

Web Interactions

Https - Html - Javascript

arnaud.nauwynck@gmail.com

Outline

Https → Load Html + Js + Css + Json + *

Js → Call Http : SPA, Rest + Json

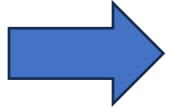
Html → execute Js <script>

Html → trigger events to Js

Js → Query/Update Html DOM

Js Frameworks, MVC, Bindings

Outline



Https → Load Html + Js + Css + Json + *

Js → Call Http : SPA, Rest + Json

Html → execute Js <script>

Html → trigger events to Js

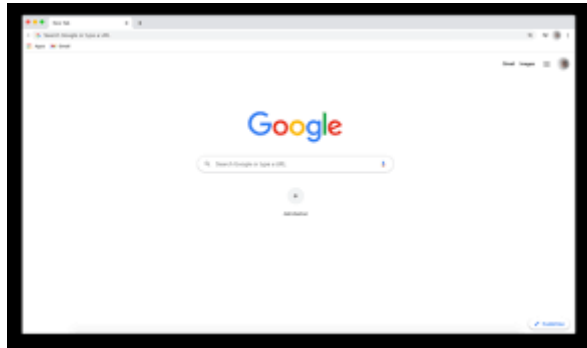
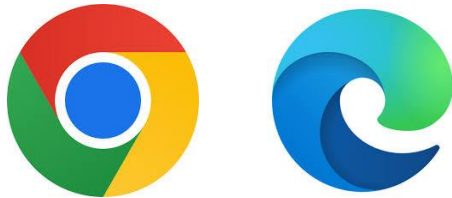
Js → Query/Update Html DOM

Js Frameworks, MVC, Bindings

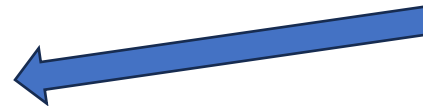
Web Browser = Http Client

Web Site = Http Server

Your PC
mydesktop.home
192.168.0.1



http Request



http Response

Http Server

DNS hostname: www.google.com

Port : 8443

Protocol: Http + TLS

Resource path: some-path

Url: https://www.google.com:8443/some-path



Https://<<hostname.domain>>:<<port>>

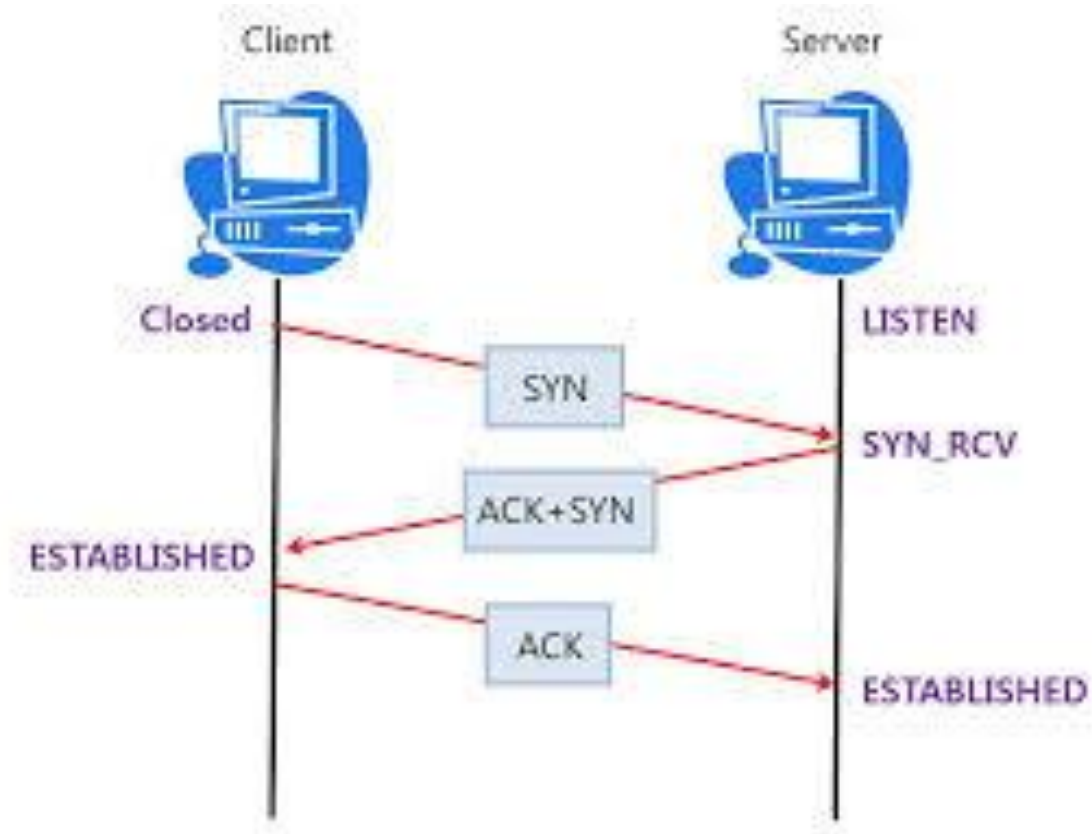
Step 1 : resolve hostname.domain to TCP-IP Address

```
$ nslookup www.google.fr
Serveur : UnKnown
Address: 192.168.0.254

Réponse ne faisant pas autorité :
Nom : www.google.fr
Addresses: 2a00:1450:4007:80c::2003
           216.58.215.35
```

