Școala
informală
de IT

Web Concepts

Agenda

- Nice to meet you! :)
- Internet & The World Wide Web
- Client-Server Architecture
- HTTP Protocol
- Web Browsers & How They Work
- Developer Tools

Nice to meet you :)


NICE TO MEET YOU
I ALREADY FORGOT YOUR NAME
memes.com

Internet

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**
- Uses the **TCP/IP protocol suite** to enable **communication**

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(total: 4.3*10^9)
  - Example: 192.168.56.17
- IPv6
  - 8 groups of 4 hexadecimal digits (numbers from 0 - 65,535) separated by colons(total: 3.4*10^38)
  - Example: 2a04:2413:8100:8080:d4d2:c098:514d:e7b2

World Wide Web

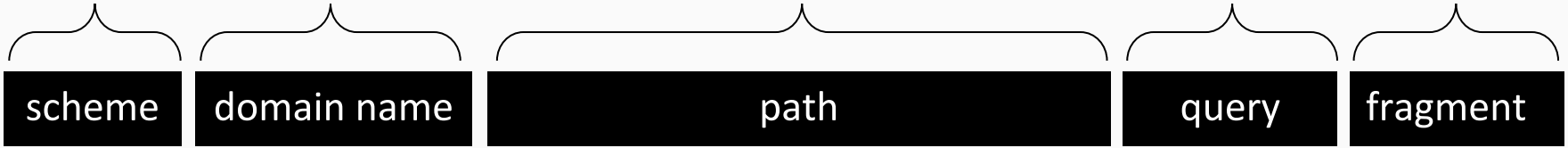
World Wide Web

- 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

