

---

---

# Studying HTTP

— Using **Chrome DevTool**, **cURL**, —  
**Postman**

---

---

# Learning Outcomes

Compare performance of HTTP/1.X and HTTP/2

Understanding HTTP Request/Response mechanisms by using

- Chrome Dev Tools

- curl command

- Postman

# Comparison HTTP/1.X and HTTP/2

Go to

<https://http2.golang.org/gopher/>  
[hertiles](#)

A grid of 180 tiled images

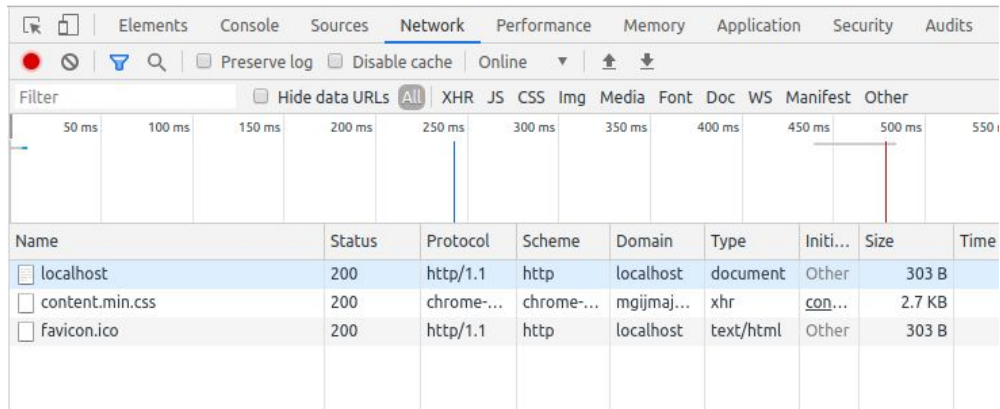
[[HTTP/2, 0s latency](#)] [[HTTP/1, 0s latency](#)]  
[[HTTP/2, 30ms latency](#)] [[HTTP/1, 30ms latency](#)]  
[[HTTP/2, 200ms latency](#)] [[HTTP/1, 200ms latency](#)]  
[[HTTP/2, 1s latency](#)] [[HTTP/1, 1s latency](#)]



# Chrome DevTools

Go to **Settings -> More Tools -> Developer Tools**

OR Press **Ctrl+Shift+I**



The screenshot shows the Chrome DevTools Network tab. The top panel displays a timeline of requests with a vertical line at 250 ms. The bottom panel shows a table of requests.

Name	Status	Protocol	Scheme	Domain	Type	Initi...	Size	Time
<input type="checkbox"/> localhost	200	http/1.1	http	localhost	document	Other	303 B	
<input type="checkbox"/> content.min.css	200	chrome-...	chrome-...	mgijmaj...	xhr	con...	2.7 KB	
<input type="checkbox"/> favicon.ico	200	http/1.1	http	localhost	text/html	Other	303 B	

## Tool Reference

<https://developers.google.com/web/tools/chrome-devtools>

<https://developers.google.com/web/tools/chrome-devtools/network/reference>

# Comparison HTTP/1.X and HTTP/2

Click on the HTTP/1 version links  
and observe performance on  
Chrome Dev Tools

Click on the HTTP/2 version links  
and observe the performance on  
Chrome Dev Tools

Compare the performance between  
the two versions for same latency

[\[HTTP/2, 0s latency\]](#) [\[HTTP/1, 0s latency\]](#)  
[\[HTTP/2, 30ms latency\]](#) [\[HTTP/1, 30ms latency\]](#)  
[\[HTTP/2, 200ms latency\]](#) [\[HTTP/1, 200ms latency\]](#)  
[\[HTTP/2, 1s latency\]](#) [\[HTTP/1, 1s latency\]](#)



# Comparison HTTP/1.X and HTTP/2

Check

the number of requests,

the file size transferred

the load time for each versions  
at the status bar of the Chrome  
DevTool as shown below

[\[HTTP/2, 0s latency\]](#) [\[HTTP/1, 0s latency\]](#)  
[\[HTTP/2, 30ms latency\]](#) [\[HTTP/1, 30ms latency\]](#)  
[\[HTTP/2, 200ms latency\]](#) [\[HTTP/1, 200ms latency\]](#)  
[\[HTTP/2, 1s latency\]](#) [\[HTTP/1, 1s latency\]](#)



