

API Testing - Métodos GET Y POST

DigitalHouse >
Coding School



**Certified Tech
Developer**
The Ultimate Degree

Índice

1. [Método GET](#)
2. [Método POST](#)

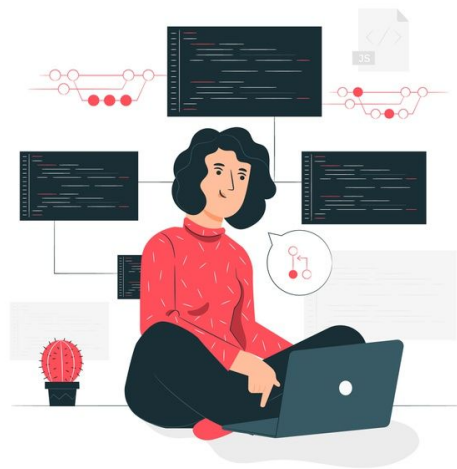
1 | Método GET

Testing método GET

Vamos a aprender a aplicar pruebas a los diferentes métodos HTTP.

Utilizaremos una solicitud GET para recuperar información de una URL específica y analizar la información obtenida a partir de los test.

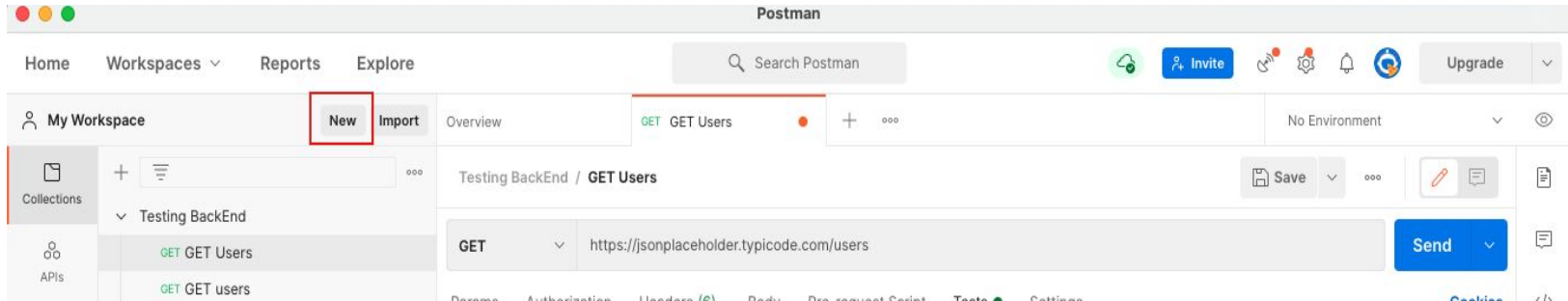
En el ejemplo analizaremos la obtención de usuarios desde un API externa.



Pasos

1

Primero, debemos crear una nueva solicitud en Postman. Para realizarlo, se debe hacer clic en la pestaña "New".



Pasos

2

El siguiente paso es crear la solicitud. ¿Cómo?

