

POSTMAN

POSTMAN



Postman es una herramienta que nos permite diseñar, probar y documentar nuestra API Rest.



Tiene una versión desktop para Windows, Linux o Mac y una versión web pero que no cuenta con la totalidad de las funciones.



Se puede descargar en <https://www.postman.com/downloads/>. Desde las últimas versiones es necesario tener una cuenta e iniciar sesión para utilizar la mayoría de las funcionalidades.

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'My Workspace' at the top, followed by 'Collections', 'Environments', and 'History'. Below these are several project entries: 'Proyecto' (marked with a star and a red '2'), 'Angular', 'Avaria', 'Cultura Rave', 'New Collection', 'NodeJS', 'Tortuguita', and 'TPPS'. Under 'TPPS', there are four items: 'POST Login', 'GET Hello' (which is selected), 'GET Hello 2', and 'POST Post Hello'. There's also a 'POST Form' entry and a 'X' button. At the top center, there's a header with '1' (count of requests), 'New' (button), 'Import' (button), and a search bar with 'GET http://localhost:8080/postman'. To the right of the search bar is a green circle with a '+' sign and the number '1'. Further right are 'Save' and 'Share' buttons. The main area shows a request configuration panel with 'HTTP' and 'http://localhost:8080/postman'. A dropdown menu for 'Method' is open, showing options: GET (selected), POST, PUT, PATCH, DELETE, HEAD, and OPTIONS. The URL 'http://localhost:8080/postman' is entered in the 'URL' field. To the right of the URL is a blue 'Send' button with a yellow border. Below the URL field are tabs for 'Headers (6)', 'Body', 'Scripts', and 'Settings'. On the far right, there's a 'Cookies' section. At the bottom, there's a 'Test Results' section showing '200 OK', response time '127 ms', size '179 B', and a 'Save Response' button. Below the results are buttons for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'Text'.

HTTP Request

- Se puede crear y ejecutar rápidamente una request.
- Se debe especificar el método HTTP y el endpoint.

HTTP Request – GET – Query Params

The screenshot shows the Postman interface for a GET request. The URL is {{TPS}} /postman?firstName=Lourdes. The 'Params' tab is selected, showing a table with two rows: 'firstName' with value 'Lourdes' and 'lastName' with value 'Valli'. The 'Send' button is highlighted with a yellow box.

Key	Value	Description	Bulk Edit
firstName	Lourdes		
lastName	Valli	mi apellido	
Key	Value	Description	

- Para agregar query parameters a una request, se pueden escribir directamente en la URL o definir en la pestaña Params.
- Se puede agregar una descripción o deshabilitarlos para ciertas ejecuciones.
- Al ejecutar la request podemos especificarle a Postman que queremos descargar el resultado.

HTTP Request – POST – Body

The screenshot shows the Postman application interface. At the top, it says "HTTP TPPS / Post Hello". Below that, the method is set to "POST" (highlighted with a red box). The URL is "{{TPPS}}/postman". On the right, there are "Save" and "Share" buttons. Underneath the method, there are tabs for "Params", "Authorization" (with a green dot), "Headers (9)", "Body" (highlighted with a yellow box), "Scripts", and "Settings". The "Body" tab has several options: "none", "form-data", "x-www-form-urlencoded" (disabled), "raw" (selected and highlighted with a yellow box), "binary", "GraphQL" (disabled), and "JSON" (highlighted with a yellow box). In the "Body" section, there is a code editor containing the following JSON:

```
1 {  
2   "firstName": "Lourdes",  
3   "lastName": "Valli"  
4 }
```

A large blue callout bubble with a downward arrow points from the "Body" section towards the bottom right. Inside the bubble, the text reads: "También podemos ejecutar un POST y enviar un body en la sección Body."

At the bottom of the screen, there are tabs for "Body", "Cookies", "Headers (5)", and "Test Results". The "Body" tab is selected. Below these tabs are buttons for "Pretty", "Raw", "Preview", "Visualize", "Text" (with a dropdown arrow), and a copy icon. The "Raw" tab is selected. The "Text" dropdown shows the JSON content: "1 Hello, Lourdes Valli!". A second blue callout bubble with a downward arrow points from the "Text" area towards the bottom right. Inside, the text reads: "Se pueden enviar varios formatos pero en general utilizamos raw JSON para ejecutar llamadas API Rest."

HTTP Request – POST – Form

The screenshot shows the Postman application interface. At the top, it says "HTTP TPPS / Form". Below that, the method is set to "POST" and the URL is "{{TPPS}}/postman/form". There are tabs for "Params", "Authorization", "Headers (8)", "Body", "Scripts", and "Settings". The "Body" tab is selected, showing "form-data" is chosen. Under "Body", there is a table with three rows:

Key	Type	Value	Description	...	Bulk Edit
firstName	Text	Lourdes		...	
lastName	Text	Valli		...	
avatar	File	⚠ archivo.jpg		...	

Below the table, there are tabs for "Body", "Cookies", "Headers (5)", and "Test Results". The "Body" tab is selected, showing a JSON response:

```
1 {  
2   "firstName": "Lourdes",  
3   "lastName": "Valli",  
4   "avatar": {  
5     "contentType": "image/jpeg",  
6     "name": "avatar",  
7     "bytes": "/9j/4AAQSkZJRgABAQEAYABgAAD/2wBDAgGBgcGBQgHBwcJCQgKDBQNDAsLDBkSEw8UHROfHh0aHBwgJC4nICIsIxwcKDcpLDAxNDQ0Hyc5PTgyPC4zL  
8     "empty": false,  
9     "size": 24928,  
10    "inputStream"  
11  }  
}
```

- Algunas veces podemos querer testear un endpoint que recibe un formulario, Postman tiene una opción que nos permite enviar los campos en este formato.

