

Tómacs Tibor

L^AT_EX

Utolsó módosítás: 2018.12.07. 10:09

```
\documentclass{article}  
\usepackage[T1]{fontenc}  
\usepackage[english]{babel}  
\begin{document}  
Hello Eszterházy Károly  
University!  
\end{document}
```




Tómacs Tibor

L^AT_EX

EGER, 2018

© Tómacs Tibor, 2018

A könyv minden részlete \LaTeX -ben készült, beleértve a borítót és az ábrákat is. Ez alól csak az Eszterházy Károly Egyetem logója, Donald Ervin Knuth és Leslie Lamport fotói, Duane Bibby rajza, valamint a 220. oldalon található animáció kivételek.

Utolsó módosítás: 2018. december 7. 10:09

Tartalomjegyzék

Előszó	XI
Jelölések	XII
1. Bevezetés	1
1.1. A \LaTeX koncepciója és jellemzői	2
1.2. \TeX -disztribúciók	2
1.3. \LaTeX editorok	3
1.4. \LaTeX használata online és mobil eszközökön	3
1.5. Telepítés	3
1.6. \LaTeX -csomagok frissítése	4
1.7. A \TeX Live verziófrissítése illetve eltávolítása	5
1.8. Fontosabb fájlkiterjesztések	5
1.9. A \TeX -rendszer fontosabb programjai	5
1.10. TeXstudio beállítások	9
2. Az első lépések	11
2.1. A \LaTeX alapfogalmai	11
2.2. Fontosabb standard dokumentumosztályok	14
2.3. Az első dokumentum elkészítése	15
3. A dokumentum nyelve	19
3.1. A babel csomag	19
3.2. A szavak elválasztása	20
3.3. Sorvégi túlcsoordulás	21
3.4. A magyar.ldf aktív karakterei	21
4. Alapvető formai elemek	24
4.1. Karakterek	24
4.1.1. Foglalt karakterek	24
4.1.2. Ékezetes betűk	24
4.1.3. Speciális betűk	25
4.1.4. Ligatúrák	25
4.1.5. Különleges karakterek	25
4.2. Szóközök	27
4.3. Központosítás	28
4.4. Betűváltozatok	30
4.4.1. Osztyályozás	30
4.4.2. Kurzív kiegyenlítés	31
4.4.3. Kiemelés	32

4.5.	Betűméretek	33
4.5.1.	Alapbetűméret	33
4.5.2.	Relatív betűméretek	33
4.5.3.	Abszolút betűméretek	34
4.6.	Térközök	34
4.6.1.	Fix méretű vízszintes térközök	34
4.6.2.	Rugalmas méretű vízszintes térközök	35
4.6.3.	Fix méretű függőleges térközök	35
4.6.4.	Rugalmas méretű függőleges térközök	36
4.6.5.	Sortávolság	36
4.7.	Törések	37
4.7.1.	Sortörések	37
4.7.2.	Oldaltörések	37
4.8.	Bekezdések	37
4.8.1.	Bekezdések balra zárása	38
4.8.2.	Bekezdések jobbra zárása	38
4.8.3.	Bekezdések középre zárása	38
4.8.4.	Többsoros idézetek	39
4.8.5.	Versek	39
4.8.6.	Párbeszéddek	40
4.9.	Tabulálás	40
4.10.	Lábjegyzetek	41
4.11.	Széljegyzetek	43
4.12.	Színek kezelése	43
4.12.1.	Színmodellek és paraméterek	43
4.12.2.	Színnevek	44
4.12.3.	Színes szöveg	44
4.12.4.	Szöveg kiemelése színes háttérrel	45
4.12.5.	Szöveg kiemelése színes aláhúzással	45
4.12.6.	Színes lapok	45
4.13.	Dátumtípusok	46
5.	Oldalak kinézete	47
5.1.	Oldalak szerkezete és méretei	47
5.2.	Oldalak nagyítása/kicsinyítése	48
5.3.	Többhasábos szedés	49
5.4.	Oldal elforgatása	49
5.5.	Méretek ellenőrzése	50
6.	Kereszthivatkozások	52
6.1.	Címkék	52
6.2.	Hivatkozás címkézett elemekre	53
7.	Listák	55
7.1.	Számozatlan listák	55
7.1.1.	Felsorolásjelek megváltoztatása	55
7.1.2.	Számozatlan listák extra térközök nélkül	57
7.2.	Leíró listák	58
7.3.	Számozott listák	59

7.3.1.	Számozott listák számozási stílusának megváltoztatása	59
7.3.2.	Hivatkozás számozott listaelemre	64
7.3.3.	Számozott listák extra térközök nélkül	65
7.3.4.	Sorfolytanos számozott listák	65
8.	Képek	67
8.1.	Kép beillesztése	67
8.2.	Hát- illetve előtérbe illesztés	69
8.3.	Külső pdf oldalak beszúrása	69
9.	Ábrák készítése	71
9.1.	Koordináta-rendszer, referenciapont és vonalvastagság	71
9.2.	Szakaszok, törött vonalak és vektorok	72
9.3.	Körvonalak	74
9.4.	Lekerekített sarkú téglalapok	76
9.5.	Bézier-görbék	76
9.6.	Útvonalak	77
9.7.	Vonalak végeinek és útvonalak csatlakozási pontjainak stílusa	80
9.8.	Betűk elhelyezése ábrában	82
9.9.	Koordináta megadása hossz mérettel	84
10.	Táblázatok	85
10.1.	Példatáblázatok	85
10.2.	Hosszú táblázatok	91
10.3.	Kiadói minőségű táblázatok	92
10.4.	Táblázatok alapvonalhoz igazítása	93
11.	Objektumok úsztatása	95
11.1.	Képek és táblázatok úsztatása	95
11.2.	Úsztatott objektumok címkézése	96
11.3.	Saját úsztatott objektumok létrehozása	98
11.4.	Úsztatás mellőzése	98
11.5.	Objektumok körbefuttatása szöveggel	98
12.	Dobozok	100
12.1.	Egysoros dobozok	100
12.2.	Bekezdésdobozok	102
12.3.	Vonaldobozok	104
12.4.	Dobozok egymásra helyezése	104
12.5.	Dobozok forgatása	105
12.6.	Dobozok nyújtása, tükrözése	105
12.7.	Dobozok átméretezése	106
12.8.	Doboz méreteinek nullázása	106
12.9.	Láthatatlan dobozok	107
13.	Verbatim, programkód, URL	108
13.1.	Verbatim	108
13.2.	Verbatim szöveg kiírása fájlba	110
13.3.	Programkódok	112

13.4. URL címek megadása	117
14. Képletek	118
14.1. Matematikai mód	118
14.2. Matematikai betűváltozatok	120
14.3. Kalligrafikus, dupla szárú betűk és fraktúrák	121
14.4. Görög betűk	122
14.5. Matematikai ékezetek	122
14.6. Műveleti jelek	123
14.7. Relációjelek	123
14.8. Közönséges matematikai jelek	125
14.9. Három pont	125
14.10. Matematikai zárójelek	126
14.11. Esetek szétválasztása	129
14.12. Matematikai jelek több szerepben	129
14.13. Változó hosszúságú vízszintes jelek	130
14.14. Gyökvonás	130
14.15. Mátrixok	131
14.16. Matematikai jelek egymásra helyezése	133
14.17. Matematikai indexek	133
14.18. Törtek, binomiális együtthatók	134
14.19. Operátorok, függvények	135
14.19.1. Nagy operátorok	135
14.19.2. „Nolimits” függvények	136
14.19.3. „Limits” függvények	136
14.19.4. Új függvények definiálása	137
14.19.5. Differenciál operátor, differenciálás	139
14.20. Képletek bekeretezése	139
14.21. Kommutatív diagramok	140
14.22. Kiemelt képletek sorszámozása	141
14.23. Képletek eltörése	142
14.24. Több képlet egymás alatt	144
14.25. Több képlet egymás alatt illesztéssel	146
14.26. Részformulák számozása	151
14.27. Oldaltörés többsoros képletekben	152
14.28. Táblázat matematikai módban	152
15. További formai elemek	154
15.1. Círil betűk	154
15.2. Gótikus írás	155
15.3. Görög betűk	156
15.4. Iniciálék	156
15.4.1. Latin iniciálé	156
15.4.2. Díszes latin iniciálé	157
15.4.3. Gótikus iniciálé	157
15.4.4. Díszes gótikus iniciálé	158
15.5. Betűk kontúrozása és árnyékolása	159
15.6. Alá- és föléhúzás egyszerre	160
15.7. Dátumtípusok automatikus toldalékolása	160

