



Școala
informală
de IT

Web Introduction



Școala
informală
de IT

Agenda

- **Let's start!**
- **Internet and TCP/IP Protocol Suite**
- **The World Wide Web**
- **Client-Server Architecture**
- **Protocols: DNS, HTTP**
- **Web Browsers**
- **Developer Tools**

Initial setup



Let's start

- Let's get to know each other!
- Accounts needed:
 - Moodle
 - Cloud 9
 - Github



Internet and the TCP / IP Protocol Suite



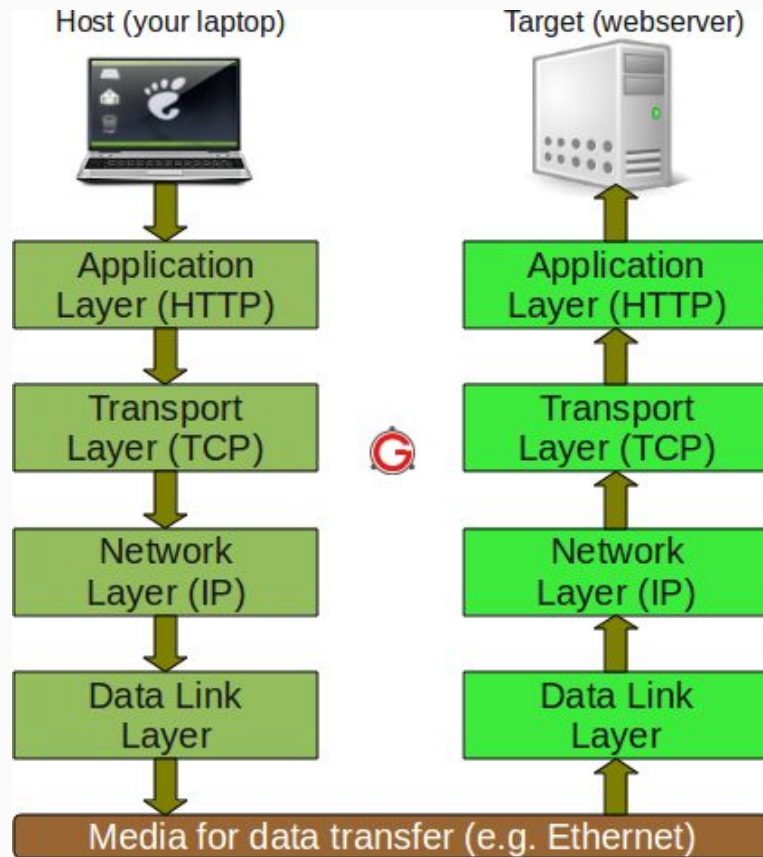
Internet

= a global network of computers that enables them to send one another small packets of digital data

- “Very fast postal system”
- Originated from **ARPAnet** in the **1960s**
- Can send **various types of data**, like web pages, email messages, and large files that might be digital videos, music files or computer programs.
- Uses the **TCP/IP protocol suite** to enable **communication**