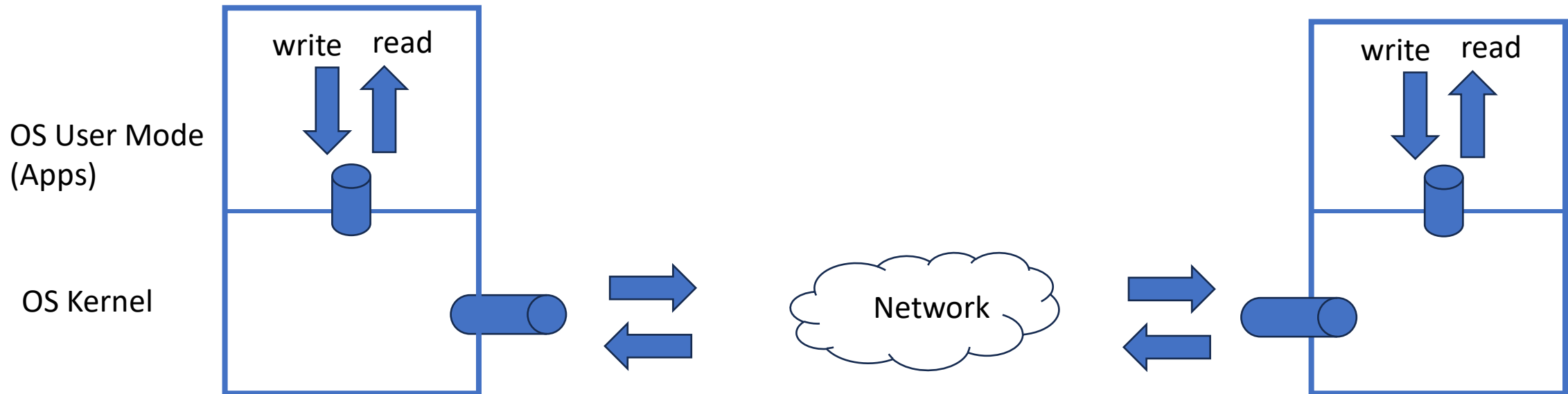
Open TCP-IP Socket

Step 2 : open TCP-IP Socket = SYN, SYN+ACK, ACK



What is a TCP-IP Socket ? (1/3)

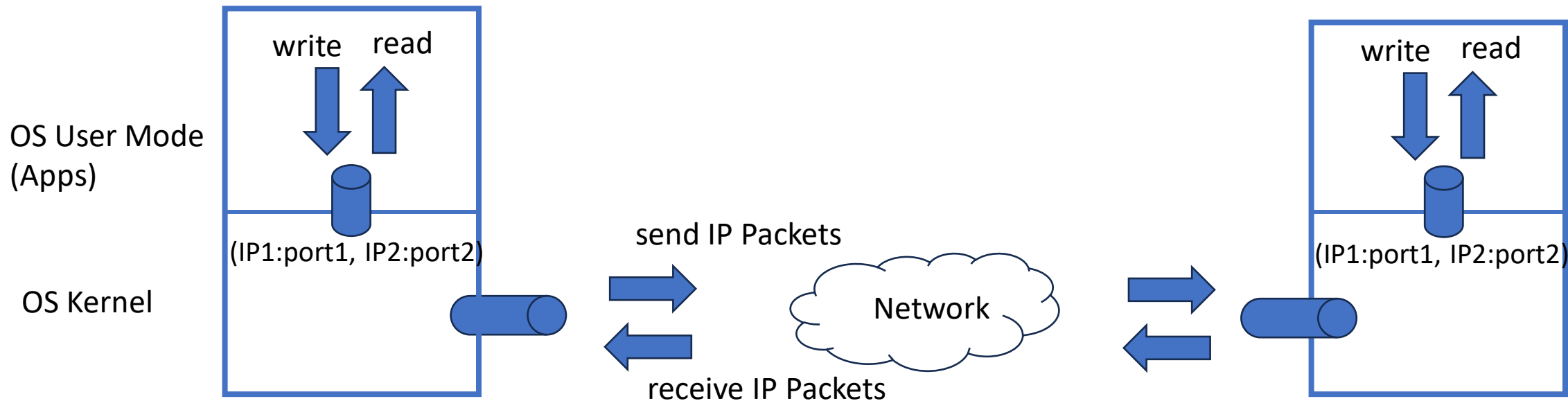
a virtual bi-direction stream



What is a TCP-IP Socket ? (2/3)

a virtual bi-direction stream

identified by 4 ids : (IP-source, Port-Source, IP-dest, Port-Dest)