184 requests	187 KB transferred	164 KB resources	Finish: 41.28 s	DOMContentLoaded: 1.52 s	Load: 40.70 s
--------------	--------------------	------------------	-----------------	--------------------------	---------------



# Comparison HTTP/1.X and HTTP/2

At the top observe the number of TCP connections established for each version

[\[HTTP/2, 0s latency\]](#) [\[HTTP/1, 0s latency\]](#)  
[\[HTTP/2, 30ms latency\]](#) [\[HTTP/1, 30ms latency\]](#)  
[\[HTTP/2, 200ms latency\]](#) [\[HTTP/1, 200ms latency\]](#)  
[\[HTTP/2, 1s latency\]](#) [\[HTTP/1, 1s latency\]](#)



☒ Use large request rows

☐ Group by frame

☒ Show overview

☐ Capture screenshots

5000 ms

10000 ms

15000 ms

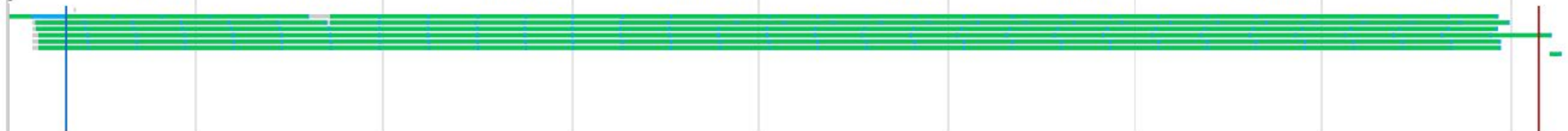
20000 ms

25000 ms

30000 ms

35000 ms

40000 ms



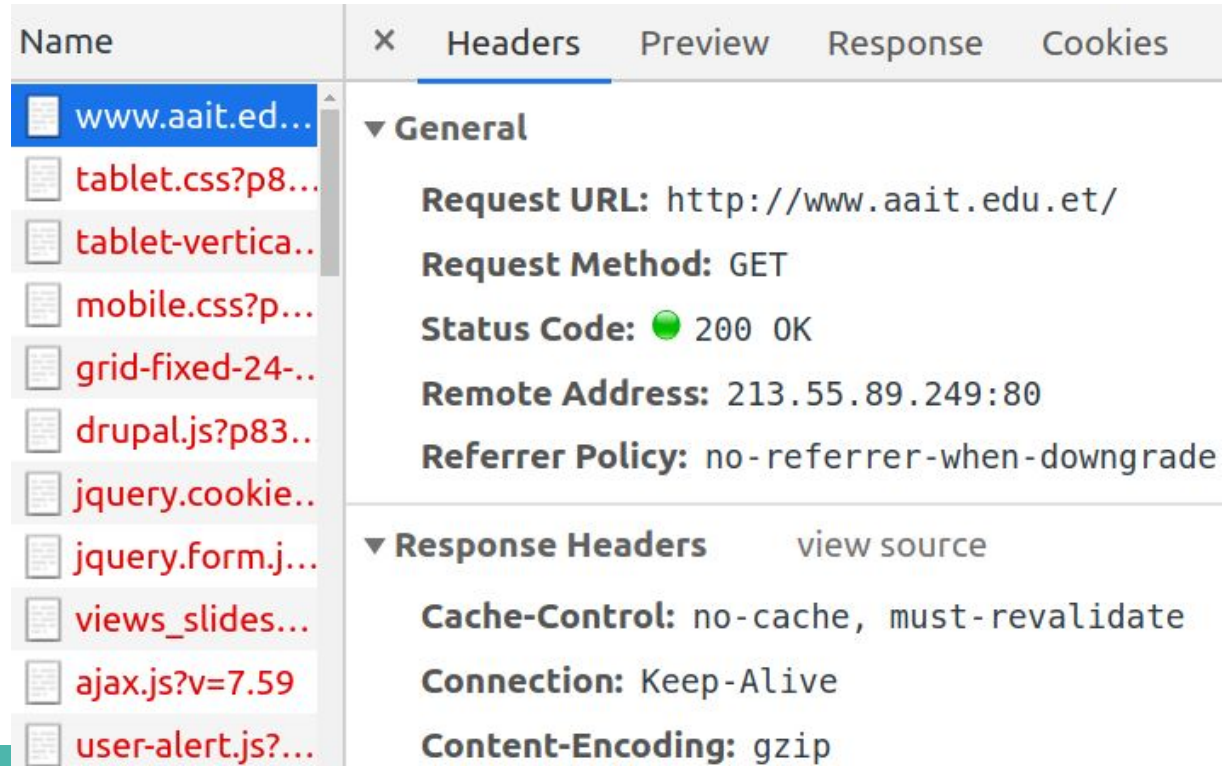


# Study HTTP Request/Response [Chrome Dev Tool]

Go to [www.aait.edu.et](http://www.aait.edu.et)

Open **Chrome Dev Tool** and select the **Network** Tab

Select the first resource and then select the Headers section



The screenshot shows the Chrome DevTools Network tab. The left sidebar lists several resources, with the first one, 'www.aait.ed...', selected. The right pane shows the 'Headers' section for this resource. The 'General' section displays the Request URL as 'http://www.aait.edu.et/', the Request Method as 'GET', the Status Code as '200 OK' (indicated by a green circle), the Remote Address as '213.55.89.249:80', and the Referrer Policy as 'no-referrer-when-downgrade'. The 'Response Headers' section is also visible, showing 'Cache-Control: no-cache, must-revalidate', 'Connection: Keep-Alive', and 'Content-Encoding: gzip'.

Name	Headers	Preview	Response	Cookies
www.aait.ed...	<b>▼ General</b> <b>Request URL:</b> http://www.aait.edu.et/ <b>Request Method:</b> GET <b>Status Code:</b> <span style="color: green;">●</span> 200 OK <b>Remote Address:</b> 213.55.89.249:80 <b>Referrer Policy:</b> no-referrer-when-downgrade			
tablet.css?p8...	<b>▼ Response Headers</b> <a href="#">view source</a> <b>Cache-Control:</b> no-cache, must-revalidate <b>Connection:</b> Keep-Alive <b>Content-Encoding:</b> gzip			
tablet-vertica...				
mobile.css?p...				
grid-fixed-24...				
drupal.js?p83...				
jquery.cookie...				
jquery.form.j...				
views_slides...				
