JavaScript

Modules 3

#Unicorn rule

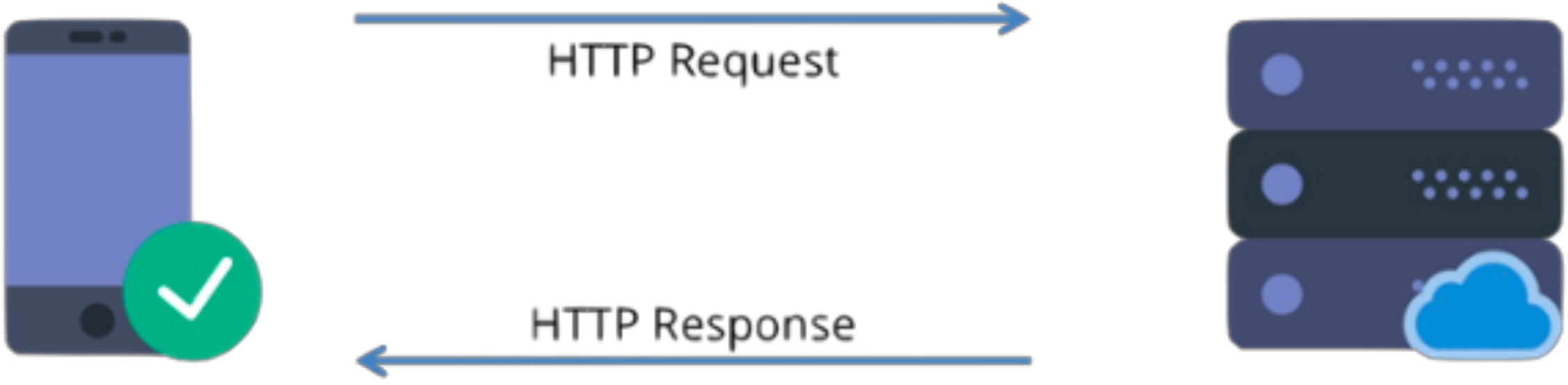
Learning objectives

Until now, you are able to create web pages, make some dynamic update using JavaScript.

But where are your data ?
So what if you want to store data ?
Or use data already stored ?

- Fetching data from a server
- Send data to a server

==> Use API <==

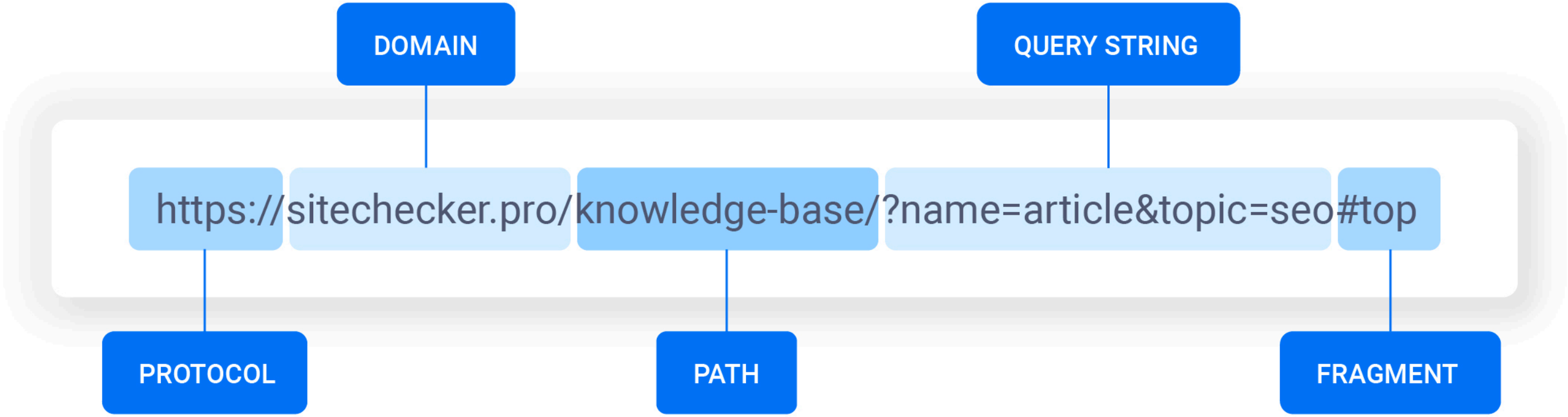

HTTP Request
HTTP Response

Application Programming Interface

What is an API?

