

Postman Fundamentals

SETTING THE STAGE







API



Application



Roman

Application
developer

Anna

API
developer





Testing and documenting API calls



Mocking data and application set up

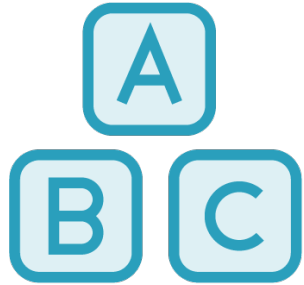
Postman **simplifies** working with APIs.

Getting Started: Postman

Getting Started: API



Goal: Increased efficiency with an API



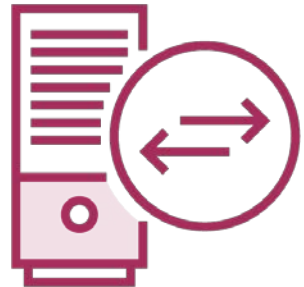
Postman basics



Testing



Collections



Mock server



Documentation



Team features

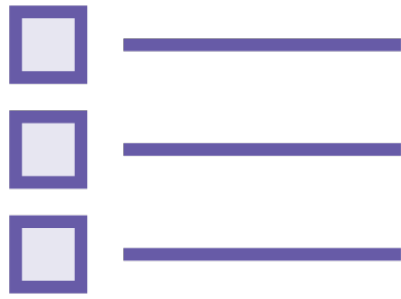
Let's get started!

Postman Basics





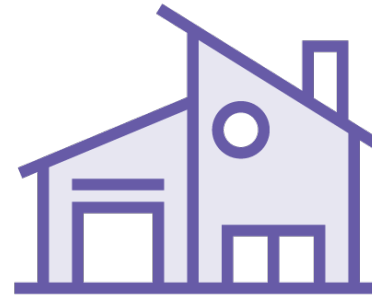
Search



Wishlist



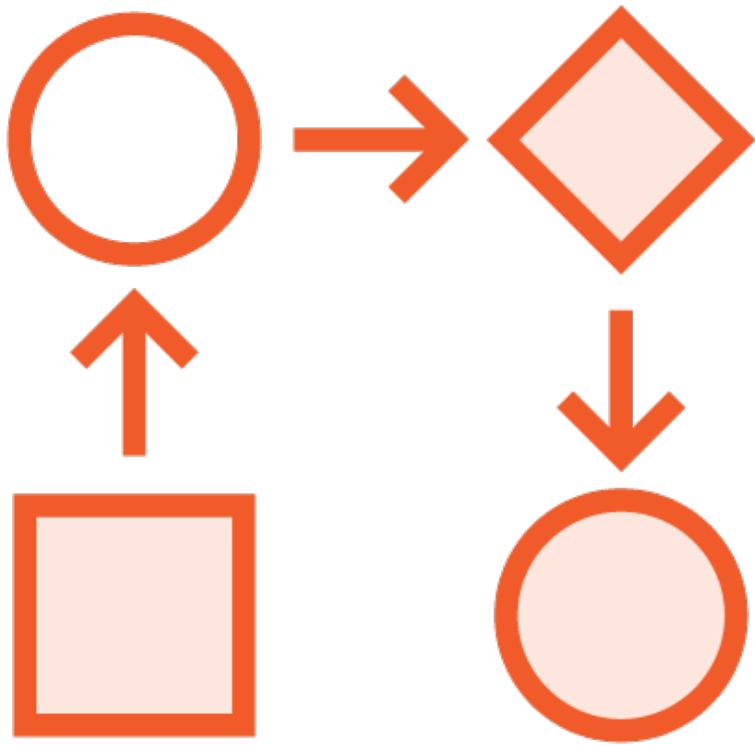
Inventory



Household



Admin



Dependent features



Basic API requests

GETting Books

Adding a New Item to Inventory

History

Authorization

Preset Headers

Possible Headers

Username

Password

Business Unit

Client ID

Existing Headers



G-Token

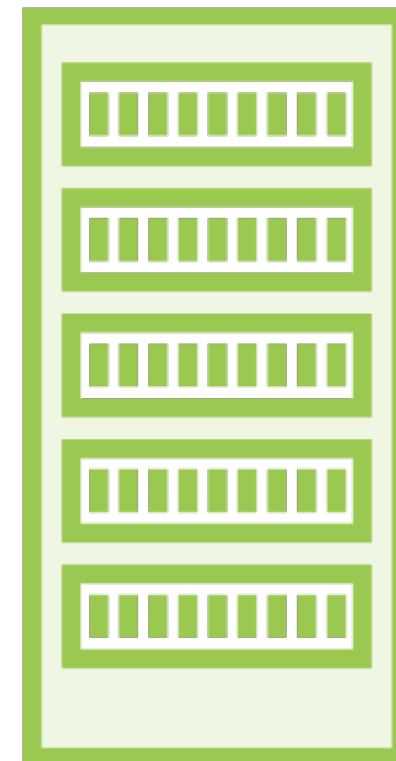


Authorization

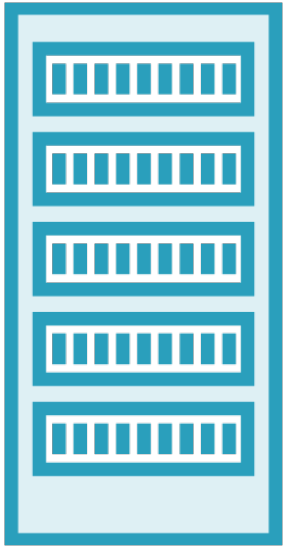


Headers

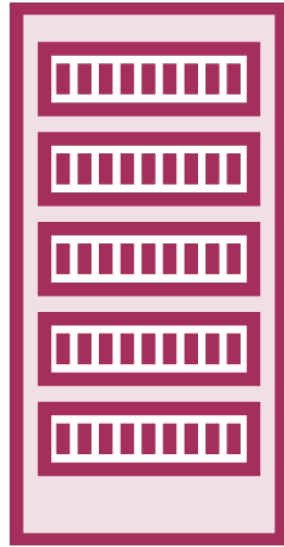
Content-Type: application/json
G-TOKEN: ROM831ESV



