***Fejlesztői dokumentáció***

A **HTML** (angolul: *HyperText Markup Language*, „hiperszöveges jelölőnyelv”) egy [leíró nyelv](https://hu.wikipedia.org/wiki/Jel%C3%B6l%C5%91nyelv), melyet [weboldalak](https://hu.wikipedia.org/wiki/Weblap) készítéséhez fejlesztettek ki, és mára már internetes szabvánnyá vált a [W3C](https://hu.wikipedia.org/wiki/World_Wide_Web_Consortium) (World Wide Web Consortium) támogatásával. Az aktuális változata az 5, mely az [SGML](https://hu.wikipedia.org/wiki/Standard_Generalized_Markup_Language) általános [jelölőnyelv](https://hu.wikipedia.org/wiki/Jel%C3%B6l%C5%91nyelv) egy konkrét alkalmazása (azaz minden 5-ös HTML dokumentum egyben az SGML dokumentumszabványnak is meg kell hogy feleljen). Ezt tervek szerint lassan kiszorította volna az [XHTML](https://hu.wikipedia.org/wiki/HTML#XHTML), amely a szintén SGML alapú [XML](https://hu.wikipedia.org/wiki/XML) leíró nyelven alapul, de még csak kevés weboldal használja.

**Használata:**

**HTML** általában szöveges állományokban található meg olyan [számítógépeken](https://hu.wikipedia.org/wiki/Sz%C3%A1m%C3%ADt%C3%B3g%C3%A9p), melyek az internethez kapcsolódnak. Ezek az állományok tartalmazzák azokat a szimbólumokat, amelyek a megjelenítő programnak leírják, hogyan is kell megjeleníteni illetve feldolgozni az adott állomány tartalmát. Megjelenítő program lehet egy [webböngésző](https://hu.wikipedia.org/wiki/Webb%C3%B6ng%C3%A9sz%C5%91) (angolul: web browser), aural böngésző (olyan, amelyik a felhasználónak felolvassa a megjelenítendő szöveget), [braille](https://hu.wikipedia.org/wiki/Braille-%C3%ADr%C3%A1s) olvasó, amely konvertálja a szöveget braille „formátumba”, [levelező program](https://hu.wikipedia.org/w/index.php?title=Levelez%C5%91_program&action=edit&redlink=1) (mint például: [Mozilla Thunderbird](https://hu.wikipedia.org/wiki/Mozilla_Thunderbird), [Microsoft Outlook](https://hu.wikipedia.org/wiki/Microsoft_Outlook), [Eudora](https://hu.wikipedia.org/w/index.php?title=Eudora&action=edit&redlink=1), [Claws Mail](https://hu.wikipedia.org/w/index.php?title=Claws_Mail&action=edit&redlink=1) stb.), valamint egyéb eszközök, például [mobiltelefon](https://hu.wikipedia.org/wiki/Mobiltelefon).

**Szimbólumok, tartalom**

Négyfajta szimbólum ([leíró elem](https://hu.wikipedia.org/wiki/HTML-elem)) található meg a **HTML**-ben:

* *strukturális* elemek, amelyek leírják az adott szöveg "célját" például <h1>A legnagyobb szöveg</h1> mint első szintű címsor (alcím).
* *prezentációs* szimbólumok, amelyek leírják, hogy az adott szöveg hogyan nézzen ki: például <b>Vastag szöveg</b> **vastag** kinézetet eredményez. (Ez a forma azonban ma már *elavultnak* számít, helyette a [CSS](https://hu.wikipedia.org/wiki/CSS) használata javasolt, ugyanis a legújabb irányelv szerint szét kell választani a tartalmat (amit a HTML kódol) és a formát (amit [CSS](https://hu.wikipedia.org/wiki/CSS)-ben szokás kódolni), és ezáltal mindenki számára elérhetővé tehetőek a weblapok: felolvasó gépeknek, amit a gyengén látók használnak, szöveges böngészőknek, és nyomtatóbarát verziót is könnyebb készíteni.)
* [*hiperszöveg*](https://hu.wikipedia.org/wiki/Hipersz%C3%B6veg)*(hypertext)* elemek, amelyek segítségével kapcsolat létesíthető a dokumentum egyes elemei és más dokumentumok között (például a <a href="https://hu.wikipedia.org/">Wikipedia</a> a [Wikipedia](https://hu.wikipedia.org/) szót mint egy kapcsolatot (angol szóval: link) a megadott [URL](https://hu.wikipedia.org/wiki/URL)-hez jeleníti meg
* *eszköz* elemek, amelyek segítségével gombok, listák, beviteli mezők hozhatók létre.

