A web működésének alapjai. Web szabványok és szabványügyi szervezetek. URI-k és felépítésük. HTTP: kérések és válaszok felépítése, metódusok, állapotkódok, tartalomegyeztetés, sütik. A web jelölőnyelvei: XML és HTML dokumentumok felépítése. Stíluslap nyelvek. JSON.

Web szabványok

- De facto szabványok: a gyakori használatból vagy a piaci elfogadottságból származnak
- De jure szabványok: helyi, állami és/vagy nemzetközi szintű szabályozók által kötelezőként előírt szabványok
- Önkéntes közmegegyezéses szabványok: különböző magánintézmények által meghatározott szabványok

Szabványügyi szervezetek

- *Web szabványokért felelős szervezetek (1): IANA*
  - A név jelentése kb. „Internet számkiosztó hatóság"
  - Az Internet működésének alapjául szolgáló kódok és számok kiosztását koordinálja
  - Az IP-címek kiosztásának globális koordinálása
  - Nyilvántartja a különféle Internet protokollokhoz használt kódokat és számokat
- *Web szabványokért felelős szervezetek (2): IETEF, az RFC sorozat*
  - A név jelentése kb. Internet mérnöki munkacsoport
  - Internet szabványokat fejlesztő nemzetközi szabványügyi szervezet
  - Az Internet szabványokhoz kötődő specifikációkat az RFC dokumentumsorozatban publikálja
  - RFC: Az RFC sorozat az Internetről szóló műszaki és szervezeti dokumentumokat tartalmaz. Minden RFC-t egy szám azonosít. A kiadott RFC-k soha nem módosulnak. A különféle hibákat hibajegyzékek javítják. Változtatások egy javított RFC írásával és kiadásával is eszközölhetők.
- *Web szabványokért felelős szervezetek (3): W3C, W3C szakmai jelentések érettségi szintjei*
  - W3C: A W3C egy nemzetközi közösség, ahol tagszervezetek, főállású alkalmazottak és a nyilvánosság munkálkodnak együtt webszabványok fejlesztésén. […] A W3C küldetése a web lehetőségeinek maximális kiaknázása. A W3C ajánlásoknak nevezett, webtechnológiákat meghatározó és webszabványoknak számító dokumentumokat publikál
  - Web mindenkinek: a web elérhető kell, hogy legyen mindenki számára, a hardverektől, szoftverektől, hálózati infrastruktúrától, anyanyelvtől, kultúrától, földrajzi elhelyezkedéstől, vagy a fizikai vagy szellemi képességektől függetlenül

    - Web mindenhol: a web elérhető kell, hogy legyen a legkülönfélébb eszközökről
    - W3C szakmai jelentések érettségi szintjei: Munkaterv: a közösség általi áttekintésre közzétett dokumentum, beleértve a W3C tagokat, a nyilvánosságot és más műszaki szervezeteket. Néhány, de nem minden munkaterv célja, hogy ajánlássá lépjen elő. Munkacsoport feljegyzésként ajánlott közzétenni azokat a munkaterveket, melyeket nem, vagy már nem szándékoznak ajánlássá előléptetni. Előzetes javaslatterv: már széles körben áttekintett dokumentum, melyet implementációs tapasztalatok szerzése céljából tesznek közzé. Javaslatterv: egy olyan dokumentum, mely megfelelő minőségű ahhoz, hogy ajánlássá váljon. Ajánlás: széles körben alkalmazható webszabvány. Munkacsoport feljegyzés: olyan dokumentum, melyet nem szánnak hivatalos szabványnak, vagy ajánlás készítése nélkül félbehagyott munkát dokumentál. Túlhaladott ajánlás: olyan specifikáció, melyet egy újabb verzió helyettesít.

URI-k és felépítésük

- *Mi az URI?*
    - Egységes erőforrás-azonosító
    - Absztrakt vagy fizikai erőforrást azonosító tömör karaktersorozat
    - Minden URI egy sémanévvel kezdődik, melyet egy ':' karakter választ el a séma-specifikus résztől
- *URI sémák*
    - file
    - http/https
    - mailto
- *URI szintaxis, a host, port, útvonal, lekérdezés és erőforrásrész-azonosító komponensek*
    - URI szintaxis: Hierarchikus felépítés. A komponensek felsorolása balról jobbra haladva fontosság szerint csökkenő sorrendben történik
    - Útvonal: Útvonal részek '/' karakterekkel elválasztott sorozata, amely lehet üres. Az első '?' vagy '#' karakterig, ezek hiányában pedig az URI végéig tart. Az állományrendszerekben megszokott módon használhatóak útvonal részként '.' és '..'
    - Lekérdezés komponens: A '?' karakter jelzi az elejét, a '#' karakterig, annak hiányában pedig az URI végéig tart. Nem hierarchikus adatokat tartalmaz. Gyakran név '=' érték formájú, '&' karakterekkel elválasztott név-érték párokat tartalmaz

