HTTP Protocol

HTTP

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

Questions?

Software University

sli.do

#java-web

Table of Contents

1. **Internet Protocol**
2. **DNS**
3. **TCP / UDP**
4. The **OSI Model**
5. **HTTP** Request
6. **HTTP** Response
7. URL-s
8. **MIME** and **Media types**
9. **HTTP** Tools
10. **HTTP/1, HTTP/2, HTTP/3**

216.58.214.46

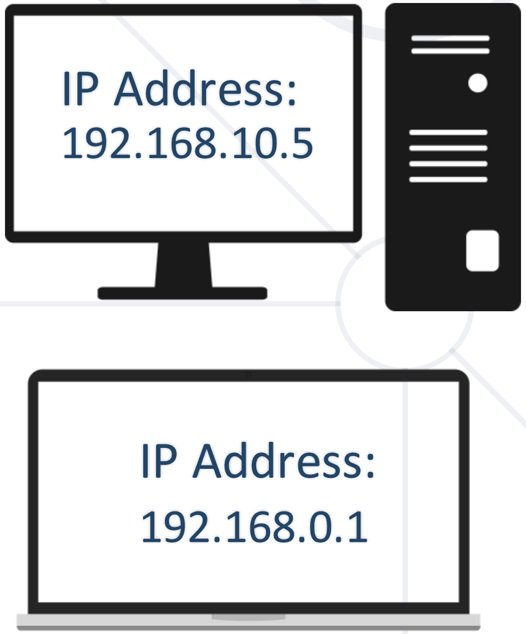
www.google.com

Internet Protocol

IPv4 and IPv6

Internet Protocol

- One of the most important protocols used in Internet communication is the **Internet Protocol** (**IP**)
- All the devices on the Internet have **addresses**
- They are called **IP Addresses**
- The IP address is **unique** to each computer or a device at the edge of the network



IPv4

- **IPv4** is a sequence of four, three-digit numbers separated by a period
  - Each number can be a number from 0 to 255
  - **IPv4** is not enough for all network devices connected to the internet
- In 1995, a new version of the internet protocol was created, it's called **IPv6**

IP Address

- An **IP Address** has many parts, organized in a hierarchy

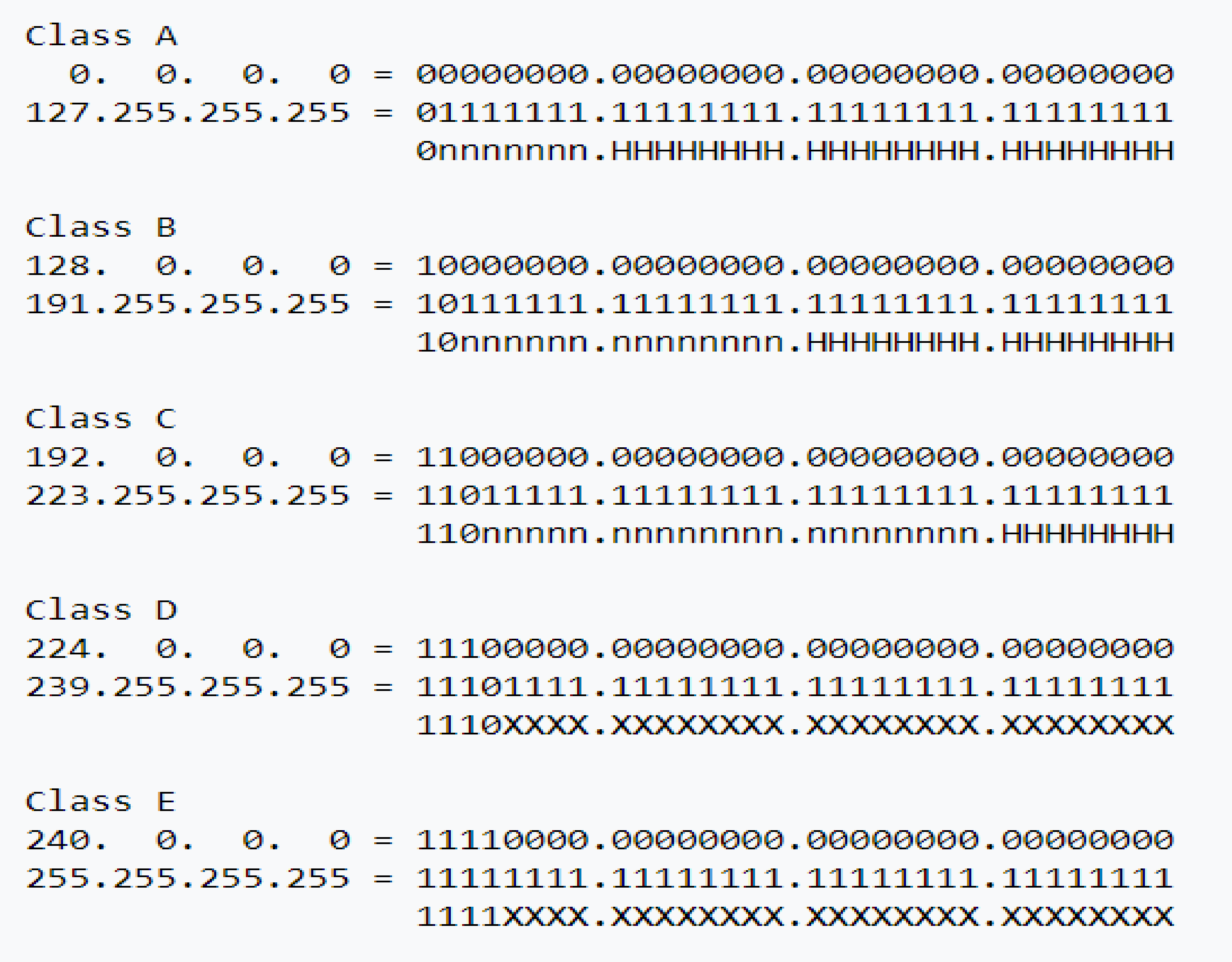
Subnetworks

192.168.14.120

Device address

- This version of IP Addressing is called **IPv4**
  - Provides more than 4 billion **32 bits** unique addresses

IP address classes



```
Class A
  0.  0.  0.  0 = 00000000.00000000.00000000.00000000
127.255.255.255 = 01111111.11111111.11111111.11111111
                  0nnnnnnn.HHHHHHHH.HHHHHHHH.HHHHHHHH

Class B
128.  0.  0.  0 = 10000000.00000000.00000000.00000000
191.255.255.255 = 10111111.11111111.11111111.11111111
                  10nnnnnn.nnnnnnnn.HHHHHHHH.HHHHHHHH

Class C
192.  0.  0.  0 = 11000000.00000000.00000000.00000000
223.255.255.255 = 11011111.11111111.11111111.11111111
                  110nnnnn.nnnnnnnn.nnnnnnnn.HHHHHHHH

Class D
224.  0.  0.  0 = 11100000.00000000.00000000.00000000
239.255.255.255 = 11101111.11111111.11111111.11111111
                  1110XXXX.XXXXXXXX.XXXXXXXX.XXXXXXXX

Class E
240.  0.  0.  0 = 11110000.00000000.00000000.00000000
255.255.255.255 = 11111111.11111111.11111111.11111111
                  1111XXXX.XXXXXXXX.XXXXXXXX.XXXXXXXX
```

What Is CIDR (Classless Inter-Domain Routing)

- Classless Inter-Domain Routing, is an IP addressing scheme that improves the allocation of IP addresses.
- It replaces the old system based on classes A, B, and C.
- This scheme also helped greatly **extend the life of IPv4** as well as slow the growth of routing tables

IPv4 Private Address Space and Filtering

- **IPv4** private address space refers to a **range** of IP addresses reserved for use within private networks. These addresses are **not** routable on the public internet, meaning routers on the internet will not forward packets with these **addresses**. Instead, they are meant for use within local networks, such as home, office, or enterprise networks

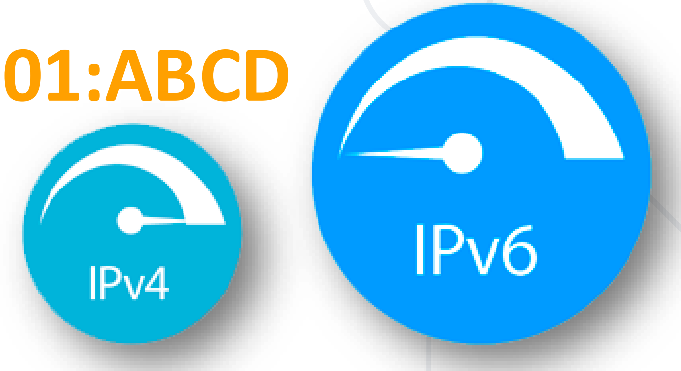
CIDR	IP address range	Class
10.0. 0.0/8	10.0. 0.0 – 10.255. 255.255	A
172.16. 0.0/12	172.16. 0.0 – 172.31. 255.255	B
192.168. 0.0/16	192.168. 0.0 – 192.168. 255.255	C

IPv4 Private Address Space and Filtering

- The **three** blocks of **IPv4** private address space are:
  - **10.0.0.0/8:** This block includes all IP addresses from 10.0.0.0 to **10.255.255.255** and is often used for large networks, such as **corporate** intranets
  - **172.16.0.0/12**: This block includes all IP addresses from 172.16.0.0 to **172.31.255.255**. It is commonly used by **medium-sized** networks
  - **192.168.0.0/16:** This block includes all IP addresses from 192.168.0.0 to **192.168.255.255** and is typically used for small **home** or **office** networks

IPv6

- **IPV6** uses **128 bits** - 340 undecillion unique addresses
  - That's more than the atoms on the surface of the Earth
- These **128** bits are organized into eight 16 bit sections
- Each 16 bit block is converted to hexadecimal and it's separated with a colon
- This is a full IPV6 address:
  - **3FFE:F200:0234:AB00:0123:4567:8901:ABCD**
- The **leading zeros** in **IPv6** can usually be left out



DNS (Domain Name System)

What is a DNS?

`www.softuni.bg`

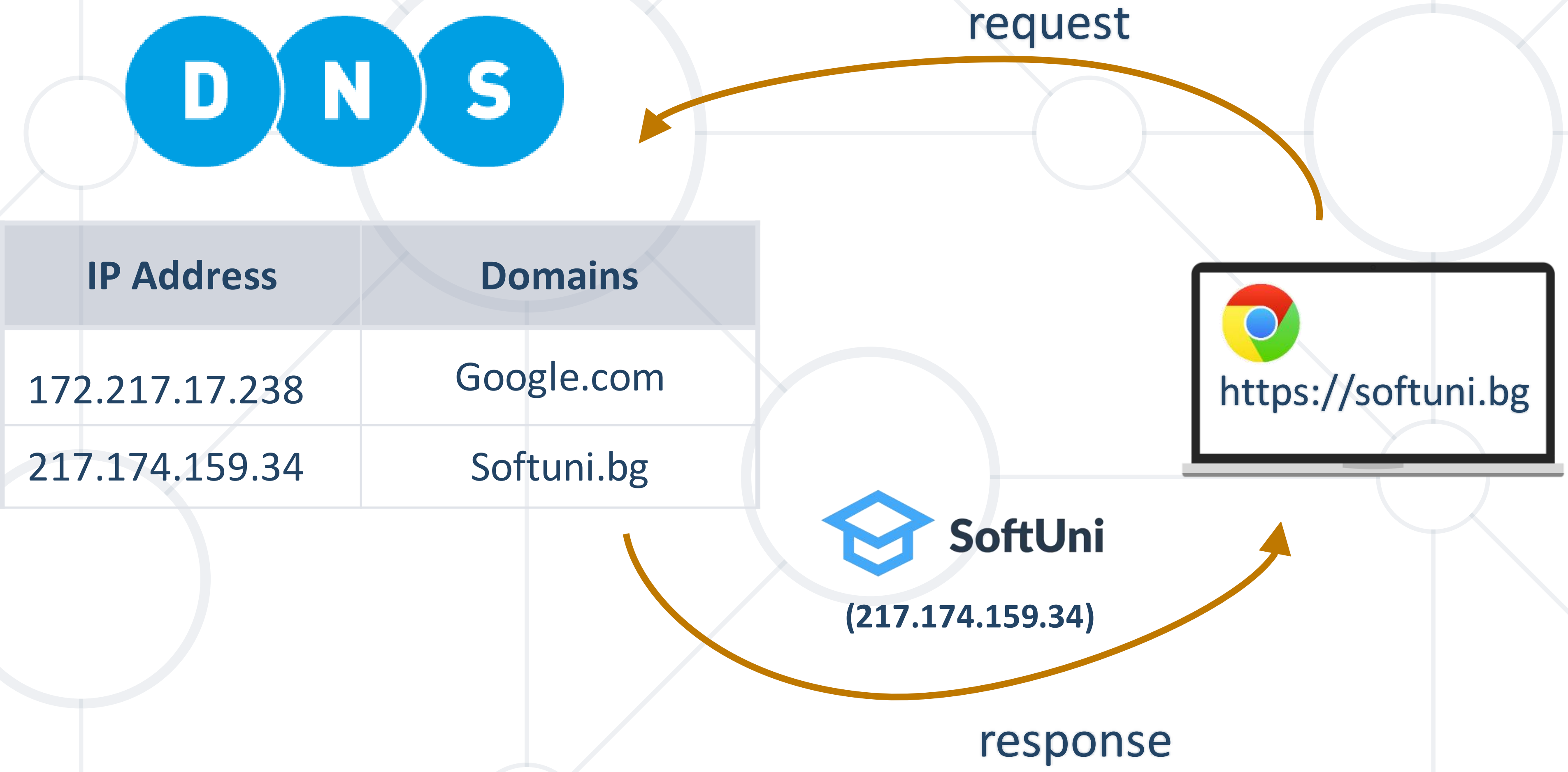
Domain name

- The **domain name** is a human way to access IP addresses for devices and websites around the world
- It is a sequence of phrases that **map** to a giant **Internet-wide database** of **IP addresses**
- When a domain name is entered in the browser, a request is made to something called a **DNS** (**Domain Name Server**)
- This server holds a cache of tons of domain names, and their matching IP addresses