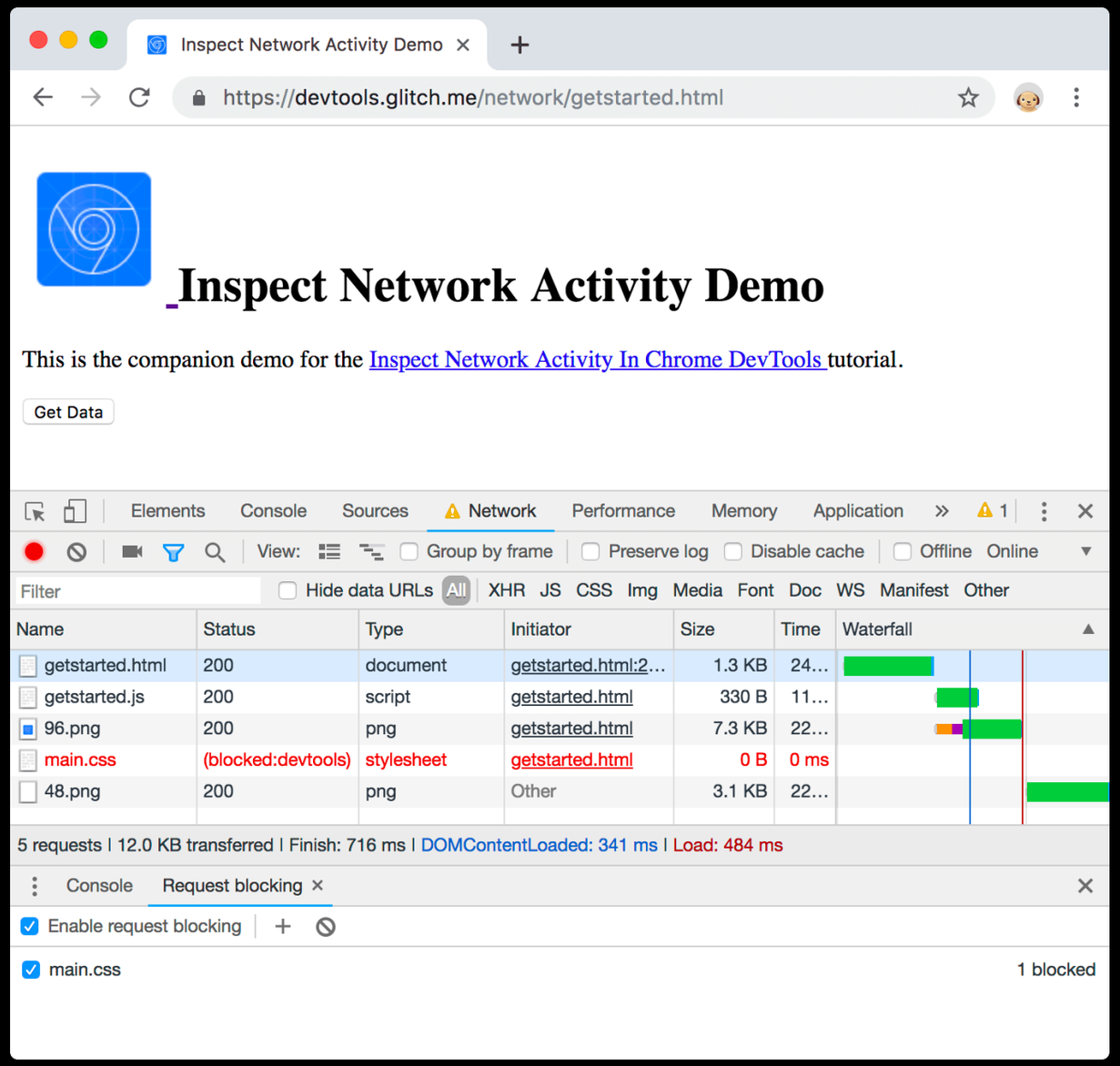
—Mulesoft

Uniform Ressource Locator



Devtools - Network

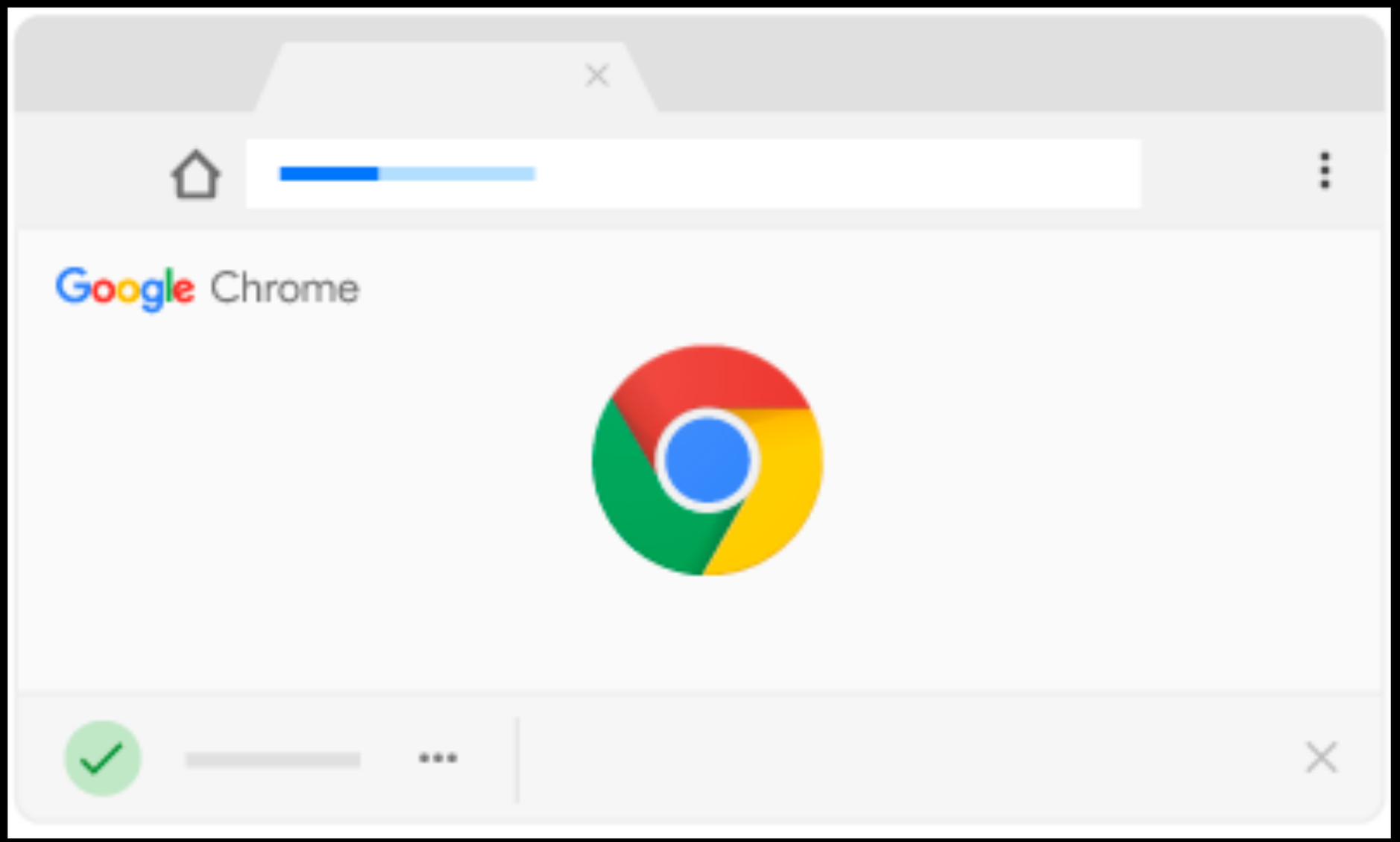
- Take the habit to have always your devtools open
- For HTTP request, take a look at “Network” tab



