Internet, WWW a URL

Počítačová síť je lokální propojení alespoň dvou zařízení, které spolu umí komunikovat. Počítačová síť může obsahovat další počítačové podsítě a tím může vznikat jakýsi hierarchický systém. Tyto počítačové sítě komunikují různými protokoly a rozumí si. Probíhá tedy výměna informací a používají stejné technické prostředky.

V komunikaci rozdělujeme dva typy komunikací, a to **peer to peer** a **client-server**. Peer to peer je způsob komunikace, kdy jsou si obě dvě strany rovny a zároveň si nepřikazují. Client-server je způsob, kdy je jedna strana autoritativní a je cílena různými klienty. Server zpracovává jejích příkazy a může je i odmítat.

Internet je celosvětový systém navzájem propojených počítačových sítí. Propojení sítí je realizováno různými způsoby (např. drátově a bezdrátově), které tu nebudeme probírat do hloubky.

Hypertextový dokument je dokument, který může být vázán na jiné dokumenty. Propojují je tzv. **hyperlinky**. Tento dokument často není lineární, ve slova smyslu, jako jsou obyčejné dokumenty, které můžete číst ze shora dolů. Tento dokument může být definován tak, že se něco může zobrazit jinde, než bylo definováno.

World wide web je soustava propojených hypertextových dokumentů. Zjednodušeně se jedná o mapu propojených dokumentů, které lze navštěvovat pomocí **webového prohlížeče**. Používá se protokol HTTP (a HTTPS) ve formátu HTML. Pro adresaci ve WWW používá URL.

Doména je částí URL adresy a určuje jednoznačný identifikátor nějakého místa (přesněji počítačové sítě či serveru) na internetu. Vznikla z důvodu, že komunikace pomocí IP adres je složitá:

- pro lidi je složité si zapamatovat (v případě IPv4) 4 skupiny čísel pro každý web, který navštěvují
- v případě HTTP se otevírá pouze jeden port - tím pádem by mohl být na jednu IP adresu registrován pouze jeden web. Tento problém by šel obejít různými dalšími porty či překonfigurací webů např. na 23.201.102.3/alza

Doména se rozděluje do několik částí, které jsou oddělené tečkami, všechny části musí být ASCII a ignoruje se velikost písmen. Základní je generická (doména 1. řádu, top level domain = **TLD**), která určuje úplný "zdroj" stránky, která je po poslední tečce. Většinou se jedná o kódy zemí a nově i další věci. Jedná se například o `.cz, .uk, .fr, .io, .website`. Je možné si zařídit svojí vlastní generickou doménu, musíte ale získat spoustu povolení a spřátelit si různé instituce. Stojí to velmi velké peníze, prostředků a času, protože své TLD musíte spravovat.

Doména, kterou si již uživatel může sám přizpůsobit a popřípadě i zakoupit se jmenuje doména 2. řádu. Určuje nějakou společnost, entitu či prostě danou stránku. Nejčastěji je doména kupována na rok, kdy první rok je levnější než ty následující. Jedná se např. o `seznam.cz, ssps.cz, google.com`.

Všechny následující domény lze označovat jako x. řádu a často se jim říká **subdoména** (sub - menší část z něčeho). Často je to nějaká podslužba dané stránky. Např. se jedná o

`email.seznam.cz, skripta.ssps.cz, calendar.ssps.cz`. S domény větším jak 4. řádu se v podstatě v praxi nelze potkat. Této služby využívají i takzvané freehostingy.

Internationalized Domain Names (IDN) je standard, který dovoluje, aby se v doménách mohli vyskytovat libovolné znaky abeced z celého světa. Doména je stále však tvořena ASCII, jen jsou znaky překódovány. Například znak `á` se přetvoří na `xn-1ca`. Pro českou doménu je vypnuto z bezpečnostních důvodů (přepis může přesměrovat na phishing stránku) a taky z konfúze, protože pro jednu doménu by bylo nutné koupit další pro zaručení bezpečnosti. Společnost CZ.NIC, která českou doménu spravuje pravidelně dělá dotazník, kde se na tuto technologii ptá. Více si o této problematice lze přečíst ZDE.

URL (uniform resource locator) je řetězec znaků s definovanou strukturou, který se používá pro adresaci na WWW. Když chceme webovou stránku navštívit, tak nenavštěvujeme pouze doménu, ale můžeme určovat další věci. Nejčastěji určujeme cestu, na které najdeme nějaký zdroj. Je rozdělen do různých částí a platí pro ně různé pravidla.

Části URL:

- **protokol** (nezáleží na velikosti písmen, povinná část, prohlížeč některé protokoly automaticky doplňuje) - např. `HTTPS, HTTP, FTP, ...`
- **doména / IP adresa** (nezáleží na velikosti písmen, povinná část)
- **port** (povinná část, prohlížeč některé porty automaticky doplňuje) - číslo od `0` do `65535`, pro HTTP se používá `80`, pro HTTPS `443`
- **umístění na serveru** - určuje místo souboru, hypertextového dokumentu, či zdroje na serveru, např. `o-skole/pedagogicky-sbor.html`
  - lomeno většinou reprezentuje přestup do jiné složky (najdi soubor pedagogicky-sbor.html v složce o-skole), ale nemusí to tak být
- **parametry** - určuje parametry se kterými může poté stránka či server pracovat, např. `?p=1` a `?name=Matěj Cajthaml&age=69`.
  - parametry vždy začínají ?, aby se rozeznali od umístění na serveru - ten otazník tedy obsahovat nemůže.
- **kotva** určuje jedno místo na stránce, na které se při načtení přeneseme, více si o ní povíme v dalších lekcích, např. `#seznam`

Často se k URL zapisuje i WWW, které k ničemu neslouží a je to normální doména 3. řádu (tzv. subdoména). Ukázky správných URL adres:

- `https://cs.wikipedia.org/wiki/Česko`
- `https://www.jakpsatweb.cz/html/url.html`
- `https://www.luxor.cz/checkout/delivery_and_payment`
- `https://ssps.cz/news/read/452`
- `https://www.presloviny.cz/kategorie/technologie/`
- `https://www.postovnisporitelna.cz/portal/ucty/postovni-ucet`

URL se postupně vyvíjí, prošli jsme pouhé základy a více informací naleznete na dokumentaci ZDE.

Vyhledávání na webu, protokoly a digitální stopa

Webový vyhledávač je automatický systém vyhledávající informace na WWW. Server automaticky prochází (crawluje) stránky a zjišťuje si z nich informace, které poté uživatelovi, který vyhledává, předá. Z různých odkazů se poté tvoří mapa (pavučina) WWW.

Webový katalog je seznam odkazů, které vedou na různé weby. Je spravován ručně.

Webové vyhledávače lze omezit pomocí souboru `robots.txt`, ve kterém lze zakázat, aby tento web navštěvovali (nebo omezit nějaké URL). Etický crawleři toto Vaše rozhodnutí budou respektovat. Není to ale povinné.

Část WWW, kam se vyhledávače nemohou dostat, říkáme **hluboký web (deep web)**. Jsou to jednoduše veškeré stránky, které potřebují například přihlášení či různou formu autentifikace. Například se jedná o internetové bankovnictví, Vaše zprávy na Instagramu, či cokoliv obdobného. **Jedná se o největší část WWW**.