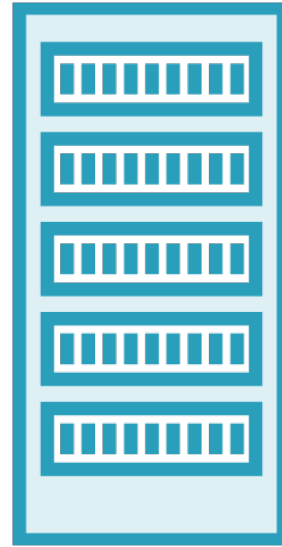
Environments



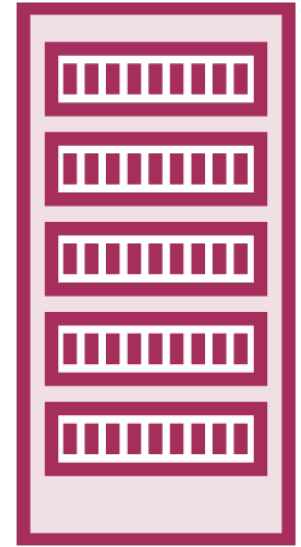
Local



Develop



Test

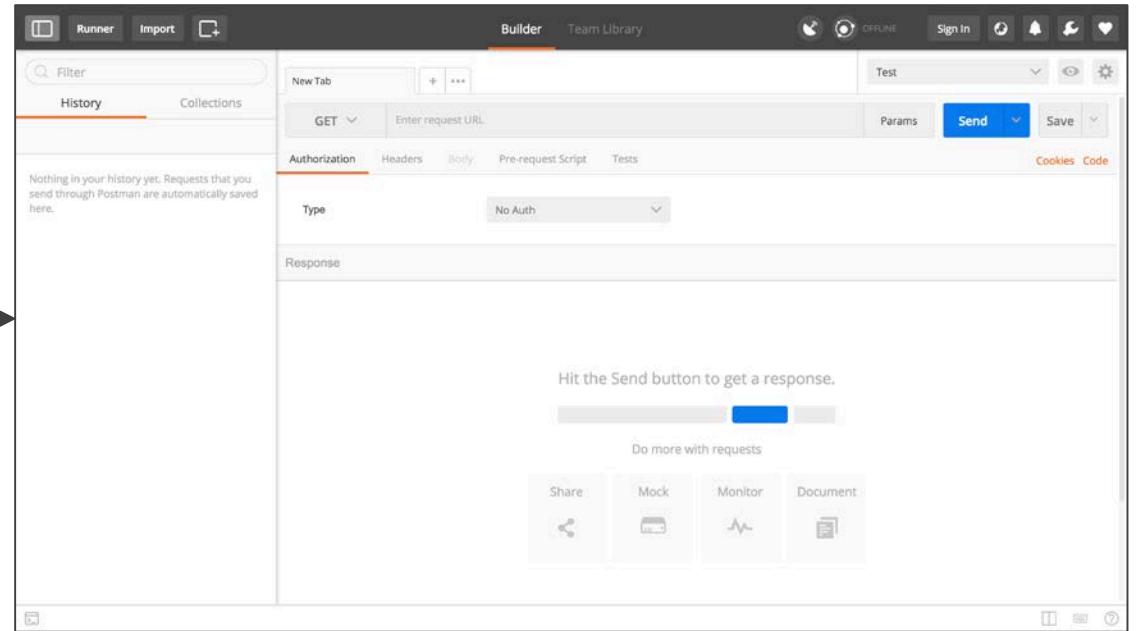


Production

Import Utility



Import cURL





Update project based on Postman tests

Generating Code

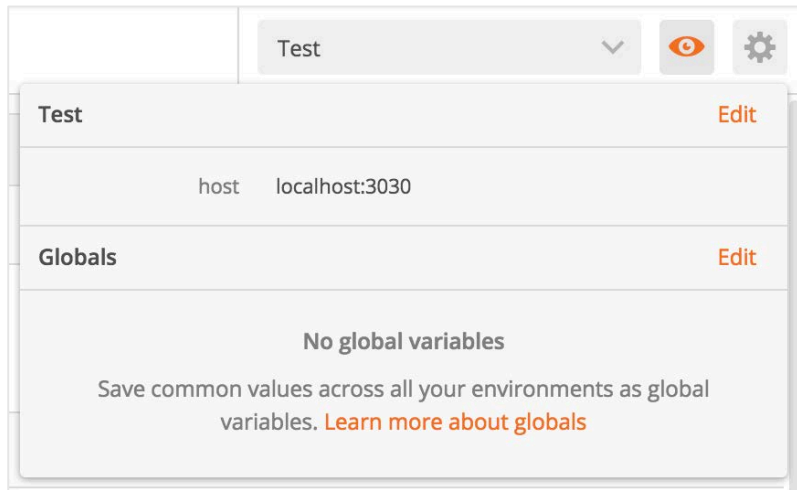
Sync



GET ▾	http://localhost:3000/books/5	Params	Send ▾
POST ▾	localhost:3000/books	Params	Send ▾
DELETE ▾	localhost:3000/books/6	Params	Send ▾

Track Requests

History		Collections
		Clear all
▼ Today		
GET	http://taylonr.com/wp-content/uploads/2016/06/cropped-IMG_20141212_180223-3.jpg	
GET	http://fonts.googleapis.com/css?family=Roboto:400,300,300italic,400italic,500,500italic,100,100italic	
GET	http://taylonr.com/	
POST	http://localhost:3000/books	
GET	http://{{host}}/books	
GET	localhost:3030/books	
GET	localhost:3000/books	
GET	localhost:3000/books	
DEL	localhost:3000/books/6	
DEL	localhost:3000/books/6	
POST	localhost:3000/books	
GET	http://localhost:3000/books/5	
POST	localhost:3000/books	
POST	localhost:3000/books	



Set Up Environments

Testing Requests



Welcome to the API Team



Released an authorization bug



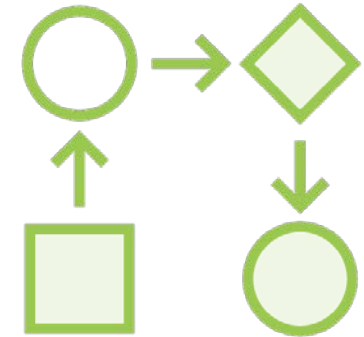
Solution?