Key points about DNS

- **Domain Names:** DNS provides a **hierarchical** naming structure, where domain names are organized in a **tree-like** hierarchy
- **DNS Servers:** DNS relies on a network of DNS servers that **store** and **manage** DNS records. These servers include recursive DNS resolvers, authoritative DNS servers, and root DNS servers
- **Types of DNS Records:** DNS records are used to **store** various types of information associated with domain names, including IP addresses (A records), mail server addresses (MX records), aliases (CNAME records), and more

DNS Example



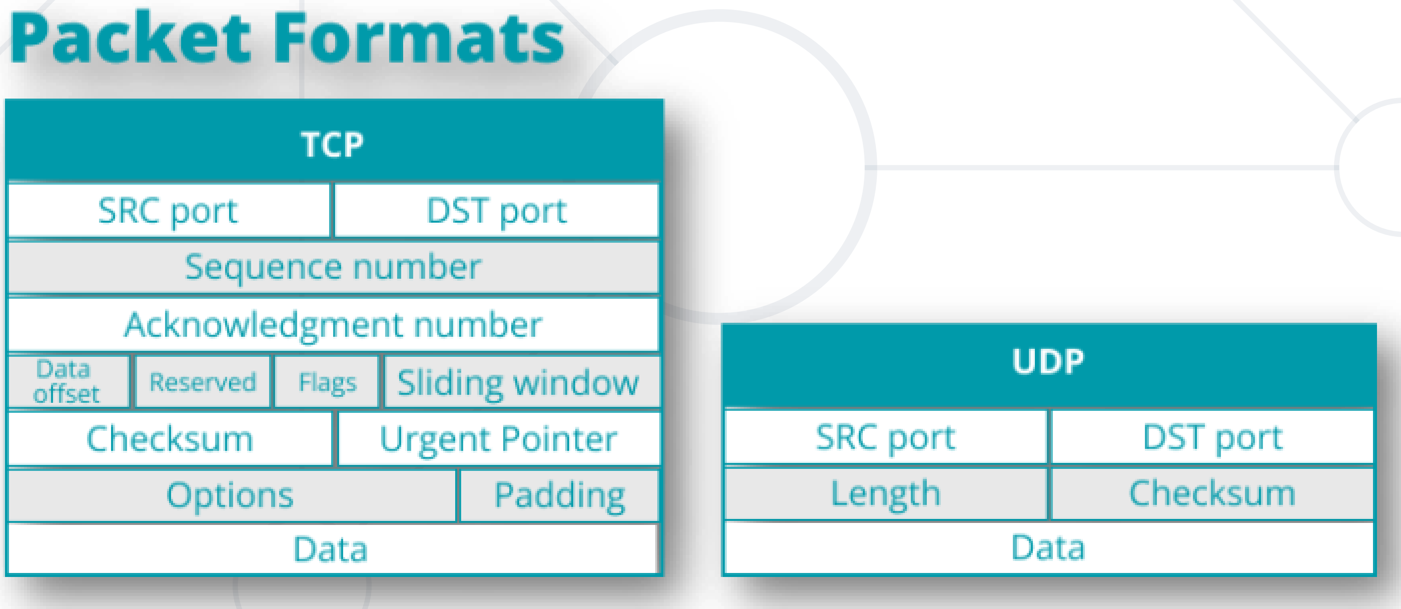
IP Address	Domains
172.217.17.238	Google.com
217.174.159.34	Softuni.bg

TCP

TCP / UDP

What is a Packet?

- A **unit** of data that is transmitted over a network
- Both **TCP** and **UDP** operate at the **transport layer** of the **OSI** (Open Systems Interconnection) model and are responsible for ensuring the **reliable** delivery of data between hosts on a network



Packet Formats

TCP

SRC port | DST port

Sequence number

Acknowledgment number

Data offset | Reserved | Flags | Sliding window

Checksum | Urgent Pointer

Options | Padding

Data

UDP

SRC port | DST port

Length | Checksum

Data

TCP Packet

- In TCP, data is transmitted in segments rather than packets. Each segment contains a **header** and **payload**
- The TCP header includes control information such as **source** and **destination** port numbers, sequence numbers, acknowledgement numbers, and
- The payload contains the **actual** data being transmitted, such as a segment of a file, a web page, or an email message
- TCP segments are reassembled into a **complete message** at the receiving end based on sequence numbers and acknowledgement messages exchanged between the sender and receiver

UDP Datagram

- In UDP, data is transmitted in **datagrams**, which are similar to packets.
- A UDP datagram consists of a **header** and **payload**
- The UDP header contains **source** and **destination** port numbers and the length of the datagram
- Unlike TCP, UDP does **not** provide mechanisms for ensuring reliable delivery, error correction, or flow control. Therefore, UDP is considered a **connectionless** protocol, and datagrams may be lost, duplicated, or delivered out of order
- UDP is often used for **real-time applications** such as voice over IP (VoIP), online gaming, and streaming media, where low latency and high throughput are more important than reliability

Reliability

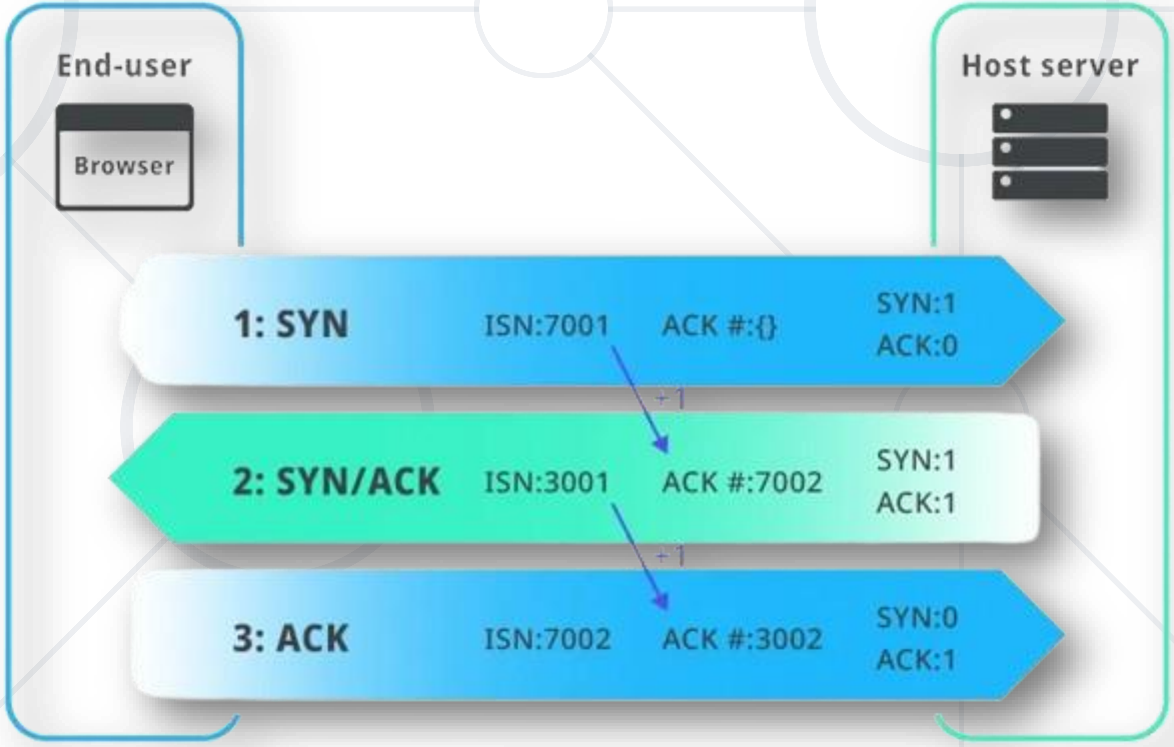
- When packets are transmitted from one location to another, they can take different paths
- When they get to the destination, they are unorganized and sometimes not complete
- So the message needs to be audited and reviewed in order to put it together in the right way
- The **Transmission Control Protocol** or **TCP** does exactly that

Transmission Control Protocol - TCP

- **TCP** uses a process, where it looks at **all the packets** in a message and **checks them**
- TCP is a **connection-based** protocol
- Using the header information in each packet, it knows
  - How many there are
  - How large they should be
  - In which order the packets should be in
- Using this checklist, it is able to rearrange the packets

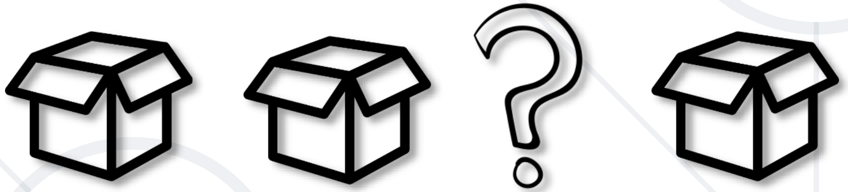
TCP 3-way Handshake

- The **3-way handshake** is a method used to establish a connection between a client and a server. It consists of three steps:
  - **SYN:** The client sends a SYN packet to the server
  - **SYN-ACK:** the server responds with a SYN-ACK packet
  - **ACK:** the client acknowledges the server's SYN-ACK packet by sending an ACK packet



Transmission Control Protocol - TCP

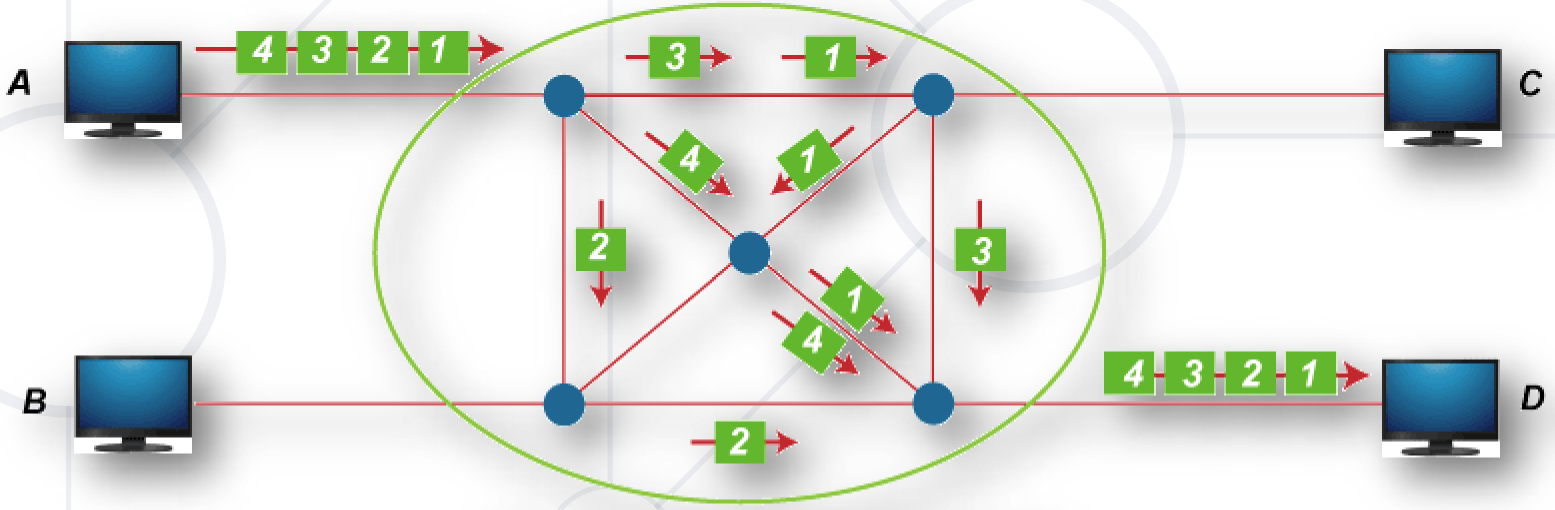
- If it finds that a packet doesn't match the expected characteristic, it is discarded
- **TCP verifies** that all the packets are
  - In the right order
  - Free of any issues



- After that it **certifies the data** and the packets are **merged** together to recreate the **original** file that was on the sender's device

Packet switching

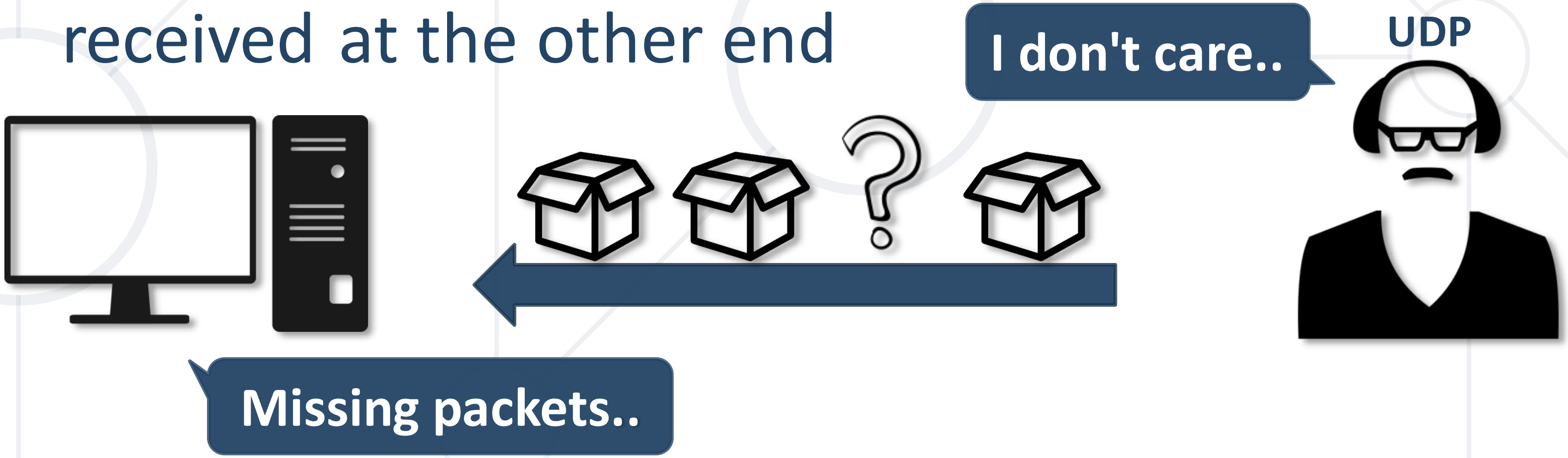
- **Packet switching:** is a method used in computer networking to **transmit** data across a network
- It enables efficient and flexible data transmission by **breaking** data into small **packets**, routing them dynamically through the network, and allowing for rerouting in case of congestion or failure