Hluboký web se často zaměňuje s **temným webem (dark webem)**. To je web, na který se nedá dostat pomocí klasických technologií a je potřeba speciální software, speciální připojení, připojení v určitý čas a podobně. Dark web stojí na anonymitě pomocí různých technologií, které lze při určitých podmínkách obejít. Na dark webu se často konají ilegální prodeje a další věci.

Do obyčejného webu patří:

- stránky školy
- wikipedia
- Facebook profil (dle nastavení soukromí)

Do hlubokého webu patří:

- virtuální škola / moodle
- Bakaláři
- zprávy v Messengeru / Instagramu
- zprávy v e-mailovém klientovi

Do temného webu patří:

- tzv. onion web, `*.onion`

Protokoly

Pro webové stránky existuje nespočet protokolů, které jsou pro jeho funkčnost nutné. My si ukážeme ty nejzákladnější z nich. Více o nich jste se měli dozvědět v předmětu PSI v 1. a 2. ročníku. Protokol je vlastně takový definovač toho, jak dvě věci mezi sebou budou komunikovat, aby si rozuměli. Mají vlastní syntaxe, sémantiku a např. synchronizaci.

HTTP

Zkratka pro **H**yper **T**ext **T**ransfer **P**rotocol. Je určen pro přenos hypertextových dokumentů (definované v WWW). Běží na portu 80, ale lze jej zřídit na jakémkoliv. Moderně se nepoužívá jenom pro přenos těchto dokumentů, ale i pro další věci a stal se takovým hlavním protokolem - je ve všech odvětvích IT.

HTTPS

Zkratka pro **H**yper **T**ext **T**ransfer **P**rotocol **S**ecure. Je to zabezpečená nástavba HTTP, používá asymetrické šifrování pomocí klíčů a certifikátů. Data jsou po cestě šifrována. Používá se port 443. Dříve se používal pouze na důvěrné informace, nyní se používá na veškerou internetovou komunikaci (je nutný všude).

IP

Zkratka **I**nternet **P**rotocol. Je určený na směrování paketů na internetu. Funguje na síťové vrstvě (komunikace mezi více sítěmi). Existují dvě verze. Jedna je IPv4, která je vyčerpaná. Proto vznikla nová IPv6, která má i mimo počet adres více výhod.

Formát IPv4: `192.168.0.1`

Formát IPv6: `2001:718::fec9:ca5`

DNS

Zkratka **D**omain **N**ame **S**ystem. Jedná se o převod doménového jména na nějaké IP. Jedná se o hierarchický systém, který určuje že např. `seznam.cz` se převede na `77.75.77.222`. Může se totiž stávat, že některé subdomény mohou vést na jiné servery, aby se celá doména nemusela nacházet na jednom serveru, což je nebezpečné. DNS neslouží jen na převod jmen, ale například i na určování e-mailových serverů a dalších záznamů pro počítače.

DNS se musí cachovat, tedy ukládat, aby se nemuseli počítače při připojení vždy ptát, kudy se mají vydat aby navštívili tuhle doménu. Nejbližší uložiště DNS je například v routeru, u ISP a dál. DNS server si ukládá svoje záznamy o daných doménách a komunikuje s ostatními, se kterými sdílí svoje záznamy.

FTP

Zkratka pro **F**ile **T**ransfer **P**rotocol. Slouží k přenášení souborů mezi počítači a servery. Podporuje nespočet operací, která každá dělá něco jiného. Využívá model klient-server a pro připojení je potřeba FTP klient, který je často i v prohlížeči.

Známí FTP klienti jsou např.: WinSCP, FreeCommander, FileZilla, ...

Digitální stopa jsou záznamy na internetu, které vytváříme při procházení WWW a internetu. Tyto záznamy se nacházejí jak v **prohlížeči** (klientovi) tak i na **serverech**. Patří do toho očekávané věci - příspěvky, komentáře, objednávky a další věci, které logicky musí server uchovávat.

Digitální stopa patří do **cenných informací**, což jsou informace, které mohou společnosti využívat k lepšímu marketingu a obecně k cílení. Dělíme je na **vědomé** (výše popsané, např. příspěvky, ...) a **nevědomé** (historie procházení, samotná návštěva stránky, kde na webu

klikáte, ...). Tyto informace nemusí být použité k špatné věci (tj. zanechané nepříteli), ale **mohou být využity k zlepšení služby**.

Pro minimalizaci zanechávání stopy je dobré si nastavit prohlížeč tak, aby **posílal webům co nejméně informací**. Poté je dobré při první návštěvě webu nastavit cookies tak (podle cookies law od EU), že **nechcete využívat soubory ke sledování**. Proti některé stopě se nelze bránit - jak se říká, to co nahrajete na internet, tam bude velmi pravděpodobně navždy.

K tématu patří taky **metadata**. Jsou to data, které popisují data (tj. **data o datech**). Jako ukázka můžou být například fotografie, kde se často ukládá, jaké zařízení fotku vyfotilo, kdy byla vyfocena, kdy byl soubor naposledy upraven a podobně. Tyto informace existují skoro ke každým datům např.: u souborů, příspěvků, e-mailů, ...

K realizaci toho, jak moc jen samotný prohlížeč posílá na internet informací (metadat), doporučuji navštívit stránku AmIUnique, na které si můžete ověřit jak unikátní jste. Webové stránky mohou například skenovat nainstalovaná písma, operační systém, verzi prohlížeče, jazyk apod.

Webový prohlížeč, hosting a certifikát

Webový prohlížeč je software, který je určený **k procházení WWW**, zjednodušeně webů. Jeho cílem je komunikovat se servery v různých protokolech, nejčastěji pomocí HTTP/S. Prohlížeč však má různé věci, které s procházením nemusí přímo souviset. Často řeší i jiné úlohy (záložky, kalendáře, e-maily, historii) a jsou nastavitelné. Hlavní částí prohlížeče je **vykreslovací (renderovací) jádro**, které slouží pro vykreslování stránek v jazyce HTML, se kterým se tento rok budeme učit.

Nejčastější jádra jsou **Gecko** (Mozilal Firefox), **Webkit** (Safari), **Blink** (Google Chrome) a např. Trident (Internet Explorer). Jádra se hlavně liší v tom, jak se webové stránky vykreslují.

Chromium je vlastně takový "prohlížeč" bez ničeho navíc. V podstatě obsahuje jen vykreslovací jádro **Blink**, které obsahuje Google Chrome. Je to velmi oblíbený nástroj, který se používá pro vykreslování webových stránek bez zbytečných věcí na více. **Velmi mnoho prohlížečů na něm staví svůj základ** a nepoužívají tím své jádra - jedná se např. o Vivaldi, Operu, MS Edge, Brave a spoustu další. Chromium prohlížeč je **opensource** a tedy každý do něj může nahlížet a upravovat jej. Chromium lze používat samostatně jako náhrada Google Chrome, ale neobsahuje například ani synchronizaci mezi zařízeními. **Hlavní nevýhodou Chromium** je to, že jej začalo využívat již tolik prohlížečů, že se stal Google gigantem, který může velmi jednoduše ovlivňovat vývoj webu - **velmi se snížila kompetitivnost na tomto trhu**. Narozdíl od Gecko a Blink, Webkit od Safari s weby velmi stagnuje.