ajax.js?v=7.59				
user-alert.js?...				



# Study HTTP Request/Response [Chrome Dev Tool]

What HTTP version is the site serving?

Study the HTTP General, Request and Response headers by referencing the following document or by searching the particular head component on Google

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>

What version of HTTP does Google use?

# Study HTTP Request/Response [curl]

Installing curl

**Ubuntu and Debian**

```
apt install curl
```

**Redhat and Centos**

```
yum install curl
```

**MacOS**

```
brew install curl
```

# Study HTTP Request/Response [curl]

Installing curl

## Windows

Available on windows 10 version 1803 or later

If not available download and install it from the following link

<https://curl.haxx.se/windows/>

Reference: <https://ec.haxx.se/>

<https://linuxacademy.com/guide/13852-understanding-curl-and-http-headers/>

# How to HTTP with curl

GET	<code>curl http://www.example.com</code>
POST	<code>curl -d "data to post" <a href="http://www.example.com">http://www.example.com</a></code>
PUT	<code>curl -T file1 http://example.com/new/res/file0</code>
HEAD	<code>curl -I http://www.example.com</code>

# How to HTTP with curl

You can ask curl to both ask for compressed content and automatically and transparently uncompress gzipped data when receiving content encoded gzip by using `--compressed`

```
curl --compressed http://example.com/
```

# How to HTTP with curl

To tell curl to do an authenticated HTTP request, you use the `-u`, `--user` option to provide username and password

```
curl --user bet:p@$w0rd http://example.com/
```

# How to HTTP with curl

You can make curl ask for a range with `-r` or `--range`. If you want the first 200 bytes out of something:

```
curl -r 0-199 http://example.com
```



# How to HTTP with curl

Sending "URL encoded" **POST** request

```
curl -d 'name=admin&shoesize=12' http://example.com/
```

**POST**ing with curl's **-d** option will make it include a default header that looks like **Content-Type: application/x-www-form-urlencoded**

However, if you **POST JSON** to a server and want to more accurately tell the server about what the content is you can do like the following

```
curl -d '{json}' -H 'Content-Type: application/json'  
https://example.com
```