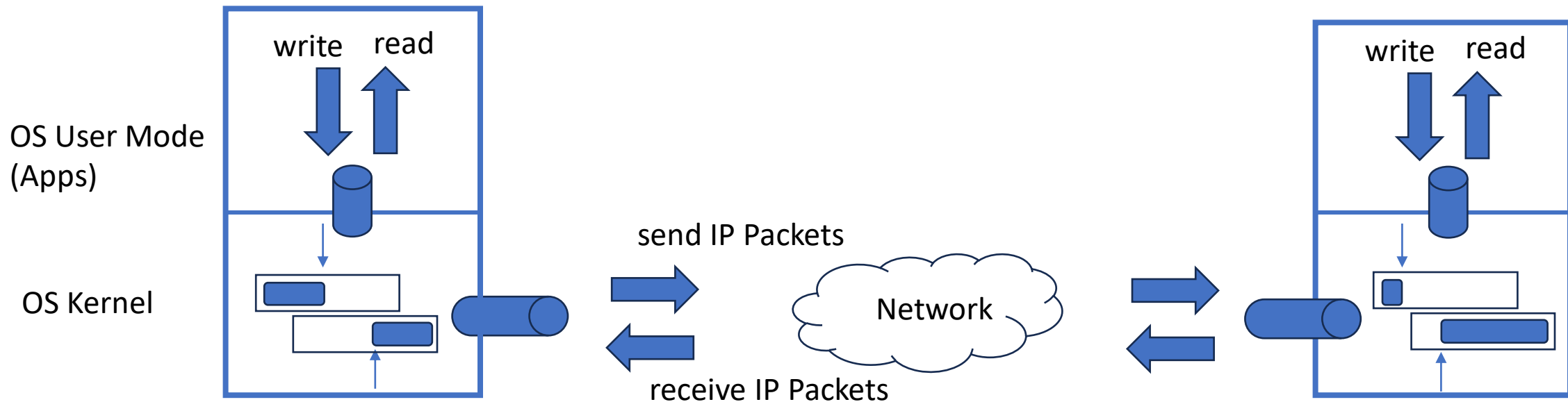


What is a TCP-IP Socket ? (3/3)

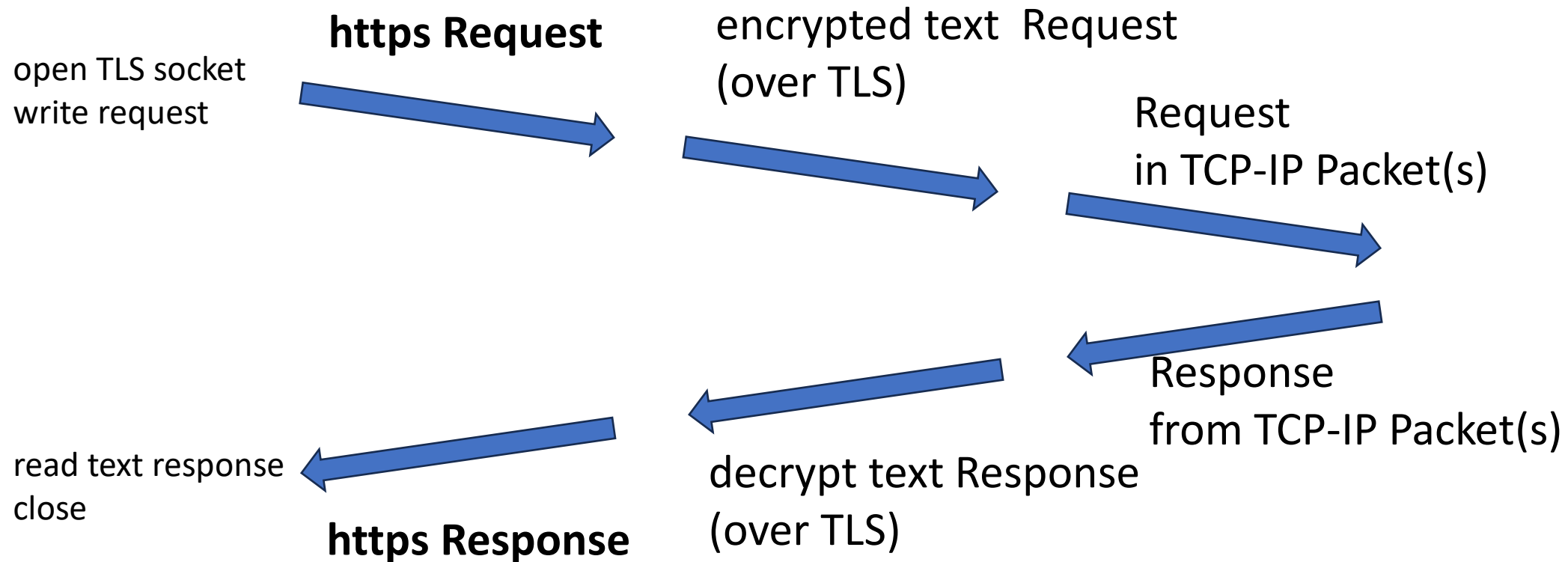
a virtual bi-direction stream

identified by 4 ids : (IP-source, Port-Source, IP-dest, Port-Dest)