Moderní prohlížeče by se měli co nejvíce přizpůsobit potřebám uživatele, jedná se např. o:

- úpravy barev
- úpravy velikosti písmen a fontů
- umí ochránit uživatele před reklamou či sledovači
- má anonymní režim
- být rychlý a nebrat zbytečné zdroje počítače
- podporuje rozšíření
- má synchronizaci

Hosting je služba, která Vám dovoluje nějak **pronajmout hardware**, který je schopen komunikovat s internetem a nebo např. něco vypočítávat. Existuje nespočet typů hostingů, my si ukážeme jen základní.

Serverhosting je pronájem celého serveru (počítače), na kterém si lze spustit jakýkoliv software. Zákazník má přístup k celému serveru a může sloužit úplně na cokoliv - např. hostování stránek, her či různých vnitřních systémů.

Webhosting je pronájem místa na cizím serveru, který je určen na hostování stránek. Jedná se o omezené místo a jsou zde také omezeny i další parametry. Server a web spravuje daná organizace, většinou nelze nastavit vlastní konfiguraci.

Freehosting je typ hostingu, který je zdarma. Často chybí technologie, databáze a další služby. Vkládá se například reklama nebo je jinak omezena. Používá se taky jako ukázka služeb daného hostingu. Často je k dispozici i doména 3. řádu zdarma.

Mezi další služby, které hostingy nabízí, patří:

- domény
- certifikáty
- databáze
- e-maily
- výpočetní služby

Pro výběr hostingu existuje nespočet kritérií. Je nutné odhadnout, kolik webová stránka zabere místa a jaké technologie budou potřeba. Mezi známe české hostingy patří:

- Roští (nabízí přímo Node.js hosting)
- Heroku (dovoluje spouštět Node.js aplikace)
- Wedos
- Endora
- Forpsi
- Stable
- Active24

Certifikát je určen pro jednoznačnou identifikaci entity (stránky, služby, uživatele) na internetu. Stojí na **asymetrickém šifrování** a podpisech. **Podepsání** je vlastně akt, kdy je certifikát podepsán (jiným) certifikátem (certifikační autoritou) a tím je schválena jeho existence - pravost.

Certifikát obsahuje **veřejný (encrypt) klíč,** který slouží k šifrování dat. Poté obsahuje **soukromý (decrypt) klíč**, který slouží k rozšifrování dat. K tomu obsahuje ještě **expiraci** daného certifikátu - jedná se o datum, po kterém by se certifikátu nemělo důvěřovat. Obsahuje další informace o tom, kdo jej **podepsal (vydal)**, **pro koho je určen**, pro jaké je domény a podobně.

Certifikační autorita je společnost, která podepisuje cizí certifikáty za určitý obnos. Poté je certifikát platný a kdokoliv si jeho pravost díky klíče certifikační autority ověřit. Tyto služby

jsou často zpoplatněné (poskytují více věcí a služeb) a nebo i zdarma, jako např. Let's Encrypt. **Certifikát může autorita zneplatnit.**

Důvěryhodnost certifikátu je založen právě na tom, že si kdokoliv může **zkontrolovat kdo jej podepsal**. Klíče pro asymetrické šifrování jsou natolik bezpečné na normální počítačích (na kvantových tomu tak být nemusí), **že je nemožné je v rozumném čase dešifrovat i se všemi PC na světě**. Přesto se používá životnost certifikátů, aby se certifikáty často měnili. Když si sám vytvoříte certifikát, tak si jej sám můžete podepsat svým certifikátem. Jedná se tzv. o **self-signed** certifikáty, které se hodí na osobní použíti a prohlížeče a další služby jej berou jako nebezpečný.

Certifikáty se používají všude - např. v HTTPS, VPN, SFTP, SSH a spoustě dalších službách.

Pro lepší představení certifikátů doporučuji tento materiál k dispozici ZDE.

Autorský zákon, nebezpečí internetu

Autorský zákon

Webové stránky jsou chráněny autorským zákonem. Jedná se tedy i o backendové projekty, které často uživatel nevidí. Často se na stránkách uvádí formulace `2022 © Cajthaml, All right reserved`, která je vlastně uveden jen pro upozornění. **Bez uvedení licence je stránka stále pod autorským zákonem**. Vašeho práva jako autora díla není možné se vzdát. Tento termín je však takovým standardem, je velmi dobré jej tam psát.

Před používáním jakékoliv věci je dobré zkontrolovat **zdroj**, **licenci** a **podmínky**, pod kterými můžete danou věc používat. V žádném případě licenci neignorujte - můžete z toho mít velké problémy.

Pro ulehčení licencování věci existují předpřipravené licence, které definují jak se věci mohou používat již předem. Jedná se například o Creative Commons, které definují různé kombinace licencí, které povolují a zakazují různé věci. Jedná se například o **zákaz používat věc pro komerční využití**, nebo že se musí po úpravě díla být použít stejná licence. Druhá věc je **fair use**, kde použijete věc v zájmu sebe, ale uvedete správně zdroj - jedná se např. pro školní účely, nemusíte daného vlastníka o použití informovat.

Nebezpečí internetu

Internet je bez pozornosti obecně velmi nebezpečné místo. Jeden z důvodu, proč tomu tak je, je většinou přisuzována **anonymita**, kdy v podstatě kdokoliv může být kdokoliv.

Nejčastější nebezpečí je závislost na internetu, kde je uživatel **závislý na samotném internetu**. Uživatel je pak na internetu schopen strávit většinu dne a není schopen bez něj žít.

Obecně je nebezpečné **zneužití osobních informací**, do čeho patří údaje, fotografie, vaše pozice nebo třeba i záznamy webkamery a nahrané zvuky. Je důležité své údaje zadávat jen tam, kde je to potřeba a můžete dané společnosti věřit.

Dále se na internetu nachází **podvody**, které mohou vést ke **ztrátě majetku, peněz, dat či zablokování zařízení**. Je důležité si prověřovat, s kým komunikujete.

Nejdůležitější je téma **kyberšikany**, které souvisí i s tématem zneužití osobních informací. Kyberšikana může stát na vydíraní a je dobré se jí bránit.

Malware

Zkratka názvu **malicious software**, který označuje nějaký nevyžádaný software nebo obecně program, který je určen k poškození počítače či uživatele. My si ukážeme jednotlivé typy.

Virus

Často je používaný jako synonymum pro malware. Samovolně se šíří bez vědomí uživatele. Obyčejně obtěžuje, ničí data a šíří se. Nejčastěji pomocí e-mailů a obecně přenosných médií.

Spyware

Určen ke sledování uživatele. Zaznamenává aktivitu uživatele a buď si vytváří log, který je jednou za čas poslán nebo aktivně posílá informace na server. Může se jednat o tzv. keylogger, tj. sledování kliku na klávesnici, ale i například pořizování snímků obrazovky či kamery.

Trojský kůň

Schovává se v zdánlivě nezávadném programu či souboru. Často při pirátění sw, her a obdob. Uživatel nemůže jednoduše zjistit zda se jedná o trojského koně - velikost souboru se značně nemění (uživatel reálnou velikost neví). Často může pomáhat zkontrolovat hash daného souboru. Obsahuje jiné typy malware. Název pochází z řecké mytologie. ZDE najdete článek na Wikipedii.