Packet Switched Network

User Datagram Protocol

- UDP does not establish a session and it does not guarantee data delivery
  - UDP is **connectionless**
- It is known as the **"fire-and-forget" protocol**
  - It sends data and it doesn't really care if the data is received at the other end



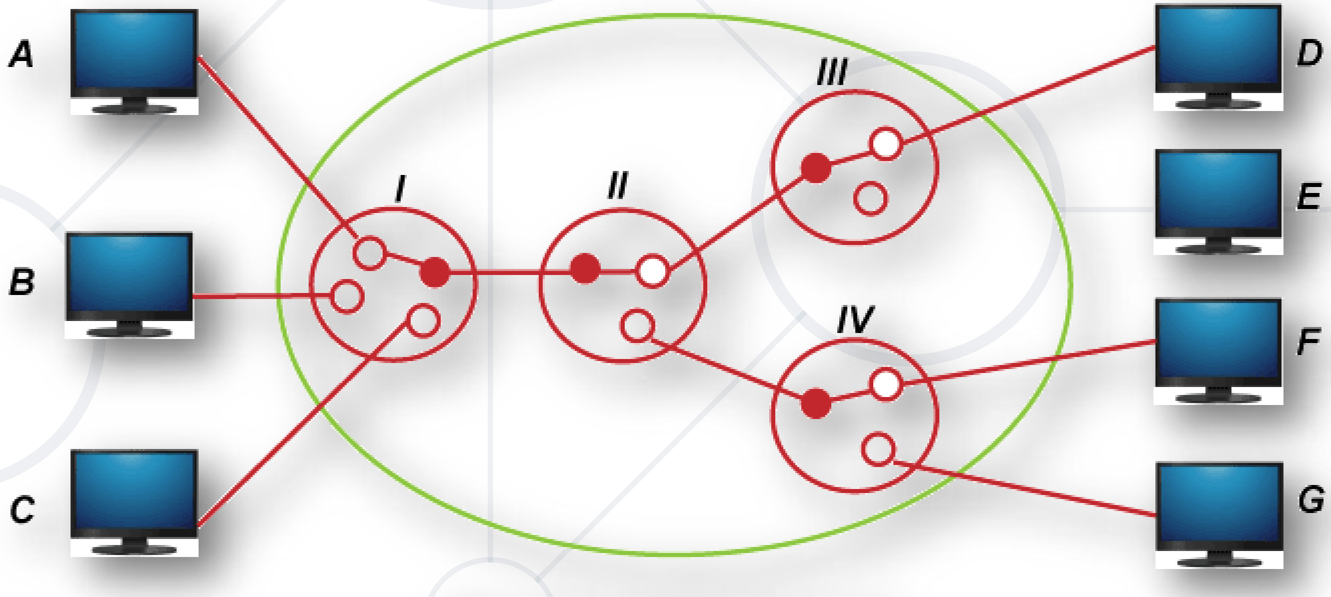
I don't care..

UDP

Missing packets..

Circuit Switching

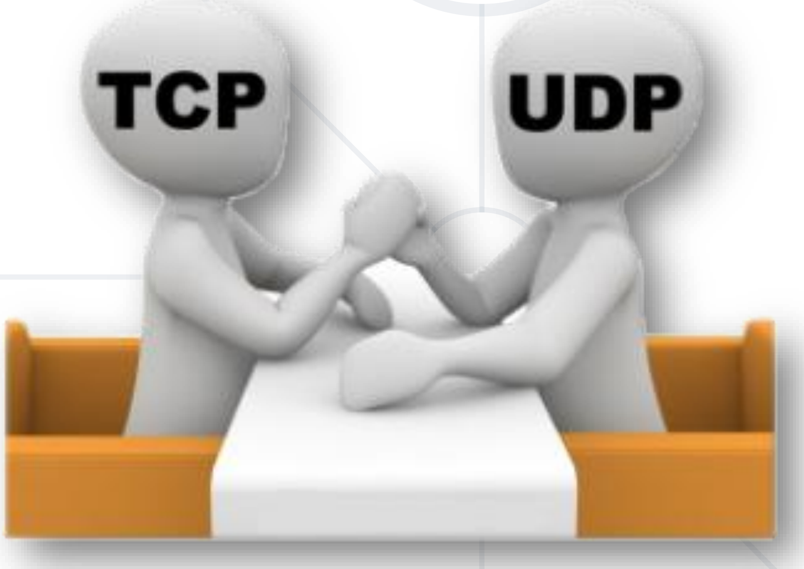
- **Circuit Switching:** provides a dedicated and uninterrupted communication path between two parties for the duration of a session, allocating fixed resources along the entire route
- The data we send **always** reach desired destination in order



Circuit Switched Network

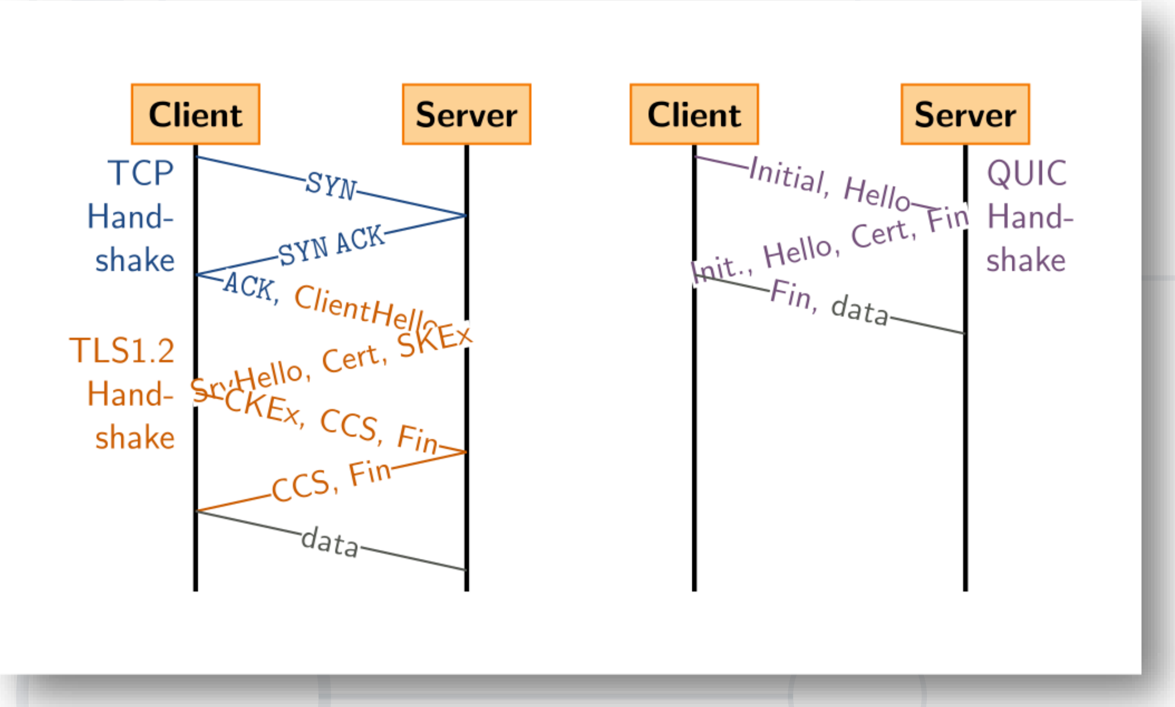
TCP vs UDP

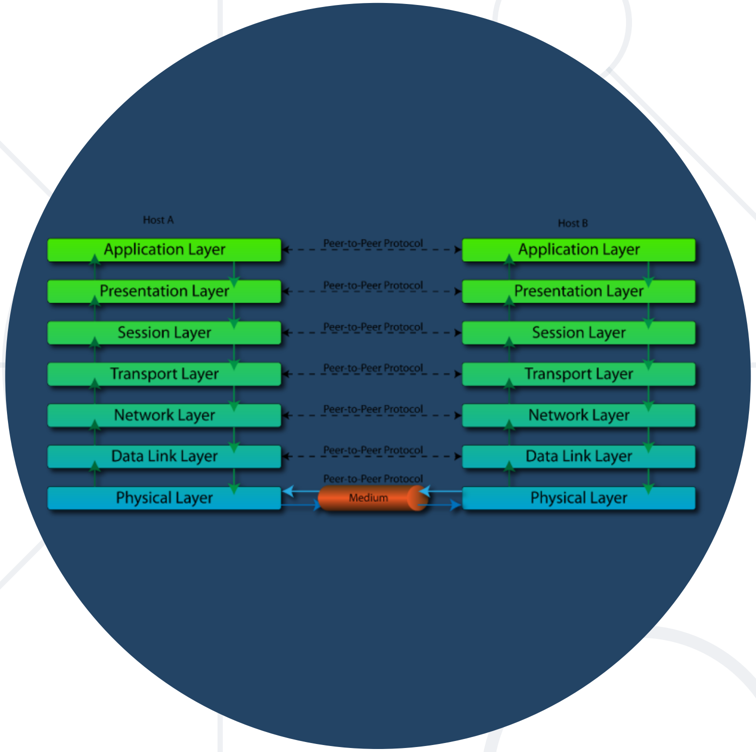
- **TCP** places **reliability** in a higher priority than speed or latency
- For instances where reliability isn't as important, but **speed** is, **UDP** is used
- UDP doesn't do excessive reliability checks, but it can send information at a faster rate
- TCP is the foundation of how a majority of data is transmitted over networks



QUIC Protocol

- **QUIC** == new transport protocol designed for **mobile-heavy** Internet usage
- Uses **UDP** as its basis, not TCP
- Packets are encrypted **individually**
- Exchange of supported protocols is a part of the initial **handshake process**





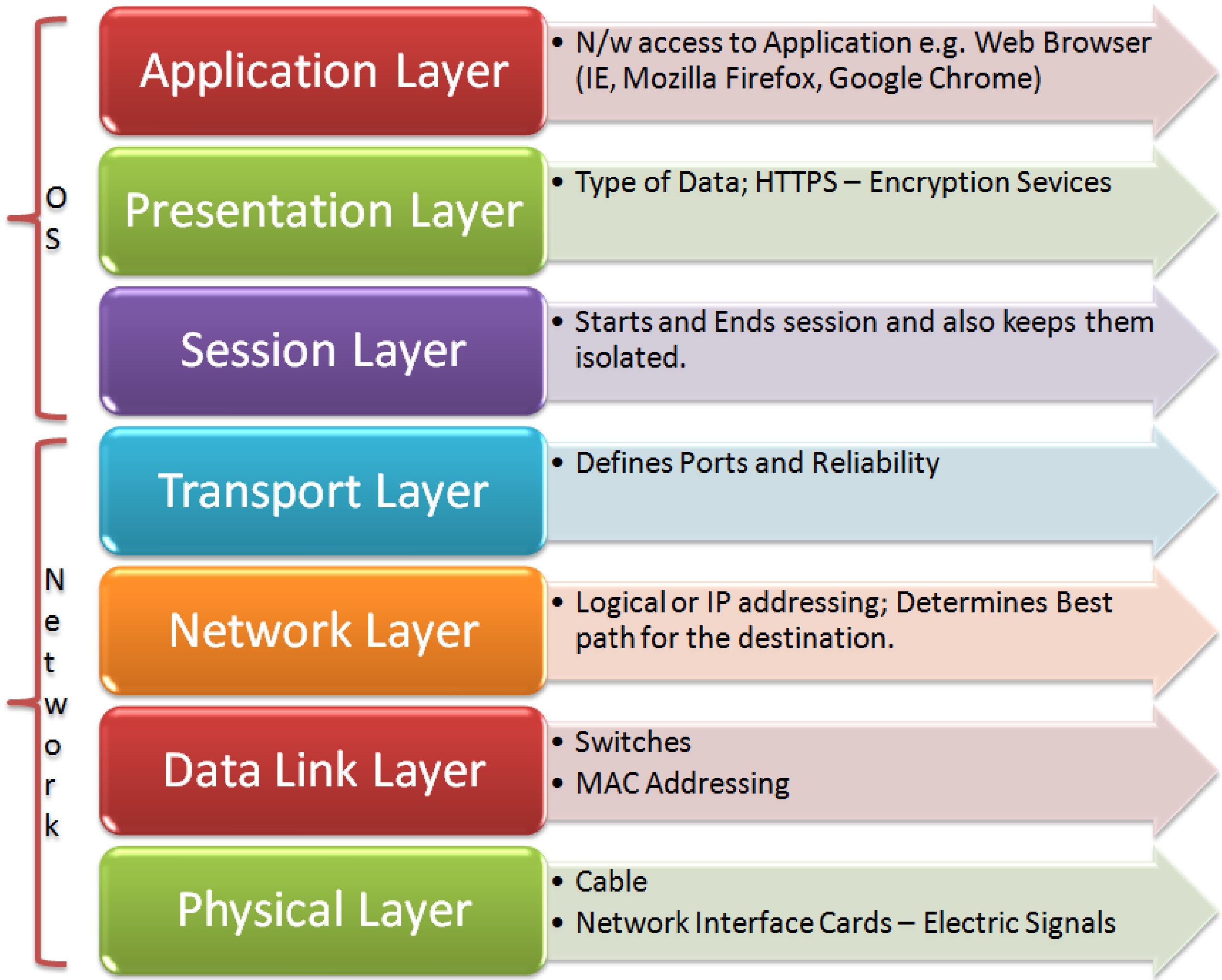
The OSI Model

What is the OSI Model?