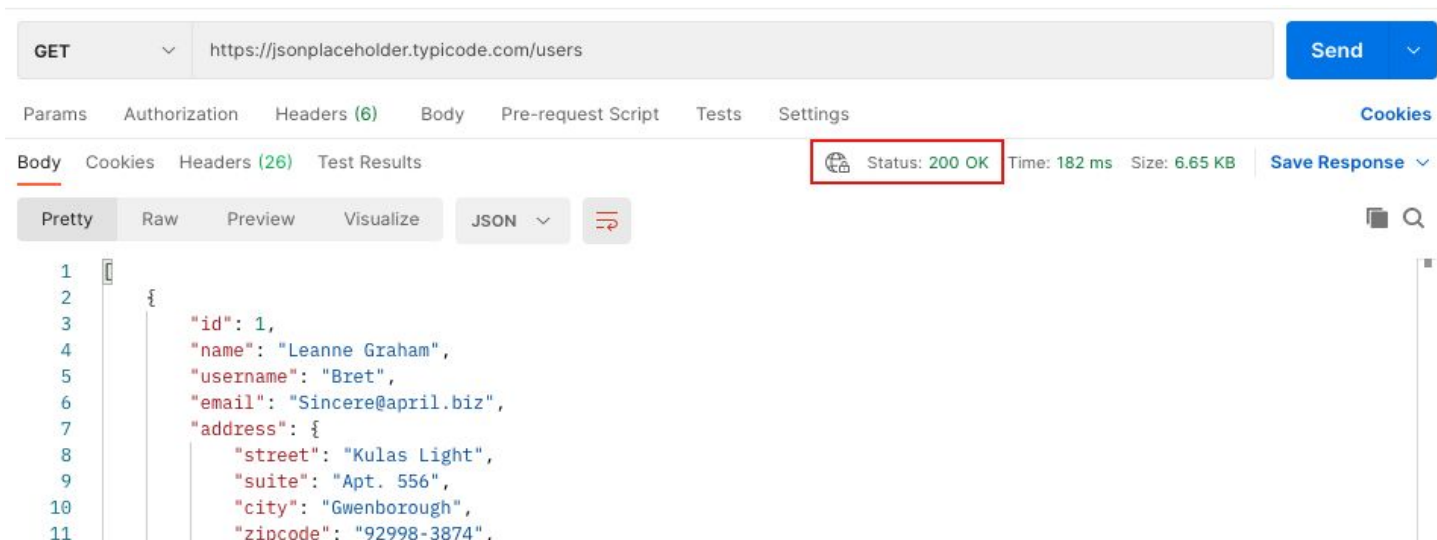
1. Configura su solicitud HTTP en GET.
2. Ingresa el enlace en la URL de la solicitud (<https://jsonplaceholder.typicode.com/users>)
3. Haz clic en ENVIAR para mandar la solicitud al servidor que aloja la URL.



Pasos

3

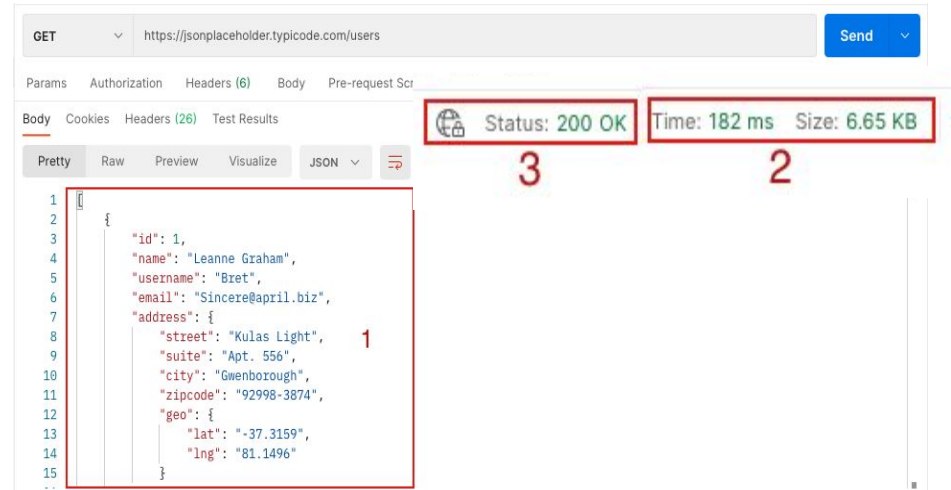
Cuando aparezca el mensaje 200 OK significa que se la solicitud se realizó correctamente.



Resultados

Podemos ver varios datos relacionados a la respuesta del servidor

1. **Response:** es la información en plano devuelta por el servidor. Con esto podemos dar una revisión temprana de los datos de la aplicación.
2. **Tiempo y tamaño de la respuesta:** con estos datos podemos ver si el sistema está cumpliendo uno de los requisitos no funcionales, tal como el rendimiento.