    - Erőforrásrész-azonosító: A '#' karakter jelzi az elejét, az URI végéig tart. Lehetővé teszi egy másodlagos erőforrás közvetett azonosítását egy elsődleges erőforrásra történő hivatkozáson keresztül. Jelentését az elsődleges erőforrás elérése során kapott lehetséges reprezentációk határozzák meg, ezek média-típusa. Hivatkozás-feloldás során mindig eltávolításra kerül
- *Abszolút URI, URI-hivatkozás, relatív hivatkozás*
    - Abszolút URI: olyan URI, amely nem tartalmaz erőforrásrészazonosítót
    - URI-hivatkozás: URI vagy relatív hivatkozás
    - Relatív hivatkozás: kb. egy URI séma-specifikus része, vagy annak egy megfelelő végszelete (lehet akár az üres karakterlánc is)

HTTP: kérések és válaszok felépítése, metódusok, állapotkódok, tartalomegyeztetés, sütik

- *A HTTP jellemzői*
    - A kliens-szerver modellen alapuló kérés-válasz protokoll
    - Állapotnélküliség: az egymást követő kérések egymástól függetlenként kezeltek
    - Kiterjeszthetőség: Például metódusok, állapotkódok, fejlécmezők
    - Általános célú: Kliensek és webszerverek közötti kommunikációhoz használják elsősorban, de elvileg tetszőleges egyéb célra felhasználható
- *Alapvető fogalmak: erőforrás, reprezentáció, tartalomegyeztetés, üzenet, payload, kliens, szerver, eredet szerver, közvetítő, proxy, munkamenet*
    - Erőforrás: egy HTTP kérés célja, melyet egy URI azonosít
    - Reprezentáció: Olyan információ, mely egy adott erőforrás múltbeli, jelenlegi vagy kívánt állapotát hivatott tükrözni. A protokollon átvihető formában van. Reprezentáció metaadatokból és reprezentáció adatokból áll
    - Tartalomegyeztetés: Egy eredet szerver számára egy erőforráshoz több, annak jelenlegi állapotát tükröző reprezentáció állhat rendelkezésre, vagy képes lehet több reprezentáció előállítására. A tartalomegyeztetés egy olyan mechanizmus, mely révén kiválasztható egy adott kéréshez legmegfelelőbb reprezentáció. Ezt a legmegfelelőbb reprezentációt kiválasztott reprezentációnak nevezzük
- *a http és https URI-séma*
    - Erőforrások adott számú TCP porton figyelő eredet szervereken történő azonosítására szolgálnak

    - Egy URI eredet szerverét a host és az opcionális port komponense azonosítja
    - Ha adott egy http vagy https URI, abból nem következik, hogy mindig egy HTTP szerver figyel az adott hoston és porton
    - Különbözőnek tekintendőek a http és a https URI-sémákon keresztül elérhető erőforrások
- *Üzenet formátum, kérések és válaszok felépítése*
    - Kétféle üzenet: Kérés. Válasz
    - Feldolgozásuk oktettek sorozataként kell, hogy történjen
    - A kérések és válaszok felépítésüket tekintve csupán az első sorukban térnek el
    - Egy kérés kezdősora az alábbi felépítésű: metódus kérés-cél HTTP-verzió CRLF. Egyetlen szóköz karakterrel kötelező elválasztani a sorban a komponenseket. A kérés-cél a cél erőforrást azonosítja, melyre a kérés vonatkozik
    - Az üzenet első sorát állapotsornak nevezzük, felépítése az alábbi: HTTP-verzió állapotkód indok_frázis CRLF. Egyetlen szóköz karakterrel kötelező elválasztani a sorban a komponenseket. Az állapotkód egy háromjegyű decimális egész szám, melyet az állapotkód tömör szöveges leírása követ
- *HTTP metódusok: GET, HEAD, POST, PUT, DELETE*
    - GET: A cél erőforrás egy aktuális kiválasztott reprezentációjának átvitelét kérelmezi. Információ lekérési célokra szolgál. Egy kliens úgy módosíthatja a GET jelentését egy kérésben a Range fejlécmező küldésével, hogy az csak a kiválasztott reprezentáció bizonyos részeinek átvitelét kérelmezi. Egy GET kérésre adott válasz gyorsítótárazható
    - HEAD: Azonos a GET metódussal, azonban a szerver nem küldhet üzenettörzset a válaszban. Úgy használható metaadatok szerzésére a kiválasztott reprezentációról, hogy reprezentáció adatok nem kerülnek átvitelre. Egy HEAD kérésre adott válasz gyorsítótárazható
    - POST: Azt kérelmezi, hogy a cél erőforrás dolgozza fel a kérésben mellékelt reprezentációt a saját szemantikájának megfelelően. POST kérésekre adott válaszok csak akkor gyorsítótárazhatók, ha explicit frissességi információt tartalmaznak
    - PUT: Azt kérelmezi, hogy a szerver hozza létre vagy helyettesítse a cél erőforrás állapotát a kérésben mellékelt reprezentáció által definiált állapottal. Egy PUT kérés sikeres végrehajtása egy adott reprezentációra arra enged következtetni, hogy egy következő GET kérés ugyanarra a cél erőforrásra egy 200 (OK) állapotkódú válaszban elküldött ekvivalens reprezentációt eredményez
    - DELETE: Azt kérelmezi, hogy az eredet szerver törölje a cél erőforrás és aktuális funkcionalitása közötti kapcsolatot. Sikeres végrehajtás

esetén a válaszban állapotkódként 200 (OK), 202 (Accepted) vagy 204 (No Content) ajánlott