- OSI model stands for Open System Interconnect
- It consists of 7 layers
  - Each layer serves the layer above it and in return, is served by the layer below it
- Understanding each layer of the model helps us with:
  - Troubleshooting
  - Communicating better with technical and non-technical individuals about any system

OSI Layers

- OSI Model consists of 7 layers:

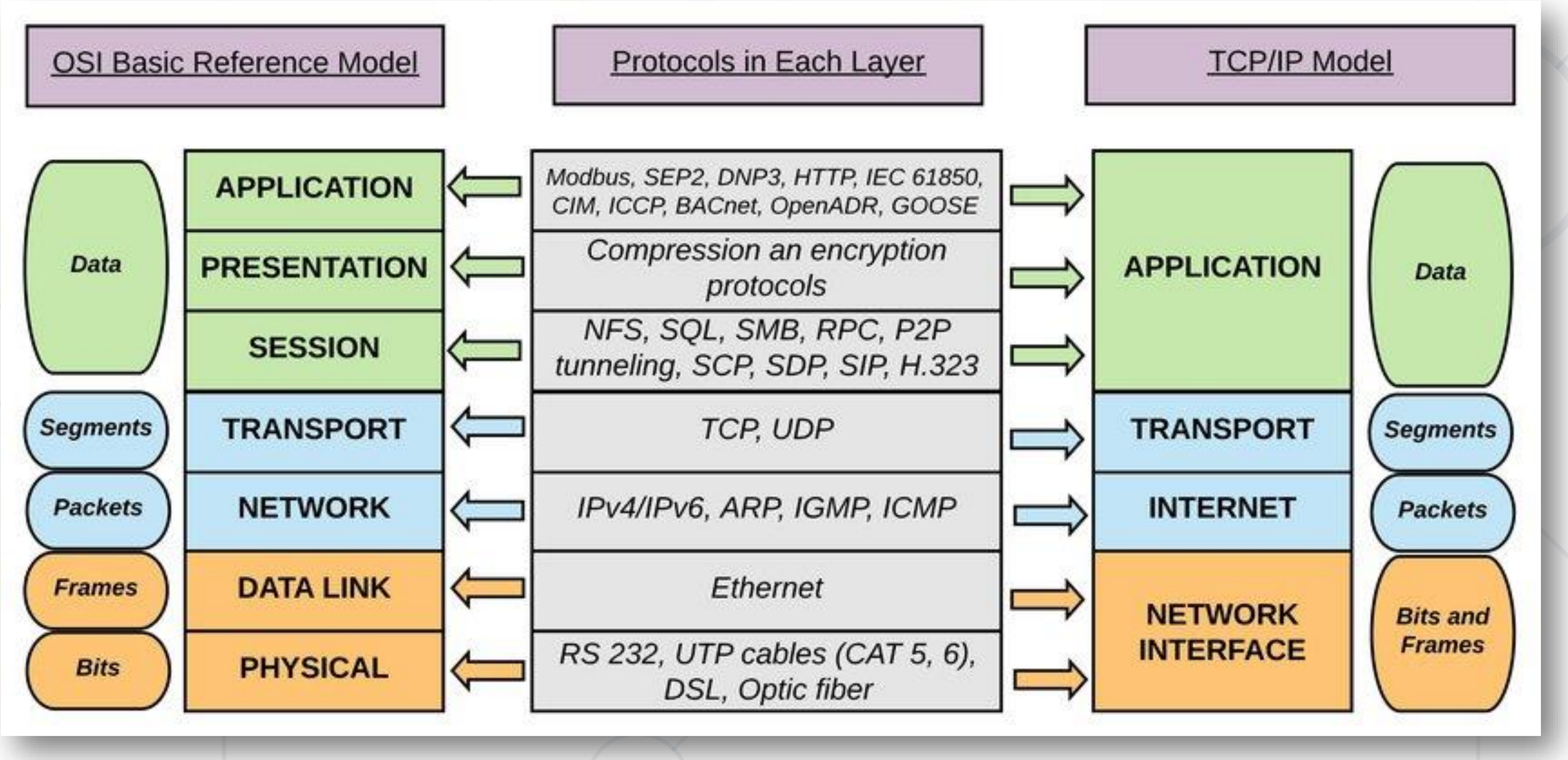


OS

Network

		Example Protocols
Application Layer	• N/w access to Application e.g. Web Browser (IE, Mozilla Firefox, Google Chrome)	HTTP, DNS, FTP, SMTP
Presentation Layer	• Type of Data; HTTPS – Encryption Sevices	TLS, SSL, compression
Session Layer	• Starts and Ends session and also keeps them isolated.	NetBIOS, PPTP, Sockets
Transport Layer	• Defines Ports and Reliability	TCP, UDP
Network Layer	• Logical or IP addressing; Determines Best path for the destination.	IP, IPsec
Data Link Layer	• Switches • MAC Addressing	ATM, Ethernet, MAC, LLC
Physical Layer	• Cable • Network Interface Cards – Electric Signals	USB, Bluetooth, 802.11a/b/g/n

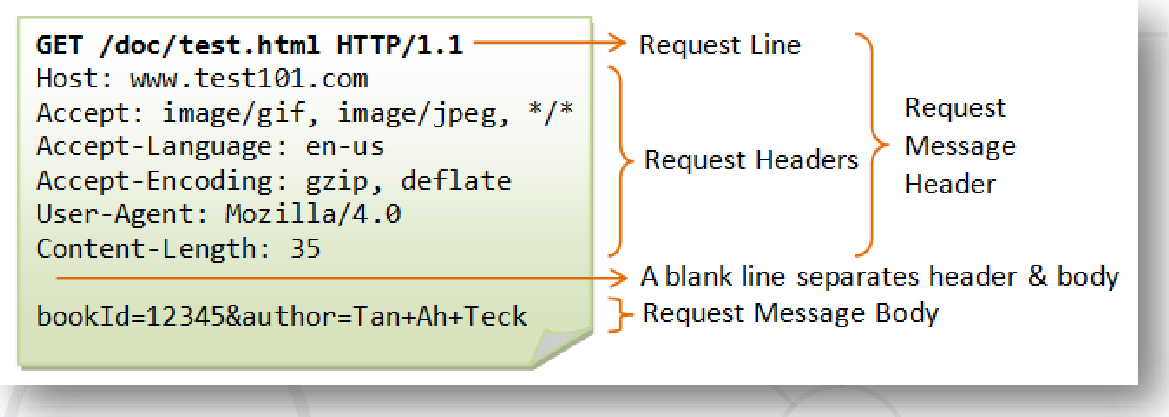
TCP/IP model mapping to OSI



	OSI Basic Reference Model	Protocols in Each Layer	TCP/IP Model	
Data	APPLICATION	*Modbus, SEP2, DNP3, HTTP, IEC 61850, CIM, ICCP, BACnet, OpenADR, GOOSE*	APPLICATION	*Data*
	PRESENTATION	*Compression an encryption protocols*		
	SESSION	*NFS, SQL, SMB, RPC, P2P tunneling, SCP, SDP, SIP, H.323*		
Segments	TRANSPORT	*TCP, UDP*	TRANSPORT	*Segments*
Packets	NETWORK	*IPv4/IPv6, ARP, IGMP, ICMP*	INTERNET	*Packets*
Frames	DATA LINK	*Ethernet*	NETWORK INTERFACE	*Bits and Frames*
Bits	PHYSICAL	*RS 232, UTP cables (CAT 5, 6), DSL, Optic fiber*		

Application Layer – 7

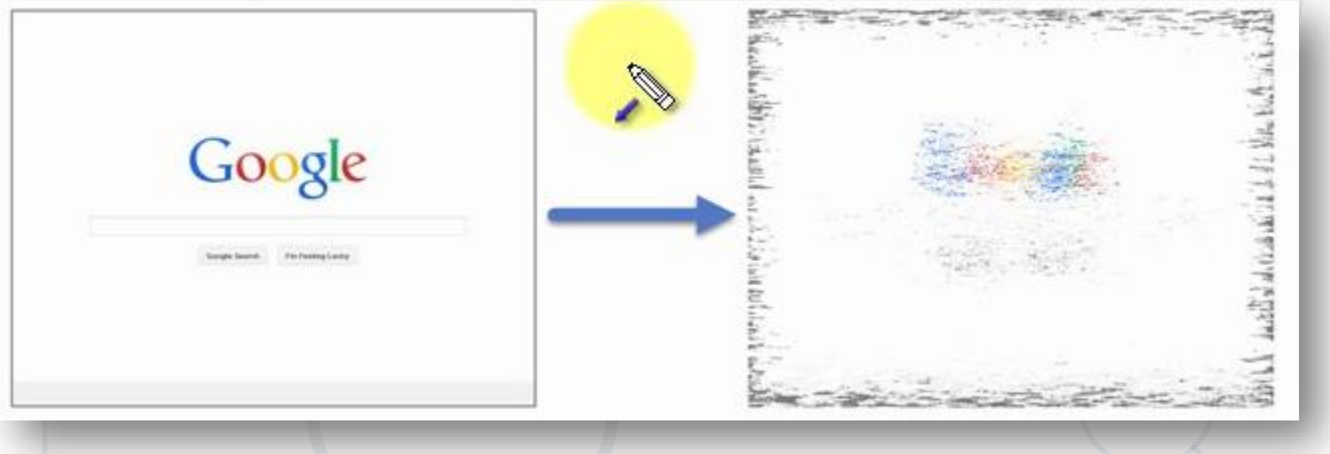
- Enables different applications like the browser to use the network and present it to the End User
- Protocol examples:
  - **Domain Name System (DNS)**
  - **File Transfer Protocol (FTP)**
  - **HyperText Transfer Protocol (HTTP)**
  - **Simple Mail Transfer Protocol (SMTP)**



```
GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35

bookId=12345&author=Tan+Ah+Teck
```

Presentation Layer – 6

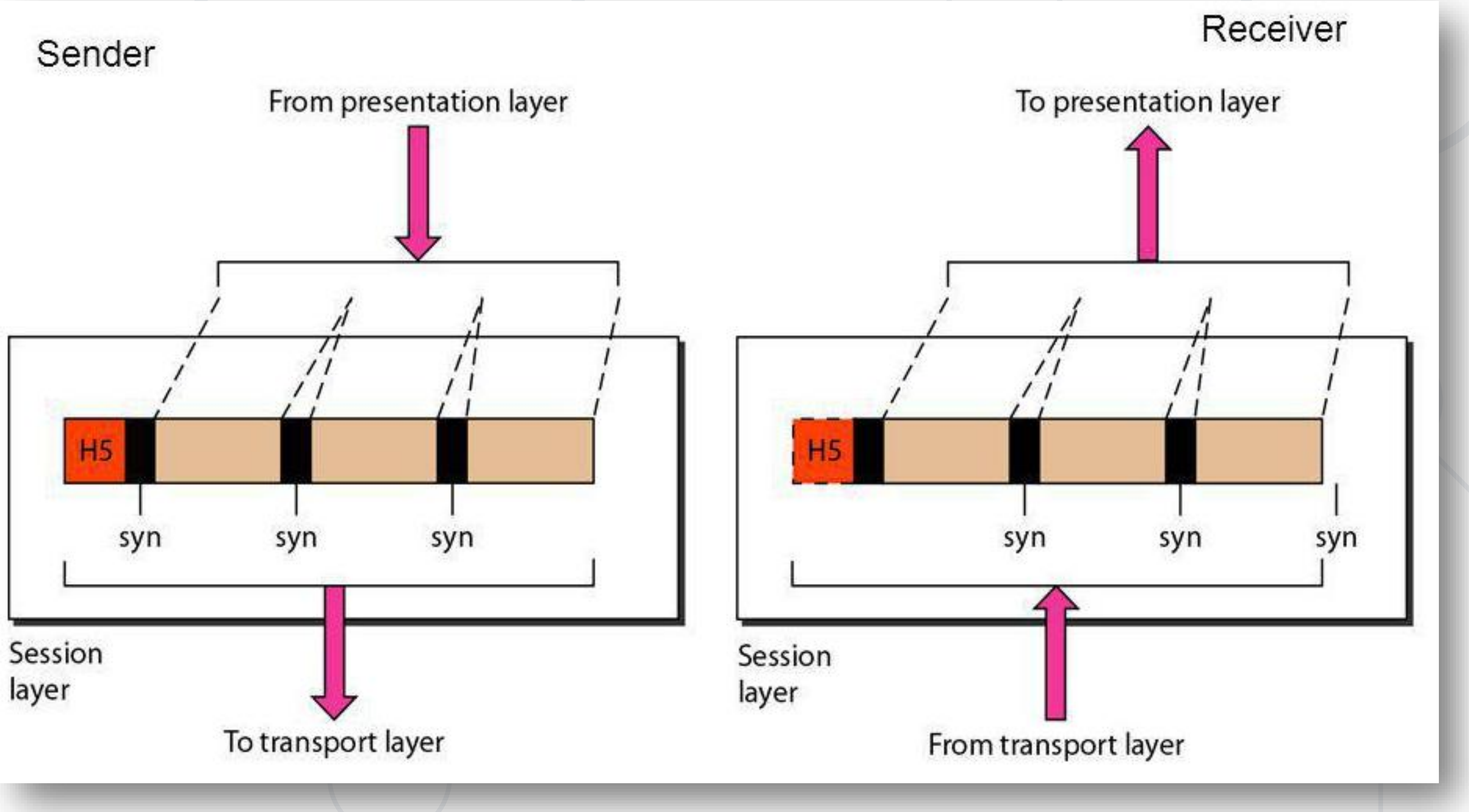
- This layer is a part of an operating system (OS)
- **Converts** incoming and outgoing **data** from one presentation format to another



- Example:
  - From clear text to encrypted (or compressed) text
  - Back to clear text

Session Layer – 5

- This layer sets up coordinates and terminates conversations
- Its services include authentication and reconnection after an interruption
- e.g.: Sockets

