**Vefforitun – glósur**

L2: Networking basics, p1 networks

* IP address er notað til að identifya vélar á netinu.
  + **Host names are only for human understanding.**
  + Host names are translated into IP addresses.
  + Dæmi:
    - Host name: Google.com – IP: 216.58.211.110
* IP addresses: Netmask
  + When does a router know whether an IP address is in the same network?
    - The IP address alone is not enough!
  + Netmask/routing prefix 216.58.211.110**/24**
    - The first 24 bit identify the network, the remaining 8 the machine.
    - Network: 216.58.2110.0
    - Machine: 110 within network 216.58.211.0
* Intermediate Servers
  + Can have different functions.
    - Routers: Direct traffic, find routes for requests.
    - Proxies: Cache and filter data, handle authentication, encapsulate networks.
  + Several server processes can run on a single machine.
    - „Server“ is used to describe both the process/software and the physical machine.
* Transmission Control Protocol (TCP)
  + Establishes and maintains a connection to the destination.
  + Cuts data into chunks (packets).
  + Manages the order of packets (packets might arrive in diffrent order than sent).
  + Manages the correctness of packets (packets might get corrupted, lost).
  + Re-assembles the packet puzzle on destination side.

L2: Networking Basics, p2 HTTP

* Hypertext Transfer Protocol (HTTP)
  + Standard web protocol
    - The WWW runs on HTTP (S)
  + Statesless: server does not know of previous HTTP requests.
  + In HTTP everything is sent and received in **clear text** (also passwords!)
    - HTTP over a secured (TLS) connection: HTTPS
* Uniform Resource Locator (URL) „Web Address“
  + <http://en.wikipedia.org/wiki/URL>
    - http = Protocol
    - en.wikipedia.org = Host name
    - wiki/URL = Path
* HTTP Methods („HTTP Verbs“)
  + HTTP knows 9 different kinds of requests.
  + Common: GET, POST, PUT, PATCH, DELETE.
    - Meira í fyrirlestrinum og seinna.
  + Standard case: GET and POST in a web browser.
* HTTP Method Properties:
  + **Safe**: does not cause any side effect on the server
    - GET
  + **Idempotent**: the effect on the server is the same if the request is executed once or multiple times
    - DELETE, PUT
  + **Cacheable**: the response to the request can be stored for future re-use.
    - GET, POST
* HTTP Response Codes
  + Thee digit integer
  + First digit: class
    - 1xx Informational
    - 2xx Success
    - 3xx Redirection
    - 4xx Client error
    - 5xx Server error
  + Clients need to understand the first digit (class)
    - Not necessarily the other two.
  + Það eru til fullt af tegundum.
    - T.d. 404 not found, 500 internal server error, ...
* HTTP is stateless!
  + HTTP is stateless but we have states in WWW.
    - You log once into email, the unse it to do several things.
    - You might be logged in automatically when you re-visit a website
    - You get advertisement depending on previously-visited websites.
* Cookies
  + Text saved on the client side (e.g. in your browser).
  + The server sets the cookie, the client returns it.
  + Cookies can be modified
    - It‘s just text stored on the client side.
  + Cookes can be deleted
    - Your application should somehow work without them as well.
  + Cookies are not evil!
    - They enable the stateful web.
    - They are just misused regularly ☺

**L5/6 CSS part 1: intro**

Nota CSS selector by class fyrir píanóið

* class Hvítir keys
* class Svartir keys

box model getur hjálpað við að sjá hvernig margin dótið er

skoða flex box fyrir positioning?

Segir að það sé usefull fyrir verkefni 1

Þarf örugglega að nota % en ekki setja gildin beint eins og ég geri

* Flexbox ætti að laga þetta?

**Java script**

* Frontend = Client side
* Backend = Server side

**Basic functions**

* function showAlert(msg) {
  + window.alert(msg);
* }
* showAlert(„Hello world“)

**Variables**

* var x = 5;
* var y = 6;
* var person = „Grischa“;
* var boolean = true;

**Access existing elements:**

* document.getElementById(id);
* document.getElementsByTagName(name);
* document.getElementsByClassName(name);
* document.body;
* document.head;
* var child = element.firstChild;

**Events**

* onclick
* onload
* onchange
* onsubmit
* oncontextmenu
* ...
* Example:
  + <button type=“button“ onclick = „callFunction()“> GO! </button>

**Objects**

* Basically það sama og dictionary í python
* var x = {y: 5, v: ‚Something‘, test: true}
* x.y
  + gefur 5
* x.v
  + gefur ‚Something‘

**JSON**

* Mjög svipað og object
* var json = ‚{„name“: „Grisha“, „age“: „31“}‘;
* Breyta object í json:
  + var json = JSON.stringify(obj);
* Breyta json í object:
  + var obj = JSON.parse(json);

**Arrays (er samt object)**

* var cars = [„Saab“, „Volvo“, „BMW“];
* cars[0] = „Saab“
* myArray.push(„new string“)
  + bætir við
* var returnVal = myArray.pop()
  + removear aftast úr arrayinu og skilar í breytuna
* myArray.length
  + lengdin

**Comparison**

* „==“ compares only the value (it performs type conversion!)
  + var x = 5
  + var y = „5“
  + x == y
    - true
* „===“ compares both the type and value
  + var x = 5
  + var y = „5“
  + x === y
    - false

Sýnir í fyrirlestri keypress, keydown, keyup

Sýnir stutt dæmi um hvernig á að spila hljóð