My Workspace

New Import

GET http://localhost: ● | POST Form | POST Post Hello | Local | TPPS | POST Login | + | Local |

Collections

Environments

History

Collections

+ ⌂ ...

Avaria

users Carpeta users que pertenece a la colección Avaria

GET findAllUsers

GET findUserById

GET findUserServicesById

POST createUser

PUT updateUser

DEL removeUserById

DEL removeAllUsers

POST login

POST token

GET findUserEventsById

POST forgotPassword

> services

> events

> Cultura Rave

TPPS

Overview Authorization Scripts Variables Runs

These variables are specific to this collection and its requests. Learn more about [collection variables](#)

Filter variables

	Variable	Initial value	Current value	...
<input checked="" type="checkbox"/>	collectionToken		ey123456	
<input checked="" type="checkbox"/>	otherVar	unValor	unValor	
	Add new variable			

Save Run Fork 0 Share ...

Carpeta users que pertenece a la colección Avaria

Collections

- La utilización de colecciones es útil cuando tenemos varios endpoints que probar.
- Podemos guardar, nombrar y agrupar las distintas requests.
- En una colección se pueden definir variables, métodos de autenticación y scripts.

My Workspace

New Import

GET http://localhost: ● | POST Form | POST Post Hello | Local | HTTPS | POST Login | + | Local

Collections

Globals

Environments

History

Local

Filter variables

	Variable	Type	Initial value	Current value	...
<input checked="" type="checkbox"/>	URL	default	http://localhost:5000	http://localhost:5000	
<input checked="" type="checkbox"/>	AVARIA_URL	default	http://localhost:8080/avaria	http://localhost:8080/avaria	
<input checked="" type="checkbox"/>	CR_token	secret		
<input checked="" type="checkbox"/>	CR_URL	default	http://localhost:8080	http://localhost:8080	
<input checked="" type="checkbox"/>	accessToken	secret		
<input checked="" type="checkbox"/>	TPPS	default	http://localhost:8080	http://localhost:8080	
Add new variable					

Save Fork 0 Share

...

The screenshot shows a user interface for managing environments and variables. On the left, there's a sidebar with tabs for Collections, Globals, and Environments (which is currently selected). The main area displays a table of variables with columns for checked status, variable name, type (either default or secret), initial value, and current value. The 'accessToken' variable is highlighted with a yellow box around its 'secret' type entry. The 'Variables' column contains checkboxes for each row.

Environments

- Se pueden definir environments para guardar variables que varían por ambiente, por ejemplo cuando queremos ejecutar request localmente o contra un servidor externo.
- Las variables se pueden definir como secret para que nos oculte los valores.

POST

{{TPPS}} /postman

Send

Params

Authorization

Headers (9)

Body

Scripts

Settings

Cookies

Auth Type

Bearer Token

The authorization header will be automatically generated when you send the request. Learn more about [Bearer Token](#) authorization.

ⓘ Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#).

Token

{{accessToken}}

Body Cookies Headers (5) Test Results

200 OK

• 129 ms

• 185 B

• Save Response

Authentication – Bearer Token

- Podemos cargar un método de autenticación.
- Postman nos ofrece muchas opciones pero una muy utilizada es el Bearer token.
- Para utilizar una variable definida en el environment o en la colección actual, escribimos el nombre entre {{ }}.

GET

{{TTPS}} /postman/Lour

Send

Params

Authorization

Headers (8)

Body Scripts

Cookies

Auth Type

Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about [Basic Auth](#) authorization.

ⓘ Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#).

Username

lourdes

Password

password



Authentication – Basic Auth

- Otro método que también puede ser bastante común es la autenticación básica por medio de usuario y contraseña.

POST

{{TTPS}} /postman/login

Send

Params Authorization Headers (7) Body Scripts ● Settings

Cookies

Pre-request

```
1 var jsonData = pm.response.json();
2 pm.environment.set("accessToken", jsonData.token);
3 pm.collectionVariables.set("collectionToken", jsonData.token);
4 pm.globals.set("globalToken", jsonData.token);
```

Post-response



Body Cookies Headers (5) Test Results

200 OK

• 123 ms

• 184 B



e.g. Save Response

...

Pretty Raw Preview Visualize

JSON

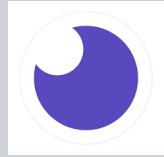


```
1 {
2   "token": "ey123456"
3 }
```

Scripts

- Otra funcionalidad muy útil son los scripts. Nos permiten ejecutar código javascript antes o después de la ejecución de una request.
- En general utilizamos el lenguaje de consulta JsonPath para acceder por ejemplo a valores dentro de la respuesta y utilizarlos más tarde.
- De esta forma podemos obtener un token de una respuesta y guardarlo en una variable.
- Se utiliza **pm.environment**, **pm.collectionVariables** y **pm.globals** para acceder a las variables definidas en un ambiente, en la colección actual o globales respectivamente.

Alternativas



Insomnia: <https://insomnia.rest/>



Bruno: <https://www.usebruno.com/>

¿Preguntas?