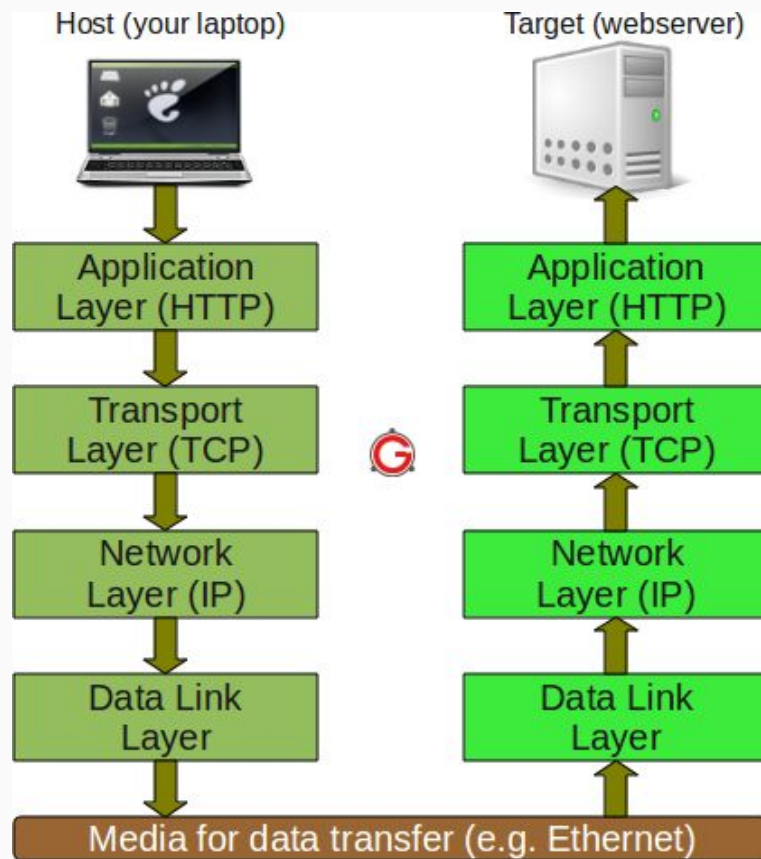
TCP / IP Protocol Suite

- **Protocol suites** have a layered architecture
 - Each layer depicts some functionality which can be carried out by a protocol
 - **Several protocols / layer**
- **TCP / IP** is the most widely known and used protocol suite
 - 4 layers



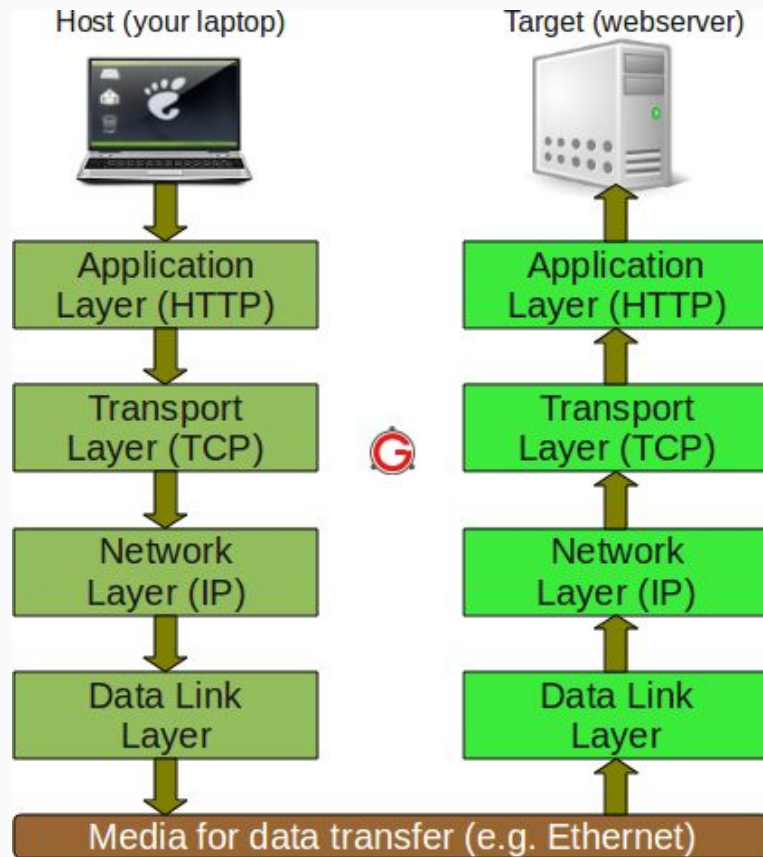
TCP / IP - Application Layer

- Includes **applications** or **processes** that use transport layer protocols to deliver the data to destination computers
- **Protocols:**
 - **HTTP** (HyperText Transfer Protocol) & **HTTPS** (HTTP Secure)
 - **FTP** (File Transfer Protocol)
 - **SMTP** (Simple Mail Transfer Protocol)