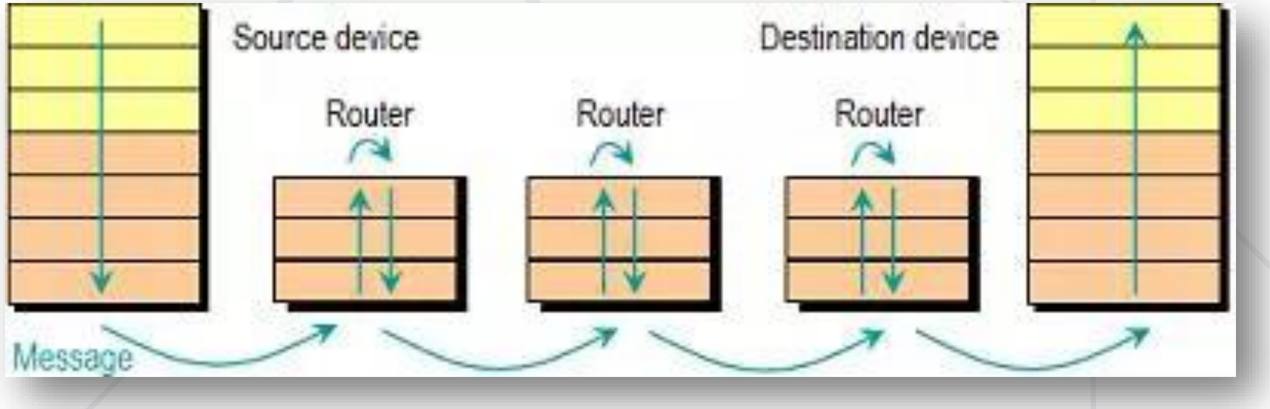


Transport Layer – 4

- Responsible for end-to-end communication over a network
- Provides logical communication between application processes
- Responsible for the management of error correction, providing quality and reliability to the end user
- Protocol examples:
  - **Transmission Control Protocol (TCP)**
  - **User Datagram Protocol (UDP)**

Network Layer – 3

- Provides the functional and procedural means of transferring packets from one node to another
- Responds to service requests from the transport layer and issues service requests to the data link layer
- Protocol examples:
  - **Internet Protocol (IP)**
  - **IPSec (IP + Auth)**

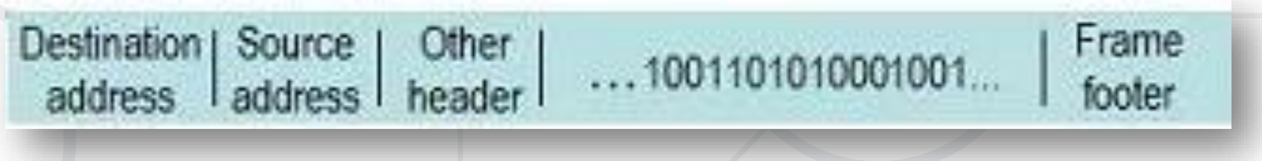


Data Link Layer – 2

- Provides node-to-node data transfer
- It **detects** and possibly **corrects** errors that may occur in the **physical layer**
- Divides into two sublayers:
  - **Medium access control (MAC)** layer - controlling how devices in a network gain access to a medium and permission to transmit data
  - **Logical link control (LLC)** layer – identifying and encapsulating network layer protocols, controls error checking and frame synchronization

Data Link Layer – 2

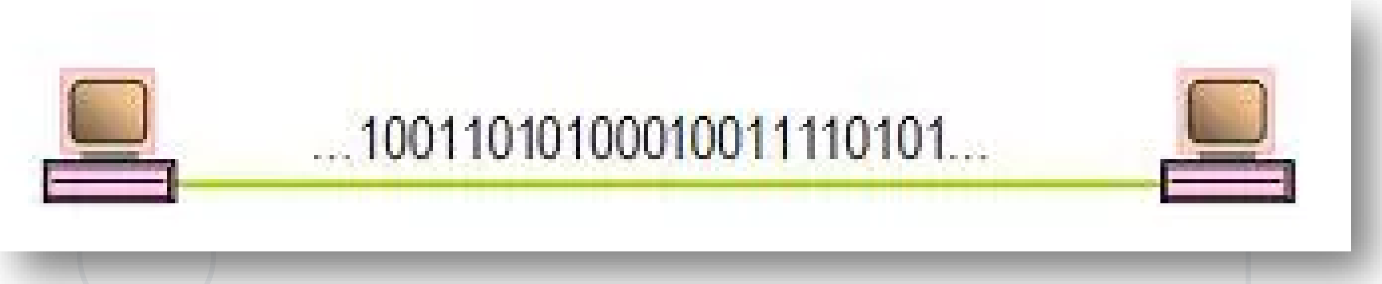
- Protocol examples:
  - **Asynchronous Transfer Mode (ATM)**
  - **Ethernet**
  - **MAC**



Destination address	Source address	Other header	...1001101010001001...	Frame footer

Physical Layer – 1

- The things you can actually physically touch
- Converts the **binary** from the upper layers into **signals**, **transmits** them over local media (electrical, light, or radio signals)
- Examples:
  - **Ethernet**
  - **USB**
  - **Bluetooth**
  - **802.11a/b/g/n**



HTTP Request

What is a HTTP Request?

HTTP Request Message

- Request message sent by a client consists of:
  - HTTP **request line**
    - Request method (**GET** / **POST** / **PUT** / **DELETE** / …)
    - Resource URI (**URL**)
    - Protocol version
  - HTTP **request headers**
    - Additional parameters
  - HTTP **request body** – optional data, e.g. posted form fields

```
<method> <resource> HTTP/<version>
<headers>
(empty line)
<body>
```

HTTP GET Request – Example

- Example of HTTP **GET** request:

```
GET /index.html HTTP/1.1
Host: localhost
<CRLF>
```

HTTP request line

HTTP request headers

The request body is empty

HTTP POST Request – Example

- Example of HTTP **POST** request:

```
POST /login.html HTTP/1.1
Host: localhost
Content-Length: 59
Content-Type: application/x-www-form-urlencoded
<CRLF>
username=testUser&password=topSecret
<CRLF>
```

HTTP request line

HTTP request headers

The request body holds the submitted form data

HTTP Response

What is a HTTP Response?

HTTP Response Message

- The **response message** sent by the HTTP server consists of:
  - HTTP response **status line**
    - Protocol version
    - Status code
    - Status phrase
  - Response **headers**
    - Provide meta data about the returned resource
  - Response **body**
    - The content of the HTTP response (data)

```
HTTP/<version> <status code> <status text>
<headers>
(empty line)
<response body – the requested resource>
```

HTTP Response – Example

- Example of HTTP **response** from the Web server:

```
HTTP/1.1 200 OK
Date: Fri, 17 Jul 2020 16:09:18 GMT+2
Server: Apache/2.2.14 (Linux)
Accept-Ranges: bytes
Content-Length: 84
Content-Type: text/html
<CRLF>
<html>
  <head><title>Test</title></head>
  <body>Test HTML page.</body>
</html>
```

HTTP response status line

HTTP response headers

HTTP response body

HTTP Response Codes

- HTTP response code classes
  - **1xx**: informational (e.g., "**100** Continue")
  - **2xx**: successful (e.g., "**200** OK", "**201** Created")
  - **3xx**: redirection (e.g., "**304** Not Modified", "**301** Moved Permanently", "**302** Found")
  - **4xx**: client error (e.g., "**400** Bad Request", "**404** Not Found", "**401** Unauthorized", "**409** Conflict")
  - **5xx**: server error (e.g., "**500** Internal Server Error", "**503** Service Unavailable")

HTTP Error Response – Example

- Example of **HTTP response** with error result:

```
HTTP/1.1 404 Not Found

Date: Fri, 17 Nov 2020 16:09:18 GMT+2
Server: Apache/2.2.14 (Linux)
Connection: close
Content-Type: text/html
<CRLF>

<html><head><title>404 Not Found</title></head>
<body>
<h1>Not Found</h1>
<p>The requested URL /img/logo.gif was not found on this server.</p>
<hr><address>Apache/2.2.14 Server at Port 80</address>
</body></html>
```

HTTP response status line

HTTP response headers

The HTTP response body

Browser Redirection

- HTTP **GET** requesting a moved URL:

```
GET / HTTP/1.1
Host: http://softuni.org
User-Agent: Gecko/20100115 Firefox/3.6
<CRLF>
```

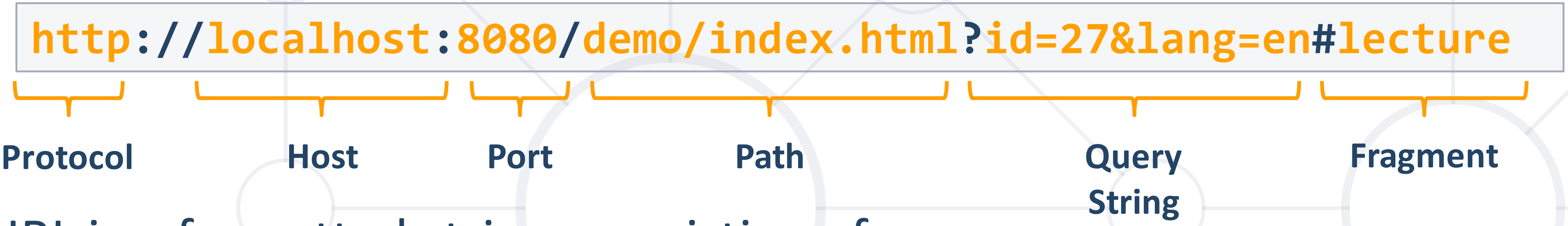
- The following HTTP response (**301** Moved Permanently) tells the browser to request another URL:

```
HTTP/1.1 301 Moved Permanently
Location: http://softuni.bg
…
```

URL

Uniform Resource Locator

Uniform Resource Locator (URL)



```
http://localhost:8080/demo/index.html?id=27&lang=en#lecture
```

- URL is a formatted string, consisting of:
  - Protocol for communicating (**http**, **ftp**, **https**...) – HTTP in most cases
  - Host or IP address (**www.softuni.bg**, **gmail.com**, **127.0.0.1**, **web**)
  - Port (the default port is **80**) – a number in range [0…65535]
  - Path (**/forum**, **/path/index.html**)
  - Query string (**?id=27&lang=en**)
  - Fragment (**#lectures**) – used on the client to navigate to some section

URL Encoding

- URLs are encoded according RFC 1738:
  - Safe URL characters: **[0-9a-zA-Z]**, **$**, **-**, **_**, **.** , **+**, *****, **'**, **(**, **)**, **,**, **!**
- All other characters are escaped by:

```
%[character hex code]
```

  - Space is encoded as "**+**" or "**%20**"

```
Наков-爱-SoftUni
```

  - URL-encoded string:

```
%D0%9D%D0%B0%D0%BA%D0%BE%D0%B2-%E7%88%B1-SoftUni
```

Char	URL Encoding
space	%20
щ	%D1%89
"	%22
#	%23
$	%24
%	%25
&	%26

Valid and Invalid URLs – Examples

- Some valid URLs:

```
http://www.google.bg/search?sourceid=navclient&ie=UTF-
8&rlz=1T4GGLL_enBG369BG369&q=http+get+vs+post
```

```
http://bg.wikipedia.org/wiki/%D0%A1%D0%BE%D1%84%D1%82%D1%83%D0%B5%
D1%80%D0%BD%D0%B0_%D0%B0%D0%BA%D0%B0%D0%B4%D0%B5%D0%BC%D0%B8%D1%8F
```

- Some invalid URLs:

Should be:
`?q=C%23+.NET+4.0`

```
http://www.google.bg/search?&q=C# .NET 4.0
```

```
http://www.google.bg/search?&q=бира
```

Should be: `?q=%D0%B1 %D0%B8%D1%80%D0%B0`

MIME and Media Types

Multi-Purpose Internet Mail Extensions

What is MIME?

- MIME == Multi-Purpose Internet Mail Extensions
  - Internet standard for encoding resources
  - Originally developed for email attachments
  - Used in many Internet protocols like HTTP and SMTP
- MIME defines several concepts
  - **Content-Type**, e.g. **text/html**, **image/gif**, **application/pdf**
    - Content **charset**, e.g. **utf-8**, **ascii**, **windows-1251**
  - **Content-Disposition**, e.g. **attachment; filename=logo.jpg**
  - Multipart messages (multiple resources in a single document)