with client/server states = buffers + "sequence numbers" (read/write)



HTTPS = Request-Response Text Protocol over TCP-IP (+TLS)



Http Request (GET, POST, PUT, ..)

<https://en.wikipedia.org/wiki/HTTP>

HTTP/1.1 example of request / response transaction [\[edit\]](#)

Below is a sample HTTP transaction between an HTTP/1.1 client and an HTTP/1.1 server running on www.example.com, port 80. [\[note 5\]](#)[\[note 6\]](#)

Client request [\[edit\]](#)

```
GET / HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

A client request (consisting in this case of the request line and a few headers that can be reduced to only the "Host: hostname" header) is followed by a blank line, so that the request ends with a double end of line, each in the form of a [carriage return](#) followed by a [line feed](#). The "Host: hostname" header value distinguishes between various [DNS](#) names sharing a single [IP address](#), allowing name-based [virtual hosting](#). While optional in HTTP/1.0, it is mandatory in HTTP/1.1. (A "/" (slash) will usually fetch a [/index.html](#) file if there is one.)

Http Response (200, 300, 400, 500 ...)

<https://en.wikipedia.org/wiki/HTTP>

Server response [\[edit\]](#)

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 155
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
ETag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Connection: close

<html>
  <head>
    <title>An Example Page</title>
  </head>
  <body>
    <p>Hello World, this is a very simple HTML document.</p>
  </body>
</html>
```

Can you GET/POST ONLY Html from Https ?

Http Requests have **2 parts**:

part 1 in text:

line 1 = VERB (GET/PUT/POST..) + path

line 2, 3, ..N = http header name=value

separator line : ". "

part 2 in any format (byte[])

loading a page (N Requests: html, png, js, svg, css..)

example:

<https://www.wikipedia.org/>

The screenshot shows the Chrome DevTools Network tab for the URL <https://www.wikipedia.org/>. The 'Network' tab is selected and highlighted with a red box. A red arrow points from this box to the first request in the list, 'www.wikipedia.org'. The list of requests includes the main document and various resources like logos, scripts, and images. A timeline at the top of the Network tab shows the duration of these requests. At the bottom, a summary bar indicates that 7 requests were made, 797 B were transferred, and the total load time was 104 ms.

Name	Status	Type	Initiator	Size	T...
www.wikipedia.org	304	document	Other	797 B	2...
Wikipedia-logo-v2@2x.png	200	png	(index):30	(memory cache)	0...
index-24c3e2ca18.js	200	script	(index):859	(memory cache)	0...
gt-ie9-ce3fe8e88d.js	200	script	(index):860	(memory cache)	0...
sprite-de847d1a.svg	200	svg+xml	(index):859	(memory cache)	0...
Wikinews-logo_sister@2x.png	200	png	(index):859	(memory cache)	0...
wikipedia.ico	200	vnd.microsoft...	Other	(disk cache)	1...








7 requests | 797 B transferred | 195 kB resources | Finish: 115 ms | DOMContentLoaded: 95 ms | Load: 104 ms