# How to HTTP with curl

You can customize request headers

To change the **Host:** header, do this:

```
curl -H "Host: test.example" http://example.com/
```

To add a **Elevator: floor-9** header, do this:

```
curl -H "Elevator: floor-9" http://example.com/
```

To switch off the **User-Agent:** header, do this:

```
curl -H "User-Agent:" http://example.com/
```

# How to HTTP with curl

With curl you set the referer header with `-e` or `--referer`, like this:

```
curl --referer http://comes-from.example.com  
https://www.example.com/
```

When a user clicks on a link on a web page and the browser takes the user away to the next URL, it will send the new URL a "referer" header in the new request telling it where it came from. That is the referer header

# How to HTTP with curl

Ask a server to only deliver a response if the resource has been modified after a particular time:

```
curl --time-cond "1 Jul 2019"  
https://www.example.org/file.html
```

This is example of conditional requests.

Conditional requests are requests that contain a condition in the sense that it asks the server to only deliver a response body if the associated condition evaluates true.

# How to HTTP with curl

curl supports HTTP/2 for both HTTP:// and HTTPS:// URLs

To ask a server to use HTTP/2, just:

```
curl --http2 http://example.com/
```

# How to HTTP with curl

## curl HTTP cheat sheet

<b>Verbose</b>	<b>hide progress</b>	<b>extra info</b>	<b>Write output</b>	<b>Timeout</b>
<code>-v</code> <code>--trace-ascii &lt;file&gt;</code>	<code>-s</code>	<code>-w "format"</code>	<code>-O</code> <code>-o &lt;file&gt;</code>	<code>-m &lt;seconds&gt;</code>
<b>POST</b>	<b>multipart formpost</b>	<b>PUT</b>	<b>HEAD</b>	<b>Custom method</b>
<code>-d "string"</code> <code>-d @file</code>	<code>-F name=value</code> <code>-F name=@file</code>	<code>-T &lt;file&gt;</code>	<code>-I</code>	<code>-X "METHOD"</code>
<b>Basic auth</b>	<b>read cookiejar</b>	<b>write cookiejar</b>	<b>send cookies</b>	<b>user-agent</b>
<code>-u user:password</code>	<code>-b &lt;file&gt;</code>	<code>-c &lt;file&gt;</code>	<code>-b "c=1; d=2"</code>	<code>-A "string"</code>
<b>Use proxy</b>	<b>Headers, add/remove</b>	<b>follow redirects</b>	<b>gzipped response</b>	<b>Insecure HTTPS</b>
<code>-x &lt;host:port&gt;</code>	<code>-H "name: value"</code> <code>-H "name:"</code>	<code>-L</code>	<code>--compressed</code>	<code>-k</code>

# Study HTTP Request/Response [Postman]

Download Postman (<https://www.getpostman.com/downloads/>)

## Get Postman for Linux

Join 8 million developers and download the **ONLY** complete API Development Environment.