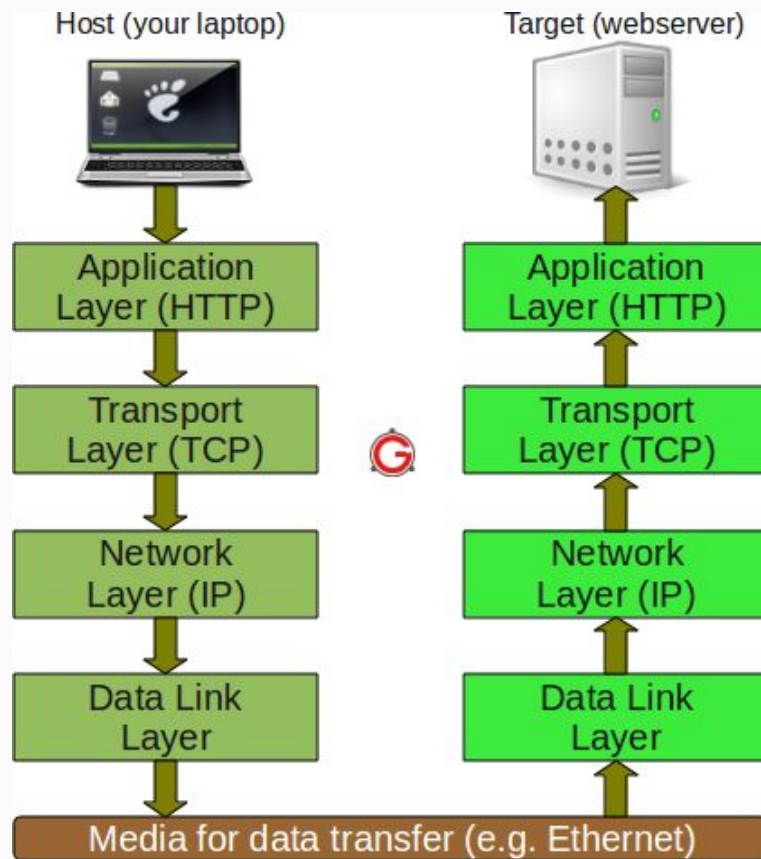
TCP / IP - Transport Layer

- Provides backbone to **data flow** between two hosts
- Protocols:
 - **TCP** = Transmission Control Protocol
 - **UDP** = User Datagram Protocol
- TCP is responsible for **breaking data down into small packets** before they can be sent over a network, and for **assembling the packets** again
- Unlike UDP, TCP is **reliable**



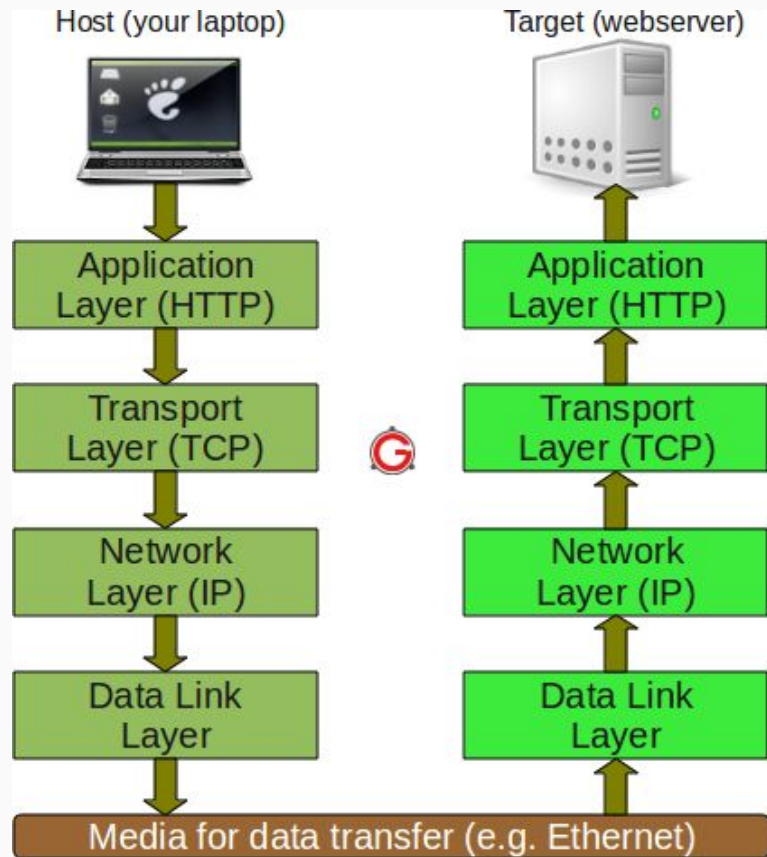
TCP / IP - Network Layer

- Also called **Internet Layer**
- Handles the **movement/routing of data on network**
- **Protocols:**
 - **IP** = Internet Protocol
- IP is responsible for **addressing, sending and receiving** the data packets over the Internet
- **All computers connected to internet have an IP address**
 - How to find it?
 - How does it look like?