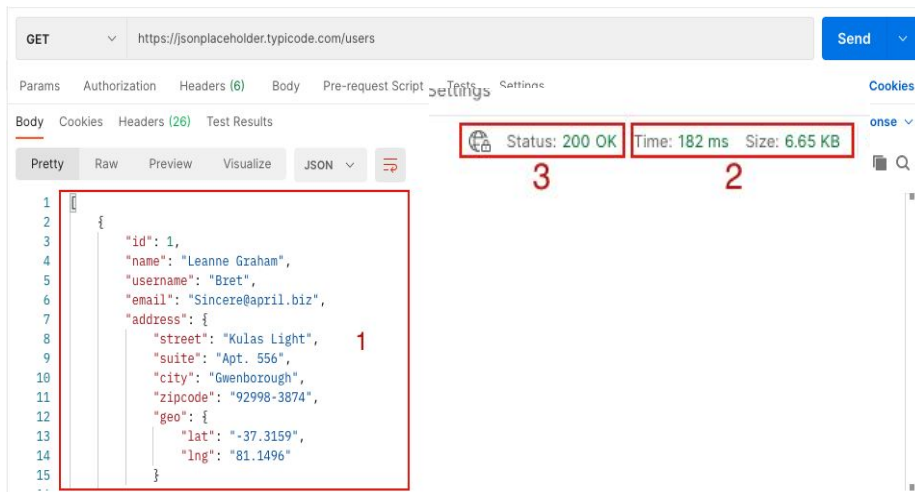


Resultados

3. Código de respuesta:

cuando solicitas información al servidor, este puede contestar distintos códigos de estado que te informan qué pasó con tu solicitud.

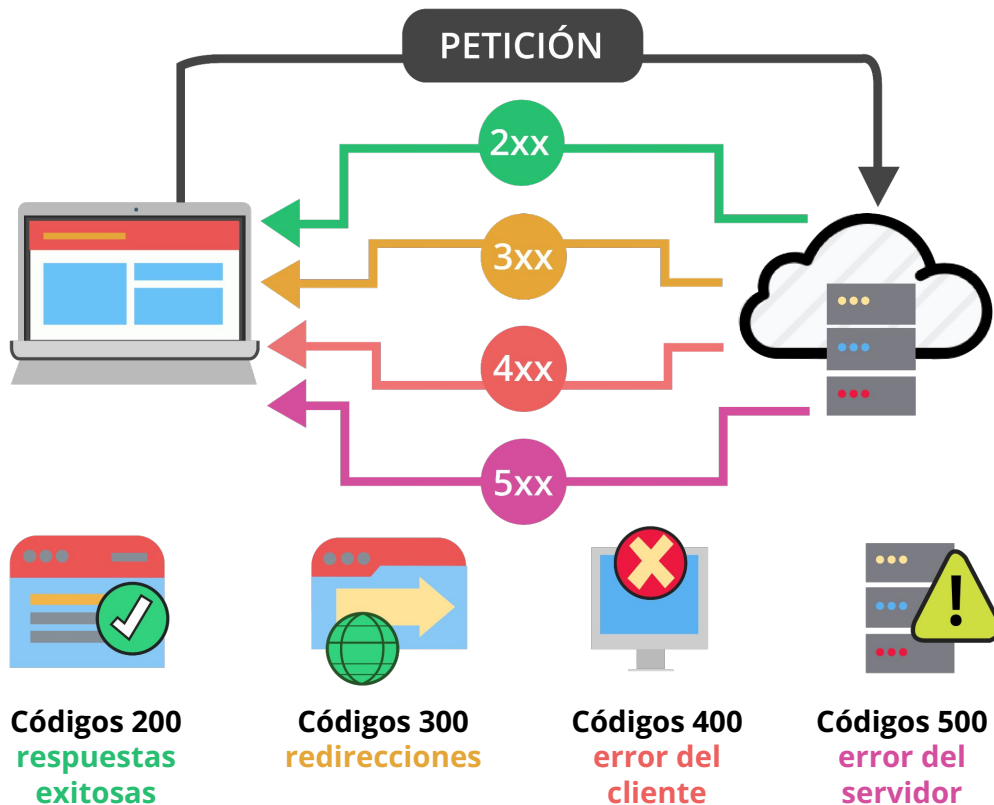
El código 200 nos indica que la solicitud se realizó con éxito.