 Download 

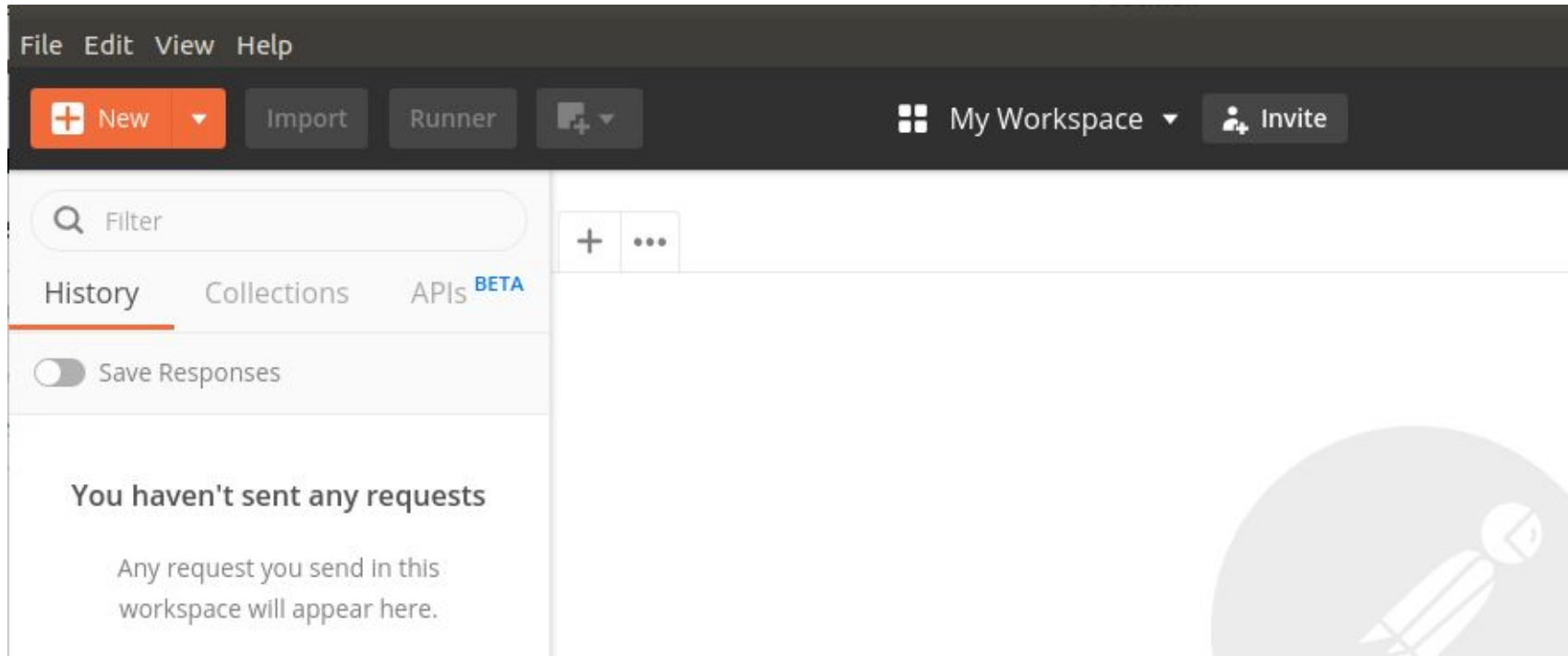
Version 7.11.0 | [RELEASE NOTES](#) | [PRODUCT ROADMAP](#)

Not your OS? Download for [macOS](#) or Windows ([x32](#) / [x64](#))



# Study HTTP Request/Response [Postman]

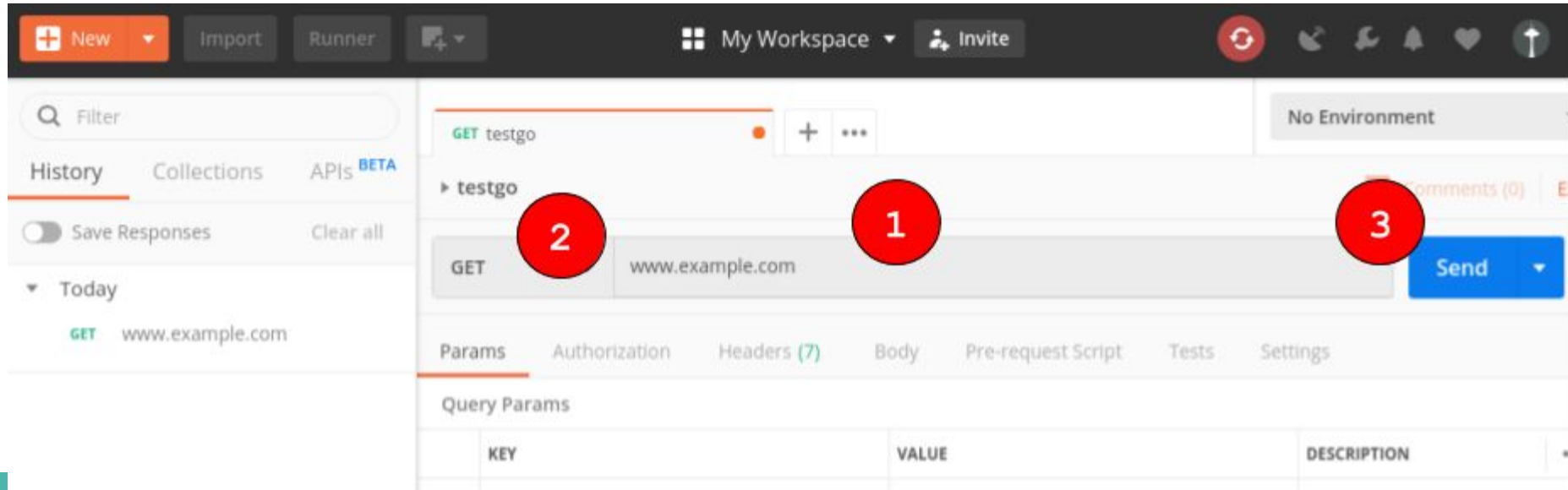
New button is where you will create a new request, collection or environment.