- *Állapotkódok, állapotkódok fajtái*
  - Háromjegyű decimális egész számok.
  - Az első számjegy az állapotkód (válasz) fajtáját határozza meg.
  - A klienseknek nem kell megérteniük minden regisztrált állapotkód jelentését
  - Az állapotkódok kiterjeszthetőek
  - Az IANA adminisztrálja az állapotkódokat
  - 1xx: Informational (információs): A végső válasz küldését megelőző előzetes választ jelez
  - 2xx: Success (siker): A szerver sikeresen megkapta, megértette és elfogadta a kérést
  - 3xx: Redirection (átirányítás): A kérés kiszolgálásához a felhasználói ágens további műveletet kell, hogy végrehajtson, ez történhet automatikusan
  - 4xx: Client Error (kliens hiba)
  - 5xx: Server Error (szerver hiba)
- *Tartalomegyeztetés, proaktív és reaktív egyeztetés*
  - Egy eredet szerver számára egy erőforráshoz több, annak jelenlegi állapotát tükröző reprezentáció állhat rendelkezésre, vagy képes lehet több reprezentáció előállítására
  - A tartalomegyeztetés egy olyan mechanizmus, mely révén kiválasztható egy adott kéréshez legmegfelelőbb reprezentáció
  - Proaktív: a szerver választja ki a reprezentációt a felhasználói ágens kifejezett preferenciái alapján. Ezt hívják szerver-vezérelt tartalomegyeztetésnek is.
  - Reaktív: a szerver választásra kínálja fel a felhasználói ágensnek a reprezentációk listáját. Ezt hívják ágens-vezérelt tartalomegyeztetésnek is
- *Sütik: definíció, felhasználások, a Set-Cookie és Cookie fejlécmezők, süti attribútumok (Expires, Max-Age, Domain, Path, Secure, HttpOnly), sütik kezelése, harmadik féltől származó sütik*
  - Süti: Egy név-érték pár és kapcsolódó metaadatok (attribútumok), melyeket egy eredet szerver egy válasz Set-Cookie fejlécmezőjében küld a felhasználói ágensnek
  - Amikor a felhasználói ágens egy Set-Cookie fejlécmezőt kap, eltárolja az attribútumaival együtt
  - A továbbiakban, amikor a felhasználói ágens egy HTTP kérést hajt végre, a Cookie fejlécmezőbe illeszti az alkalmazható, nem lejárt sütiket
  - Expires: a süti lejáratának dátumát és idejét adja meg
  - Max-Age: azt adja meg, hogy hány másodperc múlva jár le a süti

    - Domain: Meghatározza, hogy a süti mely szervereknek lesz elküldve
    - Path: A süti hatáskörét adott útvonalakra korlátozza
    - Secure: A süti hatáskörének biztonságos csatornákra korlátozása
    - HttpOnly: HTTP kérésekre korlátozza a süti hatáskörét
    - A felhasználói ágenseknek: Törölniük kell a lejárt sütiket. Az aktuális munkamenet végén törölniük kell az összes nem perzisztens sütit. Ajánlott a felhasználók számára lehetővé tenni a tárolt sütik kezelését. Például egy adott időszakban fogadott vagy egy adott tartományhoz kapcsolódó összes süti törlését. Ajánlott a felhasználók számára lehetővé tenni a sütik letiltását
    - A sütiket gyakran bírálják azért, mert lehetővé teszik a szerverek számára a felhasználók követését. Az úgynevezett harmadik féltől származó sütik különösen problémásak. Egy HTML oldal megjelenítése során egy felhasználói ágens gyakran kér le erőforrásokat más szerverekről. Ezek a harmadik félnek számító szerverek sütiket használhatnak a felhasználó követésére még akkor is, ha a felhasználó közvetlenül soha nem látogatja meg őket
- *Felhasználó követés: definíció, az alapjául szolgáló információk, a Referer fejlécmező, védekezés a követés ellen*
    - Egy lehetséges definíció: Egy adott felhasználó több különböző kontextuson keresztüli tevékenységével kapcsolatos adatgyűjtés és az ebből a tevékenységből származtatott adatok megőrzése, felhasználása vagy megosztása azon kontextuson kívül, melyben a tevékenység történt. Egy kontextus olyan erőforrások egy összessége, melyek ugyanazon fél ellenőrzése vagy több fél közös ellenőrzése alatt állnak
    - Az alábbiakon alapulhat: IP-cím. Sütik. Az ETag fejlécmező. Eszköz ujjlenyomat (operációs rendszer, képernyőfelbontás, telepített betűkészletek, …)
    - Referer fejlécmező: Lehetővé teszi a felhasználói ágens számára, hogy megadja azt az erőforrást azonosító URIhivatkozást, melyből a cél URI származik. Példa a használatra: Referer: https://www.w3.org/ Tilos a Referer fejlécmező küldése nem biztonságos HTTP kérésben, ha a mezőértékben jelzett oldal biztonságos protokollon keresztül érkezett
    - Védekezés a követés ellen: A Referer fejlécmező küldésének tiltása. Harmadik féltől származó sütik elfogadásának letiltása. Privát böngészés/inkognitómód
- *Kapcsolatkezelés: perzisztens kapcsolatok, kapcsolatok lezárása, egyidejű kapcsolatok és a sor eleji blokkolás problémája*
    - Perzisztens kapcsolatok: A HTTP/1.1 vezette be. Lehetővé teszik több kérés és válasz átvitelét egyetlen TCP kapcsolaton át. A HTTP/1.1 alapértelmezetten perzisztens kapcsolatokat használ