**Felépítés**

Egy HTML állomány három fő részre bontható:

1. A [Dokumentumtípus-definíció](https://hu.wikipedia.org/wiki/Dokumentumt%C3%ADpus-defin%C3%ADci%C3%B3) az állomány legelején, pl:  
   <!DOCTYPE html>
2. a HTML fejléc <head></head>, ami technikai és dokumentációs adatokat tartalmaz, amelyeket az internet böngésző nem jelenít meg, tehát átlag felhasználó ezeket nem látja és
3. a HTML törzs <body></body>, amely a megjelenítendő információkat tartalmazza.
4. **Karakterkódolás**
5. Egy HTML dokumentum alapértelmezésként [ISO-8859-1](https://hu.wikipedia.org/w/index.php?title=ISO-8859-1&action=edit&redlink=1), azaz nyugat-európai kódolást használ.[[1]](https://hu.wikipedia.org/wiki/HTML#cite_note-1) Gyakran előforduló hiba szokott lenni, hogy nincs beállítva a charset paraméter a fejléc content attribútumában, annak ellenére, hogy a dokumentum nem nyugat-európai kódolású szöveget tartalmaz. Magyar nyelvű oldalak esetén ilyenkor gyakran az „*ő*” és „*ű*” betűk helyett „*õ*” és „*û*” karaktereket látunk (ekkor valószínűleg "[**ISO-8859-2**](https://hu.wikipedia.org/wiki/ISO-8859-2)" vagy "**windows-1250**" volna a helyes charset érték), de az is lehet, hogy minden ékezetes betű teljesen olvashatatlanná válik, [betűszemetet](https://hu.wikipedia.org/wiki/Bet%C5%B1szem%C3%A9t) eredményezve (ha például a megjelenítendő szöveg [UTF-8](https://hu.wikipedia.org/wiki/UTF-8) kódolású). A charset paraméter értékeként több száz kódolás és érvényes alternatív név (alias) megadható.[[2]](https://hu.wikipedia.org/wiki/HTML#cite_note-2)
6. **Numerikus karakter referenciák**
7. Akármilyen kódolást használ is egy HTML dokumentum, bármilyen [Unicode](https://hu.wikipedia.org/wiki/Unicode) karakter megjeleníthető a szövegben annak [UCS](https://hu.wikipedia.org/wiki/UCS) kódjával „&#” és „;” jelek között. Például a magyar kis „ő” betű UCS kódja a [decimális](https://hu.wikipedia.org/wiki/Decim%C3%A1lis) 337 (vagy [hexadecimális](https://hu.wikipedia.org/wiki/Hexadecim%C3%A1lis) 151), ezért függetlenül a használt kódolástól, a HTML dokumentumban elhelyezett &#337; vagy &#x151; mindig az „ő” betűt fogja megjeleníteni.
8. **Karakter egyedhivatkozások**