15.8. Számok automatikus toldalékolása	161
15.9. Lorem ipsum	161
15.10. T _E X-hel kapcsolatos logók	162
15.11. Prímszámok kiírása	162
15.12. Vonalazott lapok	163
15.13. Négyzetrácsos lapok	163
15.14. Az oldal két pontjának összekötése vonallal	163
15.15. Nem vízszintes alapvonalú szöveg szedése	165
15.16. Vízjel	167
15.17. A pdf készítésének ideje óra percben	167
15.18. QR-kód	168
15.19. Vonalkód	168
15.20. Különleges bekezdések	169
15.21. Vágójelek nyomdai előkészítéshez	170
16. Strukturált művek	172
16.1. Főcím, címlap, kivonat	172
16.2. A főszöveg szintjei	173
16.3. Fattyúsorok	174
16.4. Fej- és láblécek	175
16.4.1. Alapbeállítások	175
16.4.2. Fej- és láblécek testreszabása	176
16.5. Jegyzékek	180
16.5.1. Tartalomjegyzék	180
16.5.2. Táblázatok jegyzéke	181
16.5.3. Ábrák jegyzéke	182
16.5.4. Kódok jegyzéke	182
16.5.5. Saját úsztatott objektumok jegyzéke	183
16.5.6. Jegyzékek stílusának szerkesztése	183
16.6. Tételszerű bekezdések	184
16.7. Bibliográfia	190
16.7.1. Bibliográfia készítése környezettel	190
16.7.2. BibT _E X	191
16.8. Függelék	194
16.9. Tárgymutató	195
16.10. Hosszabb művek szervezése	197
17. Elektronikus publikáció	199
18. Szakdolgozat készítése	201
19. Prezentációk	203
19.1. Témák	203
19.1.1. Teljes témák	204
19.1.2. Belső témák	205
19.1.3. Külső témák	205
19.1.4. Színtémák	205
19.1.5. Betűtípus témák	206
19.2. Keretek	206

19.3.	Egy keretben több dia	207
19.3.1.	Overlay specifikációk	208
19.3.2.	Diasorozat átlátszósága	209
19.3.3.	Overlay specifikációval rendelkező parancsok	209
19.4.	Diaváltás látványeffektekkel	211
19.5.	A prezentáció tagolása	212
19.5.1.	Címoldal	212
19.5.2.	A főszöveg tagolása	212
19.5.3.	Tartalomjegyzék	213
19.5.4.	Irodalomjegyzék	215
19.6.	Tartalmi elemek	215
19.6.1.	Listák	215
19.6.2.	Tömbök, tételszerű környezetek	216
19.6.3.	Dobozok	217
19.6.4.	Többhasábos terület	218
19.6.5.	Háttér	218
19.6.6.	Képek	219
19.6.7.	Animáció	219
19.6.8.	Videó	220
19.6.9.	Nagyítás	221
19.6.10.	Kereszthivatkozás	222
19.6.11.	Nyomógombok	223
19.6.12.	Keret ismétlése	224
20.	A \LaTeX programozása	225
20.1.	ASCII kódolás és kategória kódok	225
20.2.	Hosszúságparancsok	228
20.3.	Számlálók	229
20.4.	Vezérlő utasítások	231
20.4.1.	Feltételes utasítások	232
20.4.2.	Esetszétválasztás	235
20.4.3.	Ciklusok	235
20.5.	Parancsok definiálása	236
20.6.	Környezetek definiálása	241
20.7.	Környezet horgonyok	244
21.	Stílusfájlok írása	246
21.1.	Csomag készítése	246
21.2.	Dokumentumosztály készítése	248
22.	Fontok kiválasztása	250
22.1.	\LaTeX fontkatalógus	250
22.2.	A forrásfájl fontkódolása és a \LaTeX belső kódkészlete	250
22.3.	Globális beállítás	252
22.3.1.	Család	252
22.3.2.	Testesség	252
22.3.3.	Alak	253
22.4.	Lokális beállítás	253
22.5.	Fontcsaládnév deklarálása	254

22.5.1.	Több fontcsalád összevonása új néven	254
22.5.2.	Új fontcsaládnév deklarálása	256
22.6.	Új családosztály definiálása	257
22.7.	Új testességosztály definiálása	258
22.8.	Új alakosztály definiálása	260
22.9.	Alapértelmezett osztálykombinációk bővítése	261
22.10.	Fontok információi és tesztelése	262
22.11.	Fontváltó csomagok	264
23.	X_YLaTeX	265
23.1.	Fordítás	265
23.2.	Jellemzők	265
23.3.	Fontok betöltése	266
23.4.	Az ifxetex csomag	267
24.	További információk	268
24.1.	Hasznos csomagok	268
24.2.	Ha PDF-ben a betűk nem vektorgrafikusan jelennek meg	269
24.3.	PDF-ből kimásolt szöveg	270
24.4.	HTML oldalakon képletek megjelenítése közvetlenül LaTeX forrásból	270
24.5.	A hyperref csomag egy hibája	270
24.6.	dottedtocline=fix	270
24.7.	Ha a magyar nem alapnyelvként van beállítva	271
24.8.	A magyar.ldf téves kódolási figyelmeztetése	271
25.	Linkek	272
25.1.	Videóleckék	272
25.2.	Gyakorlatok	272
25.3.	Sablonok	273
25.4.	TeX-rendszerek	273
25.5.	Installálás nélkül, online működő TeX-rendszerek	273
25.6.	Mobil eszközökön működő TeX-rendszerek	274
25.7.	TeX-hez fejlesztett editorok	274
25.8.	Leírások	274
25.9.	Magyar tipográfiát követő segédfájlok	275
25.10.	LaTeX oldalak	275
25.11.	LaTeX fórumok	275
25.12.	LaTeX fontok	275
25.13.	Segédprogramok	275
	Irodalomjegyzék	277

Előszó

Ebben a könyvben az Eszterházy Károly Egyetem „*Számítógépes kiadványszerkesztés*” című előadásainak és gyakorlatainak bővített tananyagát foglaltam össze, mely a \LaTeX világába vezeti be az Olvasót.

Ma már a világ minden felsőoktatási intézményében ismert és standardként használt eszköz a \TeX -rendszer, melynek a \LaTeX is része. Ez tulajdonképpen egy magas szintű dokumentumleíró nyelv.

Ezzel a rendszerrel 1990-ben ismerkedtem meg. Azóta számos tananyagot, könyvet és cikket szerkesztettem vele. Több folyóirat technikai szerkesztőjeként rengeteg szerzők által elkövetett hibával találkozom, melyek számomra az oktatásban fontosak, hiszen így jobban látom, hogy mire kell a \LaTeX tanításában nagyobb hangsúlyt fektetni. Ezt igyekeztem kamatoztatni ebben a leírásban is.

A könyvet próbáltam gyakorlatias oldalról megközelíteni, ugyanakkor kézikönyvként is használható. Néha nehéz megkerülni az elméletet. Így van ez a „*Bevezetés*” című fejezettel is, ahol az Olvasó sok olyan dologgal találkozhat, amit még a gyakorlatban nem próbált ki. Erre azért van szükség, mert a későbbiekben az itt bevezetett fogalmakat gyakran fogjuk használni.