Request Cycle



Library of Tests



Integrated Workflow



Basics of testing in Postman

Pre-Built Tests

Test Syntax

```
pm.test("Status code is 200", function () {  
    pm.response.to.have.status(200);  
});
```

```
pm.test("Status code is 200", function () {  
    pm.response.to.have.status(200);  
});
```

```
pm.test("Status code is 200", function () {  
    pm.response.to.have.status(200);  
});
```

```
pm.test("Status code is 200", function () {  
    pm.response.to.have.status(200);  
});
```

```
pm.test("Status code is 200", function () {  
    pm.response.to.have.status(200);  
});
```

```
pm.test("Status code is 200", function () {  
    pm.response.to.have.status(200);  
});
```

info

globals

environment

assertions

<https://www.getpostman.com/docs/>

Basic Tests

Using Other Libraries

Available Libraries



atob

btoa

chai

cheerio

crypto-js

csv-parse

postman-collection

tv4

uuid

xml2js

NodeJS Modules



path

assert

buffer

util

url

punycode

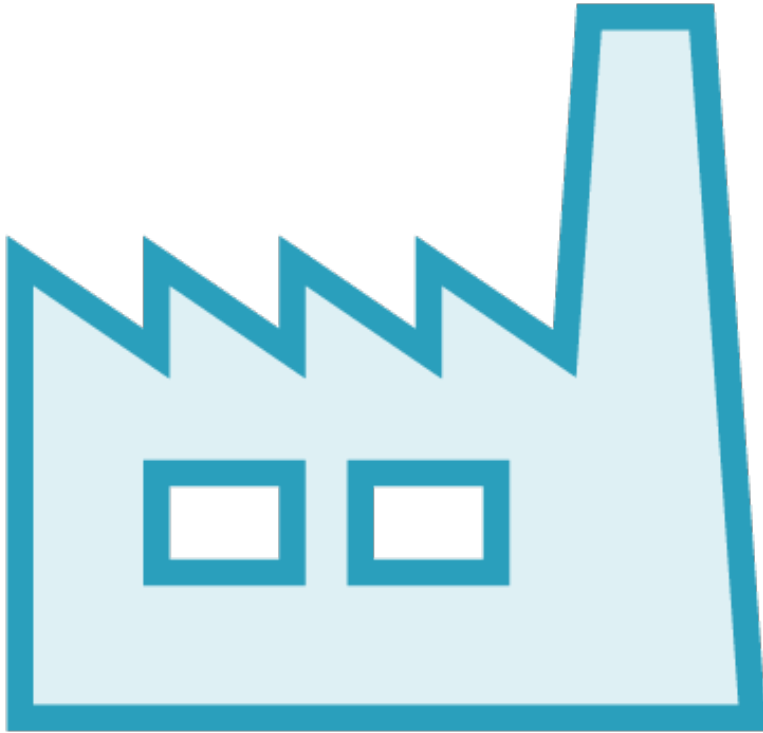
querystring

string_decoder

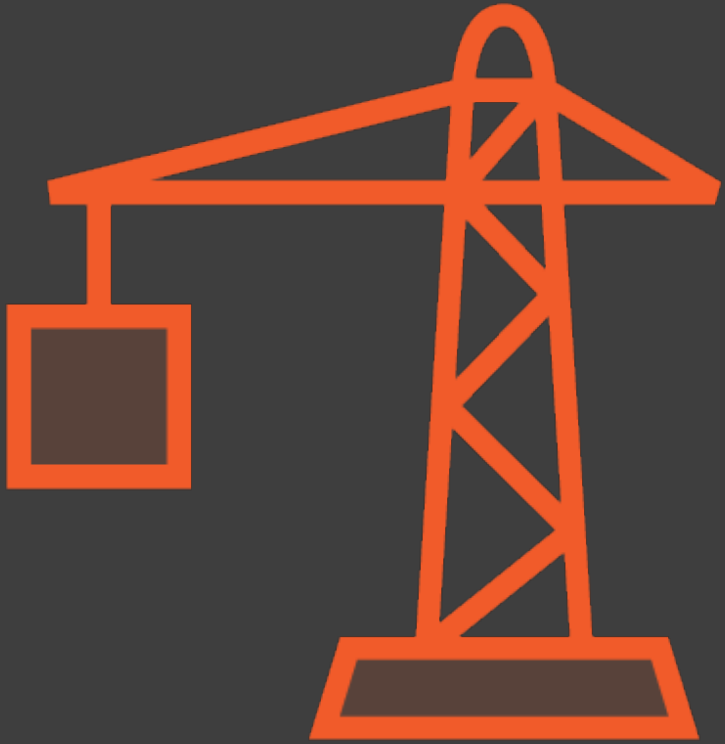
stream

timers

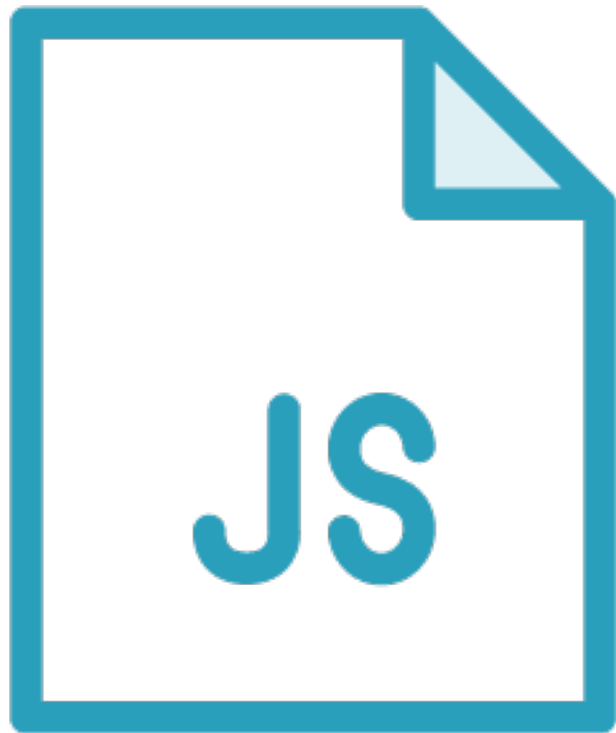
events



Pre-Built Tests



Custom Built Tests



Import Libraries



Well tested API



Maximize power of tests

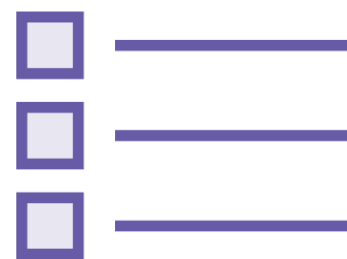
Collections



Using Tests on the Front End



Household



Wish list

Feature Requirements



Books



Users



Household



Wish list



Time consuming to set up



Postman can help



Automate API requests

Creating Collections

Collection Runner

Using Variables

Pre-request Scripts

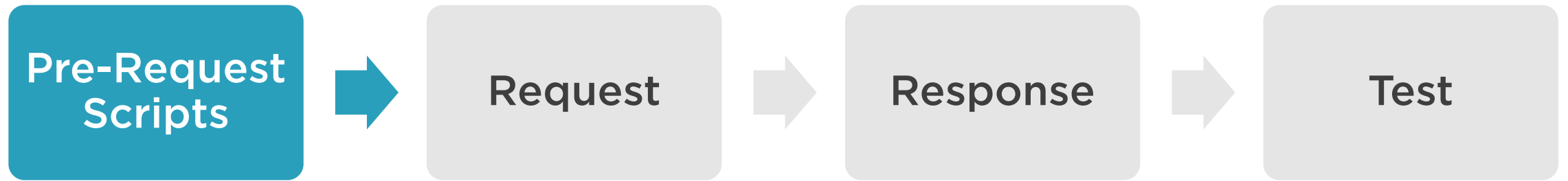
Request



Response



Test



Data Files



Data driven tests

Initializing Test Data

Basic Workflow

Set up variables

Create users

Add books

Clean up

Refactor: Loop Over Users

Refactor: Loop Over Wishlists

Scenario Tear Down

Running Your Refactored Collection



Quick application set up

Executing Tests

Testing from the Command Line

Loosely couple your services. Use Orgs to version and reuse your code. [Create a free org »](#)

★ newman public



Supercharge your API workflow
Modern software is built on APIs. Postman helps you develop APIs faster.

Using Newman, one can effortlessly run and test a Postman Collections directly from the command-line. It is built with extensibility in mind so that you can easily integrate it into your continuous integration servers and build systems.

For details on changes across v2 to v3, see the [Newman v2 to v3 Migration Guide](#)
For Newman v2.x release documentation, see the [Newman v2.x README](#).

Contents

- 1. [Getting Started](#)
 - 1. [Using Newman as a Node.js module](#)

 npm install -g newman
[how? learn more](#)

 kunagpal published 3 weeks ago

3.8.3 is the latest of 115 releases

[github.com/postmanlabs/newman](#)

Apache-2.0

Collaborators [list](#)