Resultados

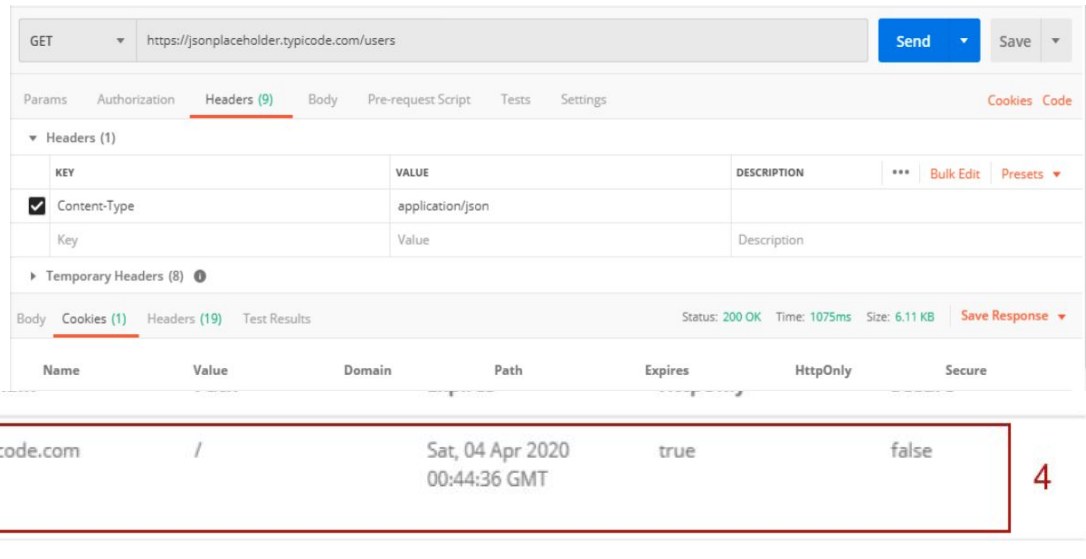
Un API nos devuelve diferentes códigos de respuesta que nos informan qué pasó con la petición. Estas respuestas se agrupan en cuatro clases.

En la imagen podemos ver las diferentes clases y su significado.



Resultados

4. **Cookies:** nos permiten ver la información relacionada con la sesión.



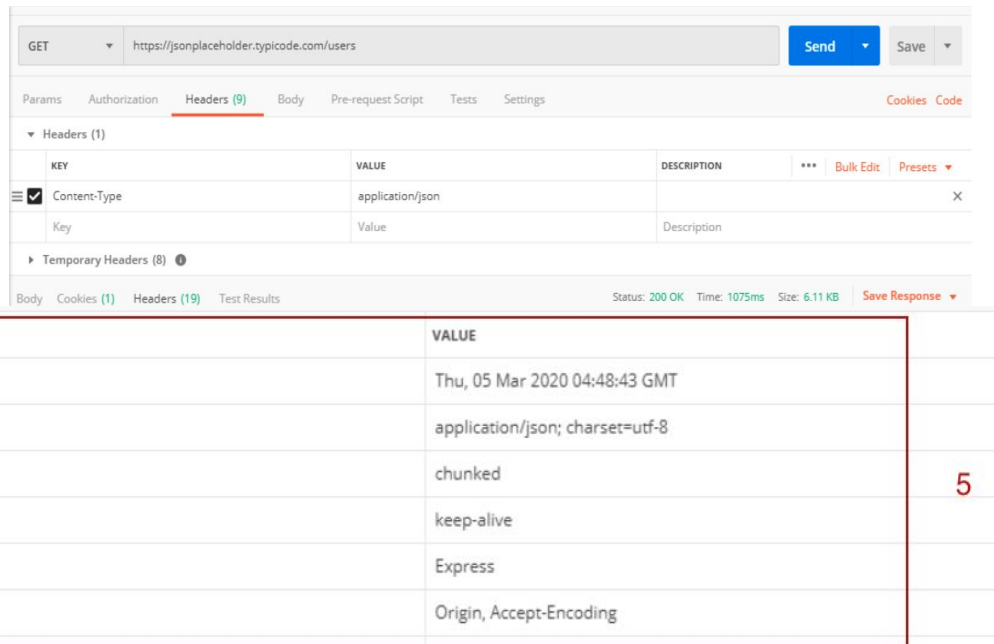
The screenshot shows a REST client interface with a GET request to `https://jsonplaceholder.typicode.com/users`. The 'Headers' tab is active, showing a single header: `Content-Type: application/json`. Below the headers, the 'Cookies' tab is selected, displaying a table of cookies. The table has columns for Name, Value, Domain, Path, Expires, HttpOnly, and Secure. A single cookie is listed: `_cfduid` with a long alphanumeric value, domain `typicode.com`, path `/`, expiration date `Sat, 04 Apr 2020 00:44:36 GMT`, and `HttpOnly: true`, `Secure: false`. The status bar at the bottom indicates a successful response: `Status: 200 OK`, `Time: 1075ms`, `Size: 6.11 KB`.