- Kapcsolat explicit lezárása: A Connection fejlécmező biztosít egy close opciót, mellyel a küldő jelezheti, hogy az aktuális kérés/válasz befejezése után lezárásra kerül a kapcsolat
- Időtúllépés: A szerverek általában van valamiféle várakozási ideje, melyen túl nem tartanak fenn tovább egy inaktív kapcsolatot
- Egyidejű kapcsolatok: A legtöbb szervert úgy tervezték, hogy képes legyen sok ezer egyidejű kapcsolatot fenntartani. A legtöbb kliens több kapcsolatot tart fenn párhuzamosan, egy szerverhez akár többet is
- Sor eleji blokkolás problémája: Jellemzően a sor eleji blokkolás problémájának elkerüléséhez használnak több kapcsolatot

A web jelölőnyelvei: XML és HTML dokumentumok felépítése

- *Mi az XML?*
  - Általános célú jelölőnyelv
  - Születése az 1990-es évek második felében
  - Napjaink egyik meghatározó, az iparban széles körben használt technológiája
  - Szűkebb értelemben: Szintaxis strukturált dokumentumok ábrázolására, mely lehetővé teszi azok automatikus feldolgozását (elektronikus dokumentum formátum)
  - Tágabb értelemben: Egy sereg közös tőről fakadó specifikációt jelent, melyeket összefoglaló néven XML családnak is neveznek
- *Jelölőnyelvek*
  - A jelölőnyelvek szöveg annotálására szolgáló számítógépes nyelvek
  - Lehetővé teszik szövegrészekhez metaadatok megadását a szövegtől jól elkülöníthető módon
- *Az XML és a HTML összehasonlítása*
  - XML: Nincs előre definiált címkekészlet, a címkék a jelentésre vonatkoznak
  - HTML: Előre definiált címkekészlet használata, a címkék a megjelenítésre vonatkoznak, tekinthető az XML egy speciális alkalmazásának (XHTML)
- *Az XML előnyei és hátrányai*
  - Előnyök: Egyszerűség, nyíltság, gyártófüggetlenség, platformfüggetlenség, univerzális adatcsere formátum, kiterjedt infrastruktúra, az iparban de-facto szabvány
  - Hátrányok: Bőbeszédű és nehézkesen használható szintaxis, nagy tárigény, bonyolultság, mindezek ellenére fontos, együtt kell élni vele
- *Dokumentum központú és adatközpontú XML*

    - Dokumentum központú: A dokumentumokat jelölésekkel megtűzdelt folyó szöveg alkotja. A dokumentumok szerkezete nagy változatosságot mutat. Lényeges az elemek sorrendje. Az ilyen dokumentumok tartalma elsősorban emberi fogyasztásra szánt. Ilyen alkalmazás például az XHTML
    - Adatközpontú: A dokumentumokat nagyszámú adatelem alkotja. Kevésbé véletlenszerű dokumentum-szerkezet. Az elemek sorrendje kevésbé lényeges. Az ilyen dokumentumok elsősorban gépi feldolgozásra szántak. Ilyen alkalmazás például az SVG

- *Mi a HTML?*
    - „A HTML a Web elsődleges leíró nyelve."
    - „egy szemantikai szintű leíró nyelv és a kapcsolódó szemantikai szintű alkalmazásprogramozási interfészek a Weben elérhető oldalak készítéséhez, melyek a statikus dokumentumoktól a dinamikus alkalmazásokig terjednek."
- *A HTML5 fejlesztése, WHATWG*
    - A Web fejlődése iránt elkötelezett közösség, mely böngészőkben implementálható szabványokat fejleszt
    - 2004-ben alapították az Apple, a Mozilla Foundation és az Opera Software programozói, akik elégedetlenek voltak a W3C a HTML fejlesztésére irányuló tevékenységével
    - Működését az irányítócsoport (Steering Group) koordinálja, melynek tagjai az Apple, Google, Microsoft és a Mozilla
    - A HTML legutóbbi verziója
- *HTML5 szabványok, eltérések a WHATWG és a W3C szabvány között*
    - Eredetileg a WHATWG fejlesztette ki a HTML5 specifikációt
    - A W3C 2007-ben kapcsolódott be a HTML5 fejlesztésébe
    - 2012 júliusa óta a WHATWG és a W3C is külön specifikációt fejleszt, melynek során eltérő fejlesztési modellt követnek
    - A W3C a HTML 5.2 specifikáció ajánlásként való kiadása után jelenleg a következő verzión dolgozik (HTML 5.3)
    - A WHATWG specifikációja soha nem lesz lezárt, folyamatosan fejlesztik („élő szabvány")
    - Eltérések: Ellentmondás. Csak a WHATWG specifikációban lévő elemek. Csak a W3C specifikációban lévő elemek. Globális attribútumok. Gépi feldolgozásra szánt adatok beágyazása
- *HTML5 elemek*
    - Az elemeknek, attribútumoknak és attribútumértékeknek meghatározott jelentése (szemantikája) van
    - A szerzők számára tilos az elemek, attribútumok és attribútumértékek a megfelelő rendeltetésüktől eltérő jelentésbeli céllal történő használata