Často se objevují i v často používaných souborech jako jsou zazipované balíčky (.rar, .zip), který například pro vybalení souborů něco udělá. Počítá se do toho i například Word se svými makry.

Počítačový červ

Šíří se po síti sám a bez vědomí uživatele. Často maže či odesílá soubory. Taktéž se často aktivuje až po čase, po tom, co se rozšíří více - tím zraní více míst / počítačů, což může vést ke větším škodám. Šíří se např. i na routerech.

Adware

Po získání tohoto malwaru se uživateli začnou ukazovat různé nevyžádané reklamy, pop-up okna a další falešné ikony, které většinou žádají uživatele něco koupit, nebo kliknout na stránku, kde bude více reklam, za které útočník dostává zaplaceno. Často je to "bezpečné", je to spíše otravné a může zpomalovat počítač.

Ransomware

Vychází z anglického slova ransom, které znamená výkupné. Po aktivaci toho malwaru se veškeré data na počítači (resp. discích) zašifrují klíčem. Na PC se uživateli ukáže informace o tom, že musí zaplatit v kryptoměně nějakou částku na určitou adresu. Za zaplacení nabízí útočník heslo k rozšifrování. Data nelze bez klíče v rozumném čase rozšifrovat - o data jste buď přišli, nebo věříte, že Vám klíč po zaplacení útočník předá. Když máte štěstí, o data nepřijdete, ale může se stát, že ransomware bude později znovu aktivován - v souborech může nadále existovat, nebo v nich bude rozšířen další malware.

Toto výkupné má často i časový limit, aby to uživatele nutilo vykonat nepromyšlenou akci. Po tomto časovém limitu budou všechny data smazána. Někdy obsahují i časový limit, který často zvedne hodnotu výkupného po uplynutí stanovené doby.

Phishing

Zkratka pro **password harversting fishing**. Jedná se o podvodnou techniku, která se používá k získávání citlivých údajů z různých míst. Často využívá **podvodné e-maily, bannery, stránky, SMS, hovory, administrátoři žádající údaje** a další. Pro důvěryhodnost používají stejný **design**, **kopírují stránky** a **snaží se působit důvěryhodně**. Pro získání údajů je použijí pro získání dalších informací či majetku a peněz. Tato metoda jako jediná využívá takzvaného sociálního inženýrství. Sociální inženýrství má za cíl "hacknout" člověka, nikoliv počítač.

Cílem tohoto úkolu jsou často nezkušení uživatele internetu jakéhokoliv věku. Často se však útočí na starší obyvatele, kteří jsou obecně více důvěřiví. Česká stránka hoax.cz phishingové útoky monitoruje a zveřejnuje, najdete je ZDE.

Ochrana na internetu

Proti nebezpečí na internetu se lze chránit různými způsoby. Nejzákladnější poznámka by měla být ta, že většina nebezpečí vzniká jen pokud to uživatel dovolí. Tedy je obecně dobré si hlídat, co vlastně děláte a pravidelně kontrolovat svůj bezpečností stav. Zjednodušeně neklikejte a nestahujte věci, které neznáte. Není dobré věřit e-mailům od adresátů, které neznáte či neočekáváte.

Je dobré mít na počítači **antivirus** či **antimalware**. **Antivirus** je program, který skenuje počítač pro nebezpečí, které jsou již dlouho známé a má svojí databázi, kterou aktualizuje. Na druhé straně, **antimalware** je program velmi podobný, ale má aktivní kontrolu nebezpečí. Kontroluje tedy často to, co daný program dělá (heuristická analýza) a má velmi obsáhlou a aktualizovanou databázi (stahuje i několikrát denně).

Další důležitá věc je mít vždy **pravidelné aktualizace věškerého softwaru**. Nové aktualizace mimo nové funkčnosti skoro vždy opravují nějaké chyby, které v předchozích verzí vznikli.

Při zadávání údajů je dobré **kontrolovat certifikáty a domény**. Neměli by se zadávat důvěrná data na vyzvání.

HTML

Hyper Text Markup Language, ve zkratce HTML je značkovací jazyk. Nejedná se tedy o klasický programovací (jako C#) nebo skriptovací (JavaScript/Lua) jazyk, ale o jazyk, který jenom definuje určité data a to v XML s vlastními definicemi tagů. Technologií XML se v předmětu WBA zabývat ale nebudeme — není nutné si to ani pamatovat.

HTML bylo představeno v roce 1990. Od té doby vznikli různé verze, ta nejdůležitější je HTML5 z roku 2012, kterou se ve WBA budeme zabývat. Tato verze se hlavně zaměřila na to, aby šel multimediální obsah spouštět na webu. Cílilo se i na to, aby webové stránky byly přehledné i pro počítače a mohli se jednodušeji crawlerovat, resp. indexovat.

HTML se používají zejména pro webové stránky, protože definují různé tagy, které vytváří na stránce různé prvky, které může uživatel vidět. Tyto soubory mají obvykle příponu `.html`.

Komentáře jsou velmi potřebné. V HTML se zapisují jedním způsobem a to pomocí sekvence `<!-- komentář -->`, kde komentář může být cokoliv. Tyto komentáře jsou víceřádkové.

Tag

Tag je značka v HTML. Jména jsou rezervovaná a určují obsah a často i chování prvku na stránce. Definujeme tím tedy to, co se v prohlížeči vykreslí. Tyto tagy se vždy zapisují v špičatých závorkách — `<tag>`.

Výše představený tag je nepárový a není moc používaný. Existuje ale párový tag, který určuje že něco někde začíná a někde končí (vymezuje kde a co). Skládá se ze dvou tagů, kde jeden je počáteční a druhý koncový. Koncový poznáme tak, že po úvodní špičaté závorce se nachází lomeno. Takový zápis tedy vypadá následovně `<tag></tag>`. Mimo tento zápis, daný tag, pokud nebyl dříve definován, neplatí.

Jednotlivé tagy lze spojovat dohromady — vkládat je do sebe, např.

```
<tag1>uz <tag>nevím</tag> co</tag1>
```

Tento zápis znamená, že vše bude označené tag1 a slovo nevím uvnitř bude označeno tagem tag.

Nějakým nedopatřením se však může stát, že se tagy pokazí a nastane křížení tagů. To znamená, že nějaký párový tag začal uvnitř jiného párového tagu, ale ukončí se až někdy mimo něj. Taková ukázka je např.

```
<tag1>uz <tag>nevim</tag1> co</tag>
```

V tomto případě má prohlížeč několik možností. První z nich je ta, že se tagy pokusí opravit. Tento zapsaný tag by se například opravil následovně

```
<tag1>uz </tag1><tag1><tag>nevim</tag></tag1><tag> co</tag>
```

Při těchto změnách se Vám ale velmi pravděpodobně může rozbít web (a případně i styly, kterými se budeme zabývat později). Proto se křížení tagů musí vyřešit již při vytváření

stránky.

Další důležitý pojem je element. Element je kompletní celek, tedy párový tag a jakýkoliv obsah uvnitř něj. Obsah může být cokoliv — další tagy, text či cokoliv jiného.