9. Az [SGML](https://hu.wikipedia.org/wiki/SGML) számos Unicode karakterhez definiál egyedi megnevezéseket,[[3]](https://hu.wikipedia.org/wiki/HTML#cite_note-3)[[4]](https://hu.wikipedia.org/wiki/HTML#cite_note-4) azonban a HTML a lehetséges SGML egyedeknek csak egy részét használja[[5]](https://hu.wikipedia.org/wiki/HTML#cite_note-5) (a használt egyedeket a dokumentum [Dokumentumtípus-definíciójában](https://hu.wikipedia.org/wiki/Dokumentumt%C3%ADpus-defin%C3%ADci%C3%B3) kell megadni, és a HTML 4.01-es verziójáig ez nem tartalmazza a közép-európai karaktereket). Emiatt a magyar ékezetes karakterek közül is csak azokra lehetséges SGML egyedhivatkozást megadni, amelyek a nyugat-európai kódkészletben is megtalálhatók. Vagyis hiába definiálja az SGML például „ő” betűhöz az „odblac” nevet, hivatalosan azt egy böngészőnek nem kell tudnia megjeleníteni egy 4.01-es HTML dokumentum esetén.
10. Az egyedek neveire az „&” és „;” jelek között hivatkozhatunk, így például az &eacute; a kis „é” betűt jeleníti meg.
11. **XHTML**
12. Az XHTML a HTML megfogalmazása [XML](https://hu.wikipedia.org/wiki/XML)-ben (a HTML [SGML](https://hu.wikipedia.org/wiki/SGML)-ben van definiálva). Gyakorlatilag nincs jelentős eltérés a két nyelv között, csak a formai követelmények lettek szigorúbbak:
13. Mindent kisbetűvel kell írni! (kivéve a DOCTYPE elemet)
14. Minden elemet le kell zárni! Az üres elemeket önmagukban egy szóközzel és egy / jellel: <br />.
15. Az elemeket csak egymásba ágyazva lehet használni!  helyett: <b><i>szöveg</i></b>
16. A jellemzőket idézőjelek közé írjuk!  helyett: <table *border="1"*>
17. A jellemzőknek legyen értéke!  helyett: <input *disabled="disabled"* />
18. **HTML5**
19. A HTML5 a következő, jelentősen átdolgozott változata a HTML-nek (Hypertext Markup Language), a web fő jelölőnyelvének. Egyik fő tervezési célja, hogy a webes alkalmazásokhoz ne legyen szükség pluginek (pl. Adobe Flash, Microsoft Silverlight, Sun JavaFX) telepítésére. A specifikáció a HTML4 és az XHTML1 új verzióját jelenti, a hozzájuk tartozó DOM2 HTML API-val együtt. A HTML5 specifikációban leírt formátumba történő migráció HTML4-ről, vagy XHTML1-ről a legtöbb esetben egyszerű, mivel a visszamenőleges kompatibilitás biztosított. A specifikáció a közeljövőben támogatni fogja a Web Forms 2.0 specifikációt is. HTML5-ben lehetőség van audio és videofájlok beillesztésére, erre a célra külön tagek vannak (<audio>, <video>). Továbbá létezik egy úgynevezett <canvas> tag, mely egy vászon. Erre [Javascript](https://hu.wikipedia.org/wiki/Javascript) segítségével lehet rajzolni, vagy képet beilleszteni. Fontos megemlíteni, hogy a HTML5-ből kivették az elrendezés megadó tulajdonságokat. Ezeket [CSS](https://hu.wikipedia.org/wiki/CSS)-ben kell megadni.