Szeretném felhívni a figyelmet a [videóleckékre](#) és a [gyakorlatokra](#), melyek jelentősen megkönnyítik az önálló tanulást.

Reményeim szerint, a kurzus elvégzése után az Olvasó természetesnek veszi majd, hogy szakdolgozatának vagy bármely más jellegű publikációjának, dokumentumának elkészítéséhez \LaTeX -rendszert használ. Ha észrevétele, megjegyzése van, kérem írjon a tomacs.tibor@uni-eszterhazy.hu címre.

A \TeX -et használók többmilliós táborának jelmondatával kívánok az Olvasónak sok \LaTeX -ben megírt igényes kiadványt!



Dr. Timács Tibor
egyetemi docens

Jelölések

A \LaTeX -ben a szerkesztés során alapvető az ún. *parancsok* használata. Ezek általános leírására a következő példában látható jelölést fogjuk használni:

```
\textbf{<szöveg>}
```

A $\langle \rangle$ jelek közé írt rész helyére olyan kódot kell beírni, melyet az adott parancs magyarázatánál adunk meg. Például ebben az esetben a $\langle szöveg \rangle$ helyére beírt betűket ez a parancs félkövéren fogja kiszedni. Konkrét *példakódot* a következő módon jelöljük:

```
\textbf{ABC}
```

Ennek eredménye így lesz jelölve:

ABC

A szerkesztési lehetőségeket ún. *csomagokkal* lehet bővíteni. Például az $\backslash euro$ parancs az *eurosym* csomag betöltésével használható, amit így fogunk jelölni:

```
\euro \in eurosym
```

A csomagokat többféle *opcióval* is be lehet tölteni. Például az $\backslash ontoday$ parancs csak a *babel* csomag *magyar* opciójával használható, amit így fogunk jelölni:

```
\ontoday \in [magyar]babel
```

A kódokban a következő példán látható módon ki fogjuk emelni az ún. *kommenteket*.

```
\title{Cím} % Itt kell beírni a címet!
```

Ha valamit *parancssorba* kell írni, azt a következő példán látható módon fogjuk jelölni:

```
latex dokumentum.tex
```

Ha egy kód valamelyik sora hosszabb mint e könyv szövegének a szélessége, akkor a kód megtörve lesz kiszedve, de annak begépelésénél az egészet egy input sorba kell írni. Az ilyen ún. *puha töréspontokat* \rightarrow módon fogjuk jelölni. Például

```
perl -x husort.pl -s gind -C circum2 -C latin2 -C separate_tags -C  $\rightarrow$   
single_symbols -C shadow_untagged -C no_vowel_equiv dokumentum.idx
```

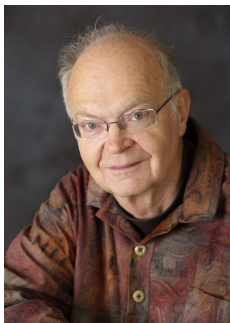
Egy program *menüjére* \blacktriangleright Menü \blacktriangleright Almenü \blacktriangleright Alalmenü formában utalunk, míg annak egy *gombját* Gomb módon jelöljük. A *billentyűzet* egy nyomógombjára Billentyű módon utalunk. Ha egyszerre több billentyűt kell megnyomni, akkor közéjük + jelet teszünk. Például Ctrl + N. A *linkeket* ilyen színű szöveggel jelöljük, míg a *videókat* a következő ábrára kattintva nézheti meg:



1. fejezet

Bevezetés

DONALD ERVIN KNUTH stanfordi matematikus 1977-ben egy olyan számítógépes programot fejlesztett ki, amely a nyomdászat minden tudását képes modellezni. Tette mind ezt azért, hogy „*A számítógép-programozás művészete*” című könyvét megfelelő formába önthesse. A programot a görög τέχνη (jelentése: művészet, mesterség; kiejtése: *techné*) szó első három betűjéből T_EX-nek keresztelte el, ami egyúttal a *text* (szöveg) szóra is utal. Így a kiejtése nem „teksz”, hanem „tekh”, mint a *technika* szóban. A T_EX márkajeleket egy egyszerű szövegfájlba **TeX** módon kell beírni.

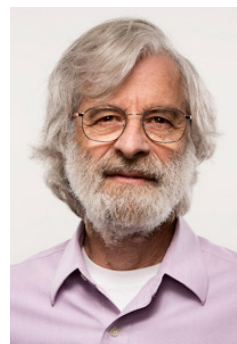


D. ERVIN KNUTH¹ Számítástechnikai hasonlaltal, a T_EX nevezhető a nyomdászat assemblerének is, mellyel minden tipográfiai feladat megoldható. Ezzel azonban csak fáradtságos úton, sok száz elemi parancs használatával tudunk dolgozni. Ezért szükség volt olyan makrócsomag létrehozására, mely magasabb szintű programozási nyelven, jóval könnyebben kezelhető.

Az első ilyen makrócsomagot maga Knuth írta, és Plain T_EX-nek nevezte el. Ennek a dokumentációját is elkészítette „*The T_EXbook*” címmel [2]. Egy másik makrócsomagot MICHAEL SPIVAK fejlesztett ki, melyet az American Mathematical Society (AMS) támogatott, és $\mathcal{A}\mathcal{M}\mathcal{S}$ -T_EX-nek nevezték el. Ez a fő hangsúlyt a matematikai képletek tipográfiájára helyezte. Magyar nyelven a Plain T_EX és az $\mathcal{A}\mathcal{M}\mathcal{S}$ -T_EX használatáról [1] ad rövid áttekintést.

Az általános tipográfiára LESLIE LAMPORT írt makrócsomagot L^AT_EX néven, aki 1989-ben visszavonult a fejlesztésétől. Ekkor a stanfordi T_EX-találkozó után létrejött egy munkacsoport, mely a L^AT_EX újraírását és kiterjesztését tűzte ki célul. 1994-ben jelent meg a L^AT_EX 2_ε (ejtsd: latekh kettő e), ami magába olvasztotta az $\mathcal{A}\mathcal{M}\mathcal{S}$ -T_EX tudását is. A L^AT_EX 2_ε folyamatosan bővül ún. csomagokkal, melyek bizonyos speciális feladatok elvégzését könnyítik meg. Mára a L^AT_EX 2_ε (a továbbiakban röviden csak L^AT_EX) vált a legnépszerűbb makrócsomaggá. A L^AT_EX márkajeleket egy egyszerű szövegfájlba **LaTeX** módon kell beírni.

Amikor T_EX-rendszerről beszélünk, akkor ezalatt az egész rendszert értjük, ami magába foglalja a Plain T_EX-et, $\mathcal{A}\mathcal{M}\mathcal{S}$ -T_EX-et, L^AT_EX-et és számos olyan dolgot, amit itt nem részletezünk, például a METAFONT nevű programot is, mely betűkészletek létrehozására alkalmas.



LESLIE LAMPORT²

¹ Forrás: <https://alchetron.com/Donald-Knuth-682882-W>

² Forrás: <http://www.brandeis.edu/commencement/honorees/lamport.html>

Évente sok ezer könyv, cikk, oktatási segédanyag, szakdolgozat, doktori disszertáció stb. jelenik meg \LaTeX -ben. Egyes tudományokban, mint a matematika, fizika, informatika, stb., a használata szabvánnyá vált, a legtöbb tudományos folyóirat csak ebben fogad el kéziratot. Magyarországon például a Typotex Kiadó minden kiadványa \LaTeX -rendszerben készül.

1.1. A \LaTeX koncepciója és jellemzői

A számítógépes szövegszerkesztő programok megjelenésével a szerzők a dokumentum megírásától annak tördeléséig mindent maguk végeznek. Azonban a legtöbben a tipográfiahoz és a szedéshez nem értenek, így sok olyan mű készül, amely nem felel meg a nyomdai követelményeknek.