Tagy jsou definované přesně a my je nebudeme využívat k jiným účelům, než byly vytvořeny.

Atributy

Jakýkoliv tag má atributy. Atributy definují nějaké další data, které popisují přesnější chování prvku. Některé atributy definují chování i pro děti (tj. pod tagy) daného tagu. Tyto tagy jsou většinou speciální pro daný tag a nikde jinde nefungují. Existují však i globální atributy, které lze nastavit v podstatě na cokoliv. Atributy se vždy určují u počátečního tagu. Atribut má vždy své jméno a hodnotu.

Atributy zapisujeme následovně:

```
<tag atribut="hodnota">text</tag>
```

Pro lepší pochopení ukážeme atributy na tagu a, který dle HTML5 definuje na stránce odkaz, na který lze kliknout. Pro určení odkazu používáme atribut `href`. Pro vytvoření odkazu na google by kód vypadal následovně:

```
<a href="https://www.google.com">odkaz na google</a>
```

Mezi zmíněné nejpoužívanější atributy se řadí id a class. Oba dva slouží na stylování a skriptování stránek. Používají se na vybírání prvků na stránce pro další práci. Identifikátor, neboli atribut id, určuje jeden prvek na stránce s daným jménem. Na stránce může být pouze jedenkrát - je unikátní. Třídy, neboli atribut class, je seznam tříd, které mohou být definovány více prvkům.

Soubory na webových stránkách

Na webových stránkách máme různé soubory a některé mají speciální význam. My si ukážeme jen hrst nejpodstatnějších z nich. Z předchozí stránky známe soubory HTML.

Jeden z nejdůležitějších souborů je tzv. index soubor. Jedná se o soubor, který se jmenuje index a má jakoukoliv příponu. Tento soubor se (v základním nastavení většiny webových hostingu) načte právě tehdy, když přistupujeme v URL na složku. Při přístupu server proskenuje složku a dle priority (obvykle např. `php > html > txt > ...`) zobrazí právě indexový soubor.

Na webových stránkách chceme například i obrázky. Nejzákladnější typy jsou rastrové obrázky, které uvnitř ukládají jednotlivé pixely a v případě komprimace nějaké zmenšené data. Druhé jsou vektorové obrázky, které místo jednotlivých pixelů ukládají rovnice, podle kterých se obrázek na počítači vykreslí. Rovnice jsou různého typu a způsobují, že se obrázky mohou nekonečně zvětšovat bez ztráty kvality. Na druhou stranu bývají (když jich máte desítky na webových stránkách), pomalé, protože se musí CPU zapojit v počítání. Pro rastrové obrázky známe přípony (resp. typy) `.jpg, .png, .gif, .webp` a pro vektorové `.svg`.

Soubory na webových stránkách vždy pojmenováváme bez diakritiky a speciálních znaků, bez mezer a nejlépe s malými písmeny. Často mají být názvy krátké a výstižné. Důvod je dobrý — protože se soubory zobrazují v URL, je často nutné aby se v nich uživatel vyznal a případě je byl schopen i napsat. Diakritika je špatná zejména na uživatelé z jiných zemích (asi byste nechtěli psát čínštinou názvy url). Špatné názvy mohou generovat i řadu problému v prohlížeči.

Základní struktura HTML

Základní struktura HTML slouží k definici stránky. Tato struktura se musí používat na všech stránkách (výjimky v podstatě neexistují). Jedná se o základních šest tagů, do kterých se vkládají další. My si jednotlivé tagy ukážeme a popíšeme.

DOCTYPE

Tento tag slouží k určení, že pracujeme s HTML a s verzí HTML5. Tento tag je docela nový a pro ostatní verze (kterými se zabývat nebudeme) je jiný. Tento tag musí být vždy jako první tag stránky z důvodu, že když se stránka načítá, aby na ně prohlížeč mohl ihned reagovat. Pro HTML5 používáme zápis `<!DOCTYPE html>`.

HTML

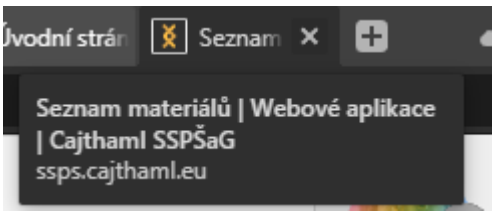
Definuje nám, odkuď kam platí to, že píšeme HTML. Pomocí HTTP se posílají různé soubory a proto je nutné definovat kořenový element. Uvnitř se nacházejí veškeré další tagy. Velmi důležitý je jeho atribut lang, který určuje jaký jazyk se na stránce primárně nachází. Z tohoto tagu prohlížeče čtou to, zda mají zobrazovat možnost překladu. Pro anglický jazyk používáme en, pro český jazyk cs. Tento atribut je velmi klíčový a proto jej musíte správně definovat v každé stránce. Seznam všech zkratek lze nalézt ZDE.

HEAD

Obsahuje definice stránky, které přímo nezobrazují prvky na stránce. Určují ale její vzhled, chování a další metadata. Nic zde definované by se v prohlížeči nemělo přímo vykreslovat.

TITLE

Určuje, jaký nadpisek aktuální stránka má. Vkládá se do hlavičky (tagu head). Tento nadpisek mohou používat crawleři a roboti, ale zobrazuje se i v "tabu" stránky. Tag může obsahovat pouze text.



META

Tyto tagy určují specifické vlastnosti stránky a vkládají se do hlavičky. Jedná se například o kódování textu, šířku pro různá zařízení, SEO či responzivitu. Tento tag má spoustu atributů, ale většina se definuje pomocí atributu name, který určuje to, co se nastavuje a content, což je jeho hodnota.

Pro české stránky většinou definujeme kódování textu na hodnotu UTF-8 pomocí atributu charset. Pro základní fungování stránek na všech zařízeních definujeme metatag s atributem name s hodnotou viewport a content na "šířku zařízení". Ukázka těchto tagů je níže.

BODY

Nejjednodušší z uvedených tagů. Jeho existence je prostá — vkládáme do něj vše, co se na stránce vykreslí, tedy jednotlivé prvky.

Základní struktura tedy vypadá takto:

```
<!DOCTYPE html>
<html>
   <head>
      <title>Smíchovská střední průmyslová školy a gymnázium</title>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
   </head>
   <body>
      <p>Dobrý den!</p>
      <p>Vyzkoušejte <a href="https://google.com">tuto úžasnou stránku</a>!</p>
   </body>
</head>
```

Když byste si tuto stránku uložili do souborového systému Vašeho počítače a otevřeli jej v prohlížeči, uvidíte dva řádky textů. Na jednom pozdrav a na druhém text s odkazem. Taky stránka obsahuje titulek, který můžete vidět v prohlížečovém tabu. Stránka by taktéž fungovala na všech zařízeních.

Základní tagy

HTML obsahuje velmi mnoho tagů a jejich vysvětlení je na spoustu textu, proto budou veškeré tagy popsány velmi stroze s tím, že si student případně neznalé tagy vyzkouší a nebo se podívá do prezentace.