A **CSS** (*Cascading Style Sheets*, magyarul: „lépcsőzetes stíluslapok”) a számítástechnikában egy [stílusleíró nyelv](https://hu.wikipedia.org/w/index.php?title=St%C3%ADlusle%C3%ADr%C3%B3_nyelv&action=edit&redlink=1), mely a [HTML](https://hu.wikipedia.org/wiki/HTML) vagy [XHTML](https://hu.wikipedia.org/wiki/XHTML) típusú strukturált dokumentumok megjelenését írja le. Ezenkívül használható bármilyen [XML](https://hu.wikipedia.org/wiki/XML) alapú dokumentum stílusának leírására is, mint például az [SVG](https://hu.wikipedia.org/wiki/SVG), [XUL](https://hu.wikipedia.org/wiki/XUL) stb.

A CSS specifikációját a [World Wide Web Consortium](https://hu.wikipedia.org/wiki/World_Wide_Web_Consortium) felügyeli.

**Áttekintés**

A CSS-t a [weblapok](https://hu.wikipedia.org/wiki/Weblap) szerkesztői és olvasói egyaránt használhatják, hogy átállítsák vele a lapok színét, betűtípusait, elrendezését, és más megjelenéshez kapcsolódó elemeit. A tervezése során a legfontosabb szempont az volt, hogy elkülönítsék a dokumentumok struktúráját (melyet HTML vagy egy hasonló leíró nyelvben lehet megadni) a dokumentum megjelenésétől (melyet CSS-sel lehet megadni). Az ilyen elkülönítésnek több haszna is van, egyrészt növeli a weblapok [használhatóságát](https://hu.wikipedia.org/wiki/Weblap-haszn%C3%A1lhat%C3%B3s%C3%A1g), rugalmasságát és a megjelenés kezelhetőségét, másrészt csökkenti a dokumentum tartalmi struktúrájának komplexitását. A CSS ugyancsak alkalmas arra, hogy a dokumentum stílusát a megjelenítési módszer függvényében adja meg, így elkülöníthető a dokumentum formája a képernyőn, nyomtatási lapon, hangos böngészőben (mely [beszédszintetizátor](https://hu.wikipedia.org/w/index.php?title=Besz%C3%A9dszintetiz%C3%A1tor&action=edit&redlink=1) segítségével olvassa fel a weblapok szövegét), vagy [Braille](https://hu.wikipedia.org/wiki/Braille-%C3%ADr%C3%A1s)-készüléken megjelenítve.

A CSS használható XML fájlok megjelenítésére is, így a strukturált dokumentumokhoz teljes körű stílust lehet megadni, befolyásolva az elrendezését, színét, betűtípusait az erre alkalmas kliensekben vagy [webböngészőkben](https://hu.wikipedia.org/wiki/Webb%C3%B6ng%C3%A9sz%C5%91).

Az elemek stílusát különböző CSS szelektorokkal lehet kiválasztani:

* *Minden elemre* – a \* szelektor használatával
* *Az elem neve alapján* – például minden 'p' vagy 'h2' [HTML-elemhez](https://hu.wikipedia.org/wiki/HTML-elem)
* *Leszármazottak alapján* – például az olyan 'a' elemekre, melyek egy 'li' elem részei, a szelektor "li a"
* class*vagy*id*attribútumok alapján* – például .*class* és/vagy #*id* a class=*"osztály"* vagy id=*"azonosító"* elemekhez

Ezeken kívül rendelkezésre áll több *pszeudo-osztály,* melyekkel további műveletekhez lehet stílust rendelni. Talán a legismertebb ezek közül a mindenek :hover, melynek stílusa akkor lép érvénybe, mikor a hozzá tartozó elem aktiválódik, például fölévisszük az egeret. Hozzá lehet fűzni a szelektorokhoz is, például a:hover vagy #elementid:hover. További ismertebb pszeudoosztályok a :first-line, a :visited vagy a :before.

A szelektorok kombinálhatók a széles körű alkalmazáshoz.

A CSS információkat a lapokhoz több módon lehet megadni:

* Szerzői stílus
  + külső CSS fájl a dokumentumból hivatkozva
  + beágyazva a dokumentumba
  + azonnali, felülírva az általános stílust egy konkrét esetre
* Felhasználói stílus
  + egy helyi CSS fájl az összes dokumentumra, melyet a felhasználó a böngésző beállításaiban adhat meg; a felhasználó megadhatja, hogy a szerző és a saját stílusai közül melyik legyen fontosabb
* Kliens stílus
  + egy alapértelmezett stílus, melyet a kliens vagy a böngésző használ az elemekhez

A CSS a szabályokat a kapcsolódások alapján súlyozza, így mikor egyszerre több szabály is érvényes lehet egy elemre, akkor a legfontosabb szabályt alkalmazza.