Stats

2,032 downloads in the last day

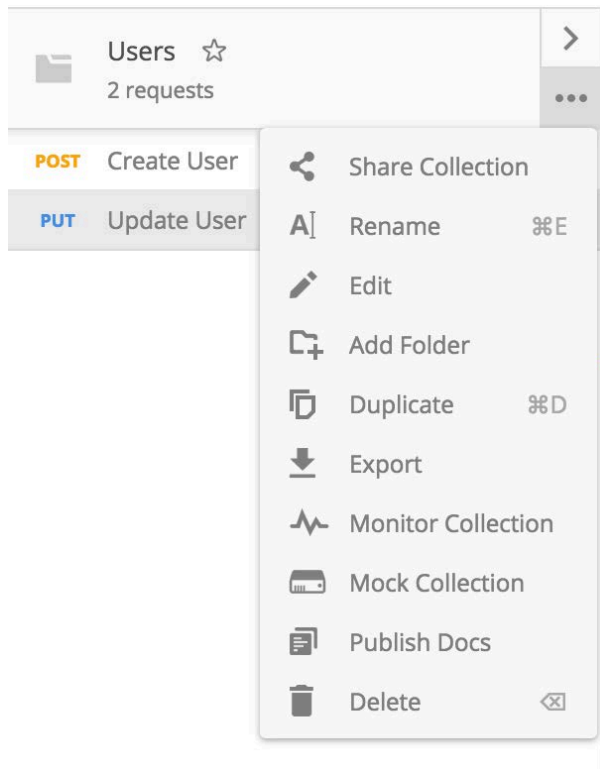
23,844 downloads in the last week

114,053 downloads in the last month

[54 open issues](#) on GitHub

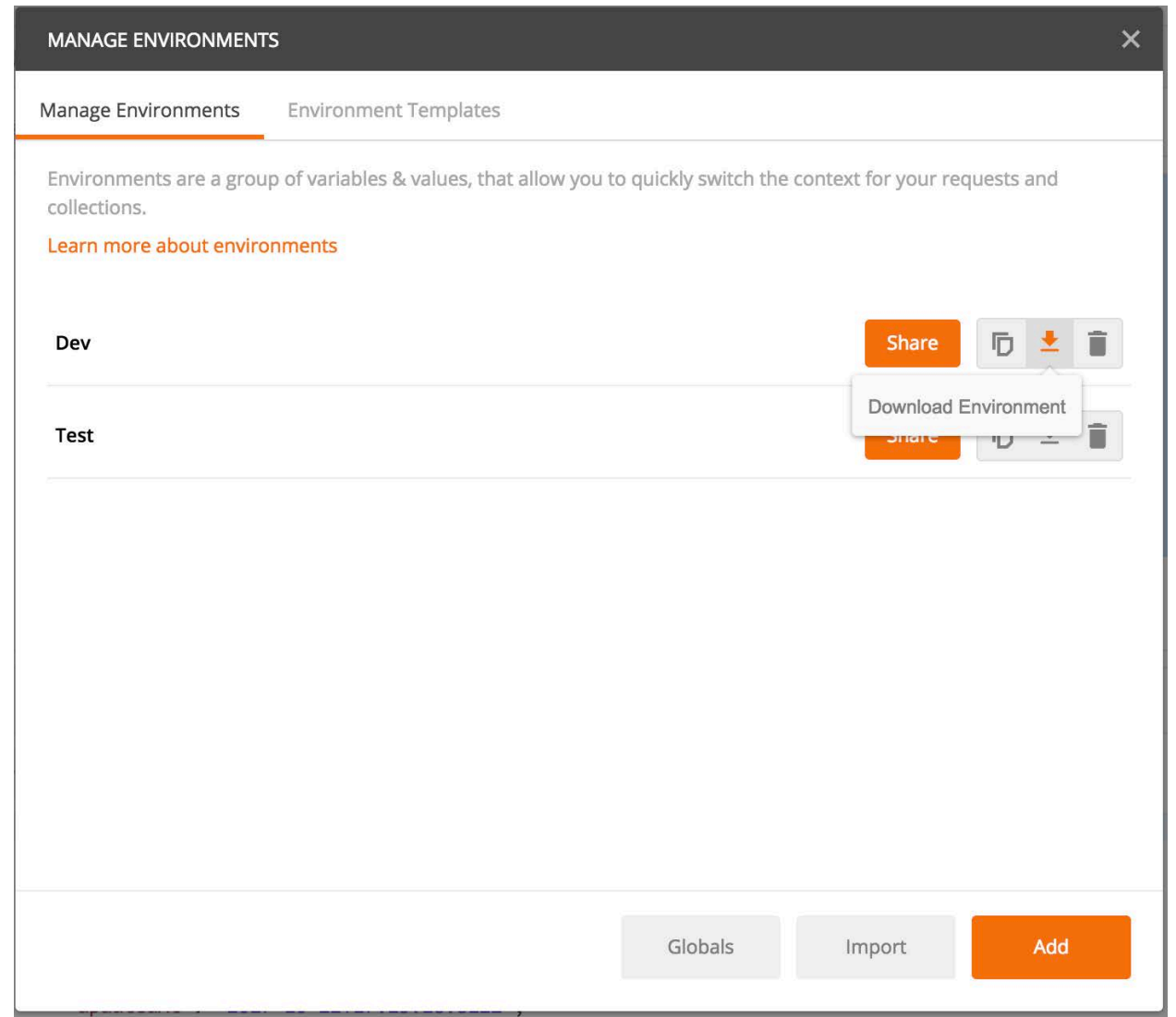


```
> npm install -g newman
```



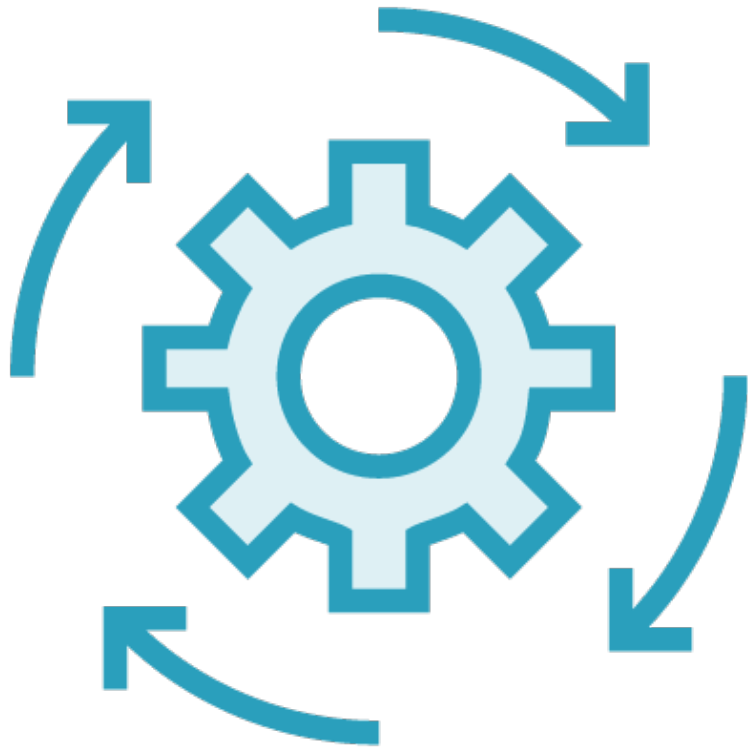
Download collection