Základní tagy:

`p` - paragraf

`br` - zalomení řádku

`b` - tlustější písmo

`i` - kurzíva

`u` - podtrhnutí

`sup` - horní index

`sub` - dolní index

`s` - přeškrtnutí

`hr` - vodorovná čára

`img` - obrázek, atribut `src` a `alt`

`link` - přidání dalšího souboru

`span` - řádkový prvek

`div` - blokový prvek

`pre` - neformátovaný text

`h1 - h6` - nadpisy, kde h1 je nejdůležitější nadpis, který by se na stránce měl objevit jen jednou

`a` - odkaz, atributy `href`, `target`

`ul / ol / li` - seznamy, `ul` = neseřazený, `ol` = seřazený

`table / tr / td / th / tbody / thead / tfoot` - tabulky, `tr` = řádek tabulky, `td` = buňka, `th` = buňka s nadpisem

`form / input / textarea / label / select / option` - formuláře

a tagy HTML základní struktury

HTML5 tagy pro určení částí webu:

`header`

`nav`

`main`

`section`

`article`

`footer`

CSS

CSS je zkratka pro **Cascading Style Sheets**, neboli kaskádové styly. Jedná se o technologii, která se používá pro stylování a designování webových stránek. Je přímo závislá na technologii HTML. CSS nemá na webových stránkách v podstatě žádnou alternativu a musí se tedy používat. Používají se různými definicemi, které HTML zobrazení upravují.

Jako stylování si můžeme představit velikost písma, barvy pozadí, textu či obrázky a rámečky. S CSS lze vytvořit spoustu věcí, ale má i své výhody a nevýhody.

Výhody:

- umíme zvolit konkrétní prvky, a to již díky třídám tak i identifikátorům
- jednou definované styly můžeme díky identifikátorům a třídám opakovaně používat
- je to moderní technologie, která se díky komunitě stále upravuje a modernizuje
- vše se upravuje na jednom místě a jednoduše, není potřeba používat další technologie

Nevýhody:

- CSS a jeho vlastnosti (více později) jsou závislé na vykreslovacím jádře. Každá stránka tedy může na všech prohlížečích vypadat jinak.
- když se načte soubor s CSS, daná pravidla, které se nevyužívají, se musí načíst také.
- CSS není zpětně kompatibilní, to znamená, že nová verze (nová pravidla) ve starších verzích (resp. prohlížečích) nebudou fungovat.

Zcela jistě by se našlo ještě více výhod a nevýhod, tyto jsou však nejzákladnější a nejdůležitější.

CSS definice se nazývají pravidla. Pravidlo se skládá ze:

- selektoru
- alespoň jedné vlastnosti, která se skládá ze:
  - jména vlastnosti
  - hodnoty vlastnosti

Vlastnosti jsou vlastně nastavení věci, které můžeme použít pro stylování. Za každou vlastnosti se musí psát středník `;`.

Zápisy CSS

CSS lze zapisovat několika způsoby. Všechny způsoby jsou založeny na vlastnostech, které jsme si ukázali.

Inline

Nejjednodušší zápis CSS je, když se CSS zapisuje do atributu style na jednotlivém tagu. Do atributu se zapisují jednotlivé vlastnosti oddělené středníkem. Vypadá to např.:

```
<p style="color: green"></p>
```

Tento způsob nepoužíváme kvůli tomu, že nevyužíváme zcela výhody CSS a ničíme kaskádu, kterou si vysvětlíme později. Tento způsob tedy využíváme jen zřídka a pro velmi (jinak nevyřešitelné) problémy. Tyto styly se načítají jako první.

External

Nejpoužívanější způsob zápisu CSS. Tento způsob využívá nový soubor, ve kterém píšeme pouze CSS. Používáme celé pravidla. Má příponu `.css` a většinou je ve vlastní složce `css`, či obdobě. Tento soubor poté nalinkujeme do HTML pomocí tagu `link`. Daný odkaz musí splňovat cestu ve složkách. Máme možnost odkazovat na aktuální složku pomocí zápisu `./` a zpětně se ve složkách pohybujeme pomocí `../`. Máme možnost i odkazovat na kořenovou URL domény a to pomocí `/`. Tento styl používáme na jakýkoliv typ webu, v našem předmětu zejména na větší projekty, práce či závěrečnou práci.

```
<!DOCTYPE html>
<html lang="cs">
<head>
    <meta charset="UTF-8">
    <title>External</title>

    <link rel="stylesheet" href="style.css"> <!-- aktuální složka, soubor style.css vedle d
    <link rel="stylesheet" href="./css/style.css"> <!-- ve složce css v aktuální složce (ob
    <link rel="stylesheet" href="../style.css"> <!-- v podřazené složce aktuální složky (ob
    <link rel="stylesheet" href="/style.css"> <!-- jsme-li např. na example.com/test/test/t
    <link rel="stylesheet" href="/../style.css">
    <link rel="stylesheet" href="./../../../style.css">
    <link rel="stylesheet" href="../style.css">
</head>
<body>
```

Embed

V tomto stylu zapisujeme CSS přímo do tagu v HTML. Na této stránce se tedy CSS na HTML ihned aplikuje. Nevýhoda je, že nelze daný styl použít na jiné stránce jinak, než že zkopírujeme toto CSS. Tento styl používáme na domácí úkoly či menší weby, kde by další CSS soubor byl zbytečný.

```
<html lang="cs">
<head>
    <meta charset="UTF-8">
    <title>External</title>

    <style>
        selektor {
            color: green
        }
    </style>
```

```
</head>
<body>
```

Import

Tento systém je používaný často na externí CSS soubory. Import je typ příkazu v CSS, který přímo v CSS nalinkuje jiný stylovací soubor dovnitř a začne jej používat. Tento systém není potřeba znát, my jej používat budeme zřídka. Lze používat v Embed nebo External zápisech. Tyto zápisy mají mnohé věci pro responzivitu, které jsou pro nás aktuálně nepodstatné.

```
<html lang="cs">
<head>
    <meta charset="UTF-8">
    <title>External</title>

    <style>
        @import  url("test.css")
    </style>
</head>
<body>
```

Komentáře se v CSS zapisuji jedním způsobem a to pomocí `/* komentář */`.

Selektory

Selektory jsou součástí již zmíněných pravidel. Jak název může vysvětlovat, slouží k tomu, aby se nějaké věci (prvky, tagy, elementy) na stránce vybrali a mohli se stylovat pomocí vlastností.