TCP / IP - Data Link Layer

- Also called **Network Interface Layer**
- Normally consists of **device drivers in the OS** and the **network interface card** attached to the system
- **Protocols**
 - **ARP** (Address Resolution Protocol)
 - **PPP** (Point to Point Protocol)



IP Addresses

- A **unique string of numbers** separated by full stops/colons that identifies each computer using the Internet Protocol to communicate over a network
- **IPv4**
 - 4 numbers from 0-255 separated by full stops
 - Example: 192.168.56.17
- **IPv6**
 - 8 groups of 4 hexadecimal digits (numbers from 0 - 221241855) separated by colons
 - Example: 2a04:2413:8100:8080:d4d2:c098:514d:e7b2

World Wide Web



World Wide Web

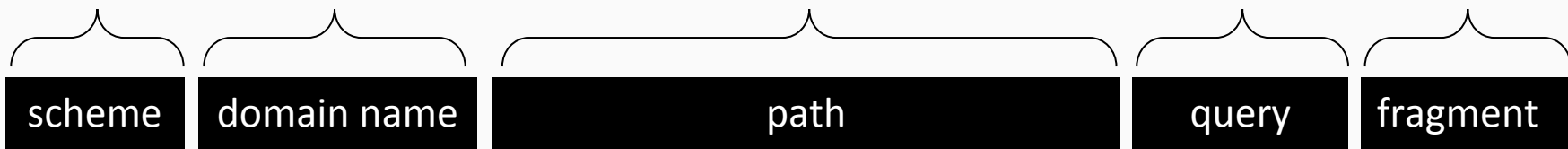
- [TED ED Short Lesson](#)
- = an **information system on the Internet** which **allows documents to be connected to other documents by hypertext links**, enabling the user to search for information by moving from one document to another
- 3 technologies:
 - **HTML**: HyperText Markup Language
 - **URI**: Uniform Resource Identifier
 - **HTTP**: Hypertext Transfer Protocol



Uniform Resource Locator (URL)

= a **reference** to a web resource that specifies its **location** on a computer network and a mechanism for **retrieving** it

`https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/GET?source=sidebar#Syntax`



1. **https** is the **scheme**
2. semicolon and two slashes (`://`) separate the scheme from the machine/domain name
3. **developer.mozilla.org** is the **machine/domain name**.
4. single slash (`/`) separates the name from the path
5. **en-US/docs/Web/HTTP/Methods/GET** is the **path**
6. question mark (`?`) separates the path from query
7. **source=sidebar** is the query (which are key-value pairs separated by **&**. Ex: `key1=value1&key2=value2`)
8. hashtag (`#`) separates the query from fragment
9. **Syntax** is the fragment

Client-Server Architecture

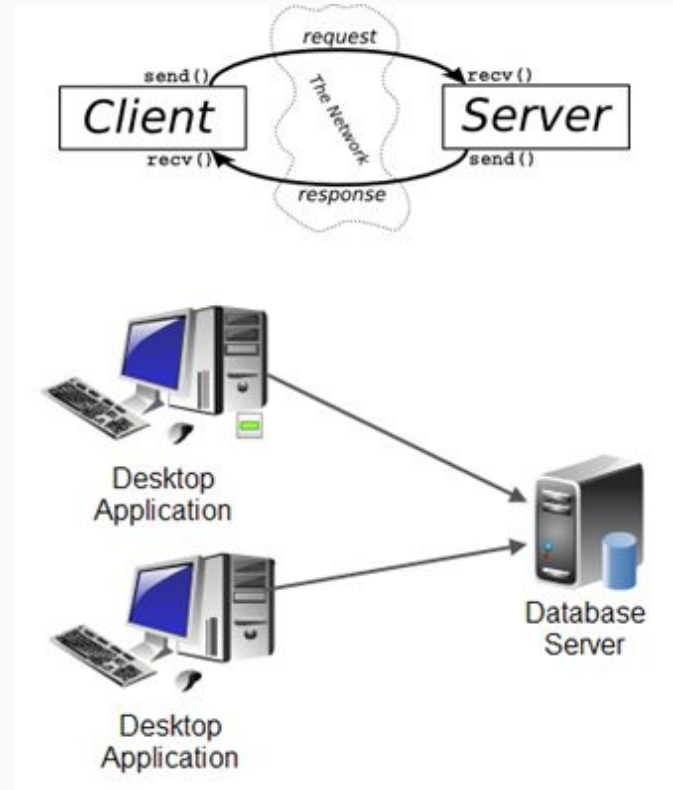


Application Architecture

- **Standalone vs distributed applications**
- **Single process architecture** (Standalone application)
 - **Command line programs**
 - **Desktop apps without network communication**
 - **Mobile apps without network communication**