A CSS használatának legfontosabb előnyei:

* Több lap vagy akár egy teljes webhely stílusait egy helyen lehet tárolni, így gyorsan és könnyen frissíthető
* Különböző felhasználókhoz különböző stílusokat lehet rendelni: például könnyen olvasható, nyomtatható stílusok
* A dokumentum mérete és komplexitása csökken, mivel nem tartalmaz információkat a megjelenítéshez

A CSS egyszerű [szintaxissal](https://hu.wikipedia.org/wiki/Szintaxis_(programoz%C3%A1si_nyelvek)) rendelkezik, csak néhány angol nyelvű kulcsszót használ a stílusok tulajdonságaihoz. A stíluslap maga a stílust leíró **szabályok** sora. Minden szabályhoz tartozik egy **szelektor** és egy **deklarációs szakasz**. Ez utóbbi kapcsos zárójelek között pontosvesszővel elválasztott **deklarációkat** tartalmaz. A deklarációk formája a következő: a **tulajdonság** neve, egy kettőspont, majd az adott tulajdonság **értéke**.

A **JavaScript** [programozási nyelv](https://hu.wikipedia.org/wiki/Programoz%C3%A1si_nyelv) egy [objektumorientált](https://hu.wikipedia.org/wiki/Objektumorient%C3%A1lt_programoz%C3%A1s), prototípus-alapú [szkriptnyelv](https://hu.wikipedia.org/wiki/Interpreteres_nyelvek), amelyet [weboldalakon](https://hu.wikipedia.org/wiki/Weblap) elterjedten használnak. Ebből fejlődött ki a [TypeScript](https://hu.wikipedia.org/w/index.php?title=TypeScript&action=edit&redlink=1), ami a JavaScript típusos változatának tekinthető.

**Története:**

Eredetileg *Brendan Eich*, a Netscape Communications mérnöke fejlesztette ki; neve először *Mocha*, majd *LiveScript* volt, később „JavaScript” nevet kapott, és szintaxisa közelebb került a Sun Microsystems [Java](https://hu.wikipedia.org/wiki/Java_(programoz%C3%A1si_nyelv)) programozási nyelvéhez. A JavaScriptet először 1997–99 között szabványosította az [ECMA](https://hu.wikipedia.org/wiki/Ecma_International) „[ECMAScript](https://hu.wikipedia.org/wiki/ECMAScript)” néven. A jelenleg is érvényes szabvány az *ECMA-262 Edition 3* (1999. december), ami a *JavaScript 1.5*-nek felel meg. Ez a szabvány egyben [ISO](https://hu.wikipedia.org/wiki/Nemzetk%C3%B6zi_Szabv%C3%A1ny%C3%BCgyi_Szervezet) szabvány is.

**A Microsoft általi átvétele**

A [Microsoft](https://hu.wikipedia.org/wiki/Microsoft) 1995-ben kifejlesztette az [Internet Explorert](https://hu.wikipedia.org/wiki/Internet_Explorer), ami a Netscape-pel való böngészőháborúhoz vezetett. A Microsoft a [Netscape Navigator](https://hu.wikipedia.org/wiki/Netscape_Navigator) JavaScript-feldolgozójának forráskódja segítségével létrehozta a sajátját, a [JScriptet](https://hu.wikipedia.org/w/index.php?title=JScript&action=edit&redlink=1).

A JScript először 1996-ban jelent meg a CSS kezdeti támogatása és a HTML néhány kiegészítése mellett. E megvalósítások merőben különböztek a Navigatorban alkalmazottaktól,[[3]](https://hu.wikipedia.org/wiki/JavaScript#cite_note-O'Reilly-2001-3)[[4]](https://hu.wikipedia.org/wiki/JavaScript#cite_note-4) ami megnehezítette a fejlesztőknek, hogy a weblapjaik mindkét webböngészőben jól működjenek, és ami a „Netscape-en működik legjobban” és „Internet Exploreren működik legjobban” széles körű használatához vezetett sok éven át.[[3]](https://hu.wikipedia.org/wiki/JavaScript#cite_note-O'Reilly-2001-3)[[5]](https://hu.wikipedia.org/wiki/JavaScript#cite_note-5)

**A JScript felemelkedése**

1996 novemberében a Netscape elküldte a JavaScriptet az ECMA International-nek, ami a sztenderddé válás kiindulópontja. Ez vezetett 1997 júniusában az első ECMAScript nyelv hivatalos kiadásához.