    - A HTML előző verzióiban rendelkezésre álló prezentációs lehetőségek többség többé nem megengedett
    - Csak a style attribútum és a style elem maradt meg, mint prezentációs jelölési lehetőség
    - A specifikációban definiált minden egyes elemnek van egy tartalommodellje
- *Document Object Model (DOM)*
    - Egy DOM fa egy dokumentum memóriabeli ábrázolása
    - A DOM egy alkalmazásprogramozási interfész (API) dokumentumok (főleg HTML és XML dokumentumok) eléréséhez és manipulálásához
    - Minden ilyen dokumentumot egy fa ábrázol, mely csomópontokból áll
    - Minden egyes csomópontot egy API-val rendelkező objektum ábrázol, így tehát manipulálható
    - A DOM interfészek Web IDL-ben kerülnek leírásra
- *HTML5 szintaxisok: HTML szintaxis, XML szintaxis*
    - Kötelező a dokumentumtípus-deklaráció
    - Az elem- és attribútumnevek kisbetű-nagybetű érzéketlenek
    - Az elemek és attribútumok neveinek megadásakor (még az idegen elemeknél is) tetszőlegesen keverhetők a kis- és nagybetűk
    - Ugyanabban a nyitó címkében soha nem fordulhat elő két vagy több olyan attribútum, melyek nevei kisbetű-nagybetű érzéketlen hasonlítás esetén megegyeznek
    - Nem idézett attribútumérték szintaxis: Ha egy, az üres karakterlánctól különböző attribútumérték nem tartalmaz egyetlen literális whitespace karaktert sem, akkor megadható az attribútumérték-határolók elhagyásával
    - Logikai attribútumok: Sok attribútum logikai. Egy logikai attribútum jelenléte egy elemen az igaz értéket ábrázolja, hiánya pedig a hamis értéket. Ha megjelenik az attribútum, akkor értéke az üres karakterlánc kell, hogy legyen, vagy egy olyan érték, mely kisbetű-nagybetű érzéketlen hasonlítás esetén megegyezik az attribútum nevével
    - Az idegen elemeknek vagy egy nyitó és egy záró címkéjük van, vagy egy önlezáróként jelölt nyitó címkéjük, mely esetben nem lehet záró címkéjük
    - Nem támogatottak a névtér-deklarációk, még idegen elemekhez sem
    - CDATA-szakaszok csak idegen tartalomban (MathML vagy SVG) használhatók
- *Miért nincs DTD a HTML5-höz?*

    - DTD-k és XML sémák nem képesek kifejezni a HTML5 által támasztott valamennyi megfelelési követelményt
- *A HTML5 dokumentumtípus-deklaráció*
    - A HTML szintaxisban a <!DOCTYPE html> dokumentumtípus-deklaráció szükséges, melynek célja mindössze annak biztosítása, hogy a dokumentum megjelenítése a szabványos módban történjen
    - Az XML szintaxisban tetszőleges dokumentumtípus-deklaráció használható, megadása nem is kötelező
- *Mi az SVG?*
    - Nyelv két-dimenziós vektorgrafika XML-ben történő leírásához
    - SVG tartalom beágyazható más dokumentumokba
- *Mi a MathML?*
    - Nyelv matematikai jelölések XML-ben történő leírásához
    - Célja lehetővé tenni a Weben matematika „felszolgálását", fogadását és feldolgozását, miként a HTML lehetővé tette ezt a funkcionalitást szöveghez
    - Leírható segítségével matematikai kifejezések megjelenítése és jelentése is
    - MathML tartalom beágyazható más dokumentumokba
- *Böngészőmotorok megjelenítési módjai*
    - Kompatibilitási mód: régi böngészők viselkedésének utánzása (emulálása) az aktuális webszabványokat sértő módon régi weboldalak megjelenítéséhez
    - Szabványos mód: a weboldalak megjelenítése az aktuális webszabványoknak megfelelően
    - Majdnem szabványos mód: néhány böngészőmotor rendelkezik egy harmadik üzemmóddal is, mely bizonyos magasságok meghatározásában tér el a szabványos módtól, mely például képek táblázatcellákban való elhelyezését érinti
    - A megjelenítési módok létezésének történeti oka, hogy a korai webszabványok nem voltak kompatibilisek az akkoriban létező böngészők viselkedésével