Download Environment





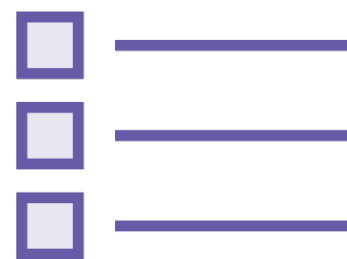
```
>newman run <collection location>  
-e <environment location>
```



Include Postman tests in build process

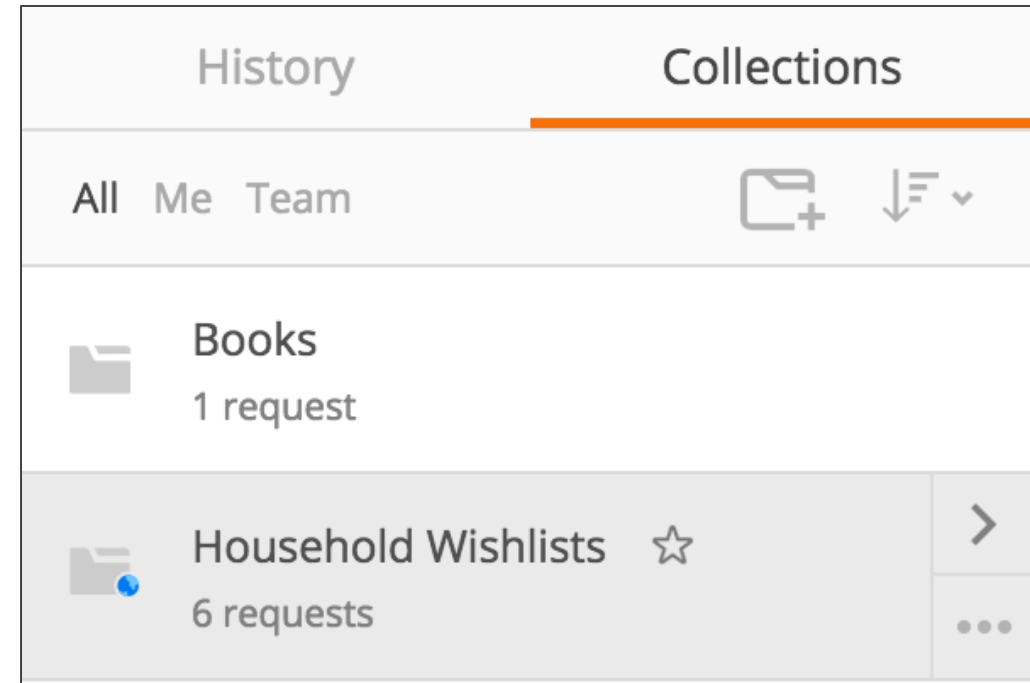


Household



Wish list

Save requests with
collections



Choose a collection or folder:

 Search for a collection or folder

All Collections

 Books

 Households

 Users

Environment Dev 

Iterations 1

Delay 0 ms

Log Responses For all requests  

Data Select File





☒ Persist Variables

Run Users

Recent Runs

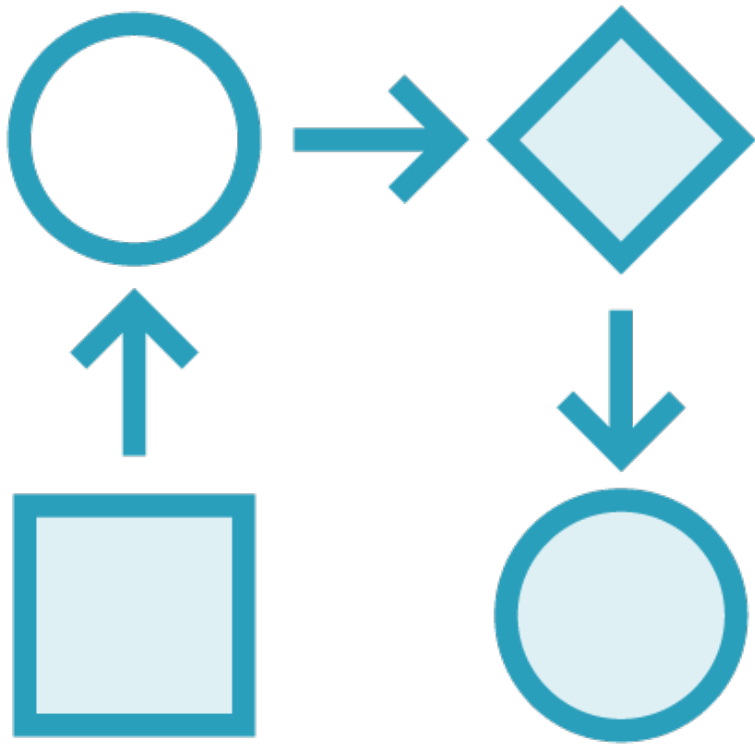
 Type to Filter

Import Test Run

	Users Dev	PASSED Today, 12:20 pm
	Users Dev	PASSED Today, 12:16 pm
	Households Dev	PASSED Today, 11:48 am
	Households Dev	PASSED Yesterday, 8:38 pm



Data driven tests



Using Postman to create workflows

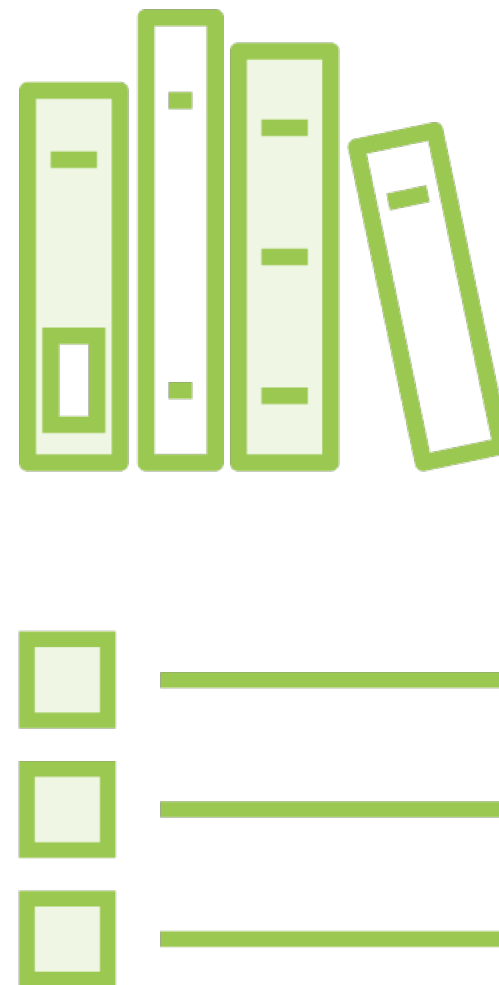
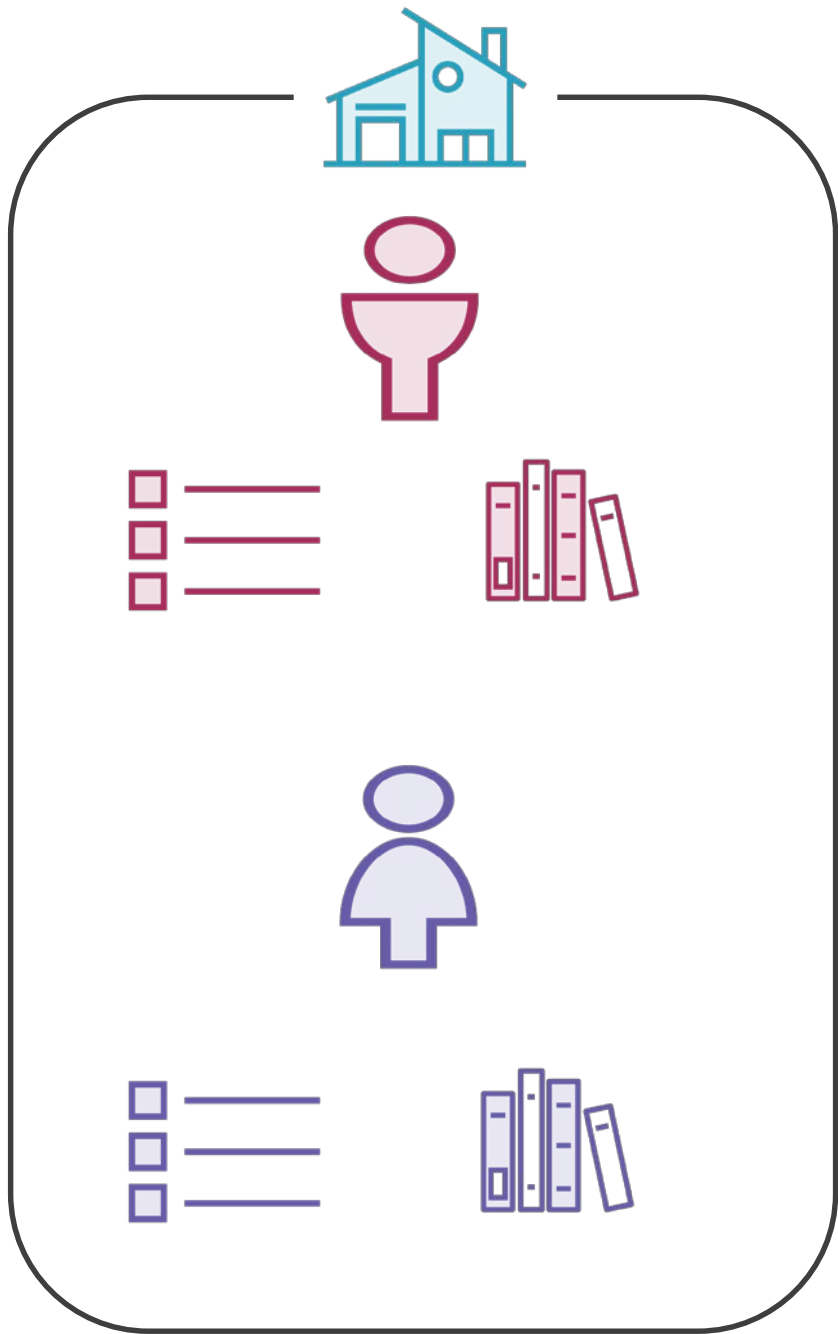
Command line tests

	executed	failed
iterations	1	0
requests	2	0
test-scripts	2	0
prerequisite-scripts	0	0
assertions	3	1
total run duration: 237ms		
total data received: 448B (approx)		
average response time: 47ms		