A \TeX SZIMBÓLUMA³

A \LaTeX koncepciója szerint, a szerző a tipográfusi munka jelentős részét a \LaTeX -re bízta, a szedési munkát pedig a \TeX végzi el. Ettől függetlenül természetesen a végső forma minden apró részletét befolyásolhatjuk, sőt saját stílusállományt is írhatunk, de ez csak tipográfiai tudással és a \LaTeX mélyebb ismeretével ajánlott.

A teljes \TeX -rendszer – így a \LaTeX is – ingyenes és nyílt forráskódú program. A \LaTeX segítségével professzionális tipográfia érhető el, beleértve a matematikai képleteket is.

Az irodalomjegyzékek, tartalomjegyzékek, szójegyzékek, lábjegyzetek és kereszt-hivatkozások automatikusan számozódnak, így állandó utólagos javítgatásokra nincs szükség.

A mai számítógépes programok közül a \LaTeX tudja a bekezdéseket a legoptimálisabban tördelni. Minden operációs rendszeren hozzáférhető, továbbá egy rendszeren megírt mű egy másik rendszeren is ugyanazt az eredményt adja, nincs áttördelési effektus. Egy kiadónak vagy egy nyomdának talán ez az egyik legfontosabb feltétel.

Nagy terjedelmű dokumentum forrása és az eredményt jelentő PDF fájl is csekély méretű, így internetes publikálásra ideális.

Ha a \TeX -et a nyomdászat assemblerének neveztük, akkor a \LaTeX egy magas szintű dokumentumleíró nyelvnek is tekinthető. A dokumentumunk \LaTeX -forrása egy szöveges állomány, mely együtt tartalmazza a kiadvány szövegét és a \LaTeX parancsait (hasonlóan egy html dokumentumhoz, csak ott nem parancsok, hanem tagek vannak). Így a szerkesztés során nem azt látjuk, amit a végén kapunk. Ez a kezdő felhasználónak hátrány, de a gyakorlat megszerzése után már előnyként fogjuk élvezni, mert ezáltal vizuális szerkesztésre nincs szükség, csak a tartalomra kell koncentrálni.

Sajnos a tipográfiai szabályoknak megfelelő új stílus kialakítása bonyolult, ezért a kezdő felhasználónak a már meglévők használata ajánlott. További hátrány, hogy leírás nélkül nem lehet boldogulni. A hibák megtalálása, javítása adott esetben nehéz lehet, de ez a gyakorlat megszerzésével illetve megfelelő editor használatával könnyebbé válik.

1.2. \TeX -disztribúciók

A \TeX -rendszer minden géptípuson és minden operációs rendszeren hozzáférhető. Az egyik legnépszerűbb \TeX -disztribúció, a Linuxon és Windowson is egyaránt működő TeX Live. Ennek van egy MacTeX nevű Mac OS-en működő verziója is. Létezik egy

³ Tervezte Duane Bibby (CTAN lion drawing by Duane Bibby; thanks to www.ctan.org).

másik T_EX-disztribúció is, a szintén népszerű MiKTeX. Ez először csak Windowson működött, de ma már telepíthető Linuxon és Mac OS-en is.

1.3. L^AT_EX editorok

A szerkesztett dokumentum forrása egy szöveges állomány, ami bármely editoron létrehozható. Jelentősen megkönnyíthetjük a szerkesztést, ha olyan editoron dolgozunk, amely a L^AT_EX-re lett optimalizálva (automatikus parancs kiegészítő L^AT_EX-parancsokhoz, PDF és forrás közötti szinkronizálás, automatikus hibakereső, parancssori programokhoz rendelt ikonok, stb.). A TeX Live és a MiKTeX tartalmaz ilyen editort, melynek a neve TeXworks. Ezen kívül még számos L^AT_EX-re kifejlesztett szerkesztő létezik. A legnépszerűbbek: Kile, TeXnicCenter, WinEdt, Texmaker, TeXstudio. Sok éves tapasztalatom alapján kimagaslóan a legjobb L^AT_EX-editornak minden tekintetben a TeXstudiot tartom.








1.4. L^AT_EX használata online és mobil eszközökön


Az [Overleaf](#) weboldal internetes böngészőben ad szerkesztési lehetőséget, és a végeredményt jelentő PDF fájlt is egy szerveren található T_EX-rendszer generálja, anélkül, hogy a saját gépünkre telepíteni kellene azt.

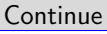
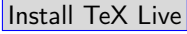

Mobil eszközökön is lehetséges a L^AT_EX használata. Például a [LaTeX Editor](#) egy Androidon futtatható ingyenes alkalmazás, amely offline is használható. Internetes kapcsolat akkor kell hozzá, ha egy hiányzó csomagot tölt le. Egy másik ingyenes Androidos alkalmazás a [VerbTeX](#). Ez offline nem használható. Fordításkor egy online elérhető szerverre telepített TeX Live rendszert használ.



1.5. Telepítés

Windows vagy Linux operációs rendszerek esetén javaslom a TeX Live és a TeXstudio együttes használatát. Itt csak a Windowsra telepítést részletezzük.

1. Töltse le a TeX Live telepítésvezérlőjét ([klikk ide](#)), majd futtassa. Ezután kövesse az utasításokat:  *Simple install (big)* →  →  →  3-szor → . A telepítés akkor fejeződik be sikeresen, ha megjelenik a „Welcome to TeX Live!” felirat. Ezután  → .

Ha a telepítés leáll a „perl.exe működése leállt...” hibaüzenettel, akkor állítsa át a TeX Live telepítésvezérlőjének (`install-tl-windows.exe`) kompatibilitását a következő módon: Kattintson a fájlra, nyomja meg a jobb egérgombot és válassza a *Tulajdonságok* menüpontot. Ezután *Kompatibilitás*, ☒ *Futtatás a következő kompatibilitási üzemmódban*, majd a legördülő listában válassza ki a *Windows XP* sort, végül . Ezután indítsa el a programot. Ha ez nem vezet eredményre akkor a következő két telepítési eljárás valamelyikét kell választani:




- a) Töltse le az `install-tl.zip` fájlt ([klikk ide](#)), csomagolja ki, majd a kicsomagolt mappában az `install-tl-advanced.bat` fájlt futtassa. Ezután  → . A telepítés akkor fejeződik be sikeresen, ha megjelenik a „Welcome to TeX Live!” felirat. Ezután .

- b) Az előző pontban kicsomagolt mappában az `install-tl-windows.bat` fájlt kell futtatni a következő módon: Nyissa meg a „Futtatás” ablakot a  +  gombokkal, írja be, hogy

```
cmd
```



majd . Az így megjelenő parancssorba írja be, hogy

```
<elérési út>\install-tl-windows -no-gui
```

majd . Gépelje be az  betűt, majd , amely elindítja a „start installation to hard disk” menüpontot. A telepítés akkor fejeződik be sikeresen, ha megjelenik a „Welcome to TeX Live!” majd a „A folytatáshoz nyomjon meg egy billentyűt...” felirat.

Ha van elég hely a merevlemezen, nem szakad meg az internetes kapcsolat, és a telepítés látszólag rendben zajlik, de a végén mégsem jut el a „Welcome to TeX Live!” feliratig, annak az lehet az oka, hogy a vírusirtó nem engedi valamelyik fájl bemásolását. Ebben az esetben kapcsolja ki a vírusirtót a telepítés idejére.

2. Töltse le a TeXstudio telepítőjét ([klikk ide](#)), indítsa el a telepítő fájlt, majd kövesse az utasításokat.






A TeXstudio néhány praktikus beállításához mentse le a [user.txprofile](#) fájlt, majd azt a TeXstudio   menüpontjával töltsse be. Néhány további beállítási lehetőségről az [1.10.](#) szakaszban lesz szó.

Minden TeX-rendszerben vannak ún. Perl programok, melyek futtatásához szükség van egy segédprogramra. Ha a TeX Live rendszert telepítette, akkor ezzel nincs további teendő, mert az tartalmaz ilyen segédprogramot. Ha MiKTeX-et használ TeX Live helyett, akkor a Perl szkript futtatót külön kell telepíteni. Erre a célra megfelel például a Strawberry Perl, mely innen telepíthető: [klikk ide](#).



Videó: Telepítés menete

1.6. L^AT_EX-csomagok frissítése

A L^AT_EX-rendszert folyamatosan bővítik újabb csomagokkal, illetve a meglévőket frissítik. Ezért célszerű néha ezeket letölteni a meglévő TeX Live rendszerünkön belül:   , majd [Update all installed](#). A „Completed” felirat megjelenése után a   menüponttal léphet ki a TeX Live Manager programból.

Ha az előző lépés nem sikerül, akkor valószínűleg a TeX Live Manager is frissítésre szorul. Ehhez töltsse le az `update-tlmgr-latest.exe` fájlt ([klikk ide](#)), majd futtassa. Ezután már frissítheti a csomagokat az előbb leírt módon.

A TeX Live Manager és a csomagok frissítését parancssorból is elvégezheti. Windows esetén a    programot futtassa. Ezután írja be a következőt:

```
tlmgr update -self -all -reinstall-forcibly-removed
```


majd .

1.7. A TeX Live verziófrissítése illetve eltávolítása

A TeX Live minden év júniusában jön ki új verzióval. Ennek telepítését az 1.5. szakaszban leírtak szerint végezheti el. Előtte az előző verziót helytakarékossági okokból célszerű – bár nem feltétlenül szükséges – eltávolítani. Ehhez Windows esetén használja a következő menüt: **Start** > **TeX Live** > **Uninstall TeX Live**.

Másik lehetőség az eltávolításra: Nyissa meg a „Futtatás” ablakot a  +  gombokkal, írja be, hogy

```
appwiz.cpl
```

majd . Válassza ki a listában a „TeX Live” sort, majd **Eltávolítás**. Mindkét módszerrel akkor fejeződik be az eltávolítás sikeresen, ha megjelenik a „*Done uninstalling TeX Live*” felirat.

1.8. Fontosabb fájlkiterjesztések

tex A Plain TeX és L^ATeX forrásfájlt jelentő szöveges állomány kiterjesztése.

bib Bibliográfiai adatbázist tartalmazó szöveges állomány kiterjesztése.

dvi (Device Independent) A TeX alaphelyzetben a forrásfájlt dvi kiterjesztésű fájlba konvertálja. Ebben csak arra vonatkozó információkat találunk, hogy a különböző fontok, képek hová kerüljenek. Raszter információkat nem tartalmaz, így közvetlenül ebből nem nyomtathatunk. Képernyőn megjeleníteni a MiKTeX részét képező YAP (Yet Another Preview), vagy a TeX Live részét képező dviout programok képesek.



ps (PostScript) Dokumentumformátum, mely a nyomdászat feltételeire lett optimalizálva.

eps (Encapsulated PostScript) Postscript alapú vektorgrafikus képformátum.


pdf (Portable Document Format) A postscript továbbfejlesztése, mely nem csak nyomtatásra, hanem monitoron való megjelenésre is optimális. Vektorgrafikus képformátum is lehet.

1.9. A TeX-rendszer fontosabb programjai


A programok áttekintéséhez tegyük fel, hogy a L^ATeX forrásfájl a `dokumentum.tex` nevet kapta, és a `"C:\első próba"` mappába tettük.

Az ismertetett programok a TeXstudioból vezérelhetők, de parancssorból is futtathatók. Utóbbi esetben Windows használata esetén nyissa meg a „Futtatás” ablakot a  +  gombokkal, írja be, hogy

```
cmd
```

majd . Az így megjelenő parancssorba írjuk be, hogy

```
cd "C:\első próba"
```

majd . A parancssori használatot csak a teljesség kedvéért írjuk le, a TeXstudio-ban ezek sokkal egyszerűbben elérhetők. A következőkben ismertetett TeXstudio beállítások a 2.12.14 verzióra érvényesek.

tex.exe

Plain T_EX forrásból készít dvi fájlt. Használata parancssorból

```
tex dokumentum.tex
```

pdftex.exe

Plain T_EX forrásból készít pdf fájlt. Használata parancssorból

```
pdftex dokumentum.tex
```

latex.exe

Ez egy ún. L^AT_EX-fordító, mely L^AT_EX forrásból készít dvi kiterjesztésű fájlt. A képeket eps formátumban kell betölteni. Kereszthivatkozások, jegyzékek esetén többször kell futtatni. Használata TeXstudióban **Eszközők** > **Parancsok** > **LaTeX**, parancssorból

```
latex dokumentum.tex
```

Ha a fordítás során hiba lép fel, akkor a fordítás egy hibaüzenettel leáll. Ha megnyomja az **Enter** gombot, akkor folytatja a fordítást a következő hibáig. Ha azt akarja, hogy a hibáknál ne álljon le a fordítás, csak naplózza azokat a `dokumentum.log` fájlba, akkor használja a latex.exe program `-interaction=nonstopmode` kapcsolóját:

```
latex -interaction=nonstopmode dokumentum.tex
```

A TeXstudio alapbeállítások esetén használja ezt a kapcsolót.

pdflatex.exe

Ez egy másik L^AT_EX-fordító, mely L^AT_EX forrásból pdf kiterjesztésű fájlt készít. A képeket jpg, png, pdf formátumban kell betölteni. Kereszthivatkozások, jegyzékek esetén többször kell futtatni. Használata TeXstudióban **Eszközők** > **Parancsok** > **PDFLaTeX**, parancssorból

```
pdflatex dokumentum.tex
```

Az `-interaction=nonstopmode` kapcsoló itt is használható, melyet a TeXstudio is használ alapbeállítások esetén.

bibtex.exe

A bib kiterjesztésű fájlban tárolt bibliográfiai adatbázist kezeli. A névsorba rendezés során az angol szabályt követi. Az ékezetes betűket akkor rakja helyesen névsorba, ha repülő ékezeteket (lásd a 4.1.2. szakaszban) használ a bib fájlban. Használata során, először a `dokumentum.tex` fájlt fordítsa a kívánt formátumba (pdf, dvi), utána TeXstudióban **Eszközők** > **Parancsok** > **Bibtex** vagy parancssorban

```
bibtex dokumentum
```

A `dokumentum.tex` fájlt ezután ismét fordítsa le.

makeindex.exe

A tárgymutatót készíti el. A névsorba rendezés során az angol szabályt követi. Használata során, először a `dokumentum.tex` fájlt fordítsa a kívánt formátumba (pdf, dvi), utána TeXstudióban **Eszközők** **Parancsok** **MakeIndex** vagy parancssorban

```
makeindex dokumentum.idx
```

A `dokumentum.tex` fájlt ezután ismét fordítsa le pdf-be vagy dvi-be.

husort.pl

A tárgymutatót készíti el. A névsorba rendezés során a magyar szabályt követi. Ez egy Perl szkript, mely nem része egyetlen standard T_EX-disztribúciónak sem. Használatához először töltsse le a `husort.pl` fájlt a szerkesztett dokumentum mappájába: [klikk ide](#). Ezután a `dokumentum.tex` fájlt fordítsa a kívánt formátumba (pdf, dvi), majd parancssorban

```
perl -x husort.pl -s gind -C circum2 -C latin2 -C separate_tags -C single_symbols -C shadow_untagged -C no_vowel_equiv dokumentum.idx
```

Ezután ismét fordítsa le a `dokumentum.tex` fájlt.

dvips.exe

A dvi kiterjesztésű fájlokat konvertálja ps-be. Használata TeXstudióban **Eszközők** **Parancsok** **DVI->PS**, parancssorból

```
dvips -o dokumentum.ps dokumentum.dvi
```

ps2pdf.exe

A ps kiterjesztésű fájlokat konvertálja pdf-be. Használata TeXstudióban **Eszközők** **Parancsok** **PS->PDF**, parancssorból

```
ps2pdf dokumentum.ps
```

latexmk.exe

Ez meghívja a `latexmk.pl` Perl szkriptet, amely a L^AT_EX forrásfájlt megfelelő számban lefordítja, futtatja a `bibtex` és a `makeindex` programokat, végül ismét megfelelő számban lefordítja a L^AT_EX forrásfájlt. Ezzel egy menetben kapunk végeredményt.

Ha pdf a cél, akkor TeXstudióból **Eszközők** **Parancsok** **Latexmk**. Parancssorból

```
latexmk -pdf dokumentum
```

Ez a `pdflatex.exe` fordítót használja. Ebben az esetben a képeket jpg, png, pdf formátumban kell betölteni.

Ha dvi a cél, akkor TeXstudióból való használatához először állítsa be a következőt: **Beállítások** **A TeXstudio beállításai** **Parancsok** Latexmk `latexmk -silent -synctex=1 %` majd **OK**. Ezután **Eszközők** **Parancsok** **Latexmk**. Parancssorból

```
latexmk dokumentum
```

Ez a `latex.exe` fordítót használja. Ebben az esetben a képeket eps formátumban kell betölteni. Arra is van lehetőség, hogy a `makeindex` helyett a `husort` programmal dolgozzon együtt a `latexmk`. Ekkor parancssorban használja a `latexmk` következő kapcsolóját:

```
-e "$makeindex='perl -x husort.pl -s gind -C circum2 -C latin2 -C ↵
    separate_tags -C single_symbols -C shadow_untagged -C no_vowel_equiv ↵
    %0 %S'"
```

A TeXstudióban való működését lásd az 1.10. szakaszban.

Ha nem akarja, hogy a fordítás minden hiba után leálljon, csak naplózza azokat, akkor használja a `latexmk` program `-silent` kapcsolóját, melyet a TeXstudio is használ alapbeállítások esetén.

Ha a `latexmk` programmal történő fordítás során keletkező munkafájlokat akarja törölni, akkor TeXstudióban **Eszközök** > **Segédfájlok törlése** illetve parancssorban

```
latexmk -pdf -c
```

vagy

```
latexmk -c
```

aszerint, hogy fordításnál használta-e vagy sem a `-pdf` kapcsolót. Ha nem csak a munkafájlokat, hanem a végeredményt jelentő pdf, ps, dvi fájlokat is törölni akarja, akkor `-c` helyett használja a `-C` kapcsolót.

latexdiff.exe

Ez meghív egy Perl szkriptet, amely két tex fájl közötti különbséget egy harmadikban mutatja meg. Például, ha a `dokumentum.tex` és a `dokumentum-rev.tex` közötti különbséget akarja megnézni, akkor parancssorban

```
latexdiff dokumentum.tex dokumentum-rev.tex > dokumentum-diff.tex
```

Az így elkészült `dokumentum-diff.tex` lefordításával a kapott pdf-ben megnézhető a különbség.

SyncTeX (Synchronize TeXnology)

Nagyobb terjedelmű dokumentum esetén rengeteg munkát meg lehet spórolni, ha a tex fájl adott pozíciójából a pdf fájl megfelelő pozíciójába tudunk ugrani és viszont. Ezt a célt szolgálja a SyncTeX program. A működéséhez használja a `pdflatex.exe`, `latex.exe` illetve `latexmk.exe` programok `-synctex=1` kapcsolóját. Természetesen dvi fájl generálása esetén ennek csak akkor van értelme, ha azt konvertálja a `dvips.exe` és `ps2pdf.exe` programok segítségével pdf-be.

A TeXstudio alapbeállítások esetén használja ezeket a kapcsolókat. Ha a fordítás befejeződött, akkor tartsa nyomva a **Ctrl** billentyűt, majd az egér bal gombjával klikkeljen a tex fájlban a megfelelő szövegrészre. Ekkor a TeXstudio átugrik a pdf fájl megfelelő részére. Ez visszafelé is működik.

texdoctk.exe (TeX Documentation Toolkit)

Ezzel a programmal a TeX-rendszer dokumentációjában kereshet. Ehhez parancssorba gépelje a következőt:

```
texdoctk
```

Célzottan is használhatja. Ha például a `geometry` csomag használatára kíváncsi, akkor TeXstudióban **Súgó** **Súgó csomagok**, és itt be kell írni a `geometry` szót, vagy parancs-sorban

```
texdoc geometry
```

1.10. TeXstudio beállítások

A következőkben ismertetett TeXstudio beállítások a 2.12.14 verzióra érvényesek.

1. Ha tárgymutatót készít, vagy BibTeX-et használ az irodalomjegyzékhez, akkor célszerű az alapértelmezett fordítót `pdflatex`-ről átállítani `latexmk`-ra:

Beállítások **A TeXstudio beállításai** **Fordítás** Alapértelmezett fordító: Latexmk

2. A TeXstudióban a `latexmk` beállítható úgy, hogy a `husort.pl` programot használja a `makeindex` helyett:

Beállítások **A TeXstudio beállításai** **Parancsok**

A „Latexmk:” utáni részt javítsa ki erre:

```
latexmk -pdf -silent -synctex=1 -e "$makeindex='perl -x husort.pl -s  
gind -C circum2 -C latin2 -C separate_tags -C single_symbols -C  
shadow_untagged -C no_vowel_equiv %0 %S'" %
```

Használata: **Eszközök** **Parancsok** **Latexmk**

Ha ezt szeretné alapértelmezett fordítónak, akkor:

Beállítások **A TeXstudio beállításai** **Fordítás**

☒ Haladó beállítások megjelenítése

Alapértelmezett fordító:

3. Ha kicsi a képernyője, és a pdf-et szeretné külön ablakban megjeleníteni, nem pedig a forrás mellett, akkor:

Beállítások **A TeXstudio beállításai** **Fordítás** PDF megjelenítő: Belső PDF néző (ablakban)

4. A TeXstudio a megnyitott zárójelet automatikusan bezárja, azaz pl. ha `{` jelet gépel be, akkor `}` fog megjelenni. Ha ezt nem akarja, akkor tegye a következőt:

Beállítások **A TeXstudio beállításai** ☒ Haladó beállítások megjelenítése

Haladó szerkesztő ☐ Zárójelpárok automatikus bezárása

5. A TeXstudio a listájában nem szereplő parancsokat kiemeli színes háttérrel, arra figyelmeztetve, hogy talán rosszul gépelte be a parancsot. Ha ezt nem akarja, akkor tegye a következőt:

Beállítások **A TeXstudio beállításai** **Szerkesztő** ☐ Helyesírás

6. Ha egy parancs fölé viszi az egeret, akkor egy súgóablak jelenik meg az adott parancsról, ami kezdőknek hasznos, de a gyors munkában zavaró lehet. Ha ezt a szolgáltatást ki akarja kapcsolni, akkor tegye a következőt:

Beállítások **A TeXstudio beállításai** ☒ Haladó beállítások megjelenítése

Haladó szerkesztő ☐ Szöveg buboréksúgójának megjelenítése a szerkesztőben

7. A TeXstudio UTF-8 fontkódolásra van beállítva. Ha ISO-8859-2 (Latin-2) kódolású forrásfájlt is rendszeresen használ, akkor a következő beállítás célszerű:

Beállítások > A TeXstudio beállításai > Szerkesztő

Alapértelmezett karakterkódolás: ISO-8859-2

Beállítások > A TeXstudio beállításai > Általános ☐ Előző munkamenet visszaállítása indításkor

Így egy Latin-2 vagy UTF-8 kódolású fájl betöltve helyes lesz az editor kódolása. Amikor nyitva hagy egy dokumentumot és úgy zárja be a TeXstudio-t, akkor újból megnyitva, a Fájl > Munkamenet > Előző munkamenet helyreállítása módon térhet vissza az előző munkamenethez. Ha a

Beállítások > A TeXstudio beállításai > Általános ☒ Előző munkamenet visszaállítása indításkor

beállítást választaná, akkor a következő történne: Ha nyitva hagy egy Latin-2 dokumentumot és úgy zárja be a TeXstudiot, akkor azt újból megnyitva, automatikusan akarja beállítani a kódolást. De csak ISO-8859-1 (Latin-1) és UTF-8 közül tud választani, így Latin-1 kódolásúként kezelné a fájlt, kivéve, ha a dokumentum tartalmazza a következő két sor egyikét:

```
%!TeX encoding = latin2
\usepackage[latin2]{inputenc}
```

8. A TeXstudio tudását szkriptekkel bővítheti:

Makrók > Makrók szerkesztése > Hozzáadás

Név (nevezze el a szkriptet)

● Parancsfájl

LaTeX tartalom (gépelje be a szkriptet)

- Általam használt szkriptek
- A TeXstudio készítőinek szkriptgyűjteménye

2. fejezet

Az első lépések

2.1. A \LaTeX alapfogalmai

Parancs

A \LaTeX -ben a dokumentum minden formázását parancsokkal végezzük. A parancs \backslash (backslash) jellel kezdődik, majd ezt követi a parancs neve, melyben ékezetes betű, szám és szóköz nem szerepelhet (kis- és nagybetű között különbséget tesz). Például a

```
 $\text{\LaTeX}$ 
```

parancs eredménye a \LaTeX logó.

Argumentum

Egy parancsnak lehet argumentuma, amit kapcsos zárójelek között lehet megadni. Például a

```
 $\text{\texttt{\textit{szöveg}}}$ 
```

a „szöveg” szót dőlten szedi ki. Kapcsos zárójelek nélkül a parancs argumentuma a soron következő első szóköztől különböző karakter, feltéve, hogy az 1 bájtos kódolású. Több bájtos kódolású karakterek például UTF-8 kódolás esetén az ékezetes betűk. Tehát

```
 $\text{\texttt{\textit szöveg}}$ 
```

a „szöveg” szóban csak az `s` betűt szedi dőlten. De ha a forrásfájlunk UTF-8 kódolású, akkor

```
 $\text{\texttt{\textit és még valami}}$ 
```

esetén, hibával fog megállni a fordítás. Több argumentum is lehet. Például

```
 $\text{\texttt{\setcounter{page}{1}}}$ 
```

az oldalszámot 1-re állítja. Ha egy parancsnak nincs argumentuma, akkor általában az utána található szóközt nem jeleníti meg. Például a

```
 $\text{\texttt{\LaTeX -forrás}}$ 
```

eredménye: \LaTeX -forrás.

Opció

Egy parancsnak lehet opciója is, de azt nem kötelező megadni. Ha nem adjuk meg, akkor az alapopció lép érvénybe. Az opció megadása szögletes zárójelek között történik. Például egy listaelem bevezethető az

```
\item
```

paranccsal, mikor is az alapértelmezett jelet teszi ki a listaelem elé, de írhatunk

```
\item[-]
```

parancsot is, melynek hatására egy kötőjelet tesz a listaelem elé. Előfordulhat, hogy egy parancsnak opciója és argumentuma is van. Például az

```
\includegraphics[width=3cm]{abra.jpg}
```

parancs betölti az `abra.jpg` képet 3 cm szélességben. Valamikor több opció is megadható. Ekkor az opciókat vesszővel kell elválasztani. Például

```
\includegraphics[width=3cm,angle=90]{abra.jpg}
```

Környezet

A `\begin`, `\end` parancspárt környezetnek nevezzük, a kettő közötti rész pedig a környezet belseje. Ezek argumentumos parancsok, melyben a környezet nevét kell megadni. Például `itemize` környezet alatt a `\begin{itemize}`, `\end{itemize}` parancspárt értjük, ami számozatlan listát készít:

```
\begin{itemize}
  \item Listaelem
  \item Listaelem
\end{itemize}
```

Blokk

Vannak olyan parancsok, melyek az utánuk lévő részre valamilyen hatást fejtenek ki. Például az `\itshape` parancs a soron következő szöveget dőlten szedi ki. Ha azt akarjuk, hogy csak egy adott részre terjedjen ki a hatása, akkor blokkba kell zárni. Blokk kapcsos zárójelekkel adható meg. Például

```
Ez egy {\itshape nem túl izgalmas} példa.
```

esetben csak a „nem túl izgalmas” lesz kiszedve dőlten. Kapcsos zárójelek helyett használhatjuk a `\begingroup \endgroup` parancsokat is, de ezt inkább stílus illetve osztályfájlok írásánál célszerű használni.

Blokkot határoz meg egy környezet is. Például

```
\begin{itemize}
  \itshape
  \item Listaelem
\end{itemize}
```

esetén az `\itshape` csak az `itemize` környezeten belül hat.

Blokkok egymásba ágyazhatók, de nem keresztezhetik egymást. Például

```
\begin{itshape}
  \begin{ttfamily}
    szöveg
  \end{ttfamily}
\end{itshape}
```

helyes, de helytelen a következő:

```
\begin{itshape}
  \begin{ttfamily}
    szöveg
  \end{itshape}
\end{ttfamily}
```

Deklarációs parancs

Ha egy parancsnak nincs argumentuma és opciója, ugyanakkor az utána található részre hatással van, akkor azt deklarációs parancsnak nevezzük. Ilyen például az előbb említett `\itshape` parancs is. Minden deklarációs parancsnak van környezet változata is. Például az alábbi két kód ekvivalens:

```
Ez egy {\itshape nem túl izgalmas} példa.
Ez egy \begin{itshape}nem túl izgalmas\end{itshape} példa.
```

Dokumentumosztály, preambulum, dokumentumtest

A \LaTeX forrásfájl szerkezete a következő séma szerint épül fel:

```
\documentclass[opciók]{dokumentumosztály}
preambulum
\begin{document}
dokumentumtest
\end{document}
```

Elsőként egy dokumentumosztályt kell betölteni a `\documentclass` paranccsal, ami a dokumentum alapstílusát határozza meg. Például az `article` dokumentumosztályt `12pt` opcióval így kell betölteni:

```
\documentclass[12pt]{article}
```

Az ezt követő részt a `document` környezetig preambulumnak nevezzük. Ide kerülhetnek azok a parancsok, melyek az egész dokumentumra hatással vannak, de megjelenítendő szöveget nem tartalmazhat. A `document` környezet belsejét dokumentumtestnek nevezzük, mely minden megjelenítendő szöveget és parancsokat tartalmaz. Az `\end{document}` parancs után írt szöveget vagy parancsokat a \LaTeX -fordító figyelmen kívül hagyja.