Name	Value	Domain	Path	Expires	HttpOnly	Secure
<code>_cfduid</code>	<code>d27e2f460f649a7000038d0196eaf1e161583369076</code>	<code>typicode.com</code>	<code>/</code>	<code>Sat, 04 Apr 2020 00:44:36 GMT</code>	<code>true</code>	<code>false</code>

4

Resultados

5. **Headers:** información sobre la solicitud procesada.



The screenshot displays a REST client interface with the following components:

- Request Bar:** Method: GET, URL: `https://jsonplaceholder.typicode.com/users`, Buttons: Send, Save.
- Navigation Tabs:** Params, Authorization, Headers (9), Body, Pre-request Script, Tests, Settings, Cookies, Code.
- Headers Tab:** Shows a table with 1 header:

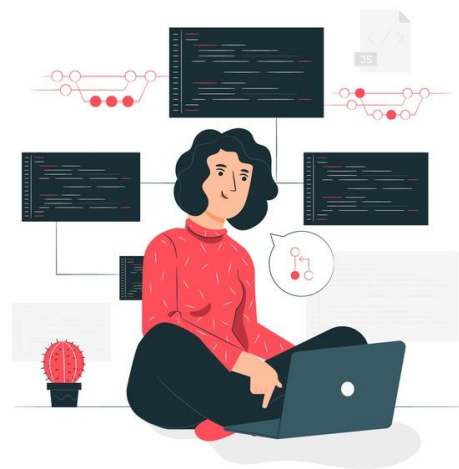
KEY	VALUE	DESCRIPTION
Content-Type	application/json	
- Temporary Headers:** Shows 8 temporary headers.
- Status Bar:** Status: 200 OK, Time: 1075ms, Size: 6.11 KB, Save Response.
- Response Headers Table:**

KEY	VALUE
Date	Thu, 05 Mar 2020 04:48:43 GMT
Content-Type	application/json; charset=utf-8
Transfer-Encoding	chunked
Connection	keep-alive
X-Powered-By	Express
Vary	Origin, Accept-Encoding

2 | Método POST

Testing método POST

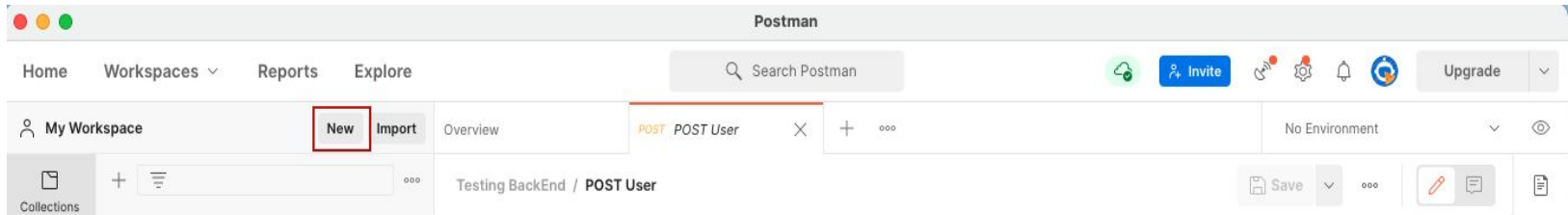
Cuando necesitamos agregar datos a nuestra aplicación utilizamos el método POST para enviar estos datos. A través de esta solicitud enviamos los datos y el API nos devuelve una respuesta que valida que la creación sea exitosa. En el ejemplo veremos la creación de un usuario y la respuesta del API.



Pasos

1

Al igual que con el método GET, se debe crear una nueva solicitud en Postman. Entonces, debemos hacer clic en la pestaña "New".