Console: No messages, No user mess..., No errors

Content-Length

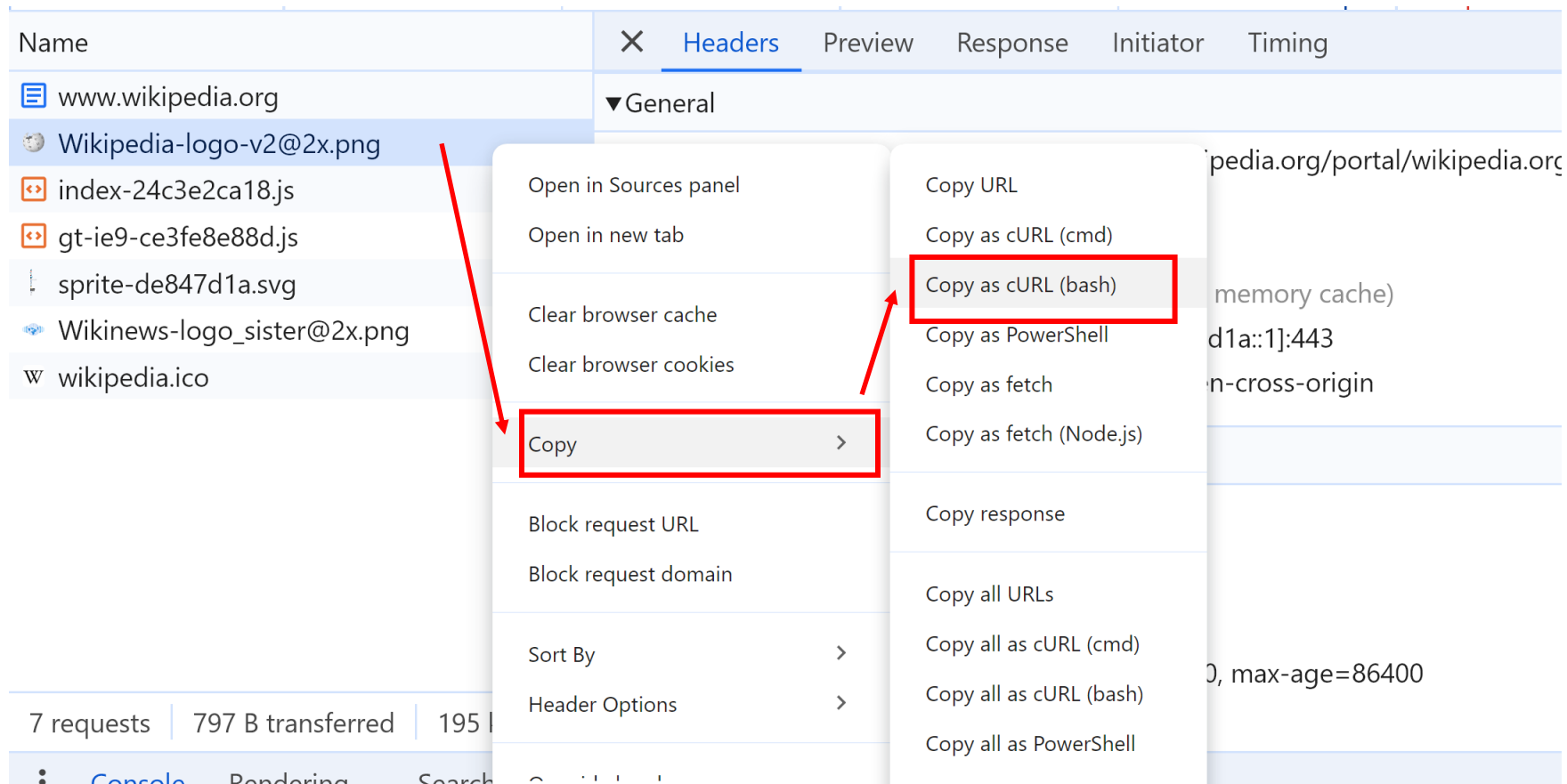
How can Http knows the full binary content response, then close the socket ?

... By using a Http Header "Content-Length"

Name	✕ Headers	Preview	Response	Initiator	Timing
 www.wikipedia.org	▼ General				
 Wikipedia-logo-v2@2x.png	Request URL:		https://www.wikipedia.org/portal/wikipedia.org/assets/img/Wikipedia-l		
 index-24c3e2ca18.js	Request Method:		GET		
 gt-ie9-ce3fe8e88d.js	Status Code:		● 200 OK (from memory cache)		
 sprite-de847d1a.svg	Remote Address:		[2a02:ec80:600:ed1a::1]:443		
 Wikinews-logo_sister@2x.png	Referrer Policy:		strict-origin-when-cross-origin		
 wikipedia.ico	▼ Response Headers				
	Accept-Ranges:		bytes		
	Access-Control-Allow-Origin:		*		
	Age:		13120		
	Cache-Control:		s-maxage=86400, max-age=86400		
7 requests	797 B transferred	195 kB resou	Content-Length:	37479	

Copy Request/Response from Browser

Often useful to reproduce Http request manually from "Postman", or "Curl"



Http Request using Shell Command "curl" (with "-v" for verbose, "--output -" for binary)

```
Cmder
$ curl 'https://www.wikipedia.org/portal/wikipedia.org/assets/img/Wikipedia-logo-v2@2x.png'
Warning: Binary output can mess up your terminal. Use "--output -" to tell
Warning: curl to output it to your terminal anyway, or consider "--output
Warning: <FILE>" to save to a file.
$
$ curl 'https://www.wikipedia.org/portal/wikipedia.org/assets/img/Wikipedia-logo-v2@2x.png'
--output file.png
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100 37479  100 37479    0     0  330k      0 --:--:-- --:--:-- --:--:-- 355k
$
```