# Study HTTP Request/Response [Postman]

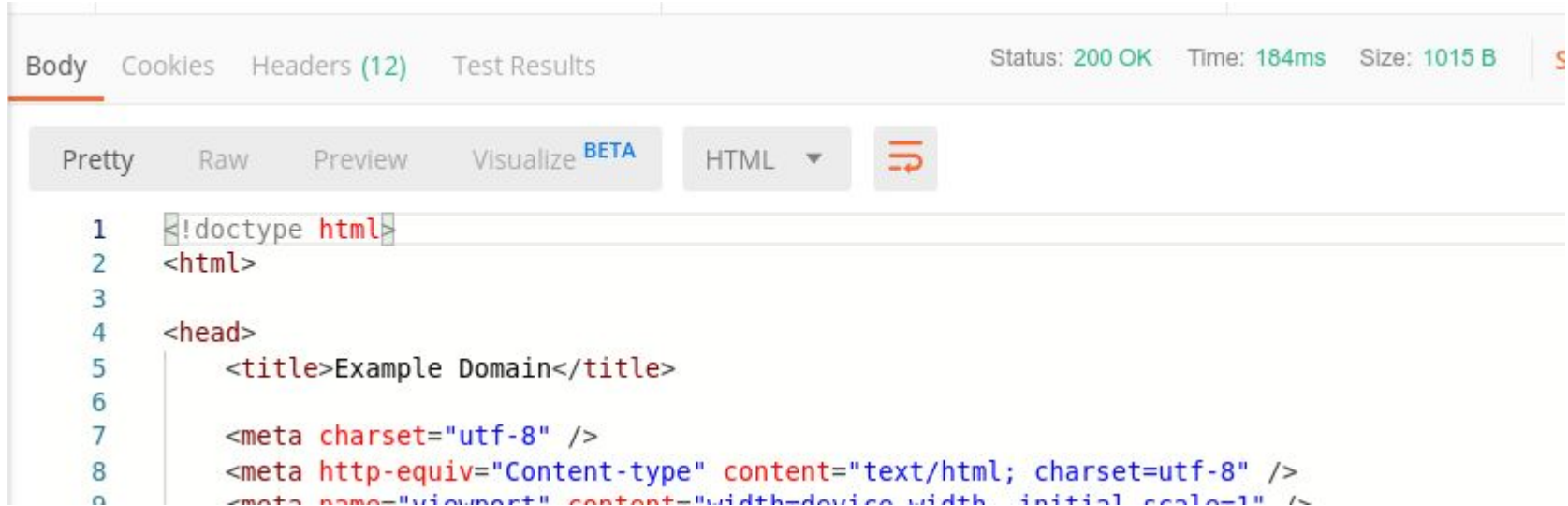
## Making Get Request

- 1 Type the **URL**
- 2 Select the **GET** method
- 3 Press the **Send** button



# Study HTTP Request/Response [Postman]

The response body will is shown at the bottom



The screenshot shows the Postman interface for an HTTP response. The top bar indicates the status is 200 OK, the time is 184ms, and the size is 1015 B. The 'Body' tab is selected, showing the response content in HTML format. The HTML code is as follows:

```
1 <!doctype html>
2 <html>
3
4 <head>
5   <title>Example Domain</title>
6
7   <meta charset="utf-8" />
8   <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
9   <meta name="viewport" content="width=device-width, initial-scale=1" />
```

# Study HTTP Request/Response [Postman]

You can also see the response header

Body Cookies Headers (12) Test Results			Status: 200 OK	Time: 364m
	KEY	VALUE		
	Content-Encoding ⓘ	gzip		
	Accept-Ranges ⓘ	bytes		
	Cache-Control ⓘ	max-age=604800		
	Content-Type ⓘ	text/html; charset=UTF-8		

# Assignment

Study postman in detail by going to the following link

<https://www.guru99.com/postman-tutorial.html>