Mock Server



Faking Out Data



API Steps

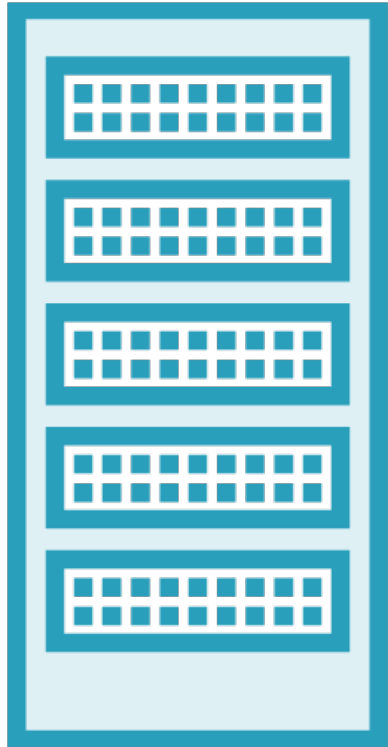
List of Households

Get Specific
Household

List of Owned
Books



API is not ready



Mock server

First Mock

API Steps

List of Households

Get Specific
Household

List of Owned
Books

Additional Responses

Mocking a Feature

Mocking Response Codes

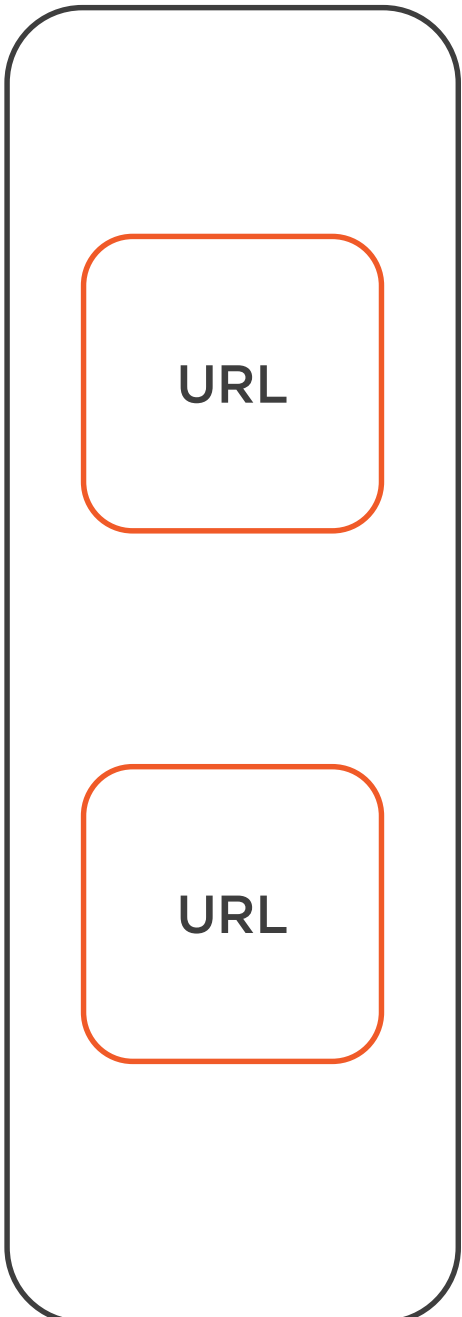
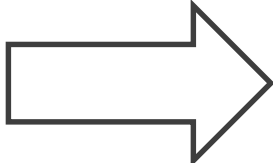
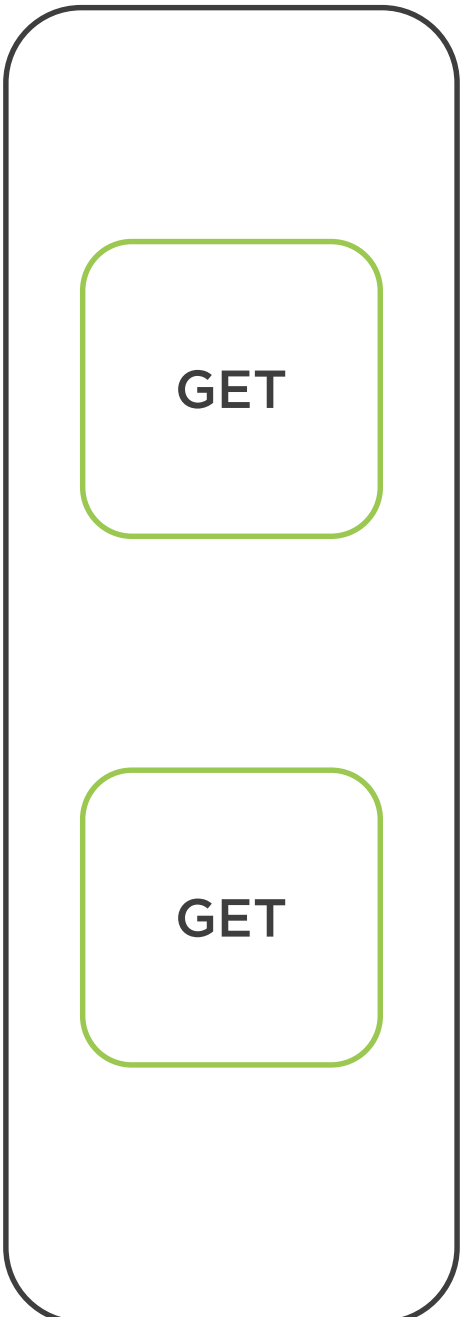
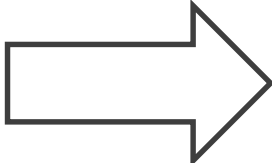
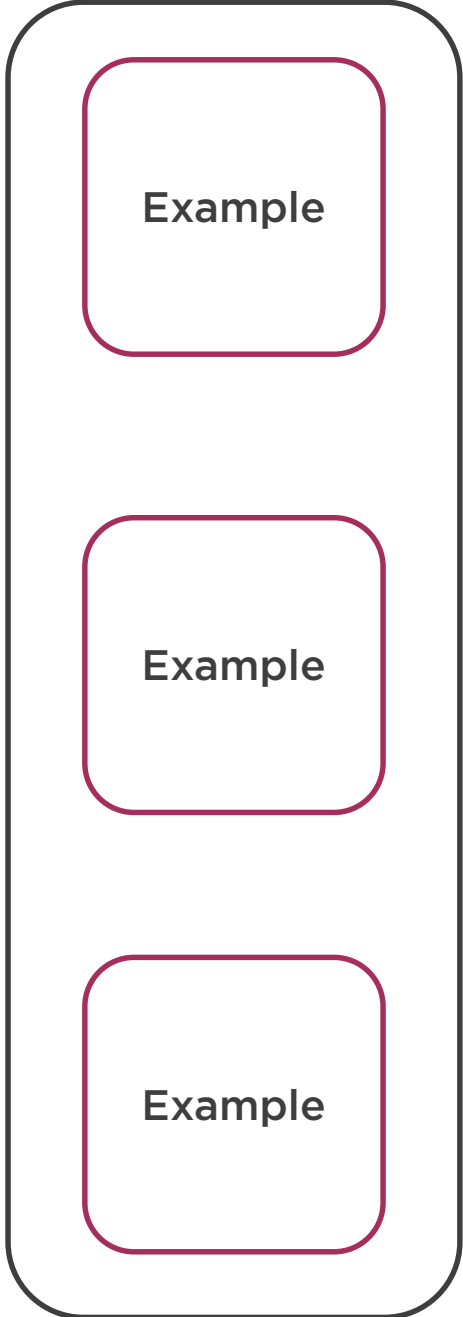
Postman Matching Algorithm