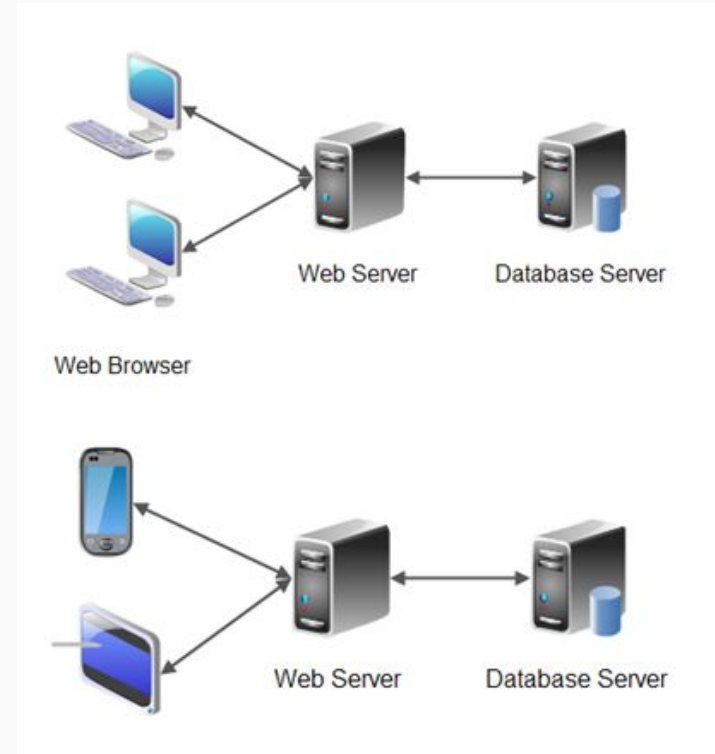
Client-Server Architecture

- A **client** is making a **request** to a **server**
- The server processes the request, and sends a **response** back to the client
- Examples:
 - Desktop application to database server communication
 - Browser to web server communication
 - Mobile to server communication



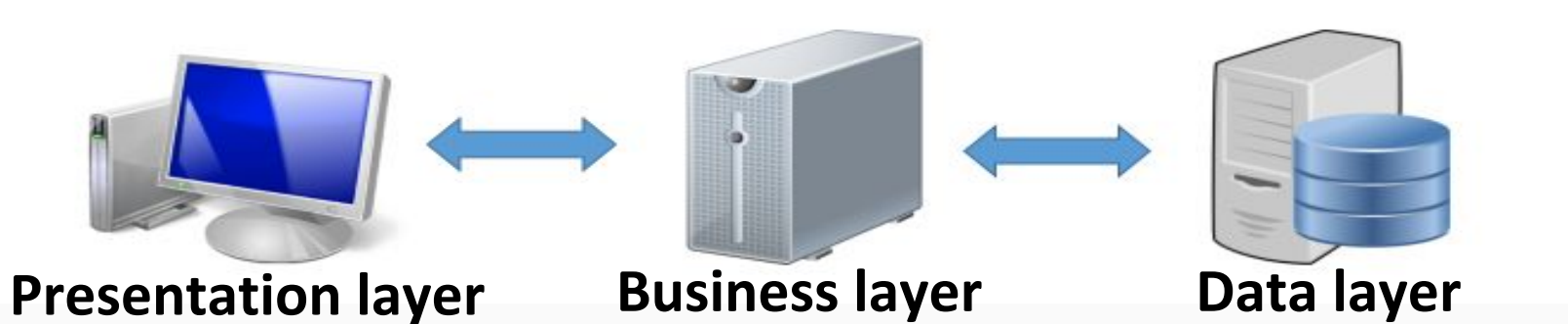
Multi-layer architecture

- Client server disadvantages
 - “Fat client”
 - Race conditions
 - Installation
- 3-layer architecture
 - Components:
 - Presentation / GUI
 - Business Logic
 - Data
 - Ex: Web and mobile apps