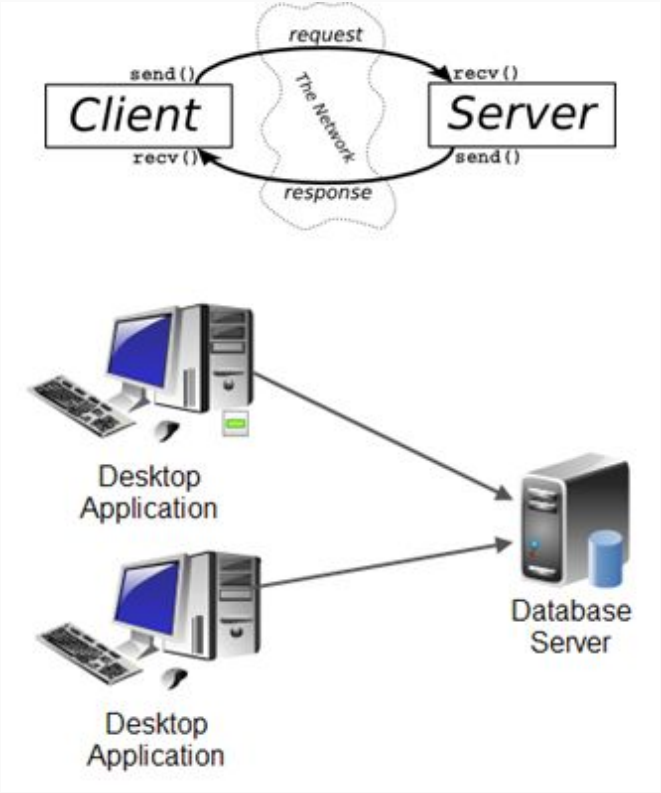
Client-Server Architecture

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

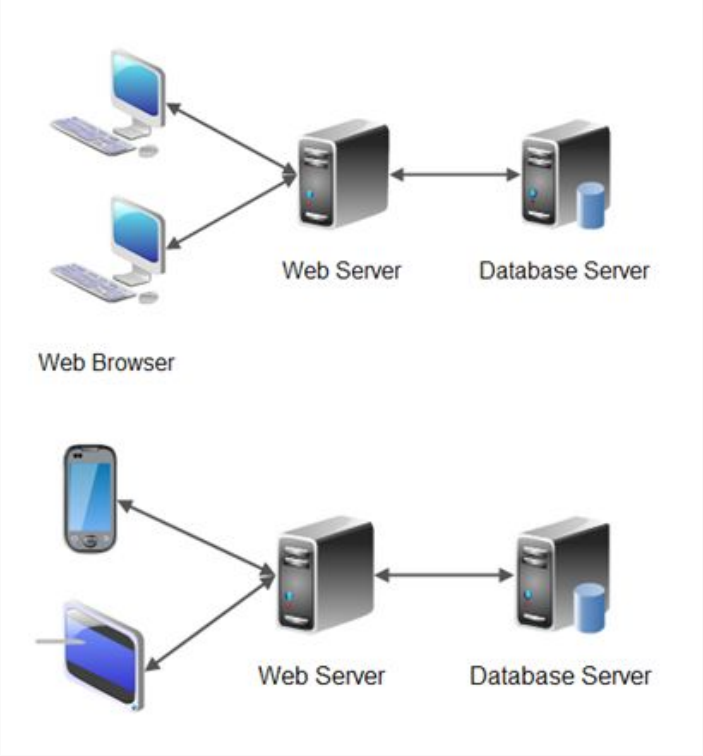


Multi-layer architecture

- Client server disadvantages
  - “Fat client”
  - Installation

=>

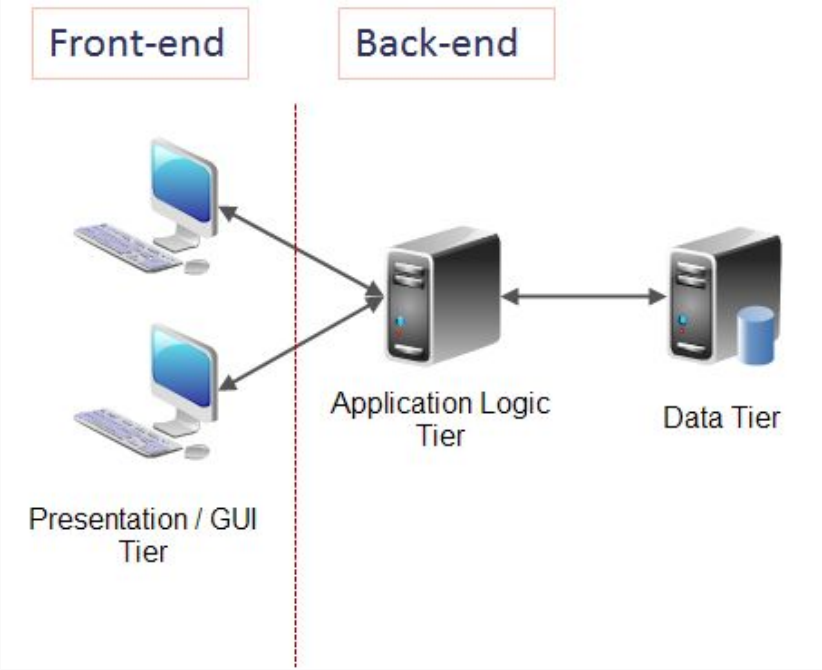
- 3-layer architecture
  - Components:
    - Presentation / GUI
    - Business Logic
    - Data
  - Ex: Web and mobile apps



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



RIA

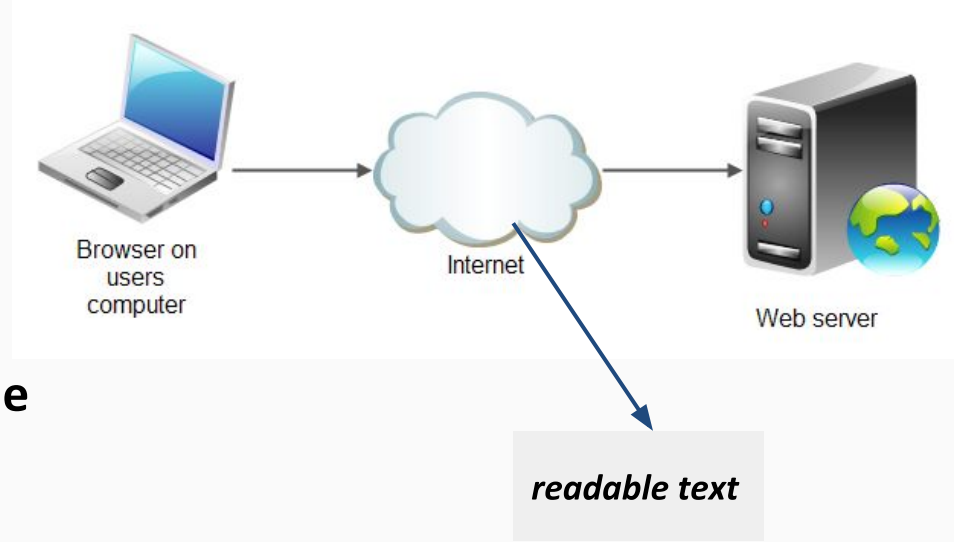
- Rich Internet Applications
  - Big part of the logic is moved on the client
  - Can be
    - SPAs
    - MPAs
- Single Page Applications
  - = a website that re-renders its content in response to navigation actions (e.g. clicking a link) without making a request to the server to fetch new HTML