A sztenderdizációs folyamat néhány évig folytatódott, közben az ECMAScript 2 1998 júniusában, az ECMAScript 3 1999 decemberében lett kiadva. Az ECMAScript 4-en a munka 2000 decemberében kezdődött.

Közben a Microsoft egyre dominánsabb lett a böngészőpiacon. A 2000-es évek elejére az Internet Explorer részesedése elérte a 95%-ot.[[6]](https://hu.wikipedia.org/wiki/JavaScript#cite_note-searchenginejournal.com-6) Ez azt jelenti, hogy a [JScript](https://hu.wikipedia.org/w/index.php?title=JScript&action=edit&redlink=1) tulajdonképp sztenderddé vált a weben.

A Microsoft eleinte részt vett a sztenderdizációban, s néhány tervezetet is megvalósított, de végül megszűnt közreműködni.

**Szerkesztés, előállítás**

A JavaScript kód vagy a [HTML](https://hu.wikipedia.org/wiki/HTML) fájlban, vagy külön (jellemzően .js kiterjesztésű) szövegfájlban van. Ezek a fájlok tetszőleges [szövegszerkesztő](https://hu.wikipedia.org/wiki/Sz%C3%B6vegszerkeszt%C5%91) (nem [dokumentumszerkesztő](https://hu.wikipedia.org/wiki/Dokumentumszerkeszt%C5%91)) programmal szerkeszthetőek.

**Futási környezete**

A JavaScript esetében a futási környezet jellemzően egy [webböngésző](https://hu.wikipedia.org/wiki/Webb%C3%B6ng%C3%A9sz%C5%91), illetve annak [JavaScript-motorja](https://hu.wikipedia.org/wiki/JavaScript-motor).

JavaScript alkalmazások futtathatók továbbá a böngészőn kívül is, [Node.js](https://hu.wikipedia.org/wiki/Node.js) vagy [Deno](https://hu.wikipedia.org/w/index.php?title=Deno&action=edit&redlink=1) segítségével, melyek [Windows](https://hu.wikipedia.org/wiki/Windows), [macOS](https://hu.wikipedia.org/wiki/MacOS) és [Linux](https://hu.wikipedia.org/wiki/Linux) alapú operációs rendszerekkel is kompatibilisek.

JavaScript programok [Windows](https://hu.wikipedia.org/wiki/Windows) alapú környezetben futtathatók továbbá a [wscript.exe](https://hu.wikipedia.org/w/index.php?title=Wscript.exe&action=edit&redlink=1) és a [cscript.exe](https://hu.wikipedia.org/w/index.php?title=Cscript.exe&action=edit&redlink=1) segítségével is.

**Inkompatibilitások**

Bár a nyelvet szabványosították, mégis részben különbözően implementálják a JavaScriptet a különböző böngészők.

**Adattípusok**

**Elsődleges elemi adattípusok**

* [String](https://hu.wikipedia.org/wiki/String) (karakterlánc)
* Number (szám, lehet tizedestört vagy egész szám is)
* [Boolean](https://hu.wikipedia.org/wiki/Boolean) (értéke igaz vagy hamis)

**Speciális típusok**

* Undefined (meghatározatlan, így nincs értéke)
* Null (kifejezetten *semmi*ként van meghatározva)

**Összetett típusok**

* Object (objektum)
* [Array](https://hu.wikipedia.org/wiki/T%C3%B6mb_(adatszerkezet)) (lista, amely több különböző adattípust is tartalmazhat)
* Set (speciális lista, amely minden értékből csak egyet tartalmazhat)