"curl" with "-v" for verbose

part 1/3: connection "*" + sent ">" Http headers

connection

sent
http headers

```
Cmder
$ curl 'https://www.wikipedia.org/portal/wikipedia.org/assets/img/Wikipedia-logo-v2@2x.png' -H '
sec-ch-ua: "Not/A)Brand";v="8", "Chromium";v="126", "Google Chrome";v="126"' -H 'Referer: https:
//www.wikipedia.org/' -H 'sec-ch-ua-mobile: ?0' -H 'User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.0.0 Safari/537.36' -H 'sec-ch-ua
-platform: "Windows"' -v --output -
* Host www.wikipedia.org:443 was resolved.
* IPv6: 2a02:ec80:600:ed1a::1
* IPv4: 185.15.58.224
*   Trying [2a02:ec80:600:ed1a::1]:443...
* Connected to www.wikipedia.org (2a02:ec80:600:ed1a::1) port 443
* schannel: disabled automatic use of client certificate
* ALPN: curl offers http/1.1
* ALPN: server accepted http/1.1
* using HTTP/1.x
> GET /portal/wikipedia.org/assets/img/Wikipedia-logo-v2@2x.png HTTP/1.1
> Host: www.wikipedia.org
> Accept: */*
> sec-ch-ua: "Not/A)Brand";v="8", "Chromium";v="126", "Google Chrome";v="126"
> Referer: https://www.wikipedia.org/
> sec-ch-ua-mobile: ?0
> User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chro
me/126.0.0.0 Safari/537.36
```

"curl" with "-v" for verbose

part 2/3: "<" received http headers

received
http headers

```
Cmdr
* schannel: failed to decrypt data, need more data
* Request completely sent off
< HTTP/1.1 200 OK
< date: Sat, 13 Jul 2024 11:19:03 GMT
< etag: "9267-617c8293c9c40"
< expires: Sat, 12 Jul 2025 12:18:17 GMT
< cache-control: s-maxage=86400, max-age=86400
< server: ATS/9.1.4
< last-modified: Mon, 06 May 2024 12:25:13 GMT
< content-length: 37479
< access-control-allow-origin: *
< content-type: image/png
< age: 16587
< x-cache: cp6016 hit, cp6016 hit/2647
< x-cache-status: hit-front
< server-timing: cache;desc="hit-front", host;desc="cp6016"
< strict-transport-security: max-age=106384710; includeSubDomains; preload
< report-to: { "group": "wm_nel", "max_age": 604800, "endpoints": [{ "url": "https://intake-logging
.wikimedia.org/v1/events?stream=w3c.reportingapi.network_error&schema_uri=w3c/reportingapi/network
_error/1.0.0" }] }
< nel: { "report_to": "wm_nel", "max_age": 604800, "failure_fraction": 0.05, "success_fraction": 0.
0}
```

curl with "-v" for verbose
part 3/3: received body

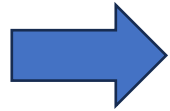
empty line
separator

received
body
(binary)

[illegible]

Outline

Https → Load Html + Js + Css + Json + *



Js → Call Http : SPA, Rest + Json

Html → execute Js <script>

Html → trigger events to Js

Js → Query/Update Html DOM

Js Frameworks, MVC, Bindings

Request as "Blocking" (Low-Level) Code / SystemCalls

Code (ex: java)

```
Socket s = // new Socket(...)  
    SSLSocketFactory.getDefault().createSocket(..)  
s.connect(...)
```

```
OutputStream out = s.getOutputStream()  
out.write("GET / HTTP 1.0\n")  
out.flush()
```

```
InputStream in = s.getInputStream()  
byte[] buffer = new byte[4096]  
int readCount = in.read(buffer)
```

```
s.close()
```

System Calls



connect()



write()



flush()



read()



close()

all blocking calls

(thread/process
suspended to wait)

In the Browser (Javascript Web Spec)
... only 1 "event loop" main Thread

JavaScript API from the Web Browser
implement ONLY
High-Level
NON-Blocking calls
(with Callbacks or Promises)

Callback for Async

```
function asyncFunc( input, resultCallback ) {  
    ...  
}
```

```
yourApp() {  
    asyncFunc(..., resultCallback ); // <= return immediatly,  
                                     // and will call later: resultCallback( ...someResult )  
  
    ... // next... called before resultCallback  
}
```

```
function resultCallback( res ) {  
    ...  
}
```


Promises for Async

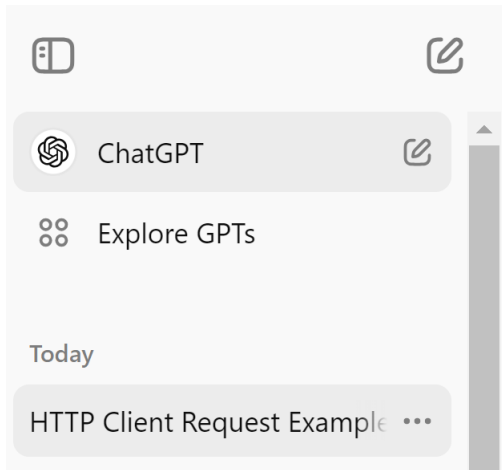
```
function asyncFunc( input ) { // return Promise
  ...
}
```

```
yourApp() {
  var promise = asyncFunc(...); // <= return immediatly, will put result in promise later
  promise.then(resultCallback);

  ... // next... called before resultCallback
}
```

```
function resultCallback( res ) {
  ...
}
```