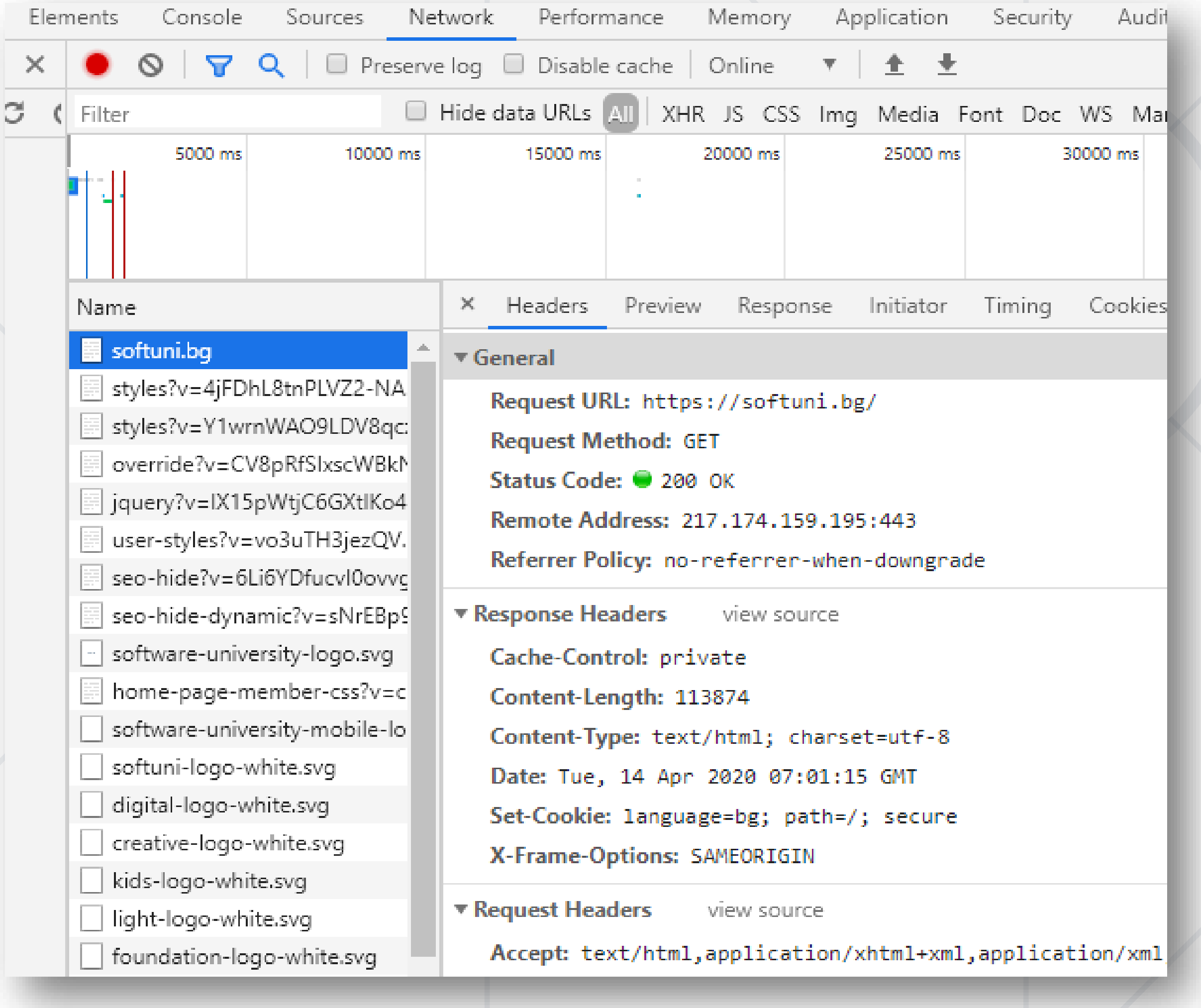
Common MIME Media Types

MIME Type / Subtype	Description
application/json	JSON data
image/png	PNG image
image/gif	GIF image
text/html	HTML
text/plain	Text
text/xml	XML
video/mp4	MP4 video
application/pdf	PDF document

HTTP Tools

Tools for Developers

HTTP Tools for Developers – Browser



softuni.bg
styles?v=4jFDhL8tnPLVZ2-NA
styles?v=Y1wrnWAO9LDV8qc
override?v=CV8pRfSlxscWBkN
jquery?v=lX15pWtjC6GXtlKo4
user-styles?v=vo3uTH3jezQV.
seo-hide?v=6Li6YDfucvl0ovvg
seo-hide-dynamic?v=sNrEBp9
software-university-logo.svg
home-page-member-css?v=c
software-university-mobile-lo
softuni-logo-white.svg
digital-logo-white.svg
creative-logo-white.svg
kids-logo-white.svg
light-logo-white.svg
foundation-logo-white.svg

▼ General

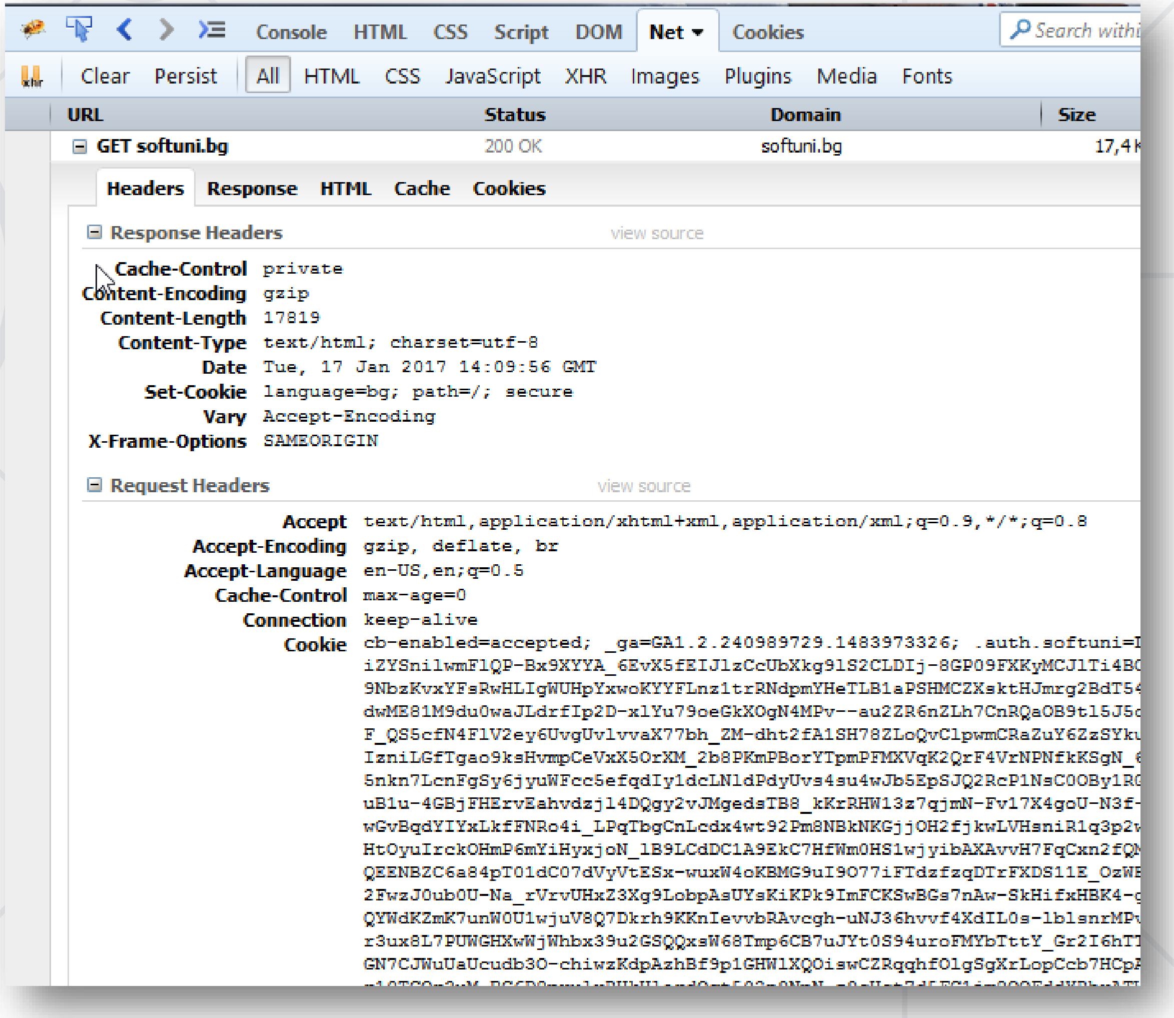
```
Request URL: https://softuni.bg/
Request Method: GET
Status Code: ● 200 OK
Remote Address: 217.174.159.195:443
Referrer Policy: no-referrer-when-downgrade
```

▼ Response Headers view source

```
Cache-Control: private
Content-Length: 113874
Content-Type: text/html; charset=utf-8
Date: Tue, 14 Apr 2020 07:01:15 GMT
Set-Cookie: language=bg; path=/; secure
X-Frame-Options: SAMEORIGIN
```

▼ Request Headers view source

```
Accept: text/html,application/xhtml+xml,application/xml
```



URL	Status	Domain	Size
GET softuni.bg	200 OK	softuni.bg	17,4

⊟ Response Headers view source

```
    Cache-Control  private
 Content-Encoding  gzip
   Content-Length  17819
     Content-Type  text/html; charset=utf-8
             Date  Tue, 17 Jan 2017 14:09:56 GMT
       Set-Cookie  language=bg; path=/; secure
             Vary  Accept-Encoding
  X-Frame-Options  SAMEORIGIN
```

⊟ Request Headers view source

```
          Accept  text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
 Accept-Encoding  gzip, deflate, br
 Accept-Language  en-US,en;q=0.5
   Cache-Control  max-age=0
      Connection  keep-alive
          Cookie  cb-enabled=accepted; _ga=GA1.2.240989729.1483973326; .auth.softuni=I
                  iZYSnilwmFlQP-Bx9XYYA_6EvX5fEIJlzCcUbXkg91S2CLDIj-8GP09FXKyMCJ1Ti4BC
                  9NbzKvxYFsRwHLIgWUHpYxwoKYYFLnz1trRNdpmYHeTLB1aPSHMCZXsktHJmrg2BdT54
                  dwME81M9du0waJLdrfIp2D-xlYu79oeGkXOgN4MPv--au2ZR6nZLh7CnRQaOB9tl5J5c
                  F_QS5cfN4FlV2ey6UvgUvlvvaX77bh_ZM-dht2fA1SH78ZLoQvClpwmCRaZuY6ZzSYku
                  IzniLGfTgao9ksHvmpCeVxX5OrXM_2b8PKmPBorYTpmPFMXVqK2QrF4VrNPNfkKSgN_6
                  5nkn7LcnFgSy6jyuWFcc5efqdIy1dcLNldPdyUvs4su4wJb5EpSJQ2RcP1NsC0OBy1RG
                  uB1u-4GBjFHErvEahvdzjl4DQgy2vJMgedsTB8_kKrRHW13z7qjmN-Fv17X4goU-N3f-
                  wGvBqdYIYxLkfFNRo4i_LPqTbgCnLcdx4wt92Pm8NBkNKGjjOH2fjkwLVHsniR1q3p2w
                  HtOyuIrckOHmP6mYiHyxjoN_lB9LCdDC1A9EkC7HfWm0HS1wjyibAXAvvH7FqCxn2fQM
                  QEENBZC6a84pT01dC07dVyVtESx-wuxW4oKBMG9uI9O77iFTdzfzqDTrFXDS11E_OzWE
                  2FwzJ0ub0U-Na_rVrvUHxZ3Xg9LobpAsUYsKiKPk9ImFCKSwBGs7nAw-SkHifxHBK4-g
                  QYWdKZmK7unW0U1wjuV8Q7Dkrh9KKnIevvbRAvcgh-uNJ36hvvf4XdIL0s-lblsnrMPv
                  r3ux8L7PUWGHXwWjWhbx39u2GSQQxsW68Tmp6CB7uJYt0S94uroFMYbTttY_Gr2I6hTT
                  GN7CJWuUaUcudb3O-chiwzKdpAzhBf9p1GHW1XQOiswCZRqqhfOlgSgXrLopCcb7HCpA
```

Chrome Developer Tools

Firebug

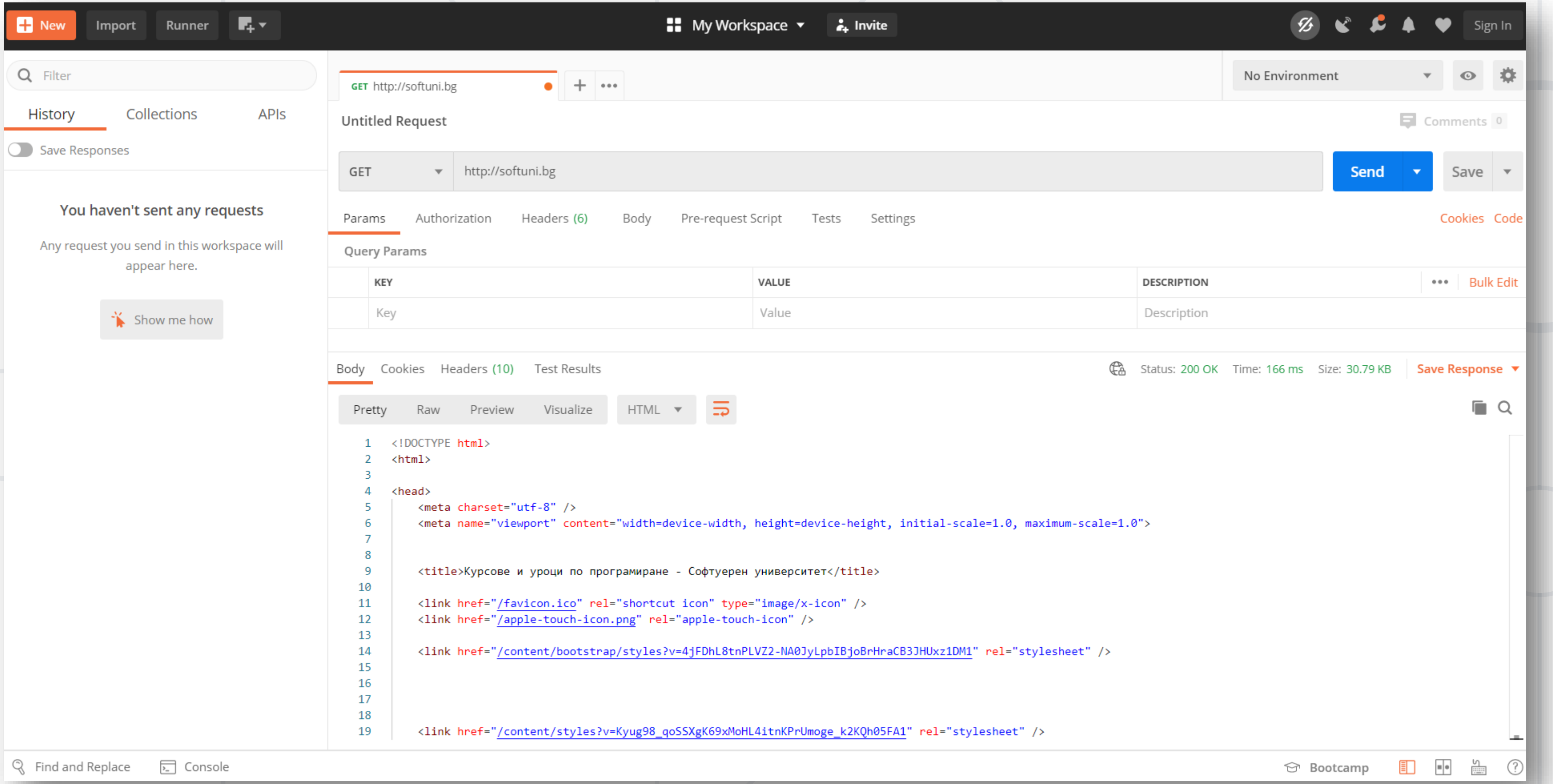
HTTP Tools for Developers – Browser

[Insomnia Rest]