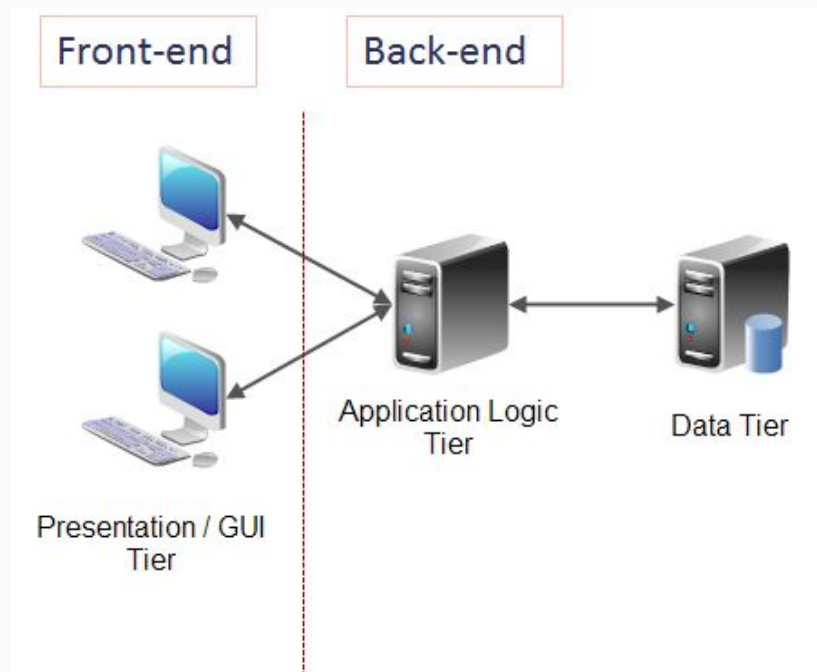


Multi-layer architecture

- **Presentation** = Application topmost level which users can access directly. Display information from other layers and communicates with business layer
- **Business** = Controls application functionality by performing detailed processing between the two surrounding layers
- **Data** = Provides data persistence (store and retrieve) mechanisms to database servers. Information is sent to business logic layer for processing and eventually back to the user.



Front-end vs Back-end

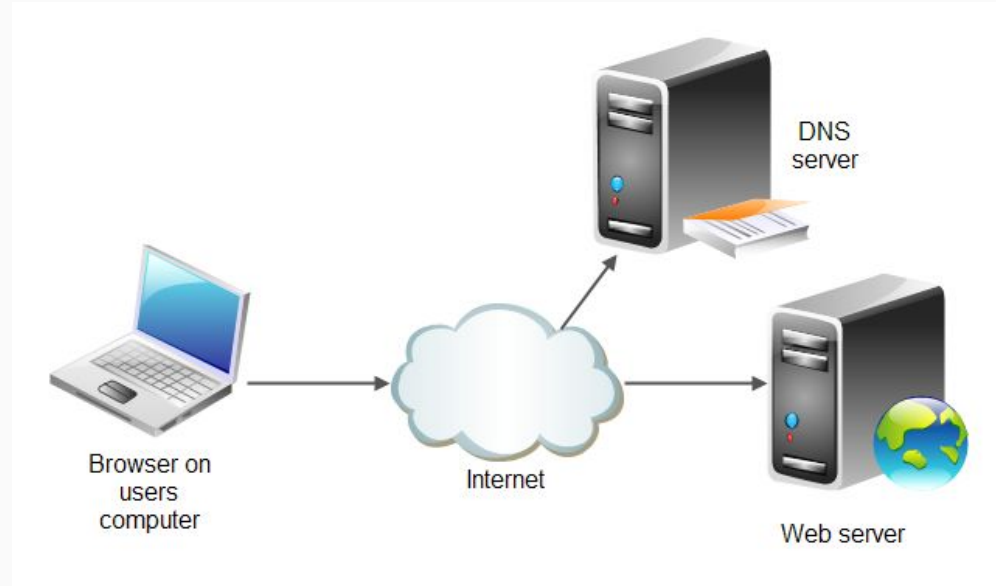


DNS



Domain Name Server

- = the Internet's equivalent of a phone book
- Domain Name Servers **maintain** a directory of domain names and **translate** them to Internet Protocol (IP) addresses