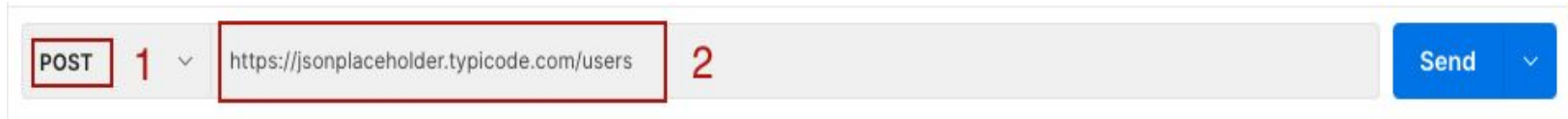


Pasos

2

El siguiente paso, es crear la solicitud. ¿Cómo?

1. Configurar su solicitud HTTP en **POST**.
2. Ingresar el enlace en la **URL** de la solicitud (<https://jsonplaceholder.typicode.com/users>).



A screenshot of an API client interface. On the left, a dropdown menu is set to 'POST' with a red box around it and a red number '1' next to it. To the right of the dropdown is a text input field containing the URL 'https://jsonplaceholder.typicode.com/users', which is also highlighted with a red box and a red number '2' to its right. On the far right, there is a blue 'Send' button with a small downward arrow.

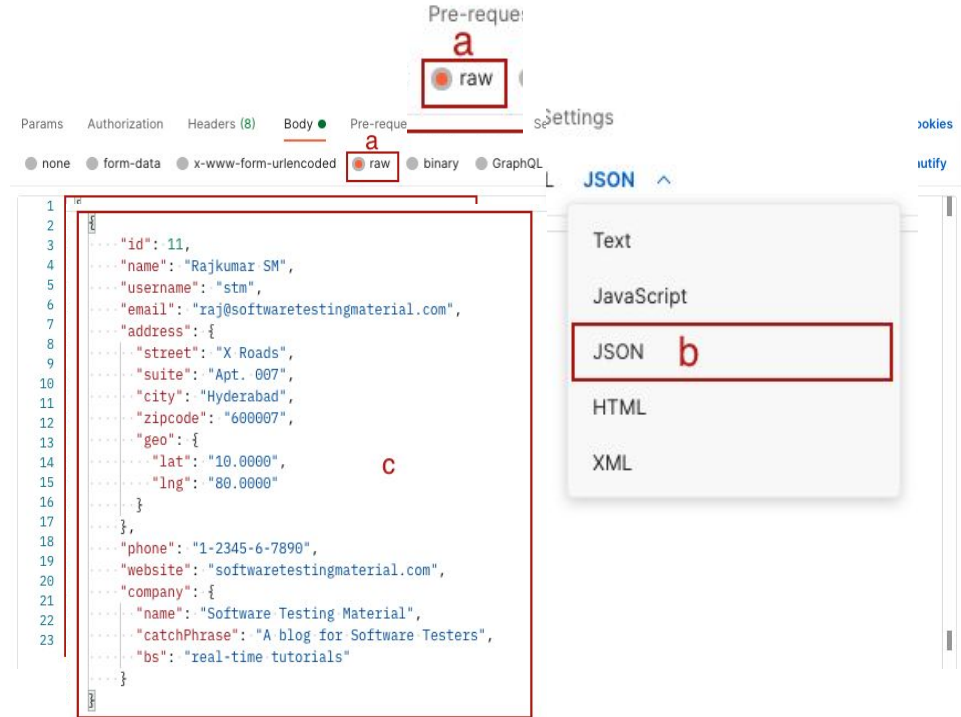
Pasos

3. Los datos para una petición POST no se los pasa por la url porque no viajan seguros: se pasan por el **BODY**. Lo podemos enviar de diferentes formas:
- Raw: se envía la información como una cadena tipo texto, a través de un archivo tipo JSON.
 - x-www-form-urlencoded: se envían los datos como si fuera un formulario.

```
{  
  "id": 11,  
  "name": "Usuario SM",  
  "username": "stm",  
  "email": "stm@digitalhouse.com",  
  "address": {  
    "street": "X Roads",  
    "suite": "Apt. 007",  
    "city": "Hyderabad",  
    "zipcode": "600007",  
    "geo": {  
      "lat": "10.0000",  
      "lng": "80.0000"  
    }  
  },  
  "phone": "1-2345-6-7890",  
  "website": "digitalhouse.com",  
  "company": {  
    "name": "Testing I",  
    "catchPhrase": "A blog for Software Testers",  
    "bs": "real-time tutorials"  
  }  
}
```