Stíluslap nyelvek

CSS

- *A CSS szerepe*
    - Strukturált (például HTML és XML) dokumentumok megjelenítésének leírására szolgáló stíluslap nyelv
    - Többféle eszközön történő megjelenítést támogat, mint például képernyők, nyomtatók, Braille eszközök

    - Szétválasztja a dokumentumok megjelenítési stílusát a dokumentumok tartalmától
    - Ilyen módon egyszerűsíti a webszerkesztést és a webhelyek karbantartását
- *A CSS fejlesztése, CSS szintek*
    - Fejlesztés: A W3C CSS Munkacsoportja fejleszti
    - CSS szintek: A szó hagyományos értelmében a CSS-nek nincsenek verziói, hanem szintjei vannak. A CSS minden egyes szintje az előzőn alapul, annak definícióit finomítja és új lehetőségeket vezet be
    - CSS Level 1: elavultnak tekintett
    - CSS Level 2: A CSS 2.1 specifikáció (egyetlen dokumentum) definiálja, javítása jelenleg fejlesztés alatt
    - CSS Level 3: Jelenleg is fejlesztés alatt áll. Moduláris felépítésű. A moduloknak is szintjeik vannak. A CSS modulok eltérő stabilitási szintűek
    - CSS Level 4: Nem létezik CSS Level 4. Önálló modulok elérhetnek a 4. vagy egy magasabb szintre, de a CSS nyelvnek már nincsenek a harmadikon túli szintjei. A CSS Level 3 kifejezést csak a korábbi monolitikus verzióktól való megkülönböztetésre használják.
- *CSS dobozmodell*
    - A CSS egy fastruktúrájú dokumentumot kap, melyet egy rajzvásznon jelenít meg egy olyan közbülső struktúrát, a dobozfát előállítva, mely a megjelenített dokumentum formázási szerkezetét ábrázolja
    - Minden egyes doboz a fában a dokumentum egy megfelelő elemet ábrázolja térben és/vagy időben a rajzvásznon
    - A CSS minden egyes elemhez nulla vagy több dobozt generál az elem display tulajdonsága által meghatározott módon
- *Szintaktikai elemek: karakterek, vezérlősorozatok, megjegyzések, deklarációs blokk, at-szabályok, stílus szabályok*
    - Karakterek: Az Unicode karakterkészlet használata
    - Vezérlősorozatok: Unicode karakterek megadásához használhatunk \hhhhhh formájú vezérlősorozatokat, ahol hhhhhh az Unicode karakter kódpontját ábrázoló legalább egy és legfeljebb 6 karakterből álló hexadecimális számjegysorozat. Speciális karakterek jelentésének elnyomásához használjuk a '\' karaktert
    - Deklarációs blokk: '{' és '}' karakterek határolják, melyek között deklarációk egy listája kötelező. A deklarációk tulajdonságnév:érték formájúak, ahol a tokenek előtt és után is megengedettek whitespace karakterek. A deklarációkat ';' karakterrel kell elválasztani
    - At-szabályok: A stíluslap feldolgozását vezérlő speciális szabályok. '@' karakterrel kezdődnek, melyet egy azonosító követ, és ';' karakterrel vagy egy deklarációs blokkal végződnek.

- *Tulajdonságok*
    - A CSS által definiált paraméterek, melyek révén a dokumentumok megjelenítése vezérelhető
    - A tulajdonságoknak neve és értéke van
    - Összevont tulajdonság: Olyan tulajdonság, mely több CSS tulajdonság értékének egyidejű beállítására szolgál
- *Értékek: azonosítók, sztringek, URL-ek, számok, százalékok, dimenziók, hosszúság mértékegységek és hosszúságok, színek, funkcionális jelölések*
    - Azonosítók: Kulcsszavak: ASCII kisbetű-nagybetű érzéketlenek. Egyéni azonosítók: Például számlálók neveként használatosak. Kisbetű-nagybetű érzékenyek
    - Sztringek: ''' vagy '"' karakterek között adhatóak meg. Nem tartalmazhatják a határoló karaktert. Új sor karakter kizárólag a \A vezérlősorozattal adható meg
    - URL-ek: Használható relatív URL, feloldásakor bázis-URL a stíluslap URL-je. Bizonyos környezetekben (például @import) elhagyható az url() függvény és az URL megadható sztringként is
    - Számok: Egészek: egy opcionális előjel karakterből és legalább egy decimális számjegy karakterből állnak. Valós számok: egy opcionális előjel karakterből, legalább egy decimális számjegy karakterből, egy opcionális decimális pont karakterből állnak, melyeket opcionálisan egy 'e' vagy 'E' karakter után egy egész szám követhet
    - Százalékok: Egy százalék egy olyan egész vagy valós szám, melyet közvetlenül egy '%' karakter követ. Egy százalék mindig egy másik mennyiséghez viszonyított
    - Dimenziók: Egy dimenzió egy olyan szám, melyet közvetlenül egy mértékegység követ. Egy adott mennyiséget ábrázolnak, mint például hossz vagy szög. Hosszúság esetén a 0 értéknél elhagyható a mértékegység
    - Hosszúságok: Relatív: egy másik hosszúsághoz képest határoz meg egy hosszúságot. Abszolút: valamilyen fizikai méreten alapul
    - Színek: 16 szín kulcsszó: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, yellow. A modern böngészőprogramok támogatni szoktak számos további szín kulcsszót is. RGB színmodell
- *Kiválasztók: típus kiválasztó, általános kiválasztó, attribútum kiválasztók (csak [att] és [att=val]), osztály kiválasztó, ID-kiválasztó, pszeudo-osztályok (csak dinamikus pszeudo-osztályok, :lang(C) és szerkezeti pszeudo-osztályok), pszeudo-elemek*
    - Típus kiválasztó: Egy CSS minősített név, a gyakorlatban tipikusan egy azonosító. A megfelelő nevű elemek illeszkednek rá
    - Általános kiválasztó: Általános kiválasztónak nevezzük a * formájú kiválasztót. Minden elem illeszkedik rá, ha nincs megadva