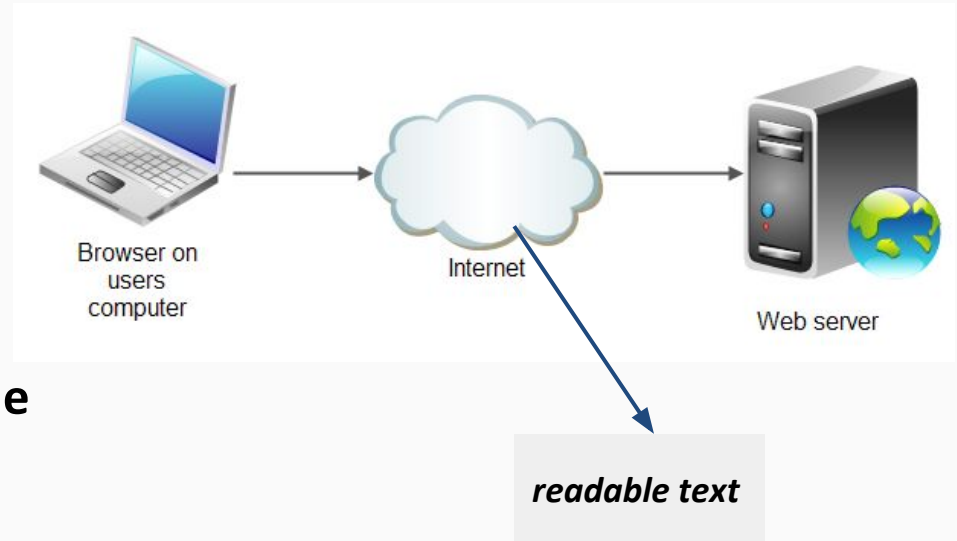


HTTP



Hyper Text Transfer Protocol

- = Client-server protocol for transferring Web resources
- It is the foundation of any data exchange on the Web!
- Important properties of HTTP
 - Request-response model
 - Text-based format
 - **Stateless**: there is no link between two requests being successively carried out on the same connection



HTTP Requests

- A request message sent by a client consists of
 - **Request line** – request method, resource URI, and protocol version
 - **Request headers** – additional parameters
 - **Body** – optional data
 - E.g. posted form data, files, etc

```
<request method> <resource> HTTP/<version>  
<headers>  
  
<body>
```

HTTP Request Methods

- A set of HTTP request methods indicate the desired action to be performed for a given resource
- These request methods are sometimes referred to as **HTTP verbs**
- The most common request methods are mapped on **CRUD**:
 - **Create** - HTTP **POST**
 - **Read** - HTTP **GET**
 - **Update** - HTTP **PUT**
 - **Delete** - HTTP **DELETE**

HTTP Responses

- A response message sent by a server consists of
 - **Status line** – protocol version, status code, status phrase
 - **Request headers** – metadata
 - **Body** – the contents of the response (the requested resource)

HTTP/<version> <status code> <status text>
<headers>

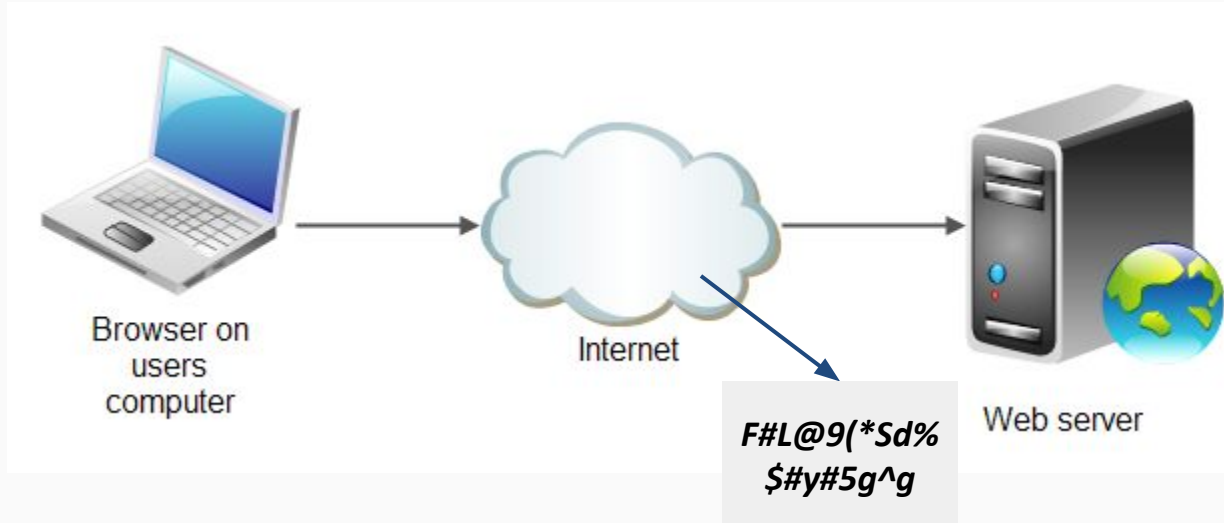
<response body – the requested resource>