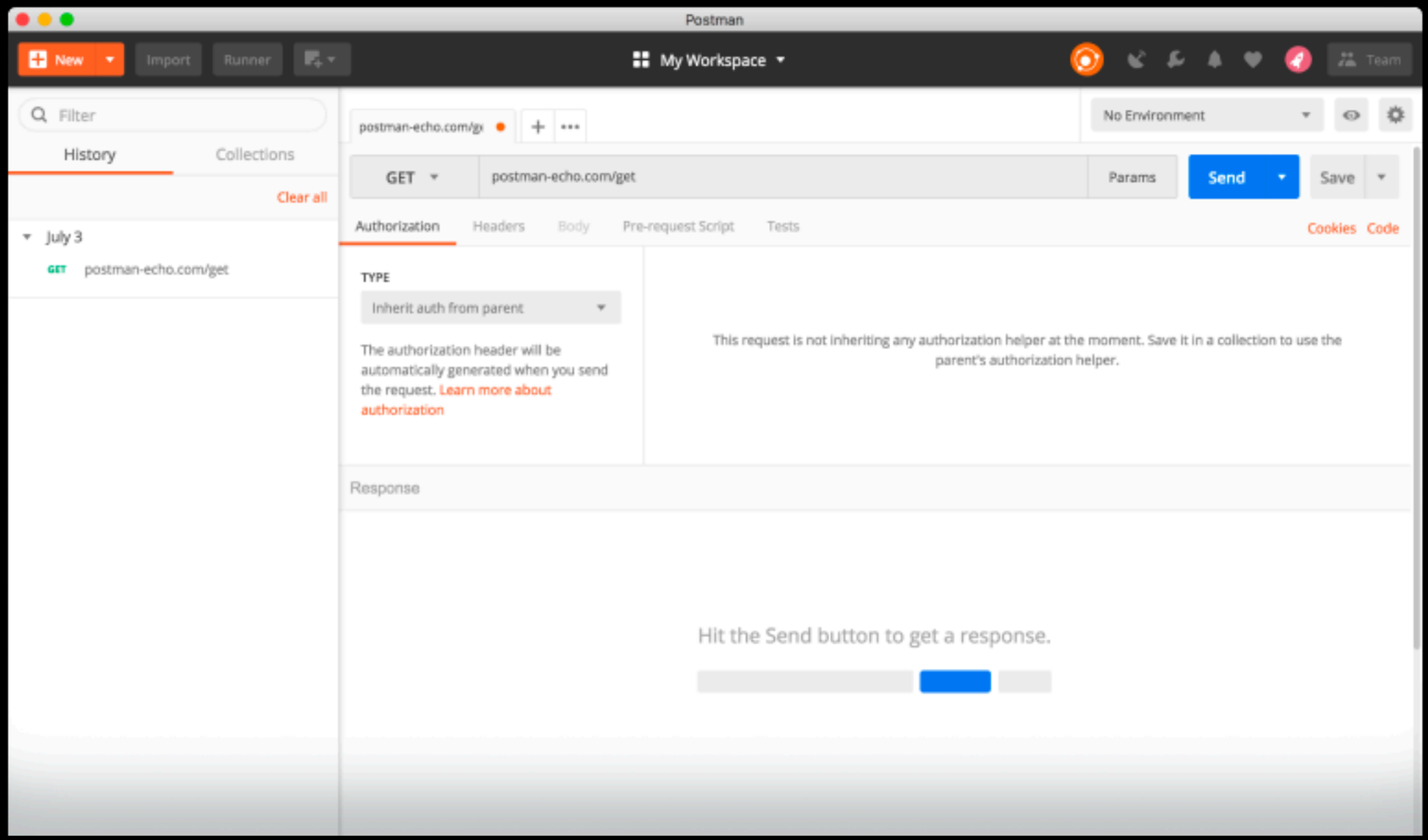
🐶 Let’s get a dog 🐕

https://dog.ceo/dog-api/

Browser



Postman


Postman
New
Import
Runner
My Workspace
Team
Filter
History
Collections
Clear all
July 3
GET postman-echo.com/get
postman-echo.com/gx
No Environment
GET
postman-echo.com/get
Params
Send
Save
Authorization
Headers
Body
Pre-request Script
Tests
Cookies
Code
TYPE
Inherit auth from parent
The authorization header will be automatically generated when you send the request. Learn more about authorization
This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.
Response
Hit the Send button to get a response.

Terminal

kevin — -bash — 101×15

```
Kevins-MBP:~ kevin$ curl https://dog.ceo/api/breed/germanshepherd/images/random
{"message":"https:\/\/images.dog.ceo\/breeds\/germanshepherd\/n02106662_6931.jpg","status":"success"}
Kevins-MBP:~ kevin$
```