[Postman]

[RESTClient]

HTTP Tools for Developers – Postman



GET http://softuni.bg

Untitled Request

You haven't sent any requests

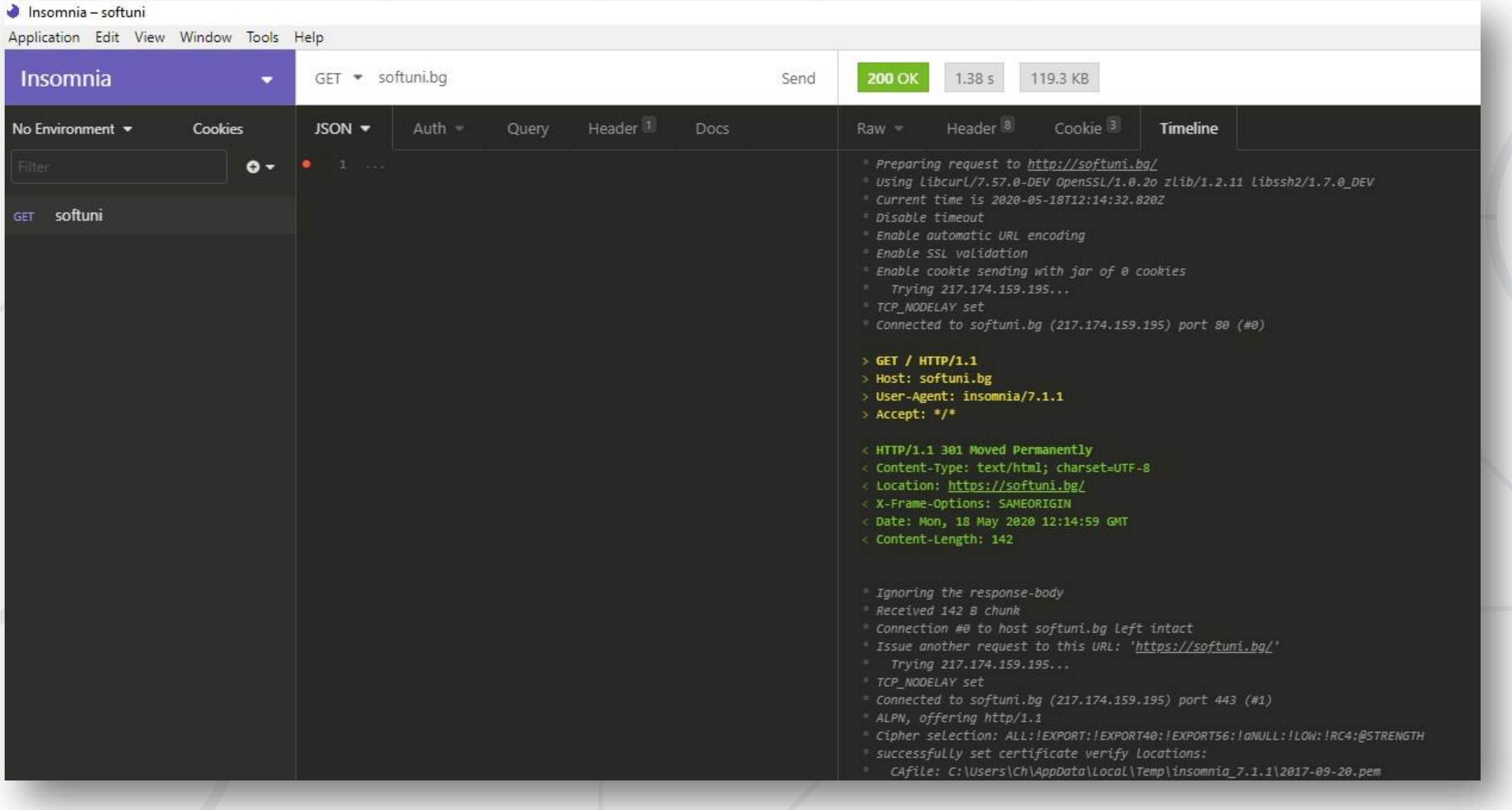
Any request you send in this workspace will appear here.

Status: 200 OK Time: 166 ms Size: 30.79 KB

```html
<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, height=device-height, initial-scale=1.0, maximum-scale=1.0">


    <title>Курсове и уроци по програмиране - Софтуерен университет</title>

    <link href="/favicon.ico" rel="shortcut icon" type="image/x-icon" />
    <link href="/apple-touch-icon.png" rel="apple-touch-icon" />

    <link href="/content/bootstrap/styles?v=4jFDhL8tnPLVZ2-NA0JyLpbIBjoBrHraCB3JHUxz1DM1" rel="stylesheet" />




    <link href="/content/styles?v=Kyug98_qoSSXgK69xMoHL4itnKPrUmoge_k2KQh05FA1" rel="stylesheet" />
```

[Postman](https://www.postman.com)

HTTP Tools for Developers – Insomnia



```
* Preparing request to http://softuni.bg/
* Using libcurl/7.57.0-DEV OpenSSL/1.0.2o zlib/1.2.11 libssh2/1.7.0_DEV
* Current time is 2020-05-18T12:14:32.820Z
* Disable timeout
* Enable automatic URL encoding
* Enable SSL validation
* Enable cookie sending with jar of 0 cookies
*   Trying 217.174.159.195...
* TCP_NODELAY set
* Connected to softuni.bg (217.174.159.195) port 80 (#0)

> GET / HTTP/1.1
> Host: softuni.bg
> User-Agent: insomnia/7.1.1
> Accept: */*

< HTTP/1.1 301 Moved Permanently
< Content-Type: text/html; charset=UTF-8
< Location: https://softuni.bg/
< X-Frame-Options: SAMEORIGIN
< Date: Mon, 18 May 2020 12:14:59 GMT
< Content-Length: 142


* Ignoring the response-body
* Received 142 B chunk
* Connection #0 to host softuni.bg left intact
* Issue another request to this URL: 'https://softuni.bg/'
*   Trying 217.174.159.195...
* TCP_NODELAY set
* Connected to softuni.bg (217.174.159.195) port 443 (#1)
* ALPN, offering http/1.1
* Cipher selection: ALL:!EXPORT:!EXPORT40:!EXPORT56:!aNULL:!LOW:!RC4:@STRENGTH
* successfully set certificate verify locations:
*   CAfile: C:\Users\Ch\AppData\Local\Temp\insomnia_7.1.1\2017-09-20.pem
```

Insomnia Rest

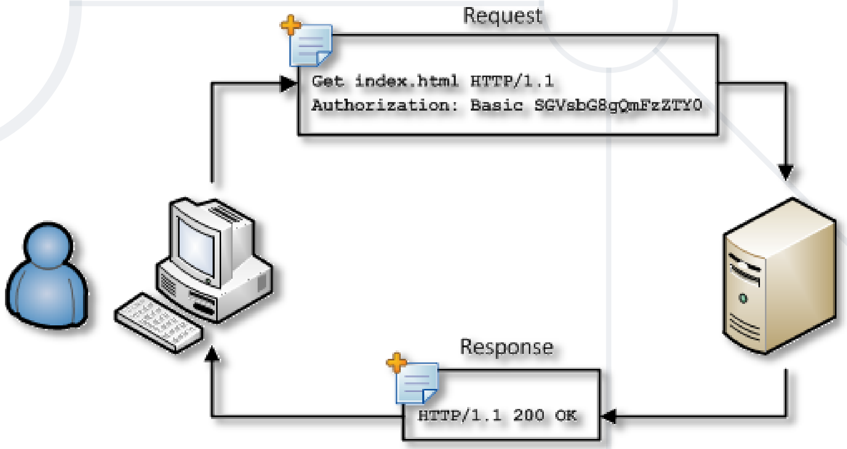
HTTP

HTTP/1, HTTP/2, HTTP/3

Hypertext Transfer Protocol

What's HTTP/1

- **HTTP/1** is the basic protocol that facilitates communication between clients and servers on the web
  - **HTTP/1** is stateless, meaning each request from a client to a server is independent and unrelated to previous requests. Fast & Optimized Meets modern web usage requirements
  - **HTTP/1** messages are text-based, consisting of plain text headers and, optionally, a message body
  - The first standardized version of HTTP, HTTP/1.1, was published in early 1997

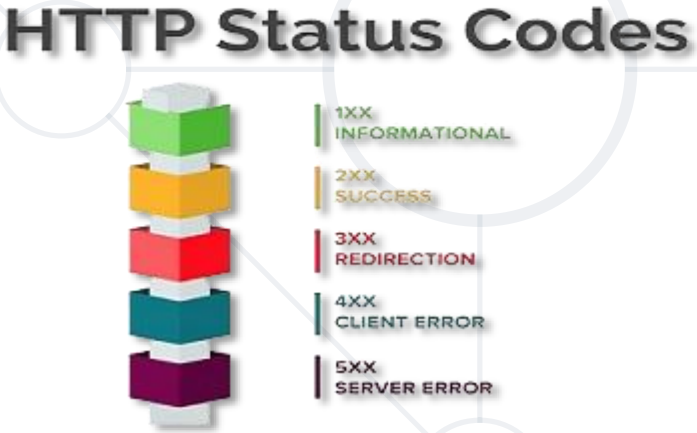


```
Get index.html HTTP/1.1
Authorization: Basic SGVsbG8gQmFzZTY0
```

```
HTTP/1.1 200 OK
```

HTTP/1 – Methods, Headers, Status Codes

- **HTTP/1** defines several methods (also known as verbs) that indicate the desired action to be performed on a resource. Common methods include **GET**, **POST**, **PUT**, **DELETE**, etc
- **HTTP/1** requests and responses include headers, which provide metadata about the request or response



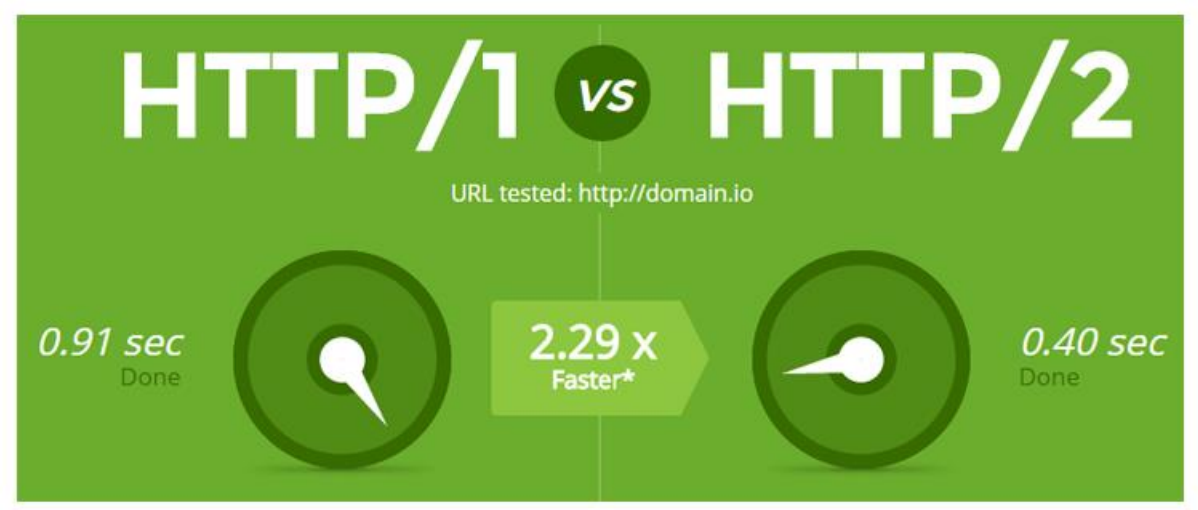
- **HTTP/1** uses status codes to indicate the outcome of a request. These codes are three-digit numbers sent by a server in response to a client's request. They can signify success (2xx), redirection (3xx), client error (4xx), or server error (5xx)

What's HTTP/2

- **HTTP/2** (originally named **HTTP/2.0**) major revision of the **HTTP** network protocol used by the **World Wide Web**
  - Supported by most of the popular web browsers (Chrome, Mozilla, Opera...)
  - Fast & Optimized. Meets modern web usage requirements.
  - Completely Backwards-Compatible