alapértelmezett névtér. Elhagyható olyan egyszerű kiválasztóból, mely további komponenseket is tartalmaz
    - [att]: Az att attribútummal rendelkező elemek illeszkednek rá
    - [att=érték]: Olyan elemek illeszkednek rá, melyek att attribútumának értéke pontosan érték
    - Osztály kiválasztó: HTML dokumentumoknál a [class~=érték] attribútum kiválasztó helyett használható az ekvivalens .érték kiválasztó
    - ID-kiválasztó: #azonosító formájú kiválasztó, az adott azonosítójú elem illeszkedik rá. Az azonosítót egy ID típusú attribútum kell, hogy szolgáltassa a dokumentumban
    - Dinamikus pszeudo-osztályok: olyan pszeudoosztály, melyet egy elem megszerezhet vagy elveszíthet, miközben a felhasználó interakcióban van a dokumentummal
    - :lang(C): A C nyelvű szöveget tartalmazó elemek illeszkednek rá. XML dokumentumokban a nyelvet az xml:lang attribútum határozza meg, HTML dokumentumokban a nyelv megadható a lang és az xml:lang attribútummal is
    - Szerkezeti pszeudo-osztályok: Amikor megállapításra kerül, egy elem helye a testvéreinek listájában, akkor csak az elemeket kell a listában figyelembe venni. A listában az elemek számozása egytől történik
    - Pszeudo elemek: Lehetővé teszik a dokumentumok olyan részeinek kiválasztását, melyek más módon nem hozzáférhetők. Minden kiválasztókban legfeljebb egy pszeudo-elem megengedett
- *Kiválasztók: kombinátorok (leszármazott kombinátor, gyermek kombinátor, szomszéd testvér kombinátor, általános testvér kombinátor)*
    - Leszármazott kombinátor: Egyszerű kiválasztók két sorozatát elválasztó whitespace. Ha P és Q egyszerű kiválasztók két sorozata, akkor a P Q kiválasztóra a Q-ra illeszkedő olyan elemek illeszkednek, melyek a P-re illeszkedő elemek leszármazottai
    - Gyerek kombinátor: Egyszerű kiválasztók két sorozatát elválasztó '>' karakter. Ha P és Q egyszerű kiválasztók két sorozata, akkor a P > Q kiválasztóra a Q-ra illeszkedő olyan elemek illeszkednek, melyek a P-re illeszkedő elemek gyermeke
    - Szomszéd testvér kombinátor: Egyszerű kiválasztók két sorozatát elválasztó '+' karakter. Ha P és Q egyszerű kiválasztók két sorozata, akkor a P + Q kiválasztóra a Q-ra illeszkedő olyan elemek illeszkednek, melyek a P-re illeszkedő elemet követnek közvetlenül a dokumentumban
    - Általános testvér kombinátor: Egyszerű kiválasztók két sorozatát elválasztó '~' karakter. Ha P és Q egyszerű kiválasztók két sorozata, akkor a P ~ Q kiválasztóra a Q-ra illeszkedő olyan elemek

illeszkednek, melyek a P-re illeszkedő elemet követnek (nem feltétlenül) közvetlenül a dokumentumban