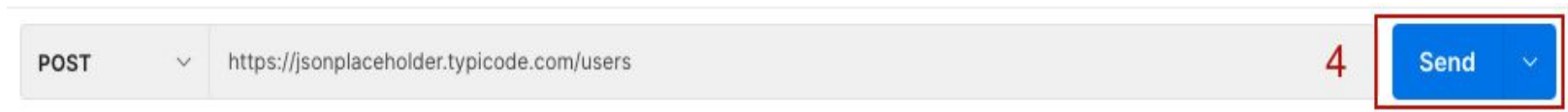
Pasos

- En este ejemplo enviaremos los datos en formato Raw. Para ello haz clic en el cuerpo de la solicitud y selecciona la opción "raw" (a), luego "Json" (b). Finalmente copia y pega el ejemplo brindado en la diapositiva anterior en el body (c).



Pasos

5. Haz clic en **ENVIAR** para mandar la solicitud al servidor que aloja la URL.

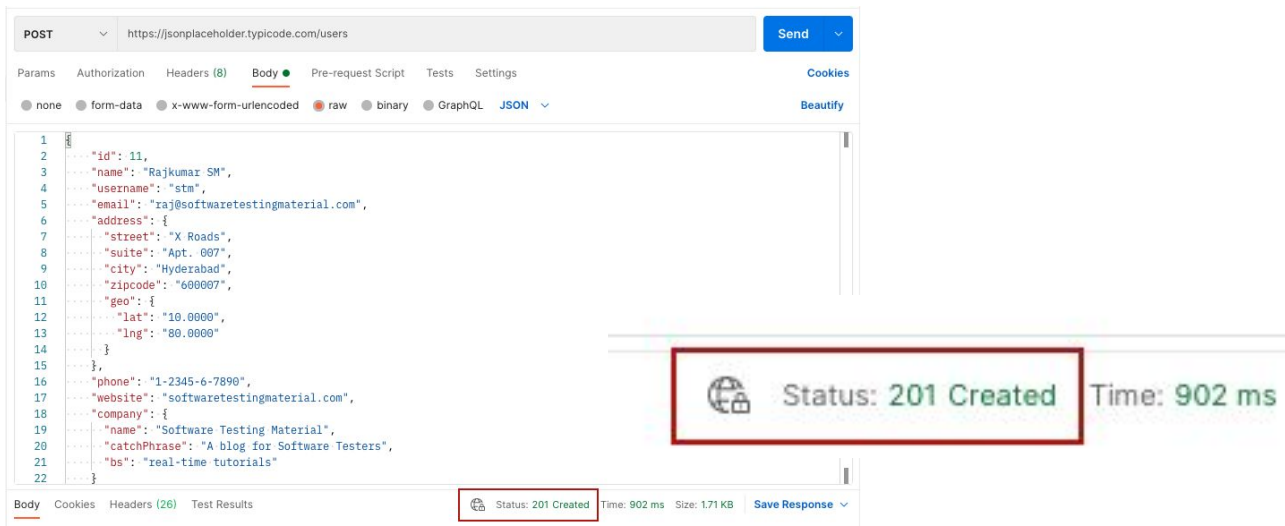


The image shows a horizontal bar representing an API client interface. On the left, the word "POST" is displayed next to a small downward arrow. In the center, the URL "https://jsonplaceholder.typicode.com/users" is entered. On the right side of the bar, there is a red number "4". To the right of the bar, there is a blue button with the text "Send" and a small downward arrow. This button is highlighted with a red rectangular border.

Pasos

3

Si aparece el mensaje **201 CREATED**, significa que se la solicitud se realizó correctamente.