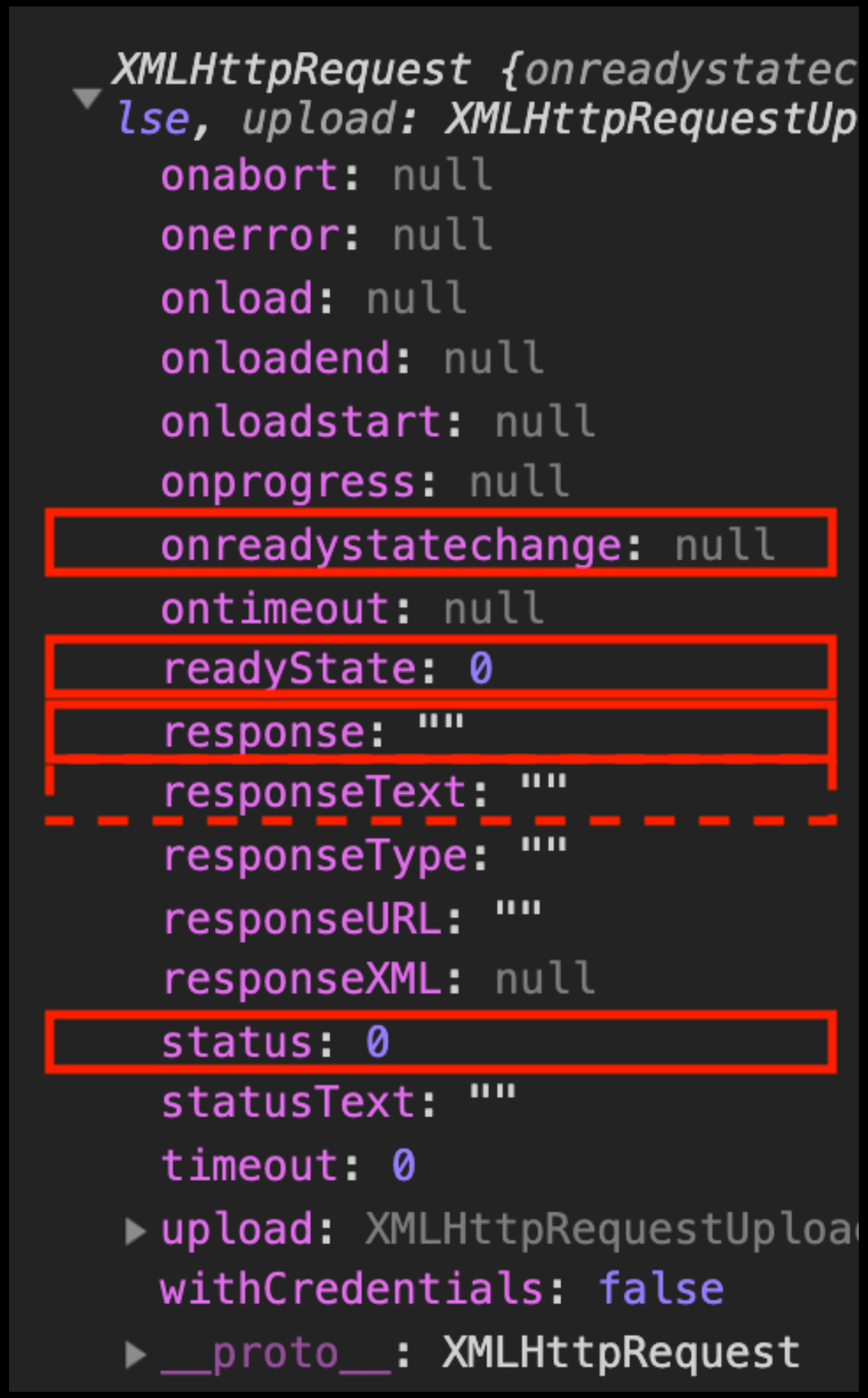
Request using JavaScript

XMLHttpRequest

```
1  // Create XHR Object
2
3  const xhr = new XMLHttpRequest();
```

XHR instance

XHR instance



ReadyState

```
1  // Read the state
2
3  console.log(xhr.readyState)
```

Value	State	Description
`0`	`UNSENT`	Client has been created. `open()` not called yet.
`1`	`OPENED`	`open()` has been called.
`2`	`HEADERS_RECEIVED`	`send()` has been called, and headers and status are available.
`3`	`LOADING`	Downloading; `responseText` holds partial data.
`4`	`DONE`	The operation is complete.

Add event listener

```
1  // Add event listener
2
3  xhr.onreadystatechange = function() {
4    console.log(xhr.readyState)
5  }
```

Open the request

```
1  // Open request
2
3  xhr.open('GET', 'https://pokeapi.co/api/v2/pokemon/pikachu/')
```

Send the request

```
1   // Send request
2
3   xhr.send()
```

Complete example (get)

```
1   // COmplete example
2
3   const xhr = new XMLHttpRequest();
4
5   xhr.onreadystatechange = function () {
6     if (xhr.readyState === 4 && xhr.status >= 200 && xhr.status < 300) {
7       console.log(xhr.response);
8     }
9   }
10
11  xhr.open('GET', 'https://jsonplaceholder.typicode.com/posts/1');
12
13  xhr.send();
```

