

























 Quiz #2 Introducing Postman









Your First API Request

-  Task: Create a workspace
-  Task: Create a collection
-  Task: Get books from the Library API
-  Request-Response pattern
-  Quiz #3: Your First API Request









Request Parameters

-  Variables in Postman
-  Query parameters
-  Task: Search books by genre
-  Task: Multiple query parameters
-  Path Variable
-  Task: Get a book by id
-  Quiz #4: Query vs. path parameters

Sending data with POST

-  Task: Add a book
-  Task: Add an authorization header
-  Task: Use Postman Auth instead!
-  Quiz #5: Sending Data With Postman

Introduction to variables and scripting

-  Variables in Postman (Continued)
-  Setting variables programmatically
-  Task: Your first script
-  Task: Grab the new book id

Task: Get a book by id

Someone keeps visiting the library daily, asking whether "Ficciones" by Jorge Luis Borges is available.

When you fetched all the books in the library, you may have noticed that each book has a unique **id** value. This **id** can always be used to identify the book, even if its other properties are changed.

Since this person keeps asking about "Ficciones", you've jotted down that the unique **id** of this book is **29cd820f-82f9-4b45-a7f4-0924111b5b89**

(Don't believe us? You can always search for "Ficciones" with the **search** query parameter: **GET /books?search=ficciones**)

Get a book by **id**

According to the [API documentation](#), we can get a specific book by hitting the path **GET /books/:id**, where we replace **:id** with the book's id.

1. Hover on your Postman Library API v2 Collection, click the three dots icon and select **Add request**. Name your new request **"get book by id"**.

2. Make sure the request method is set to **GET**, and paste in this endpoint as the **request URL**: **{{baseUr1}}/books/:id**

Postman automatically adds a "Path Variables" editor in the Params tab of the request for any path variables in the request URL prefixed with a colon **:**

3. In the **Params** tab of the request, paste the **id** for "Ficciones" (**29cd820f-82f9-4b45-a7f4-**

HTTPPostman Library API v2 / **get books by id**

GET{{baseUr1}}/books/:id

ParamsAuthorizationHeaders (5)BodyPre-request ScriptTestsSettings

Query Params

	Key	Value
	Key	Value

Path Variables

	Key	Value
	id	29cd820f-82f9-4b45-a7f4-0924111b5b89

4. **Save and Send** your request

You should get a **200 OK** response with a single JSON object that represents the "Ficciones" book. At the time of this example, the book is not checked out:

BodyCookiesHeaders (10)Test Results

PrettyRawPreviewVisualizeJSON

```
1 {
2   "id": "29cd820f-82f9-4b45-a7f4-0924111b5b89"
3   "title": "Ficciones",
4   "author": "Jorge Luis Borges",
5   "genre": "fiction",
6   "yearPublished": 1944,
7   "checkedOut": false,
8   "isPermanentCollection": true,
9   "createdAt": "2022-03-30T00:54:52.606Z"
10 }
```

Debugging requests in the Postman Console

You used Postman's path variable helper in the **Params** tab of the request to add a path variable nicknamed **:id** to the request URL in a human-friendly way. Postman replaces **:id** with the value you specify for **id** in the Path Variables editor.

You can always view the raw request sent to the API by opening the **Postman Console** in the lower left of Postman. All requests you make and their responses are logged in the Postman Console. Scroll to the bottom to expand the most recent request.

EnvironmentsHistoryCollectionsRecent

Postman Library API v2

GET get booksGET get fiction booksGET get book by idPOST add bookPATCH checkout bookDEL delete bookPOST skill check

GET {{baseUr1}}/books/:id

ParamsAuthorizationHeaders (6)Body

Query Params

	Key	Value
	Key	Value

BodyCookiesHeaders (10)Test Results

PrettyRawPreviewVisualizeJ

```
1 {
2   "message": "Book with id 'd7547"
3 }
```

OnlineConsole

GET https://library-api.postmanlabs.com/booksGET https://library-api.postmanlabs.com/books?genze=fiction&checkedOut=falseGET https://library-api.postmanlabs.com/books/d7547af6-9666-4ecc-8f7a-2c7b81c023aeGET https://library-api.postmanlabs.com/books/d7547af6-9666-4ecc-8f7a-2c7b81c023ae

You can see that Postman has inserted the book **id** as a path parameter in place of the **:id** placeholder when making the request. Cool!

If you run into any errors when making API calls, always check the Postman Console and ensure the raw request was sent as expected. *A common error is adding accidental white space in your query or path parameter values.*

Great job!

You now know how to refine your requests with **query** and **path variables**, and how to debug your requests using the **Postman Console**. Test your knowledge in a quick quiz before moving on.