**Fontosabb objektumok**

* Date (dátum)
* RegExp ([reguláris kifejezés](https://hu.wikipedia.org/wiki/Regul%C3%A1ris_kifejez%C3%A9s))
* Function (függvény)
* Math (matematikai függvények gyűjteménye)

A string típusú adatokat [aposztróf](https://hu.wikipedia.org/wiki/Aposztr%C3%B3f) ('...'), [idézőjel](https://hu.wikipedia.org/wiki/Id%C3%A9z%C5%91jel) ("...") vagy backtick (`...`) pár közé írjuk. Mindhárom megoldás lehetséges, így az éppen nem használt jeleket is tartalmazhatja a szöveg.

**A JavaScript alapobjektumai és függvényei**

|  |  |
| --- | --- |
| **Megnevezés** | **Leírás** |
| Array | tömb objektum |
| Boolean | logikai értékeket leíró és azok kezelését szolgáló beépített objektum |
| Date | dátumkezelésre szolgáló beépített objektum |
| eval() | JavaScript kód végrehajtása a paraméterként megadott karakterláncból (Biztonsági kockázatot rejtő, veszélyes függvény!) |
| Math | matematikai függvények használatát biztosító beépített objektum |
| MimeType | MIME típusok kezelésre szolgáló beépített objektum |
| Number | számok megvalósítását szolgáló beépített objektum |
| parseFloat() | a numerikus értéket képviselő objektumból kinyeri a lebegőpontos számot |
| parseInt() | a numerikus értéket képviselő objektumból kinyeri az egész számot |
| RegExp | reguláris kifejezések kezelésre szolgáló beépített objektum |
| String | sztringek megvalósítását és kezelését szolgáló beépített objektum |
| BigInt | nagy egészek tárolására alkalmas adattípus |

**Math.**

|  |  |
| --- | --- |
| **Kód** | **Leirás** |
| Math.abs(x) | Visszatér a szám abszolút értékével |
| Math.acos(x) | Visszatér a szám arkusz koszinuszával (radián) |
| Math.asin(x) | Visszatér a szám arkusz szinuszával (radián) |
| Math.atan(x) | Visszatér a szám arkusz tangensével (radián) |
| Math.cbrt(x) | Visszatér a szám köbgyökével |
| Math.ceil(x) | Visszatér a nem egész szám felfelé kerekített értékével |
| Math.cos(x) | Visszatér a szám koszinuszával (radián) |
| Math.E | Visszatér az 'e' számmal (kb. 2,71) |
| Math.floor(x) | Visszatér a nem egész szám felfelé kerekített értékével |
| Math.LN2 | Visszatér a 2 'e' alapú logaritmusával |
| Math.LN10 | Visszatér a 10 'e' alapú logaritmusával |
| Math.log(x) | Visszatér a szám 'e' alapú logaritmusával |
| Math.max(x, y, z, ...) | Visszatér a legnagyobb számmal |
| Math.min(x, y, z,.....) | Visszatér a legkisebb számmal |
| Math.PI | Visszatér a 'pi' számmal (kb. 3,14) |
| Math.pow(x, y) | Visszatér a szám hatványával 'x' az alap és az 'y' a kitevő |
| Math.random() | Visszatér egy random számmal 0 és 1 között |
| Math.round(x) | Visszatér a nem egész szám kerekített értékével |
| Math.sin(x) | Visszatér a szám szinuszával (radián) |
| Math.sqrt(x) | Visszatér a szám négyzetgyökével |
| Math.tan(x) | Visszatér a szám tangensével (radián) |

**JavaScript események**

**Az oldal egészére vonatkozó események**

|  |  |  |
| --- | --- | --- |
| **Esemény** | **Eseménykezelő** | **Bekövetkezése** |
| Load | onLoad | Az oldal minden objektuma letöltődése után |
| Resize | onResize | Dokumentum átméretezésekor |
| Scroll | onScroll | Dokumentum görgetésekor |
| Unload | onUnload | Dokumentum eltávolítása esetén ablakból vagy frame-ből. Érvényes BODY, FRAMESET elemekre. |