HTTP

Hyper Text Transfer Protocol

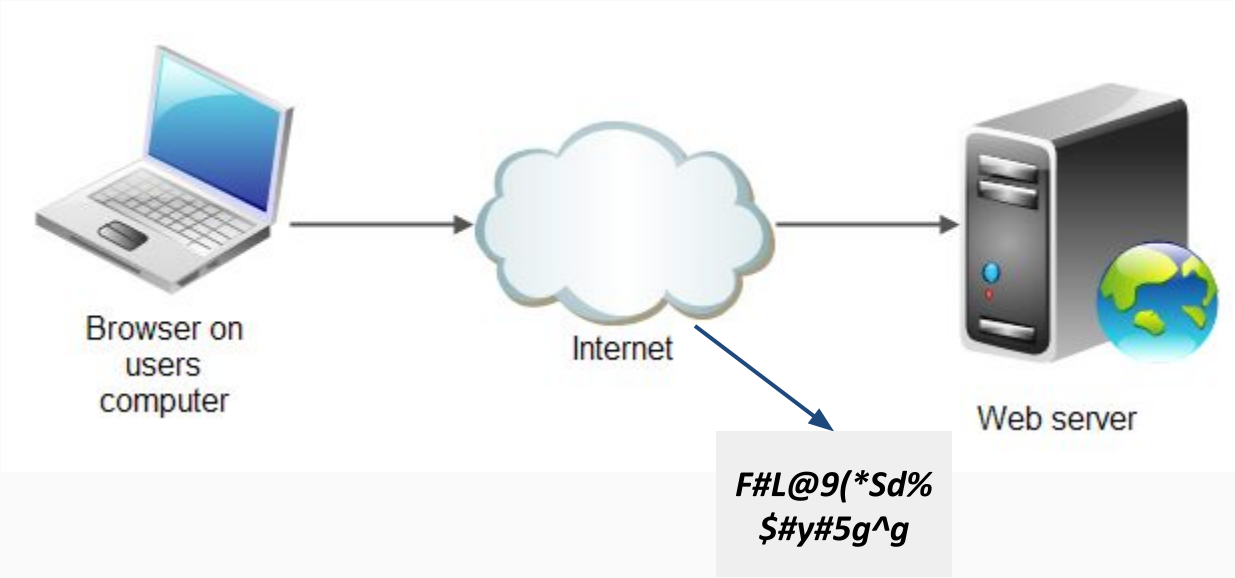
- = Client-server protocol for transferring Web resources
- It is the foundation of any data exchange on the Web!

- 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



HTTPS

- **HTTPS - HyperText Transfer Protocol Secured**
  - **Is the standard today!**



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>

<request 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)