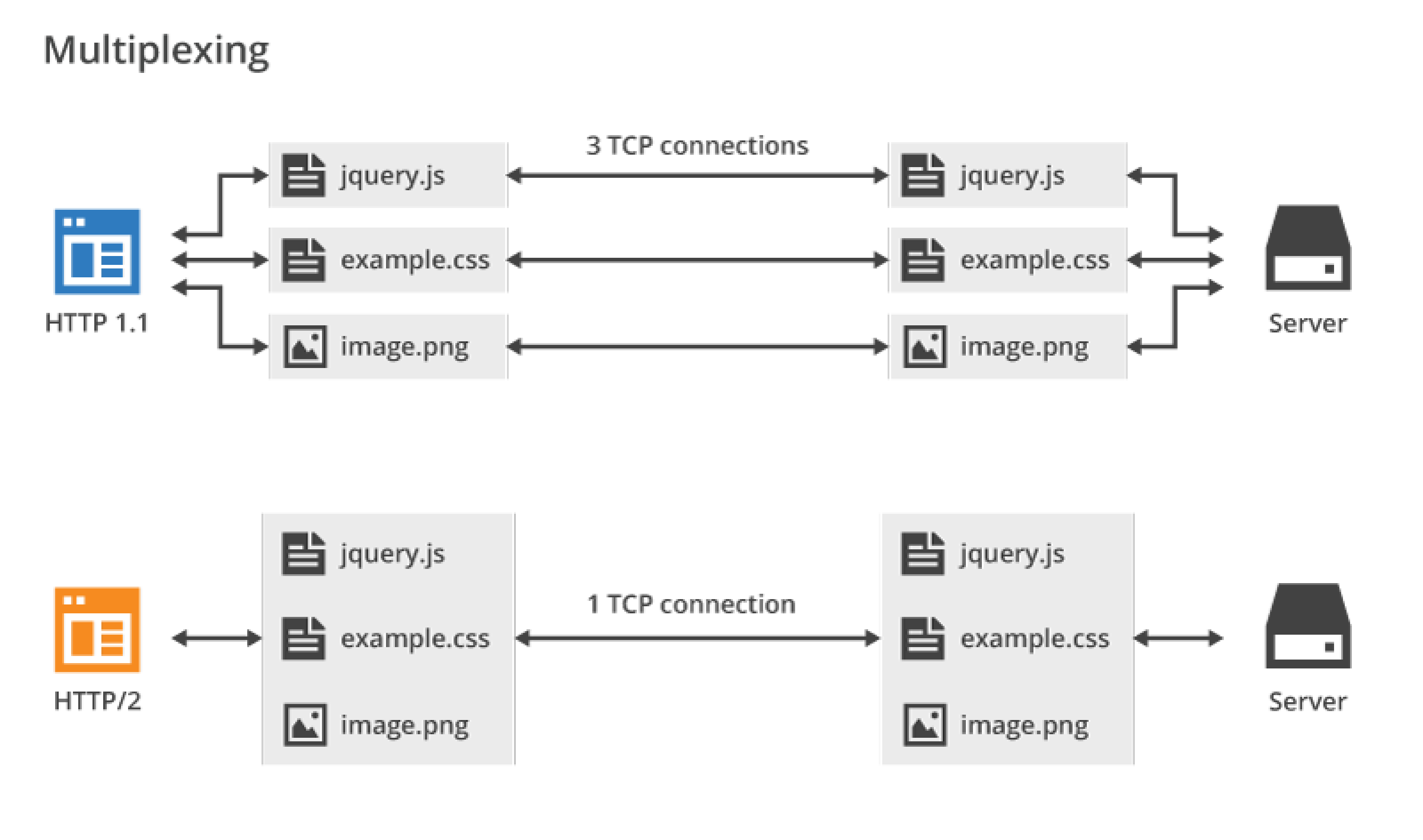
What's New?

- **HTTP/2** is meant to erase the need of maintaining complex server infrastructures in order to perform well.
- **HTTP/2** communicates in binary data frames.
- **HTTP/2** introduces several new important elements
  - HTTP/2 Multiplexing
  - HTTP/2 Header Compression
  - HTTP/2 Server Push



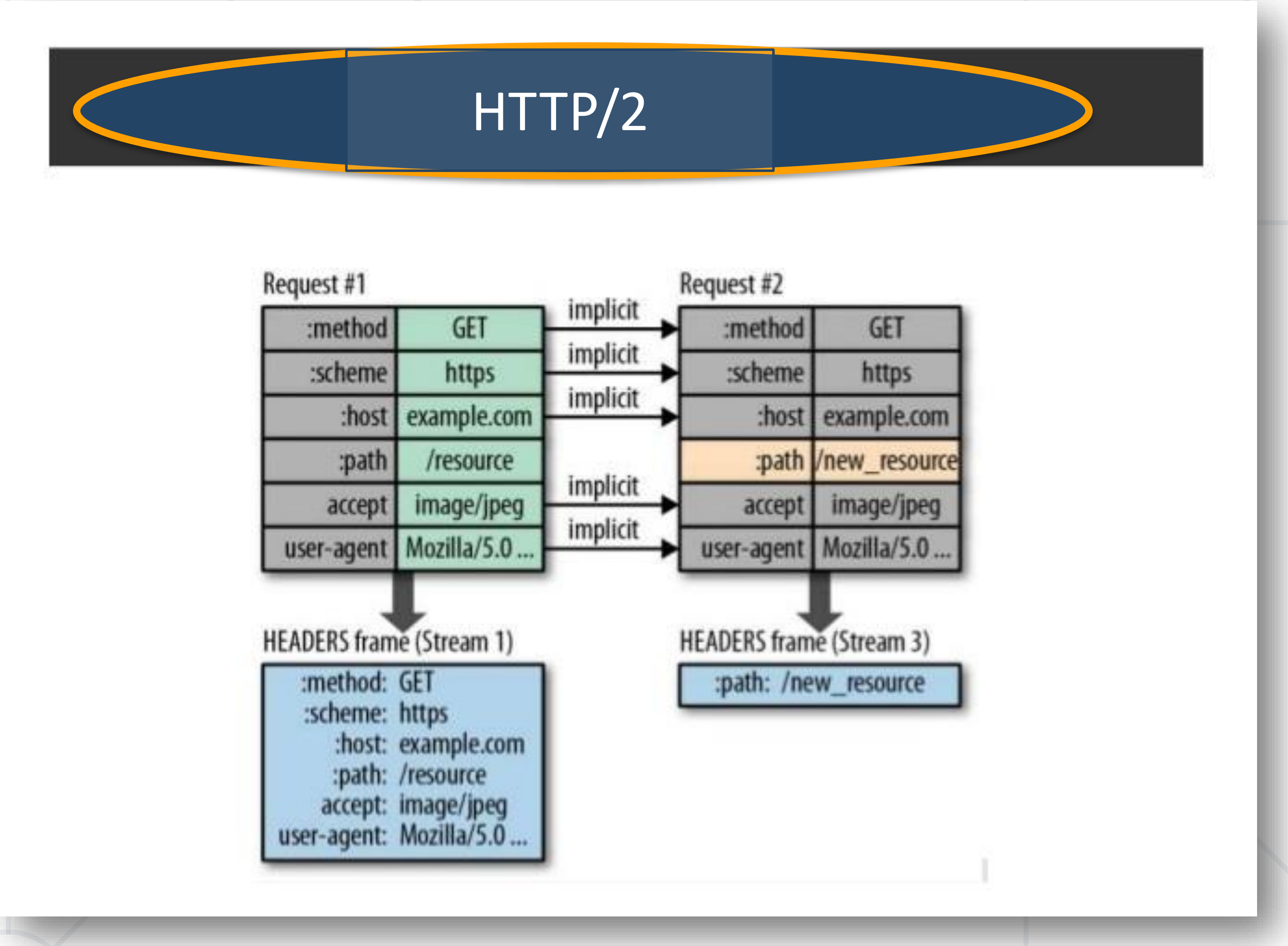
HTTP/2 Multiplexing

- The art of handling multiple streams over a **single** TCP connection.



HTTP/2

- **HTTP/2** maintains a **HTTP Header Table** across requests
- Optimizes communication drastically
- The process is essentially a **de-duplication**, rather than compression



HTTP/2

Request #1

:method	GET
:scheme	https
:host	example.com
:path	/resource
accept	image/jpeg
user-agent	Mozilla/5.0 ...

Request #2

:method	GET
:scheme	https
:host	example.com
:path	/new_resource
accept	image/jpeg
user-agent	Mozilla/5.0 ...

HEADERS frame (Stream 1)

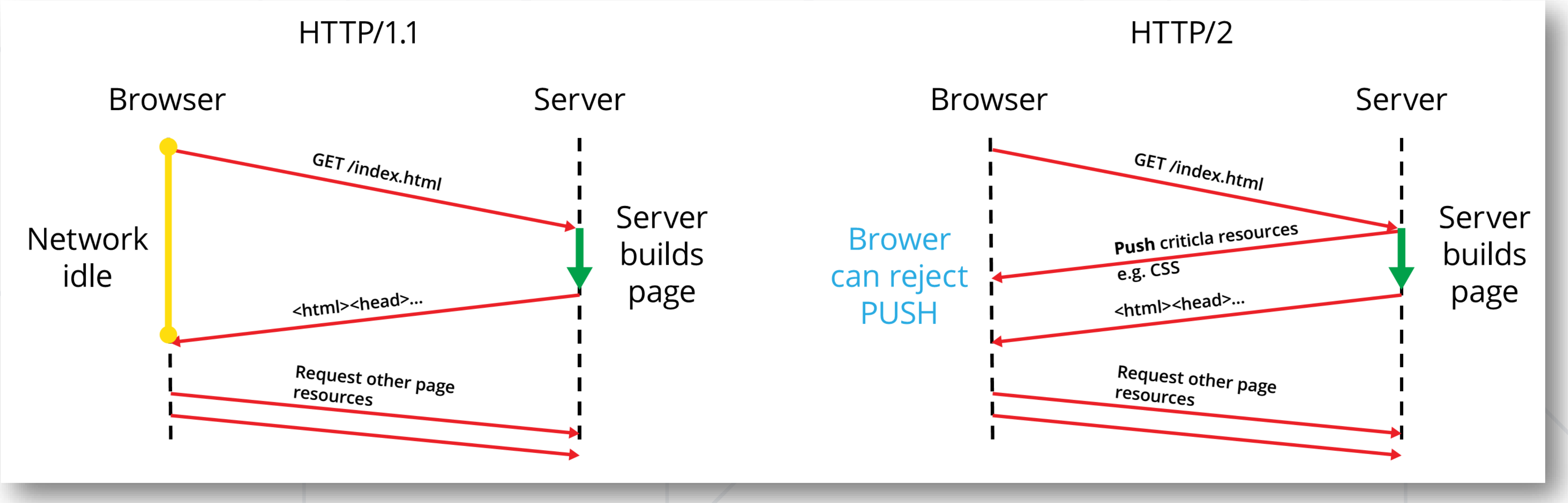
```
    :method:  GET
    :scheme:  https
      :host:  example.com
      :path:  /resource
     accept:  image/jpeg
 user-agent:  Mozilla/5.0 ...
```

HEADERS frame (Stream 3)

```
:path:  /new_resource
```

HTTP/2 Server Push

- **HTTP/2 Server Push** is the process of sending resources to clients, without them having to ask for it



What's HTTP/3

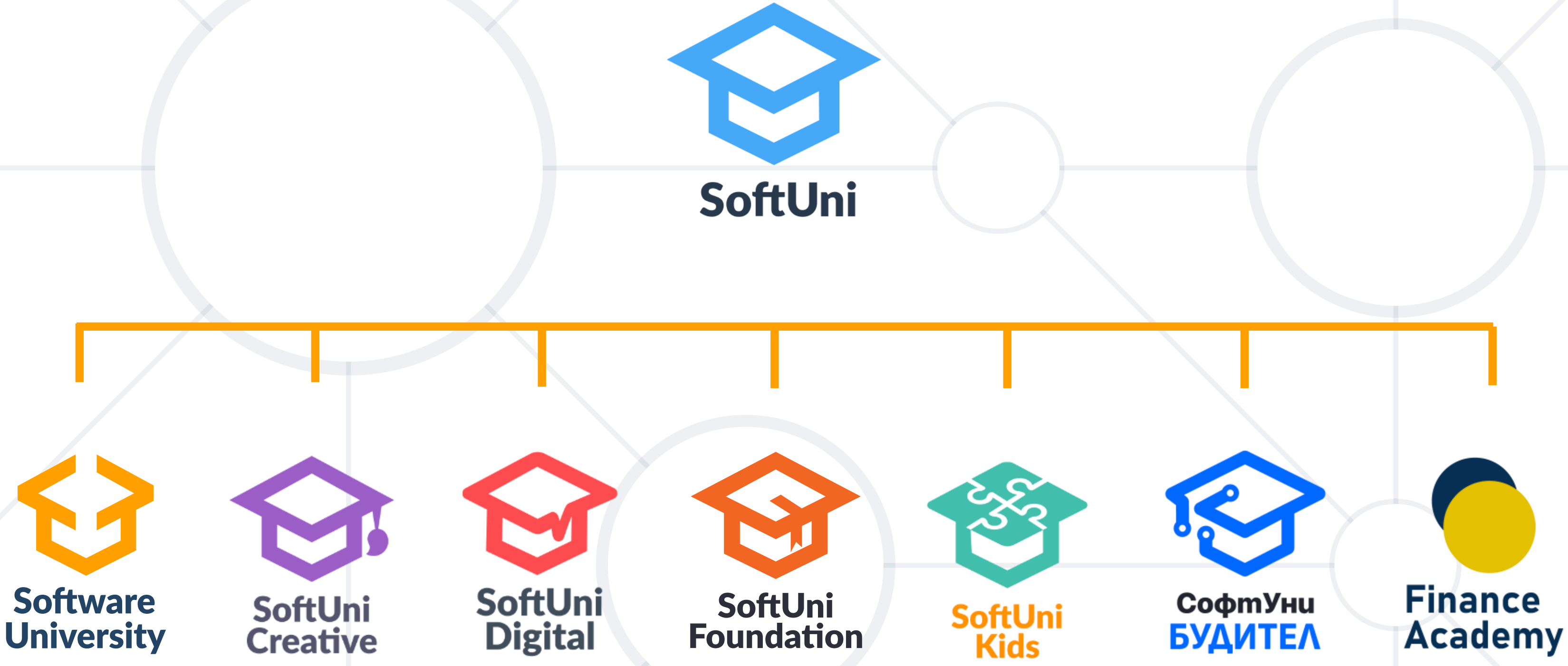
- **HTTP/3** is a new standard in development that will affect how web browsers and servers communicate
  - Significant upgrades for user experience
- **Performance**, **Reliability**, and **Security**
- **HTTP/3** runs on **QUIC**, a new transport protocol designed for **mobile-heavy** Internet usage

Summary

- Internet Protocol
- DNS
- TCP/UDP
- The OSI Model
- HTTP:
  - Request
  - Response
  - Tools
  - Versions
- URL
- MIME and Media types

Software University

Questions?



SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

СофтУни БУДИТЕЛ

Finance Academy

© SoftUni – https://about.softuni.bg. Copyrighted document. Unauthorized copy, reproduction or use is not permitted.

SoftUni Diamond Partners

SmartIT

Coca-Cola HBC
Bulgaria

SUPER
HOSTING
.BG

createX

Postbank
Решения за твоето утре

DRAFT
KINGS

VIVACOM

AMBITIONED

INDEAVR
Serving the high achievers

SOFTWARE
GROUP

PHAR
VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg