CSS-szintaxis

CSS alkalmazása HTML-re

Külső stíluslap alkalmazása	<link rel="stylesheet" type="text/css" href="stilus.css" />
Médiatípus meghatározása (a deklarációk csak a megadott médiumon történő megjelenítésre lesznek érvényesek):	media="screen" (all, braille, handheld, monochrome, print, projection, screen, speech, tv)
Ugyanezzel az attribútummal lehet bizonyos kijelzőméretek esetén más és más stíluslapot aktiválni:	<link href="tablet.css" rel="stylesheet" type="text/css" media="only screen and (min-width:481px) and (max-width:768px)" />
HTML-lapon belüli stílusok	<head> <style type="text/css"> … </style> </head>
Külső CSS importálása a fejléc style-tagjébe (minden további deklaráció elé kell tenni):	<style type="text/css"> @import "stilus.css" … </style>

Konkrét kiválasztó médiaspecifikus formázása:
A bekezdések betűmérete minden kijelzőn 20px legyen…
	<style type="text/css">
	p {
	font-size: 20px
	}
	@media "print"
	p {
	display: none
	}
	</style>

… kivéve nyomtatásban, ahol el kell rejteni:	
HTML-tagre vonatkozó stílus	<body> Tartalom </body>

Általános szintaxis	szelektor {
	tulajdonság: érték;
	tulajdonság: érték mértékegység;
	tulajdonság: érték
	}

Többszörös kiválasztás (ÉS-kapcsolat)	szelektor, szelektor {
	tulajdonság: érték
	}

Kiválasztók (szelektorok)

Tagek
Az előre definiált sztenderd HTML-tagek megcímzése.
	body {
	font-family: Calibri, serif
	}

Osztályok
Ha egy lapon belül több HTML-elemre akarunk azonos formázásokat alkalmazni, osztályhoz kell rendelnünk.
 	
HTML:
 	
CSS:
 	
Másfajta elemet is be lehet sorolni az osztályba:
	
	 …
	
	.idezet {
	font-size: 75%;
	letter-spacing: 2px
	}
	
	<h1 class="idezet">…</h1>

Azonosítók
Ha egy weboldalon belül több elemre akarunk azonos formázásokat alkalmazni, azonosítókhoz is rendelhetjük őket. A lapon belül csak egyszer szerepelhetnek.
 	
HTML:
 	
CSS:
	
	 …
	
	#header {
	height: 160px
	}

Univerzális kiválasztó
Minden elem kiválasztása.
	* {
	margin: 0;
	padding: 0
	}

Gyermekkiválasztó
Egy elemen belül elhelyezett másik elem megcímzése.
 	
HTML:
 	
CSS:
 	
A következő beágyazás esetén azonban ez a CSS-címzés nem működne:
 	
Az egyszerű szóköz azonban olyan sarjkiválasztó, amely átlép minden köztes sarjelemen:
	
	 Én vagyok a szülőelem, aki egy gyermekelemet tartalmaz.
	
	p > strong {
	text-decoration: underline
	}
	
	 Én vagyok a szülőelem, aki egy sarjelemet tartalmaz.
	
	p strong {
	text-decoration: underline
	}

Testvérkiválasztó Csak az adott elem után közvetlenül álló másikat címezi meg.	
HTML:	Fejezetcím Bekezdés Bekezdés

Ez a CSS-kiválasztó csak a fejezetcímet követő első bekezdést címezi meg:	h1 + p {
	color: red
	}

Attribútumkiválasztó Segítségével olyan elemek választhatók ki, amelyek bizonyos attribútumot hordoznak, sőt akár egy konkrét értékkel ellátott attribútumot horodzó elemek is.	

Ez az összes linket címezi meg (a hivatkozási pontokat nem):	a[href] {
	color: red
	}

Ez csak a kapcsolat.html-re mutató linkeket címezi meg:	a[href="kapcsolat.html"] {
	color: red
	}

Ez azokat a linkeket választja ki, amelyek céloldala *tartalmazza* a megadott karakterláncot; azaz minden olyan linket, amelynek célja ezen domain alatti lapra hivatkozik.	a[href*="timetodesign.hu"] {
	color: red
	}

Látszólagos kiválasztók	
Olyan kiválasztók, amelyek nem mint HTML-elemek léteznek, hanem a megjelenítő eszköz – pl. böngésző – által meghatározott körülmények között keletkeznek.	

Dinamikus látszólagos kiválasztók
Első betű kiválasztó:
	p:first-letter {
	font-size: 200%
	}

Első sor kiválasztó:	p:first-line {
	font-weight: bold
	}

Első gyermek kiválasztó (a gyermekelemnek csak az első előfordulását címezi meg):	p:first-child {
	font-size: 200%
	}

Elem előtti kiválasztó tartalom (itt: kép) beszúrására:	h2:before {
	content: url(img/strigula.png)
	}

Elem utáni kiválasztó tartalom (itt: szöveg) beszúrására:	h2:before {
	content: '…'
	}

Linkek státuszainak kiválasztói
rollover-státusz:
	a:hover {
	color: green
	}

fókusz-státusz:	a:focus {
	color: red
	}

megkattintott státusz:	a:active {
	color: black
	}

meglátogatott link:	a:visited {
	color: pink
	}

Öröklődés

A szülőelemek tulajdonságait gyermekelemeik öröklik. Ezt a tulajdonságot pl. a html-be ágyazott valamennyi elem örökli:
 	
Nem minden tulajdonság öröklődik tovább. Részletek a CSS-referenciákban.
	html {
	font-family: Calibri, sans-serif
	}

Kaszkádolás

A kiértékelés során a megjelenítő a következő helyeken szereplő stílusdeklarációkat veszi figyelembe (növekvő prioritással): Azaz a böngésző először is a saját alapértelmezései alapján formázza meg a dokumentumot. A külső stíluslapok deklarációi azonban felülírják a böngészőjéit. A konkrét HTML-lap fejlécében deklarált stílusok felülírják a külső CSS-ekben foglaltakat. Végül a konkrét HTML-hez kapcsolt deklarációk felülírják a fejlécben foglaltakat. Más szóval minél közelebb szerepel a stílusdeklaráció a taghez, annál erősebb.	1. A böngésző alapértelmezése. 2. Külső stíluslap. 3. A HTML-fejléc <style>-tagje. 4. Inline-deklarációk (kokrét elem style="" attribútuma).

A kiértékelésnek ez a prioritási rendje az !important utasítással írható felül. Ha a következő sor szerepel a külső CSS-ben…	p {
	color: grey !important
	}

… akkor a következő HTML-bekezdés is szürke lesz: Igaz, a taghez közelebbi !important is felülírja a távolabbit, így a bekezdés mégis pirosra festhető:	Bekezdés Bekezdés

Mértékegységek

Színek	HTML Color Names pl. purple Hexadecimális kód: #RRGGBB (2-2 hexadecimális karaktere az egyes szín-komponensek (piros, zöld, kék) ereje) pl: #FF00FF ennek rövidítése: #F0F RGB-meghatározás: rgb(RRR,GGG,BBB) 0-tól 255-ig terjedő decimális számok pl: rgb(255,0,255) vagy 0%-tól 100%-ig terjedő értékek pl: rgb(100%,0%,100%) RGBA-meghatározás: rgba(RRR,GGG,BBB,A) Így megadható az alpha-csatorna is (áttetszőségi érték), 0-tól 1-ig. pl: rgba(255,0,255,0.5)

Méretek

Relatív mértékegységek Pixel:	font-size: 12px A pixel mérete függ a kijelző felbontásától. A képernyők általában 72 – 96 pixel / hüvelyk (Zoll, inch) felbontásúak.
EM és EX:	font-size: 1.2em font-size: 1.2ex Az m ill. az x betű magasságához képesti érték. 1 em vagy ex a betűméret 100%-ának felel meg.
Százalék:	width: 75% Egy máshonnan örökölt mérethez képesti érték.
Abszolút mértékegységek SI-mértékegységek:	font-size: 1cm font-size: 10mm font-size: 1in (Zoll, inch = 25,4 mm) Elvileg nem függenek a kijelző felbontásától.
Tipográfiai pont (= 1/72 hüvelyk):	font-size: 12pt
Tipográfiai pica (= 12 pont):	font-size: 1pc

CSS-tulajdonságok

Betűtulajdonságok

Betűtípus Vesszőkkel elválasztva alternatívák adhatók meg arra az esetre, ha a kliens gépén nincs meg az első, ill. nincs meg benne valamelyik ékezetes vagy speciális írásjel.	font-family: 'Times New Roman', Times, serif betűcsaládok: serif, sans-serif, cursive, monospace
Betűstílus	font-style: italic oblique (ugyanaz) normal (felülír egy másonnan örökölt értéket)
Betűváltozat (kiskapitális)	font-variant: small-caps normal inherit (örökölt érték)
Betűvastagság	font-weight: bold lighter, light, normal, bold, bolder vagy a százasok 100-tól 900-ig (nem mindegyik betűtípusban van 9 különböző súlyosság, sőt a legtöbben jó, ha 2-nél több van)
Betűméret Orientációként: az alap szövegméret 16px, h1: 36px, h2: 24px, h3: 21px, h4: 18px, h5: 16px, h6: 14px	font-size: 16px vagy xx-small, x-small, smaller, small, medium, large, larger, x-large, xx-large
Sormagasság	line-height: 20px
Fentiek egyetlen deklarációban is megadhatók: font: 'Times New Roman', Times, serif, italic, small-caps, bold, 16px/20px	

Szövegtulajdonságok

Szín	color: grey
	HTML Color Names
	hexakód
	rgb(RRR,GGG,BBB) //0-255//
	rgb(RRR%,GGG%,BBB%) //0%-100%//
	rgba(RRR,GGG,BBB,A) //R, G, B: 0-255 A: 0.1 – 1//

Betűköz	letter-spacing: 2px
Szóköz	word-spacing: 10px
Szöveg-igazítás	text-align: left, center, right, justify, start (balról-jobbra írásnál ugyanaz mint a left) end (balról-jobbra írásnál ugyanaz mint a right)

Első sor bekezdése	text-indent: 40px Negatív értékeket is felvehet.
Szöveg-dekoráció	text-decoration: underline (aláhúzott) overline (föléhúzott) line-through (áthúzott) blink (villogó)
Szöveg-transzformáció	text-transform: uppercase (csupa nagybetűs) lowercase (csupa kisbetűs) capitalize (minden szó első betűje nagybetűs)
Szövegárnyék	text-shadow: 2px 2px 2px #333 paraméterei rendre: eltérés jobbra, eltérés lefelé (negatív értéket is felvehetnek), elmosottság, szín
Sortörés és űrök kezelése	white-space: normal (hosszú sorok tördelése) nowrap (tördelés megszűntetése) pre (nincs tördelés és megtartja az eredeti white-space-eket)
Föggőleges igazítás Konténeren vagy táblázatcellán belüli szöveg függőleges igazítása. Szövegkörnyezetben felső / alsó indexbe teszi a kiválasztott szöveget (de méretét nem csökkenti).	vertical-align: bottom, middle, top sub, baseline, super text-bottom, text-top
Felsorolások	
Strigula rendezetlen felsorolásnál: rendezett felsorolásnál: egyéni kép mint strigula: listaelem bekezdése:	list-style-type: disc, square, circle decimal, decimal-leading-zero, lower- és upper- roman, alpha, greek és latin hebrew, armenian, georgian, cjk-ideographic, hiragana, katakana, hiragana-iroha, katakana-iroha list-style-image: url(img/strigula.png) list-style-position: inside, outside
A deklarációk itt is csoportosíthatók: list-style: square inside url(img/strigula.png)	
Táblázatok	

Táblázatcím alul / felül	caption-side: bottom, top
Szélesség	width: 600px Százalékos meghatározással rugalmas táblázat is kialakítható. table-layout: fixed (azonos oszlopszélességek), auto, inherit Megadja, hogy a a böngésző hogyan kezelje a szélességeket.
Keret	border-collapse: separate (cellaközök vannak – ez az alapértelmezés) collapse (nincsenek cellaközök; a cellakeretek egybeesnek) border-spacing: Cellaköz mértéke, megfelel a cellspacing="" attribútumnak.
Üres cellák kezelése	empty-cells: show, hide Megjelenjenek –e az üres cellák.
Hivatkozások	
Státusz-kiválasztók Alapállapot (megegyezik az a címzéssel) Fókusz-státusz Rollover-státusz Megkattintott státusz Meglátogatott státusz	 a:link {} a:focus {} a:hover {} a:active {} a:visited {}
Háttér	

Háttérszín	background-color: grey
	HTML Color Names
	hexakód
	rgb(RRR,GGG,BBB) //0-255//
	rgb(RRR%,GGG%,BBB%) //0%-100%//
	rgba(RRR,GGG,BBB,A) //R, G, B: 0-255 A: 0.1 – 1//

Háttérkép elérése: ismétlődése: Görgetése	 background-image: url(img/bg1.png); background-repeat: repeat, no-repeat, repeat-x, repeat-y background-attachment:

Pozíciója	scroll (az alapértelmezés) fixed (a háttér marad a helyén) local (csúszkákkal ellátott dobozban a tartalommal együtt gördül) background-position top, bottom, left, right, center
Fentiek egyetlen deklarációban is megadhatók: background: grey url(img/bg1.png) repeat-x fixed	
További tulajdonságok	
Láthatóság	visibility: ▪ collapse – táblázat sorának elrejtése (a helye megmarad) ▪ hidden – elem elrejtése (a helye megmarad) ▪ inherit – örökölt érték ▪ visible – látható
Megjelenítés További megjelenítések: marker, table-caption, table-cell, table-column, table-column-group, table-footer-group, table-header-group, table-row, table-row-group	display: ▪ none – elem és összes gyermekelemének elrejtése (helyet sem foglal) ▪ block – dobozszerű megjelenítés (inline elemek konvertálása) ▪ inline – soron belüli megjelenítés (block elemek konvertálása) ▪ inline-block – soron belüli doboz megjelenítés ▪ inline-table – táblázat soron belüli megjelenítése ▪ list-item – felsoroláselemként történő megjelenítés ▪ run-in – befutó megjelenítés; az elem az utána köv. blokk első soron belüli eleme lesz ▪ compact – kompakt megjelenítés; az elem az utána köv. blokk elembe soron belüli elemként kerül; ha nem fér be, akkor külön blokkba kerül
Dinamikus tartalmak	

Karakterláncok	p:after {
	content:'szöveg'
	}

Képek	p:before {
	content: url(strigula.png)
	}

Idézőjelek	p:before {
	content: open-quote
	}
	
	p:after {
	content: close-quote
	}

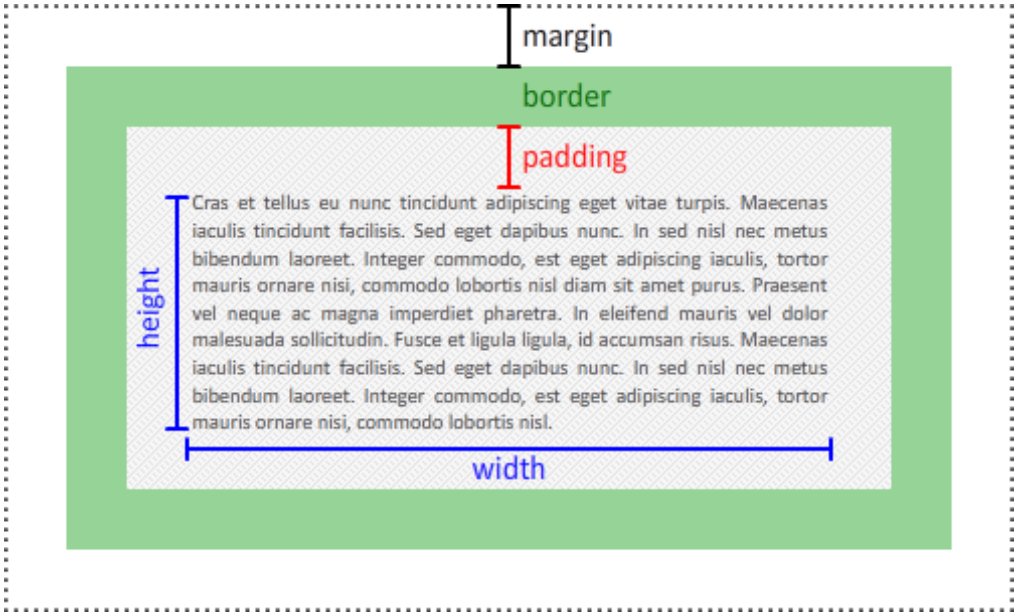
Számláló (rugalmasabbá teszi a fejezetek vagy	

bekezdések számozását)
 	
változó deklarálása és nullázása:
 	
változó megnövelése és kiírása:
	
	body {
	counter-reset: fejezetsz
	}
	h2:before {
	counter-increment: fejezetsz;
	content: counter(fejezetsz) ". fejezet:"
	}

Kurzor beállítása	cursor: alias, all-scroll, auto, cell, context-menu, copy, crosshair, default, help, inherit, move, no-drop, not-allowed, pointer, progress, text, vertical-text col-resize, row-resize e-resize, ew-resize, n-resize, ne-resize, nesw-resize, ns-resize, nw-resize, nwse-resize, s-resize, se-resize, sw-resize, w-resize

Doboz-modell

Minden HTML-elemet téglalapként kell felfogni, amely a következő tulajdonságokkal rendelkezik:



A szöveget tartalmazó <p>-tag is egy ilyen doboz. A kitöltés (padding) az a távolság, amennyivel tartalmát – jelen esetben a szöveget – befelé tolja.

A doboz szélességét (width) és magasságát (height) a kitöltés megtoldja; ha pl. egy 400px széles és 300 px magas doboznak 20px kitöltést adunk, a ténylegesen renderelt doboz 440 px-t fog elfoglalni széltében és 340 px-t hosszában. Ekkora lesz tehát az a terület, amelyet a háttérszín – vagy mint fent – a háttérkép kitölt.

A doboznak keret (border) adható, ami szintén kifelé növekszik és a testvérelemeket eltolja. Létezik olyan körvonal (outline) is, ami a testvérelemeket nem tolja el, hanem elfedi.

A margó (margin) a külső távolság; ez adja meg, hogy a doboz testvérelemeitől és szülőelemétől mekkora távolságot tartson. Tehát erre a dobozra vonatkoztatva ugyanazt az eredményt érjük el, ha szülőjének adunk paddinget. Ha a szülőnek paddinget, ÉS a gyermeknek margint adunk, a kettő összeadódik.

A padding, a border, az outline és a margin mind a négy irányba külön-külön is megadható az angol irányszavak (-top, -right, -bottom, -left) hozzáfűzésével.

A múltban sok problémát okozott az InternetExplorer 5 és 5.5 sajátos felfogása a doboz-modellről, de a 6. verzió óta a Microsoft is a W3C ajánlását követi (ebben a kérdésben), így ma már nincs szükség hackekre.

Kitöltések	
	padding-top: 10px; padding-right: 30px; padding-bottom: 10px; padding-left: 30px;
Az előbbi egy deklarációban (az óramutató járása szerinti sorrendben):	padding: 10px 30px 10px 30px
Ugyanez 3 paraméterrel (fent, jobbra és balra azonos, lent)	padding: 10px 30px 10px
Ha a függőleges és vízszintes értékek azonosak, elég csak két érték (függőleges és vízszintes):	padding: 10px 30px
Ha mind a 4 irányban azonos a kitöltés, elég 1 érték:	padding: 20px

Ugyanezt a logikát követik…

Külső margók	***Keretek***
margin-top: 10px; margin-right: 30px; margin-bottom: 10px; margin-left: 30px;	border-top-width: 10px; border-right-width: 30px; border-bottom-width: 10px; border-left-width: 30px;
margin: 10px 30px 10px 30px	border-width: 10px 30px 10px 30px
margin: 10px 30px 10px	border-width: 10px 30px 10px
margin: 10px 30px	border-width: 10px 30px
margin: 20px	border-width: 20px

A keretek paraméterei:

szélesség:	border-width: 5px numerikus érték vagy thin, medium, thick
szín:	border-color: grey
stílus:	border-style: ridge dashed (szaggatott), dotted (pontozott), double (kettős), groove (domborzat-árnyékolt 1), hidden (rejtett), inherit (örökölt), inset (homorú), none (nincs keret), outset (domború), ridge (domborzat-árnyékolt 2), solid (egyszerű; alapértelmezés)

Fentiek egy deklarációban is megadhatók (de abban mindhárom tulajdonságnak értéket kell adni): Így a keret minden oldalon azonos stílusú lesz. Az irányszavak beszúrásával azonban akár minden oldalon eltérő szélességű, színű és stílusú keretünk lehet, azaz a CSS2 összesen 16 további keret-tulajdonságot ismer: Ez a részletesség pl. akkor hasznosul, amikor a dobozokat egymástól vízszintes vagy függőleges egyenesekkel választjuk el.	border: 5px grey ridge border-top: 2px red dashed; border-right: 4px yellow dotted; border-bottom: 6px green double; border-left: 8px blue groove; border-top-width border-top-color border-top-style border-right-width border-right-color border-right-style border-bottom-width border-bottom-color border-bottom-style border-left-width border-left-color border-left-style
A körvonalak paraméterei:	
A körvonalak annyiban különböznek a kerettől, hogy a testvérelemeket nem tolják el, hanem elfedik, tehát az elemek *elhelyezkedésére* semmilyen hatással nincsenek. Szélességük, színük és stílusuk minden irányban azonos.	outline: 2px red dashed … annyit tesz, mint: outline-width: 2px; outline-color: red; outline-style: dashed;
Méretezés A szélesség és a magasság megadja, hogy mekkora legyen az elem. ***Méret alsó korlátozása*** A minimális szélesség és a minimális magasság megadja, hogy *legalább* mekkora legyen az elem. Ezt a méretet akkor is megtartja, ha a tartalom nem tölti ki és akkor is, ha a böngészőablakot lekicsinyítjük. ***Méret felső korlátozása*** A maximális szélesség és maximális magasság megadja, hogy az elem *maximálisan* mekkora lehet. Ezt a méretet akkor sem lépheti túl, ha a tartalom nem fér el benne („túlcsordul”) és akkor sem, ha a böngészőablakot megnöveljük.	width: height: min-width: min-height: max-width: max-height:
Túlcsordulás	

Ez a tulajdonság adja meg, mi történjék azzal a tartalommal, ami nem fér el egy dobozban, pl. egy width vagy egy max-width tulajdonság miatt.

overflow:

- visible: a tartalom látható (a dobozon kívül) – az alapértelmezés
- hidden – elrejti a túlcsorduló tartalmat
- scroll – MINDKÉT csúszkát megjeleníti (amelyik nem szükséges, az inaktív)
- auto – csak a szükséges csúszkát jeleníti meg (a függőlegeset, a vízszinteset, mindkettőt vagy egyiket sem)
- inherit – örkölt érték

Maguk a csúszkák nem növelik meg az elem méretét.