Verb

URL

Response Code



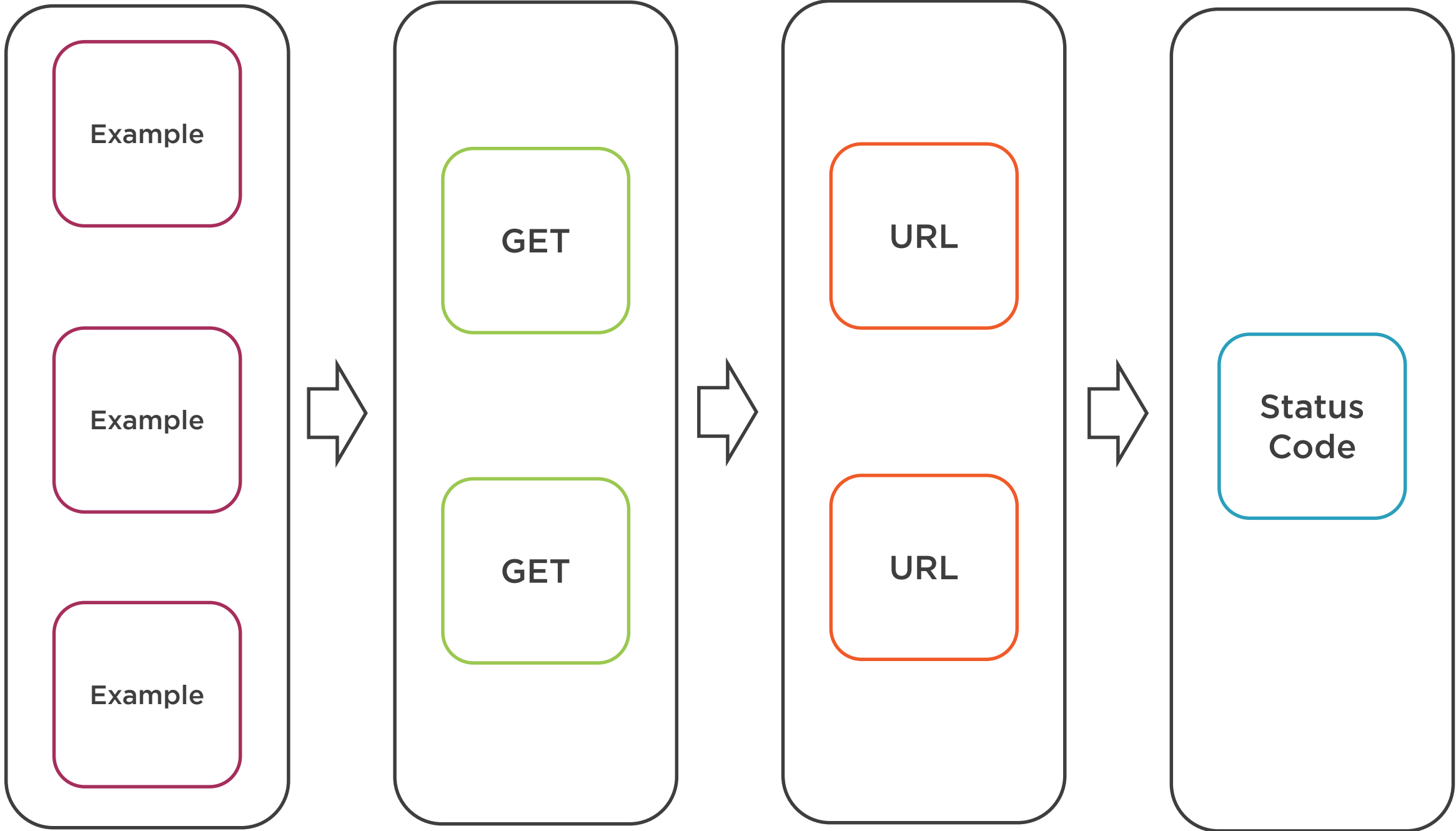
URL Matching

Exact Match

Remove Slashes

Lowercase Match

Strip Alpha-numeric





Examples are scored



Does not match on headers

Review

Single Mock

Multiple Mocks

Mock a Feature

Response Codes

**Matching
Algorithm**

Postman breaks API
dependency

Documentation



The Need for Documentation



Testing the *API*



Paying API consumers

Required Documentation

URL

Headers

Request Body

Sample Responses

Postman will make this
simple!

Documenting a Collection

Detailed Request Documentation

Publishing Documentation



Organize requests for documentation

Applying a Custom Theme



Basic Documentation



Sample Responses



Published
Documentation