Selektování tagu

Vybere veškeré tagy na stránce, které jsou daného typu (daného názvu). Na ně poté aplikuje dané vlastnosti. Níže uvedená ukázka vybere všechny tagy b na stránce a nastaví jim tmavě oranžovou barvu textu.

```
b {
    color: darkorange;
}
```

Selektování všeho

Vybere všechny tagy na stránce. Vybírá i poletující text a cokoliv jiného.

```
* {
    color: blue;
}
```

Selektování pomocí tříd

Najde všechny tagy na stránce, které obsahují v atributu class uvedenou třídu. Daný element může obsahovat i další třídy. Níže uvedená ukázka vybere například prvek `<i class="kocka yoko bambi test"></i>`, protože obsahuje třídu `kocka`. Oproti tagu je před názvem třídy tečka. Třídy tedy nesmí obsahovat tečku, mezery a další speciální tagy.

```
.kocka {
    color: yellow;
}
```

Selektování pomocí identifikátoru

Vybere jeden tag na stránce, který má atribut ID stejného jména. Níže uvedená ukázka vybere prvek `<div id="test"></div>`. Identifikátor se pozná pomocí hashtagu v názvu. Identifikátory tento znak (a ani žádné další speciální) nesmí obsahovat.

```
#test {
    color: pink;
}
```

Vnořovací

Slouží k vybrání prvků dle posloupnosti v hierarchickém systému HTML. Poznáte jej díky mezeře mezi tagy, identifikátory, třídy či dalšími selektory. Zápis `div .information` znamená, že se najdou na stránce takové tagy, které splňují že jsou tagu typu `div` a uvnitř v nich (a klidně v dědí dětech, tedy KDEKOLIV uvnitř) mají jakýkoliv tag, který má třídu `.information`. Těchto částí může být nekonečně mnoho.

```
p span {
    color: green;
}
```

Seskupovací

Seskupí více selektorů tag, že se na stránce najdou takové prvky, které dané selektory splňují a je na ně použité stejné pravidlo (resp. vlastnosti). Může znovu obsahovat nekonečno těchto částí. Části (selektory) se rozlišují čárkou mezi nimi.

```
span, b, p .test span {
    color: green;
}
```

Výše uvedená ukázka obsahuje jeden seskupovací selektor, který se skládá ze selektorů `span`, `b` a `p .test span`.

Násobný

Na stránce se velmi často stává, že potřebujeme na stránce vybrat takové prvky, které mají více tříd najednou, mají daný tag a nebo třeba třídu a identifikátor. K vybrání takového prvku používáme již známe věci — tag dáme na začátek selektoru a potom určujeme jednotlivé třídy tečkami. Skládáme je za sebe bez mezer. Identifikátor můžeme dát před třídy nebo za ně.

```
span#main.green.important {
    color: green;
}
```

Výše uvedená ukázka vybere tag span, který má identifikátor `main` a má dvě třídy a to `green` a `important`. Ukázkový selektor je ZCELA ROZDÍLNÝ oproti např. selektoru `span #main.green .important` — je důležité si na mezery dávat pozor.

Přímý dědic

Velmi podobný vnořovacímu selektoru, rozdíl je že místo mezer používáme `>` a rozdílné chování stojí na tom, že místo toho aby KDEKOLIV uvnitř našel následující selektor, tak ho bude hledat přímo jako dítě daného tagu a dovnitř se vnořovat již nebude.

```
span > b {
    color: green;
}
span#main > .green.important > b {
    font-weight: bold;
}
```

Vedlejší potomci

Vybere poslední prvek v selektoru (tj. poslední dítě), které splňuje, že ho předchází nějaký jiný prvek. Např. `.test ~ .test2` vybere každý prvek s třídou `test2`, který předchází prvek s třídou `test`.

```
p ~ a {
    color: gold;
```

```
}
```

Vedlejší přímí potomci

Podobný selektor jako vedlejší potomci, jen předcházející prvek musí být přímo před daným prvkem.

```
p + a {
    color: gold;
}
```

Atributy

Selektorem můžeme vybrat veškeré prvky, které splňují nějaké pravidlo s atributy. Atribut může mít jakékoliv jméno a hodnotu, kterou můžeme zkontrolovat.

```
[attr] // prvek má daný atribut
[attr="value"] // prvek má daný atribut s danou hodnotou
[attr*="ssps"] // prvek má daný atribut a uvnitř obsahuje hodnotu
[attr^="https://"] // prvek má daný atribut a hodnota začíná na hodnotu
[attr$=".org"] // prvek má daný atribut a hodnota končí na hodnotu
[attr~="link"] // prvek má daný atribut a uvnitř obsahuje hodnotu link, chová se stejně, ja
```

Seznam všech selektorů pro atributy naleznete ZDE.

Vlastnosti

Z WBA (a určitě i dalších zdrojů) víme, že v CSS existuje velmi mnoho vlastností. V prezentacích 3. - 9. je jich ukázáno spoustu a proto zde vypíši pouze jejich názvy a rozdělení. Student si případně neznalé vlastnosti vyzkouší a nebo se podívá do prezentace.

Formátování textu:

- `color` — barva textu
- `font-weight` — tloušťka písma
- `font-size` — velikost písma
- `font-family` — určení fontu / písma
- `font-style` — typ písma (např. italics)
- `text-align` — zarovnání písma
- `text-transform` — transformování písma
- `text-decoration` — dekorace textu
- `text-shadow` — stín písma
- `word-spacing` — mezery mezi slovy

- `letter-spaccing` — mezery mezi písmeny
- `line-height` — výška řádku

Blokové styly:

- `display` — určení typu zobrazení
- `background-color` — barva pozadí
- `width` — šířka
- `height` — výška
- `padding` — vnitřní odsazení
- `margin` — vnější odsazení
- `border` — ohraničení
- `border-radius` — zakulacení rohů
- `outline` — ohraničení zvenku (nezvětšuje prvek)

Další:

- `list-style-image` — obrázek odrážek
- `list-style-type` — typ odrážek
- `list-style-position` — pozice odrážek
- `box-shadow` — stín prvku (uvnitř i z venku)
- `background-image` — určení obrázkového pozadí
- `background-position` — pozice pozadí
- `background-repeat` — opakování pozadí
- `background-size` — velikost obrázku
- `background-attachment` — umístění obrázku
- `overflow` — nastavení chování při přetečení
- `visibility` — nastavení viditelnosti prvku
- `opacity` — průhlednost
- `transform` — úpravy zobrazení prvku
- `filter` — filtr přes prvek
- `border-collapse` — spojení ohraničení u tabulek

Flexbox:

- `flex-wrap` — zalomení řádků
- `flex-shrink` — nastavení zmenšovatelnosti prvku
- `flex-grow` — nastavení zvětšovatelnosti prvku
- `flex-basis` — nastavení základní velikosti
- `flex-direction` — směr umístění prvků
- `justify-content` — zarovnání prvků na hlavní ose
- `align-items` — zarovnání prvků na vedlejší ose
- `align-self` — zarovnání aktuálního prvku v rodiči

- `gap` — mezery mezi prvky (row-gap, column-gap)
- `order` — určení pozice prvku v rodiči

Grid:

- `gap` — viz. výše
- `grid-template-columns` — určení sloupců, je možné použít opakování repeat
- `grid-template-rows` — určení řádků, je možné použit opakování repeat
- `grid-row-start` — určení začátku řádku
- `grid-row-end` — určení konce řádku
- `grid-column-start` — určení začátku sloupce
- `grid-column-end` — určení konce sloupce
- `justify-items` — zarovnání prvků uvnitř buňky
- `align-items` — zarovnání prvků uvnitř buňky
- `justify-content` — zarovnání celého gridu v místě, které mu je přiděleno

Jednotky

V CSS se používá mnoho jednotek, které definuji velikost a další věci. Jednotky obecně zařazujeme do kategorií absolutní či relativní. Absolutní jsou ty, které se velikosti zařízení či na ničem jiném nezáleží.