**Egéresemények**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Esemény** | **Eseménykezelő** | **Bekövetkezése** | **Menete** | **Érvényes** |
| Click | onClick | Az adott elemre való egérkattintáskor | MouseDown > MouseUp > Click (Többszörös kattintásnál a *detail* attribútum értéke minden kattintásnál megnövekszik eggyel.) | A legtöbb elemre. |
| MouseDown | onMouseDown | Egérgomb lenyomása az adott elem felett | - | A legtöbb elemre. |
| MouseUp | onMouseUp | Egérgomb felengedése az adott elem felett | - | A legtöbb elemre. |
| MouseOver | onMouseOver | Az egérkurzor az adott elem fölé kerülése esetén. | - | A legtöbb elemre. |
| MouseMove | onMouseMove | Az egérkurzor mozog az adott elem fölött. | - | A legtöbb elemre. |
| MouseOut | onMouseout | Az egérkurzor az adott elemet elhagyja. | - | A legtöbb elemre. |

Egy **lakásfoglaló oldal** célja az, hogy megkönnyítse a lakások és ingatlanok bérlését vagy megvásárlását keresők és az ingatlanokat kínálók számára. Ezek az oldalak egyfajta online platformként működnek, ahol a kereslet és a kínálat találkozik. A következőkben részletesen bemutatom, hogy mire jó egy lakásfoglaló oldal, milyen funkciókat kínál, és miért vált elengedhetetlenné a modern ingatlanpiacon.

**1. Az alapvető cél: kereslet és kínálat összekötése**

A lakásfoglaló oldalak alapvető funkciója, hogy kapcsolatot teremtsenek a lakást keresők (bérlők vagy vásárlók) és az ingatlant kínálók (tulajdonosok, bérbeadók, ingatlanirodák) között. Ez egy egyszerű, gyors és hatékony megoldást biztosít, amely időt takarít meg mindkét fél számára, és minimalizálja a kommunikációs akadályokat.

**2. Mire jó egy lakásfoglaló oldal?**

**a) Széles választékot kínál az ingatlanokból**

Egy lakásfoglaló oldal lehetőséget ad arra, hogy a felhasználók egyszerre több száz vagy akár több ezer ingatlan között böngésszenek. Ezek az oldalak szűrőkkel és kategóriákkal könnyítik meg a keresést, például:

* **Elhelyezkedés:** Város, kerület, utca vagy akár földrajzi régió.
* **Árkategória:** Min/max ár megadása, napi/heti/havi bérleti díj szerint.
* **Lakás jellemzői:** Méret, szobák száma, erkély, garázs, bútorozottság, stb.
* **Ingatlan típusa:** Kiadó vagy eladó lakás, családi ház, nyaraló, garzon, stb.

**b) Kényelmes és gyors foglalási folyamat**

A lakásfoglaló oldalak egyik legnagyobb előnye, hogy a teljes folyamat – a lakás kiválasztásától kezdve a foglalásig – online lebonyolítható. A felhasználó:

1. Böngészhet a lakások között.
2. Megtekintheti az elérhető időpontokat.
3. Közvetlenül foglalhat időpontot a lakás megtekintésére vagy bérlésére.
4. Előlegfizetést vagy foglalót küldhet online, ami biztosítja a tranzakció gyors lezárását.

**c) Átlátható információk nyújtása**

Egy jól működő lakásfoglaló oldal részletes információt nyújt az ingatlanokról. Ide tartoznak:

* Fényképek és videók az ingatlan állapotáról.
* Pontos leírás az ingatlan jellemzőiről.
* Elérhető közszolgáltatások (pl. közlekedés, boltok, iskolák).
* Átlátható árak és extra költségek (pl. közös költség, rezsi).

Források: <https://hu.wikipedia.org/wiki/HTML>

<https://hu.wikipedia.org/wiki/JavaScript>

https://hu.wikipedia.org/wiki/CSS