CSS-pozícionálás

A pozíció és a kapcsolódó tulajdonságok és értékeik

Pozícionálás

position:

- static – A sztatikus pozícionálás alapértelmezés; az elemek normális folyama. Az elem ott jelenik meg, ahol éppen tartunk a dokumentumban.
- relative – A relatívan pozícionált elem a balról és fentről tulajdonságokkal elmozdítható (a sztatikus helyzetéhez képest). Az utána következő elem elhelyezése azonban ugyanúgy történik, mintha ez a helyén maradt volna, tehát a relatív pozícionálás az elemek folyamát nem módosítja.
- absolute – A legközelebbi nem-static pozícionálású szülőelemhez képesti pozícionálás; itt a balról és fentről való távolságon kívül a lentről és jobbról tartandó távolság is megadható. A kódban utána következő elem elfoglalja a megüresedett helyet.
- fixed – A rögzített pozícionálás az abszolúthoz hasonlóan kiemeli az elemet az elemfolyamból, de az elmozdítás mindig a bodyhoz képest történik.

Doboz elmozdítása

fentről számított távolság:
balról számított távolság:
lentről számított távolság:
jobbról számított távolság:

Hogy *mihez képest* történik az elmozdítás, azt a position: értéke határozza meg.

top: 150px
left: 150px
bottom: 150px
right: 150px

A top és a left a doboz bal felső sarkának, a bottom és a right a jobb alsó sarkának helyzetére vonatkozik. Mind a 4 tulajdonság negatív értéket is felvehet, amivel a doboz akár képen kívülre is helyezhető.

Lebegés

Elem körülfuttatása a forráskódban utána következő elemekkel. Hasonló az img tag align="" attribútumához.

```
h1 {
float: right
}
```

- left – az elem baloldalt áll (körülfuttatás jobbra)
- right – az elem jobboldalt áll (körülfuttatás balra)
- none – az alapértelmezés

Lebegés törlése: azt az elemet, amellyel körülfuttattuk az előzőt, ki lehet vonni ezen viselkedés hatálya alól.

```
p {
clear: left
}
```

- left – az elem nem futja körül az előzőt jobbra
- right – az elem nem futja körül az előzőt balra
- both – az elem nem futja körül az előzőt

Azok az elemek, amelyeknél törölték a lebegést, maguk körülfuttathatók maradnak (ha kapnak egy floatot).

Rétegek Ha több elemet úgy pozícionálunk, hogy egymást fedjék, a kódban később álló elem eltakarja a korábban állót. Az elemek z-tengelyen elfoglalt pozícióját a z-index-szel változtathatjuk meg. A z-tengelyt a kijelzőre merőleges tengelyként képzelhetjük el. Minél magasabb egy elem z-indexe, annál „közelebb” van a nézőhöz. Ha két elem ugyanazt a z-indexet kapja, akkor megint a kódbeli sorrend lesz a döntő.	z-index: 1

Kommentálás
 	
kód strukturálására, magyarázására, deklarációk hatályon kívül helyezésére (kikommentelés)
 	
az egész soros kommentek elhelyezése még gyorsabb
	/* - - - - - B E K E Z D É S E K - - - - - */
	
	/*
	p {
	margin-top: 10px;
	margin-left: 20px;
	}
	*/
	
	p {
	margin-top: 10px;
	// margin-left: 20px;
	}