Http Request as Async (=Non-Blocking) Code



ChatGPT ▾

give me sample javascript code for http client request



Sure! Here are some sample JavaScript codes for making HTTP client requests using various popular methods:

Using Fetch API

The Fetch API is a modern way to make HTTP requests. It is built into the browser and is very straightforward to use.

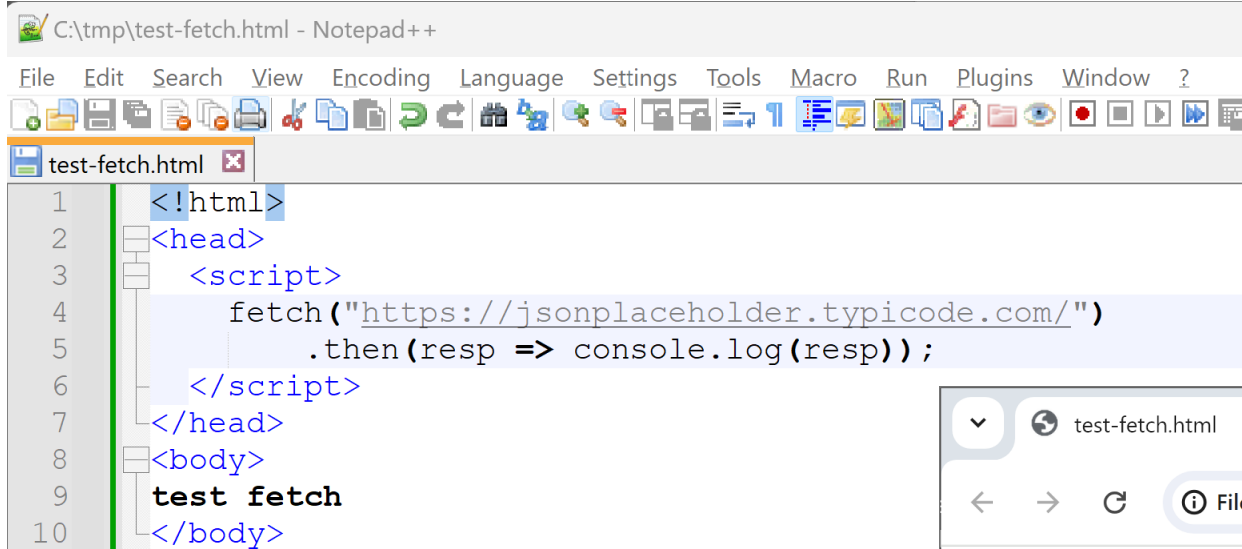
GET Request

javascript

Copy code

```
fetch('https://jsonplaceholder.typicode.com/posts')
  .then(response => response.json()) // Parse the JSON from the response
  .then(data => console.log(data))   // Log the data to the console
  .catch(error => console.error('Error:', error)); // Log any errors
```