Web Browsers

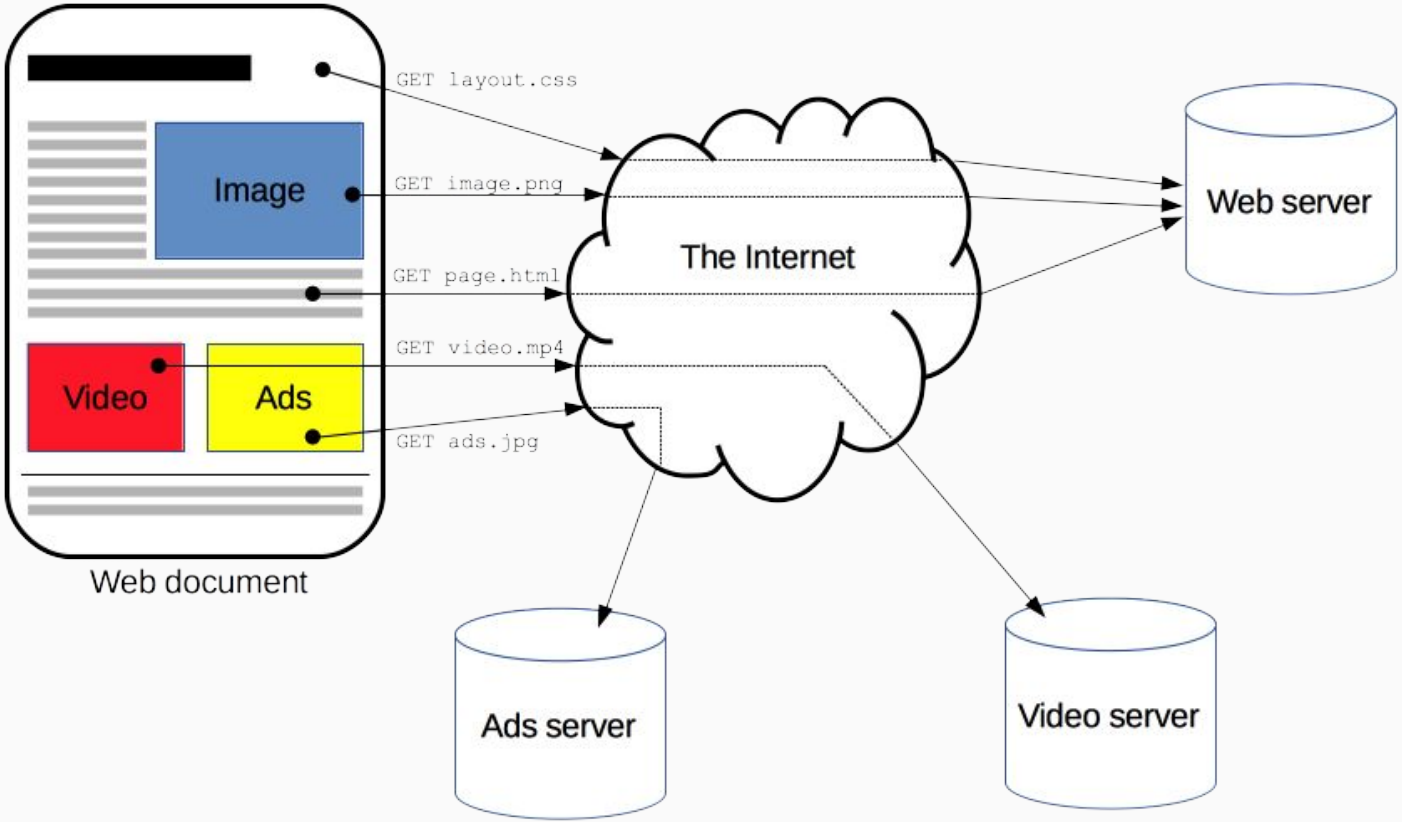
Web Browsers

- A web browser is a client-side software application for retrieving information resources on the WWW
- Browsers examples (including mobile devices):
  - Google Chrome
  - Mozilla Firefox
  - Microsoft Edge
  - Apple Safari
  - Opera
  - Microsoft Internet Explorer
  - ...

How The Web Browsers Work

How The Web Browsers Work

1. Get HTML
2. Parse head
   - Make requests to get the referenced files *in the order of their appearance*
   - => place .css and font files first
3. Parse body
   - Create DOM and CSSOM, merge them and render the page
   - Make requests to get the referenced files
   - => for most use cases, place .js files as the last tags in the body


GET layout.css
Image
GET image.png
GET page.html
Video
Ads
GET video.mp4
GET ads.jpg
Web document
The Internet
Web server
Ads server
Video server

Developer Tools

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