moe/v4/anime/2

GET

https://api.jikan.moe/v4/anime/2

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

BodyCookiesHeaders (12)Test Results

Status: 404 Not Found

PrettyRawPreviewVisualizeJSON

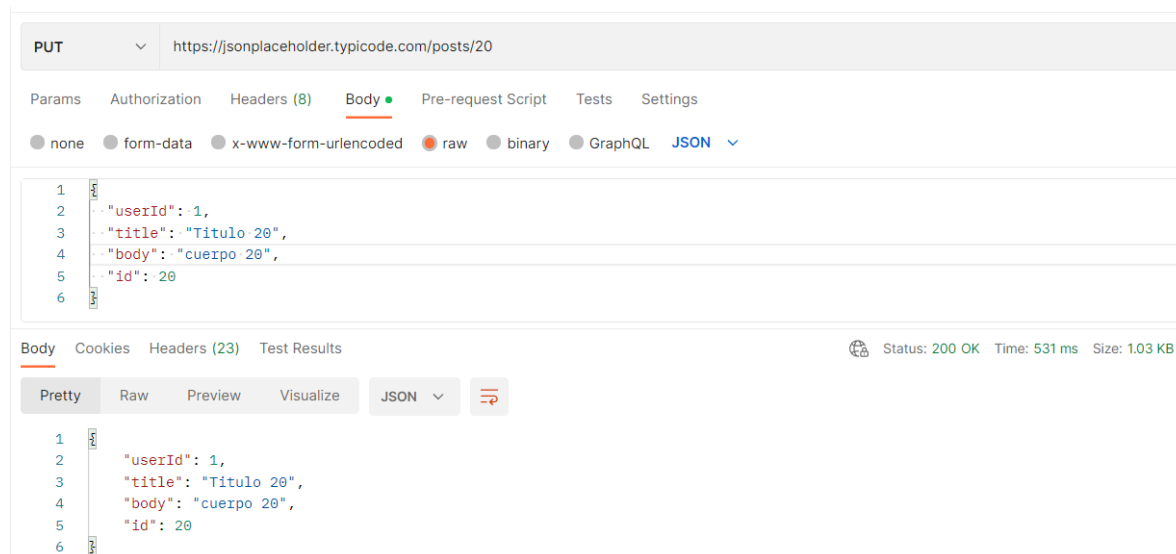
```
1  {
2    "status": 404,
3    "type": "BadResponseException",
4    "message": "Resource does not exist",
5    "error": "404 on https://myanimelist.net/anime/2/"
6  }
```

## Actualizar publicaciones:

### Feature:

```
1  >> Feature: Actualizar publicacion
2      AS usuario de la web de JSON PLACE HOLDER
3      WANT actualizar informacion en las publicaciones
4      BECAUSE quiero tener información actualizada
5
6  >> Scenario Outline: Actualizar publicacion por id
7      Given url 'https://jsonplaceholder.typicode.com/posts/' + <id>
8      And request { "userId": 1, "title": <title>, "body": <body>}
9      When method PUT
10     Then status 200
11     And match response.title == title
12     And match response.id == parseInt(id)
13     And match response.body == body
14
15     Examples:
16     | id | title      | body      |
17     | 1  | Titulo 1   | cuerpo 1   |
18     | 5  | Titulo 5   | cuerpo 5   |
19     | 16 | Titulo 16  | cuerpo 16  |
20     | 20 | Titulo 20  | cuerpo 20  |
```