- *Kiválasztók: specifikusság, a specifikusság meghatározása*
  - Specifikusság: Kiválasztókhoz és deklarációkhoz specifikusság meghatározása. A specifikusság egy háromelemű (a, b, c) vektor, ahol a, b és c nemnegatív egészek. A vektorok rendezése lexikografikusan történik
  - Specifikusság meghatározása: a specifikusság egy (a, b, c) vektor, ahol: a a kiválasztóban előforduló ID-kiválasztók száma. b a kiválasztóban előforduló attribútum kiválasztók és pszeudo-osztályok száma. c a kiválasztóban előforduló típus kiválasztók és pszeudo-elemek száma
- *Stíluslap eredet: felhasználói ágenstől származó, felhasználótól származó, szerzőtől származó*
  - Felhasználói ágenstől származó: A felhasználói ágensek biztosítanak alapértelmezett stíluslapot. Például a Firefox böngészőben az alábbi módon férhetünk hozzá az alapértelmezett stíluslaphoz
  - Felhasználótól származó: A felhasználó megadhat saját stíluslapot adott dokumentum megjelenítéséhez
  - Szerzőtől származó: (X)HTML esetén a link fejléc elemmel adható meg a dokumentumhoz külső stíluslap. (X)HTML esetén használhatjuk a style fejléc elemet is, mellyel közvetlenül ágyazhatunk be stílus szabályokat a dokumentumba
- *A kaszkád*
  - Több különböző deklaráció szolgáltathatja egy tulajdonság értékét egy elemhez
  - Ezek a deklarációk különböző eredetűek is lehetnek
  - A kaszkád az a folyamat, melynek során meghatározásra kerül, hogy a vonatkozó deklarációk közül melyik határozza meg egy adott elem egy adott tulajdonságának értékét
- *Szabályok sorrendje*
  - Akkor számíthat a sorrend, ha egy elemre egynél több azonos specifikusságú stílus szabály vonatkozik
  - Ezek közül mindig a sorrendben utolsó a „legerősebb”
- *Öröklés*
  - Az öröklés a tulajdonságértékek továbbadását jelenti a szülő elemektől a gyermek elemekhez
  - Bizonyos tulajdonságok öröklöttek, ami azt jelenti, hogy az értékük öröklés révén kerül meghatározásra, feltéve, hogy a kaszkád nem eredményez egy értéket
  - A specifikáció minden egyes tulajdonsághoz meghatározza, hogy öröklött-e

    - Egy tulajdonság kaszkádolt értékeként az inherit kulcsszó az öröklést kényszeríti ki
- *Kezdőértékadás (kezdőérték)*
    - Minden tulajdonságnak van egy kezdőértéke, melyet a CSS specifikációk határoznak meg. A kezdőérték előírható úgy, hogy az felhasználói ágenstől függ
    - Ha a kaszkád nem eredményez egy értéket és a tulajdonság nem öröklött, akkor a tulajdonság meghatározott értéke a kezdőérték
    - Egy tulajdonság kaszkádolt értékeként az initial kulcsszó azt eredményezi, hogy a kezdőérték lesz a meghatározott érték

JSON

- *Mi a JSON?*
    - Könnyűsúlyú szöveges nyelvfüggetlen adatcsere formátum
    - Strukturált adatok ábrázolására szolgál
    - Ember számára is könnyen olvasható és írható formátum
    - Szoftverek által könnyen generálható és feldolgozható
    - Az ECMAScript programozási nyelvből származik
- *Mi az ECMAScript/JavaScript?*
    - ECMAScript: A JavaScript programozási nyelv szabványosítása. A jelenleg aktuális a 9-es számú kiadás. A jelenleg fejlesztés alatt álló következő verzió az ECMAScript 2019
    - JavaScript: A JavaScript kifejezést használják az ECMAScript különböző gyártók általi megvalósításaira
- *A JSON és az ECMAScript összehasonlítása*
    - A JSON az ECMAScript szintaxisán alapul, de nem teljesen kompatibilis vele
- *A JSON és az XML összehasonlítása*
    - A JSON az XML alternatívájaként használható adatcseréhez
    - Nagyjából ugyanazokat az előnyöket kínálja, mint az XML, azonban annak hátrányai nélkül
    - A JSON és az XML közös jellemzői: Egyszerűség (egyértelműen a JSON a nyerő). Az ember számára is könnyen írható és olvasható formátumok. Szoftverek által könnyen generálható és feldolgozható formátumok (egyértelműen a JSON a nyerő). Interoperabilitás. Nyíltság. Önleíró adatábrázolás. Univerzális adatcsere formátumok
    - A fő különbség az, hogy a JSON adat-orientált, az XML pedig dokumentum-orientált
    - Adatszerkezetek ábrázolásához a JSON tökéletes választás
    - Dokumentum-középpontú alkalmazásokhoz az XML-t használjuk
- *Primitív típusok: sztringek, számok, logikai értékek, null*

    - Sztringek: Unicode karakterek sorozatai, melyeket idézőjelek (U+0022) határolnak. Bármely karaktert tartalmazhatják. Speciális karakterek megadásához rendelkezésre állnak a szokásos escape szekvenciák. A BMP-hez tartozó Unicode karakterek megadhatóak \unnnn módon, ahol nnnn a karakterkód négy hexadecimális számjeggyel ábrázolva
    - Számok: Nincs korlátozás a számok tartományára és pontosságára. A gyakorlatban célszerű szem előtt tartani az interoperabilitást
- *Strukturált típusok: tömbök, objektumok*
    - Tömbök: Tetszőleges számú érték rendezett sorozata (lehet üres). Az elemek különböző típusúak is lehetnek
    - Objektumok: Tetszőleges számú név-érték párból állnak. A név tetszőleges sztring, az érték tetszőleges JSON érték. A név-érték párokra a tag elnevezést is használjuk