test-fetch.html

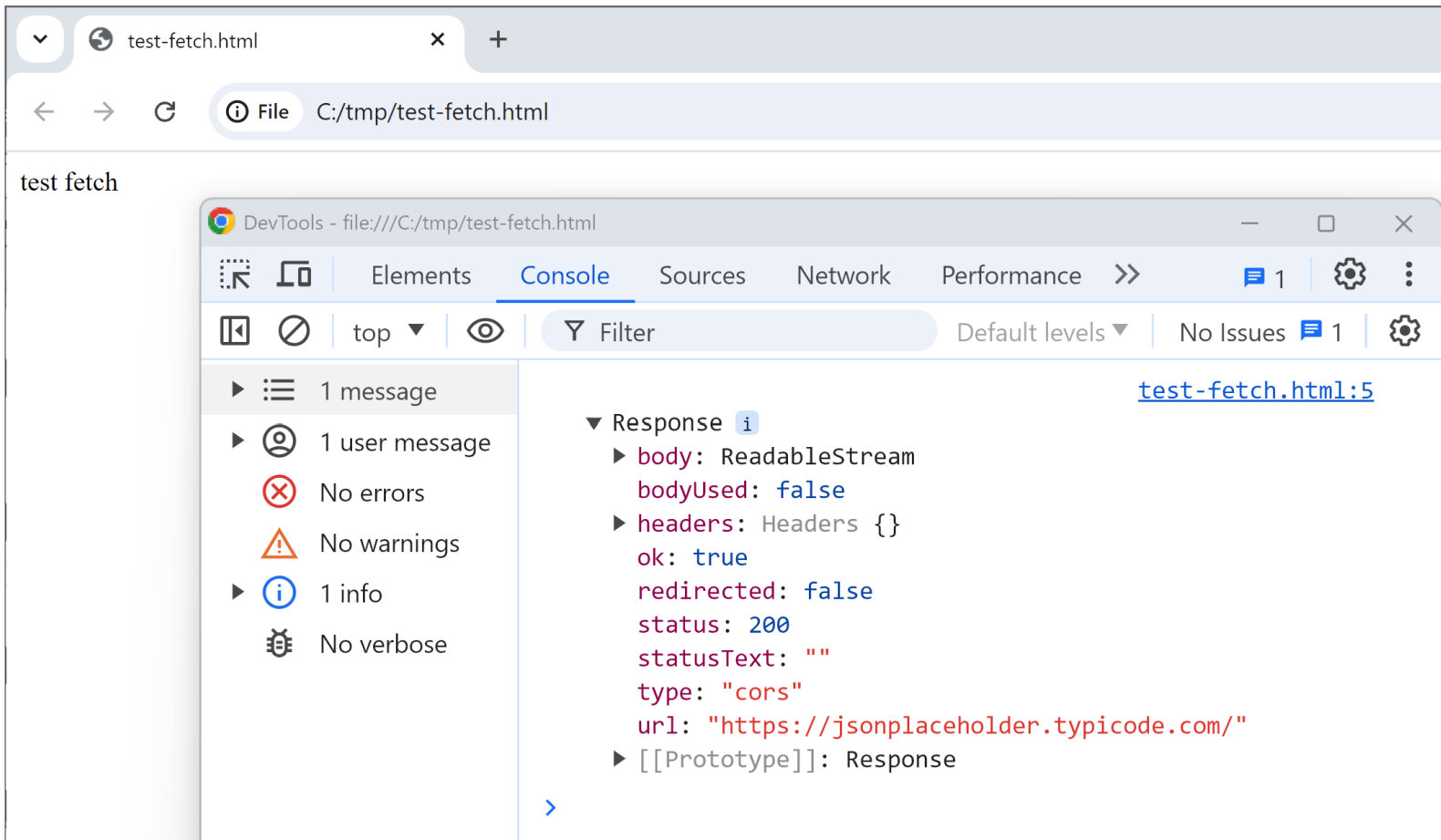


```
C:\tmp\test-fetch.html - Notepad++  
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?  
test-fetch.html  
1 <!html>  
2 <head>  
3 <script>  
4   fetch("https://jsonplaceholder.typicode.com/")  
5     .then(resp => console.log(resp));  
6 </script>  
7 </head>  
8 <body>  
9   test fetch  
10 </body>
```

(1/3) Edit Html code, save file

(2/3) open browser

(3/3) Open DevTools (F12)
open Console



using NodeJS (same Javascript interpreter V8 as Chrome/Edge)

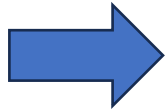


```
$ node
Welcome to Node.js v20.10.0.
Type ".help" for more information.
> fetch("https://jsonplaceholder.typicode.com/").then(resp => console.log(resp))
Promise {
  <pending>,
  [Symbol(async_id_symbol)]: 59,
  [Symbol(trigger_async_id_symbol)]: 30
}
> Response {
  [Symbol(realm)]: null,
  [Symbol(state)]: {
    aborted: false,
    rangeRequested: false,
    timingAllowPassed: true,
    requestIncludesCredentials: true,
    type: 'default',
    status: 200,
    timingInfo: {
      startTime: 4452.7895999999785,
```


Outline

Https → Load Html + Js + Css + Json + *

Js → Call Http : SPA, Rest + Json



Html → execute Js <script>

Html → trigger events to Js

Js → Query/Update Html DOM

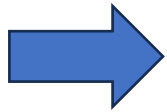
Js Frameworks, MVC, Bindings

Outline

Https → Load Html + Js + Css + Json + *

Js → Call Http : SPA, Rest + Json

Html → execute Js <script>



Html → trigger events to Js

Js → Query/Update Html DOM

Js Frameworks, MVC, Bindings

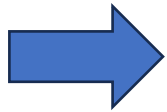
Outline

Https → Load Html + Js + Css + Json + *

Js → Call Http : SPA, Rest + Json

Html → execute Js <script>

Html → trigger events to Js



Js → Query/Update Html DOM

Js Frameworks, MVC, Bindings

Outline

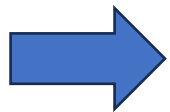
Https → Load Html + Js + Css + Json + *

Js → Call Http : SPA, Rest + Json

Html → execute Js <script>

Html → trigger events to Js

Js → Query/Update Html DOM



Js Frameworks, MVC, Bindings