### Pruebas con Postman:



## Pruebas con karate:

```
Run: TestKarate x
Tests passed: 9 of 9 tests - 13 sec 664 ms

Test Results
13 sec 664 ms


[THYMELEAF] * [* {src}] [1000]: com.intuit.karate.template.KaScriptAttrProcessor
[THYMELEAF] * Element Model Processors by [matching element and attribute name] [precedence]:
[THYMELEAF] * [{script} {ka:scope,data-ka-scope}] [1000]: com.intuit.karate.template.KaScriptElem
[THYMELEAF] * [* {ka:set,data-ka-set}] [1000]: com.intuit.karate.template.KaSetElemProcessor
[THYMELEAF] TEMPLATE ENGINE CONFIGURED OK
13:39:44.108 [Test worker] DEBUG org.thymeleaf.TemplateEngine - [THYMELEAF] TEMPLATE ENGINE INITIALIZED

HTML report: (paste into browser to view) | Karate version: 1.4.0.RC2
file:///C:/Users/nunez/Escritorio/Nueva%20carpeta/C1-2023-QA-Auto-Reportes-Karate/PruebasKarate/build/karate
=====

BUILD SUCCESSFUL in 18s
2 actionable tasks: 1 executed, 1 up-to-date
1:39:44 p. m.: Execution finished ':test --tests "runners.TestKarate"'.
```

## Reporte Generado con karate framework:

### Resumen:



Karate Labs

Features

2023-03-29 01:39:44 p. m.


2

0

Tags | Timeline

Feature	Title	Passed	Failed	Scenarios	Time (ms)
src/test/java/features/ActualizarPublicaciones.feature	Actualizar publicacion	4	0	4	2956
src/test/java/features/BuscarAnime.feature	Obtener anime	5	0	5	4598

### Resumen obtener anime por id:



Karate Labs

5

0

Scenarios

2023-03-29 01:39:43 p. m.

[1.1:15] Obtener anime por id

[1.2:16] Obtener anime por id

[1.3:17] Obtener anime por id

[1.4:18] Obtener anime por id

[2:20] Obtener anime por id inexistente

Summary | Tags | Feature: src/test/java/features/BuscarAnime.feature | Obtener anime AS usuario de la web de anime WANT buscar nombres y características de animés BECAUSE ver mucho anime

Scenario: [1.1:15] Obtener anime por idms: 1069

7Given url 'https://api.jikan.moe/v4/anime/' + 11

8When method GET1062

9Then status 2000

10And match response.data != null2

11And match response.data.mal\_id == parseInt(id)2

12And match response.data.url != null1

Scenario: [1.2:16] Obtener anime por idms: 851

7Given url 'https://api.jikan.moe/v4/anime/' + 51

8When method GET846

9Then status 2000

10And match response.data != null2

11And match response.data.mal\_id == parseInt(id)1

12And match response.data.url != null1

Scenario: [1.3:17] Obtener anime por idms: 827

7Given url 'https://api.jikan.moe/v4/anime/' + 162

8When method GET821

9Then status 2000


10And match response.data != null1

11And match response.data.mal\_id == parseInt(id)1

12And match response.data.url != null1



## Resumen actualizar publicaciones:

 Karate Labs Scenarios 2023-03-29 01:39:36 p. m.	4	0
Summary   Tags   Feature: <code>src/test/java/features/ActualizarPublicaciones.feature</code>   Actualizar publicacion AS usuario de la web de JSON PLACE HOLDER WANT actualizar informacion en las publicaciones BECAUSE quiero tener información actualizada		
Scenario: [1.1:16] Actualizar publicacion por id ms: 1855		
7 Given url 'https://jsonplaceholder.typicode.com/posts/' + 1		173
8 And request ( "userId": 1, "title": Titulo 1, "body": cuerpo 1)		105
9 When method PUT		1386
10 Then status 200		0
11 And match response.title == title		116
12 And match response.id == parseInt(id)		68
13 And match response.body == body		6
Scenario: [1.2:17] Actualizar publicacion por id ms: 355		
7 Given url 'https://jsonplaceholder.typicode.com/posts/' + 5		4
8 And request ( "userId": 1, "title": Titulo 5, "body": cuerpo 5)		1
9 When method PUT		339
10 Then status 200		0
11 And match response.title == title		2
12 And match response.id == parseInt(id)		7
13 And match response.body == body		2

## Conclusion:

Karate framework, es una herramienta que cualquiera puede usar incluso si no sabe programar, es muy bueno como su sintaxis es básicamente la misma que un gherkin, permitiendo realizar automatizaciones de manera rápida y sencilla.