Absolutní:

- `px` — pixel na obrazovce
- `cm` — definice centimetru, existuje ještě `mm`
- `in` — palec, 1in ≐ 2.54cm
- `pt` — bod, 1pt = 1/72in
- `pc` — pica, 1pc = 12pt

Relativní:

- `%` — procentní hodnota rodiče
- `em` — velikost textu, 1em = aktuální velikost fontu
- `rem` — velikost textu hlavního prvku stránky
- `ch` — šířka znaku nuly pro aktuální písmo
- `vw` — 1vw = 1% šířky zařízení
- `vh` — 1vh = 1% výšky zařízení
- `fr` — jedna část (zlomek) v gridu

Barvy v CSS můžeme určit několika způsob:

- **NÁZEV:** dle názvů, které naleznete ZDE.
- **RGB:** `rgb(x, y, z)`, kde jednotlivé znaky jsou čísla od 0 do 255, kde hodnoty `0, 0, 0` je černá barva, 255, 255, 255 je bíla.
- **RGBA:** `rgba(x, y, z, a)`, kde a je číslo od 0 do 1, kde 0 znamená zcela průhledná barva a 1 neprůhledná.

- **HEX:** `#abcxyz`, kde jsou po dvou vždy značí hexadecimální číslem pro červenou, zelenou a modrou barvu, stejně jako u rgb, určuje se hodnota 0 až 255 ale v hexadecimální zápisu.
- **HEXA:** `#abcxyzde`, kde de značí průhlednost, stejně jakou o rgba.
- a další

Pozicování

Umístění na webové stránce slouží k uložení prvků stránky na jinou pozici než mu určila stránka nebo jiné rozložení. Všechny různé umístění se ovládají pomocí vlastností `top`, `bottom`, `left` a `right`. Nyní si ukážeme veškeré základní umístění:

Statické pozicování je to klasické pozicování které je základně nastaveno. Je to jediné pozicování neumí pracovat s vlastnostmi `top`, `bottom`, `left` a `right`.

Relativní pozicování je velmi podobné tomu statickému a v funguje na něm pozicování pomocí vlastnosti `top`, `bottom`, `left` a `right`. Prvek se umístí podle pozicování statického a výše zmíněnými vlastnostmi jej odsadíme od této pozice.

Fixované pozicování se váže na velikost obrazovky a při scrollování na webové stránce prvek zůstane umístěn dle obrazovky. Toto pozicování také způsobí že prvek se umístí na jinou úroveň stránky a tím pádem neovlivňuje žádné další prvky.

Absolutní pozicování způsobí že se to zachytí na nejbližší nadřazený prvek na kterému je nastaveno nějaké pozicování tedy kromě toho statického. Pozicování znova způsobí, že neovlivňuje žádné další prvky.

Lepící pozicování je vlastně přechod dvěma stavy pozicování a to relativním a fixovaným. Dokud stránka nesplní odsazení daného lepícího prvků bude se prvek chovat jako relativním pozicováním. Až prvek splní odsazené trošku fixované pozicování a tady prvek bude zachycen dle velikosti obrazovky.

Na stránce taky můžeme změnit vrstvu prvku ručně a to pomocí vlastnosti z-index. Vlastnosti předáváme hodnotu číselnou která určuje v jaké vzdálenosti prvek je. Čím větší číslo tím výše bude prvek na ose Z umístěn.

Pseudotřídy a pseudoelementy

Pseudotřídy jsou vlastně speciální selektor, který ukazuje na nějaký stav prvku na stránce. Jako ukázku si můžeme uvést například najetí myši na prvek nebo například zda je prvek n-tý v rodiči. Pseudotřídy zapisujeme pomocí `:` a dokonce je můžeme navazovat.

Nejzákladnější typ pseudotříd jsou navigační, které reprezentují nějakou pozici v stromové struktuře stránky, do kterých patří například:

- `:nth-child(4)` — čtvrté dítě
- `:nth-child(2n)` — každé druhé dítě
- `:first-child` — první dítě
- `:only-child` — prvek, který je jediné dítě

- `:first-of-type` — první prvek daného tagu

Další typ jsou aktivovatelné pseudotřídy které se aktivují po nějaké uživatelské aktivitě:

- `:hover` — najetí na prvek
- `:active` — prvek je aktivní
- `:focus` — prvek je zabraný (např. TABem)

Často používaným selektorem je negace, která zneguje zadaný selektor. Selektor používáme pomocí `:not(jinySelektor)`.

Seznam všech pseudotříd naleznete ZDE.

Pseudoelementy se často zaměňují za pseudotřídy, samotné pseudoelementy jsou speciální selektory, které vybírají neexistují prvky na stránce. Neexistujícími prvky se myslí např. první písmeno, první řádek, prvek uvnitř jako první, a další. Pro pseudoelementy používáme `::`.

Jako ukázku můžeme uvést:

- `::selection` — určení stylů vybraných částí textu uživatelem
- `::first-line` — vybrání první řádky
- `::first-letter` — vybrání prvního písma
- `::after` — vybere prvek uvnitř jako poslední
- `::before` — vybere prvek uvnitř jako první

Seznam všech pseudoelementů naleznete ZDE.

Priorita stylování

Jedna vlastnost kaskády je určování priority stylování. Priority se počítají dle následujícího schématu, kdy první prvky mají větší prioritu než nižší:

- !important
- inline styl
- identifikátory
- třídy, atributy, pseudotřídy
- tagy a pseudoelementy
- styl rodiče

Responzivita

Responzivita je způsob úpravy stránek tak, aby byly zobrazitelné na všech zařízeních. Jedná se např. o desktop, handheldy, tablety, hodinky a další. Při tvorbě responzivity se často zaměřujeme zejména na různé šířky zařízení.

V minulosti se stránky responzivní nedělali, protože přenosné zařízení nebyly tak běžné (resp. neměli internet). Často se taktéž tvořili separátní stránky pro telefony a desktop (viz. např. Alza).

Při responzivitě často mluvíme o tom, že stránky tvoříme tekutě, tedy, že se různé prvky vlévají do kontejnerů. Používáme taktéž relativní jednotky (`vh`, `%`, `vw`, ...), které práci zjednodušují. Taktéž je nutné používat flexbox a grid. Pro responzivitu se hodí používat DevTools, které se na jakékoliv zařízení umí změnit. V praxi jako nejmenší šířku používáme 320px. Pro responzivitu je nutné zapnout v `meta` tagu `viewport` se šířkou dle zařízení.

Pro tvorbu používáme tzv. media queries, které při určité podmínce zaktivují pravidla na stránce. Podmínka může být např. šířka či výška zařízení. Responzivitu nejčastěji píšeme na konci stránky (resp. za jednotlivými pravidly), aby měli větší prioritu.

```
@media (min-width: 200px) {
// pravidla se zaktivují pokud je stránka větší jak 200px
}

@media (max-width: 900px) {
// stránka menší jak 900px
}

@media (min-width: 900px) and (max-width: 1200px) {
// mezi 900 a 1200px
}

@media print {
// když se stránka tiskne
}
```

Většinou stránky tvoříme návrhem desktopu a poté se budujeme k menším zařízením. Jde však i opak. Důležité je určit breakpointy, což jsou body stránky, které určují, kdy se prvky začnou přeuspořádat, těch je dobré mít tak 4, ale každý projekt je jiný.