Complete example (post)

```
15    const xhr2 = new XMLHttpRequest();
16
17    xhr2.onreadystatechange = function () {
18      if (xhr2.readyState === 4 && xhr.status >= 200 && xhr.status < 300) {
19        console.log(xhr2.response);
20      }
21    }
22
23    xhr2.open('POST', 'https://jsonplaceholder.typicode.com/posts');
24    xhr2.send('userId=1&title=My title&body=my body post very cool');
```

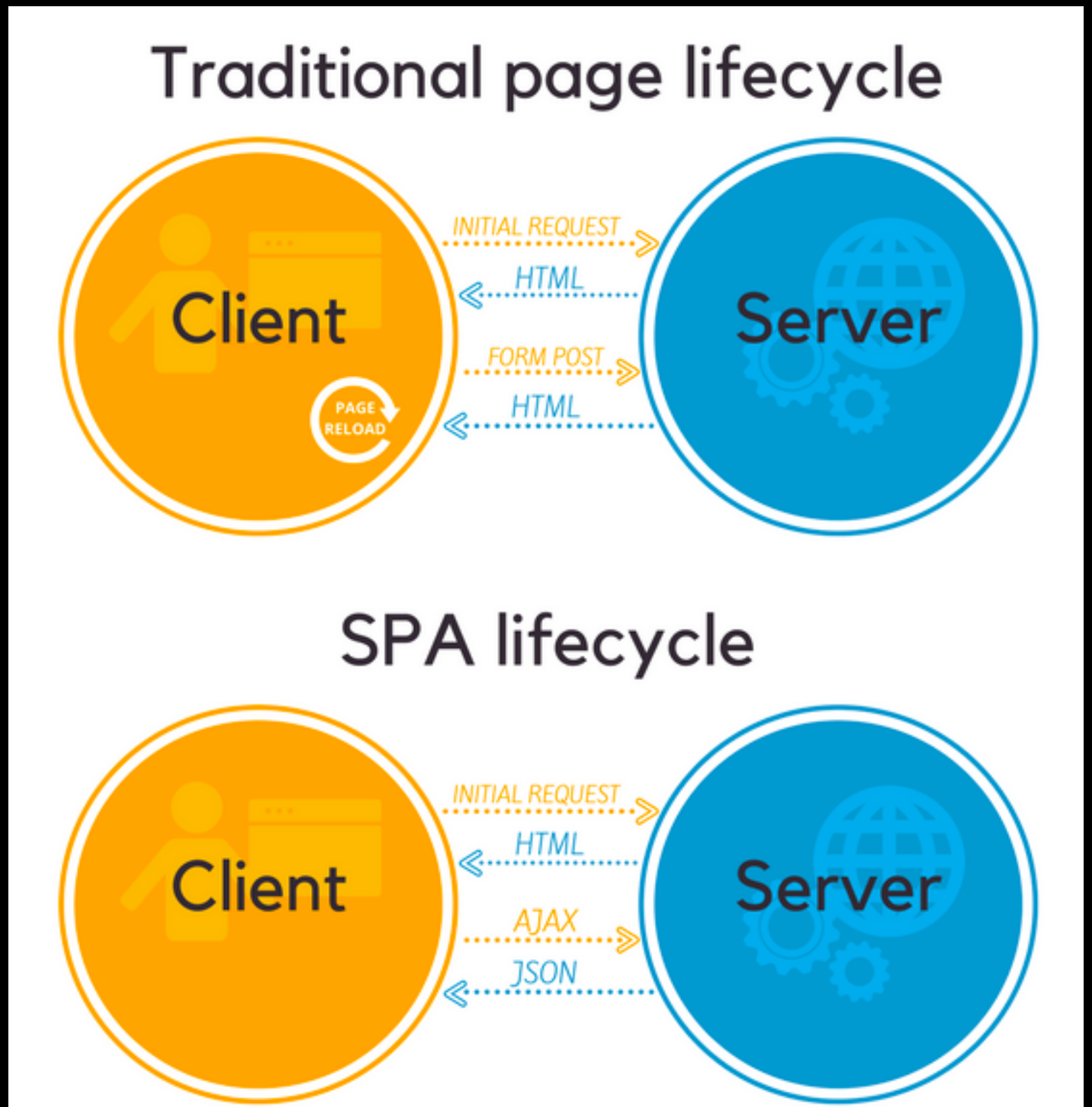
AJAX

AJAX is a developer's dream, because you can:

- Update a web page without reloading the page
- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

–W3School

Asynchronous Javascript And XML



Your turn