HTTP Response Status Codes

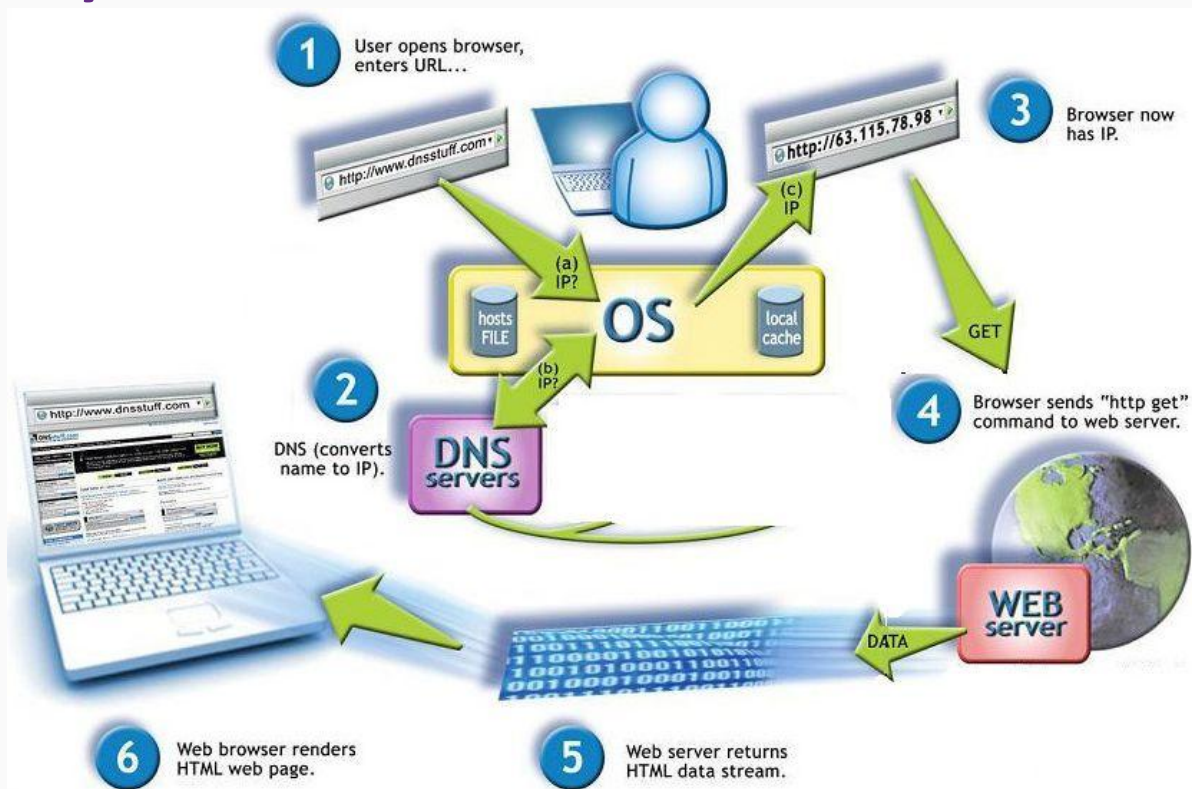
- HTTP response status codes indicate whether a specific HTTP request has been successfully completed
- They are grouped in five classes:
 - **Informational** - **1xx** (100 Continue)
 - **Successful** - **2xx** (200 Success, 201 Created, 204 No Content)
 - **Redirects** - **3xx** (302 Found, 304 Not Modified)
 - **Client errors** - **4xx** (400 Bad Request, 401 Unauthorized, 404 Not Found)
 - **Server errors** - **5xx** (500 Internal Server Error, 503 Service Unavailable)

HTTPS

- **HTTPS** - HyperText Transfer Protocol **Secured**
- **SSL** - Secure Sockets Layer
 - The SSL protocol is used to encrypt data for secure data transmission



WWW Lifecycle



Web Browsers



Web Browsers

- A web browser is a **client-side software application** for **retrieving, presenting and traversing information resources on the WWW**
- **Most popular browsers**
 - Google Chrome
 - Mozilla Firefox
 - Microsoft Internet Explorer
 - Microsoft Edge
 - Apple Safari
 - Opera

Developer Tools

DEMO

Resources

<http://www.bbc.co.uk/webwise/guides/what-is-the-internet>

<http://www.thegeekstuff.com/2011/11/tcp-ip-fundamentals>

<http://webfoundation.org/about/vision/history-of-the-web/>

<https://code.tutsplus.com/tutorials/http-the-protocol-every-web-developer-must-know-part-1--net-31177>