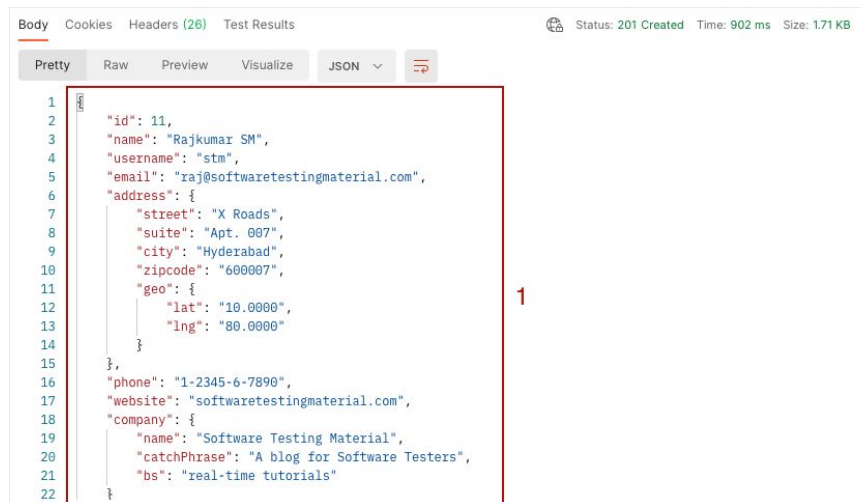


Resultados

Podemos ver varios datos relacionados a la respuesta del servidor

1. **Response:** es la información en plano devuelta por el servidor, la cual nos sirve para validar si la creación fue exitosa.

Generalmente este método devuelve los datos del usuario creado o un mensaje de creación exitosa.

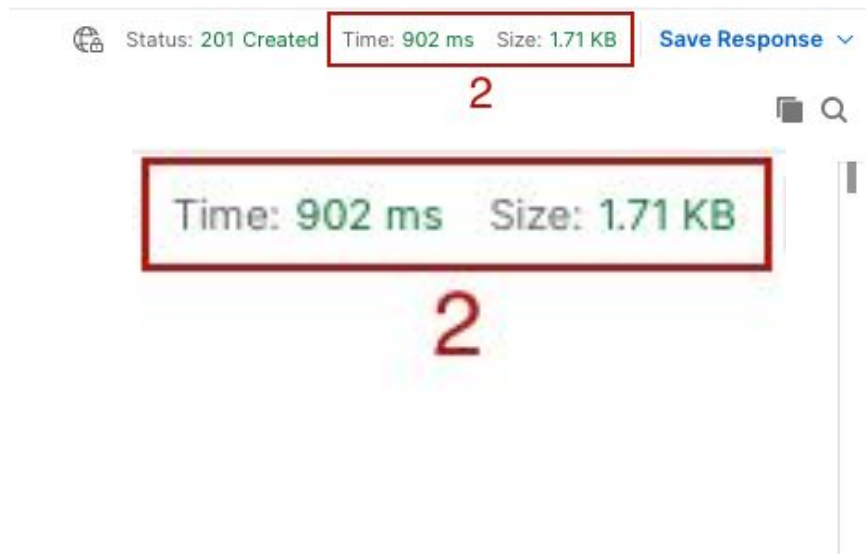


```
1  {
2    "id": 11,
3    "name": "Rajkumar SM",
4    "username": "stm",
5    "email": "raj@softwaretestingmaterial.com",
6    "address": {
7      "street": "X Roads",
8      "suite": "Apt. 007",
9      "city": "Hyderabad",
10     "zipcode": "600007",
11     "geo": {
12       "lat": "10.0000",
13       "lng": "80.0000"
14     }
15   },
16   "phone": "1-2345-6-7890",
17   "website": "softwaretestingmaterial.com",
18   "company": {
19     "name": "Software Testing Material",
20     "catchPhrase": "A blog for Software Testers",
21     "bs": "real-time tutorials"
22   }
23 }
```

Resultados

Podemos ver varios datos relacionados a la respuesta del servidor

2. **Tiempo y tamaño de la respuesta:** con estos datos podemos ver si el sistema está cumpliendo uno de los requisitos no funcionales, tal como el rendimiento.



Resultados

Podemos ver varios datos relacionados a la respuesta del servidor

3. **Código de respuesta:** cuando solicitas información al servidor, este puede contestar distintos códigos de estado que te informan qué pasó con tu solicitud. El código de respuesta 201 nos indica que la creación fue exitosa.



DigitalHouse>
Coding School