Csomag

A dokumentumosztály képességeit, stílusát csomagokkal bővíthetjük. Ezeket a preambulumban kell betölteni a

```
\usepackage[opciók]{csomag neve}
```

paranccsal. Például

```
\usepackage[a5paper]{geometry}
```

az oldalt A5 méretre állítja. Ha nincs opció vagy alapopciókat használunk, akkor a szögletes zárójelek nem kellenek. Például

```
\usepackage{listings}
```

esetén programkódokat tudunk megjeleníteni. Ha több opciót is betöltünk, akkor azokat vesszővel kell elválasztani. Például

```
\usepackage[paperwidth=105mm,paperheight=75mm]{geometry}
```

esetén az oldal szélessége 105 mm és az oldal magassága 75 mm lesz. Ha alapopciókkal több csomagot is betöltünk, akkor az a következő módon is megtehető:

```
\usepackage{<csomag1>,<csomag2>,<csomag3>}
```

Például

```
\usepackage{listings,fancyhdr}
```

betölti a `listings` és a `fancyhdr` csomagokat, amit így is meg lehetett volna tenni:

```
\usepackage{listings}
\usepackage{fancyhdr}
```

Komment

Ha a forrásállományba ún. kommentet akar elhelyezni, vagyis amit a \LaTeX -fordító figyelmen kívül hagy, akkor azon szöveg elejére írjon `%` jelet. A komment vége sortörés. Például:

```
% Ez a szöveg nem jelenik meg fordítás után!
Ez megjelenik, % de ez megint nem!
```

Ha több sorból álló részt akar „kikommentezni”, akkor használja a `comment` csomag `comment` környezetét. Például

```
Ez megjelenik,
\begin{comment}
de ez nem,
és ez sem!
\end{comment}
Ez ismét megjelenik!
```

2.2. Fontosabb standard dokumentumosztályok

Korábban láttuk, hogy elsőként egy dokumentumosztályt kell betölteni, ami a dokumentum alapstílusát határozza meg:

```
\documentclass[<opciók>]{<dokumentumosztály>}
```

Itt három standard dokumentumosztályt említünk meg, melyek a legtöbb esetben megfelelnek az igényeinknek.

article Előadások, meghívók, kisebb jelentések, programdokumentációk, publikációk stb. készítéséhez használhatjuk. Főbb opciói:
10pt, **11pt**, **12pt** A dokumentum alap betűmérete. Alapopció: 10pt

- a4paper**, **a5paper**, **b5paper**, **letterpaper** Lapméret. Alapopció az angoloknál szabványos levélpapír méret: **letterpaper**. Fontos, hogy bármelyik méretet is választja, a fizikai lapméret minden esetben A4 lesz, amennyiben az alapbeállításokkal telepítette a TeX-rendszert. Ezek az opciók csak a kiválasztott lapméretnek megfelelő margókat állítják be. Ha fizikailag is be akarja állítani a lapméretet, akkor a **geometry** csomagot kell használnia (lásd az 5.1. szakaszt).
- oneside**, **twoside** Egy- illetve kétoldalas szedés. Alapopció: **oneside**.
- twocolumn** Kéthasábos szedés.
- notitlepage**, **titlepage** Címlap nincs, van. Alapopció: **notitlepage**.
- draft** Jelzi a sorvégi túlsordulásokat és az ábráknak csak a doboza jelenik meg.
- final** Nem jelzi a sorvégi túlsordulásokat és az ábrákat megjeleníti. Ez alapopció.
- report** Beszámolók, értekezések, diplomamunkák készítéséhez használhatjuk. Opciói ugyanazok, mint az **article** dokumentumosztály esetében. Alapértékek: **10pt**, **letterpaper**, **oneside**, **titlepage**, **final**. A részek és fejezetek ebben az osztályban mindig új oldalon kezdődnek. Eerre vonatkozó opciók:
- openright** A részek és fejezetek páratlan sorszámú oldalon kezdődjenek, s ennek érdekében akár üres oldalt is hagyjon.
- openany** A részek és fejezetek nyitó oldalszáma bármilyen lehet, nem csak páratlan.
- book** Könyvek írásához használhatjuk. Opciói megegyeznek a **report** dokumentumosztályéval. Alapértékek: **10pt**, **letterpaper**, **twoside**, **titlepage**, **openright**, **final**.

2.3. Az első dokumentum elkészítése

Nyissa meg a TeXstudiot és abban egy új dokumentumot **Fájl** **Új**. Írja be a következőket:

```
1 \documentclass{article}
2 \begin{document}
3 Hello World!
4 \end{document}
```

Ezt mentse el **Fájl** **Mentés**. A megjelenő ablakban a mentés előtt hozzon létre egy új mappát az előzetesen kiválasztott helyen az **Új mappa** gombbal. Lépjen be a létrehozott mappába, majd a fájl nevének megadása után **Mentés**. Fontos, hogy ezt minden dokumentum esetén tegye meg, azaz minden dokumentum külön mappában legyen, ugyanis egy dokumentumhoz több fájl is fog tartozni, melyeket célszerű egy helyen tartani és projektként kezelni.

Fordítsa le az így elkészített forrásfájlt **Eszközök** **Fordítás és megjelenítés**. Ez alap esetben a pdflatex.exe fordítót használja. Fordítás után megjelenik a pdf, melyen a „Hello World!” mondat látható 10 pt betűmérettel és a lap alján oldalszámozás van. A lap mérete A4 lesz, de a margók az angoloknál szabványos levélpapír mérethez lesznek igazítva.

Korábban láttuk, hogy az 1. sorban betöltött **article** dokumentumosztálynak az alap betűméretre három opciója van (**10pt**, **11pt**, **12pt**), a lapméretre pedig többek között van egy **a4paper** opciója. Az előbb azért jelent meg a dokumentum 10 pt be-

tűmérettel, mert a 10pt alapopció. Így ha át akar térni A4 lapméretnek megfelelő margókra és 12 pt betűméretre, akkor az 1. sort egészítse ki az alábbi módon:

```
1 \documentclass[a4paper,12pt]{article}
2 \begin{document}
3 Hello World!
4 \end{document}
```

Most írjon ékezetes betűket is a forrásba. Például a 3. sort javítsa ki így:

```
1 \documentclass[a4paper,12pt]{article}
2 \begin{document}
3 Hello Eszterházy Károly University!
4 \end{document}
```

A TeXstudio alaphelyzetben UTF-8 kódolású fontokat használ, így ezt az információt meg kell adni a forrásban is. Pontosabban ez nem feltétlenül szükséges, mert *2018-tól a L^AT_EX alapesetben UTF-8 kódolású fájlokat kezel*. Ennek ellenére mégis célszerű megadni annak érdekében, hogy régebbi telepítésű rendszeren is leforduljon a forrásfájl. Ehhez a 2. sorban töltsse be az `inputenc` csomagot `utf8` opcióval:

```
1 \documentclass[a4paper,12pt]{article}
2 \usepackage[utf8]{inputenc}
3 \begin{document}
4 Hello Eszterházy Károly University!
5 \end{document}
```

Ha a TeXstudio nyugat-európai ISO 8859-1 (Latin-1) vagy kelet-európai ISO 8859-2 (Latin-2) kódolásra van beállítva, akkor az `inputenc` csomagot mindenképpen be kell tölteni `latin1` illetve `latin2` opcióval.

Ékezetes betű nem csak billentyűzetről vihető be, hanem parancsként is. Például, ha az „a” betűre egy vesszőt akar tenni ékezetként (azaz „á” betűt szeretne), akkor használhatja a `\'{a}` parancsot. Ezt a megoldást repülő ékezetnek nevezzük. (Bővebben lásd a 4.1.2. alszakaszban.) Javítsa ki a 4. sort az alábbi módon:

```
1 \documentclass[a4paper,12pt]{article}
2 \usepackage[utf8]{inputenc}
3 \begin{document}
4 Hello Eszterh\'{a}zy K\'{a}roly University!
5 \end{document}
```

Természetesen így nagyon körülményes magyar szöveget begépelni, ráadásul a forrás olvashatatlan. Ez a megoldás csak akkor indokolt, ha egy ékezetes betű nincs a billentyűzeten, vagy ha egy olyan fájlba ír ékezetes betűket, aminek a felhasználójáról nem lehet tudni, hogy a saját forrását milyen kódolással fogja szerkeszteni.

A L^AT_EX úgy működik, hogy az `inputenc` csomag `utf8` opciója – illetve 2018-tól már e nélkül is – az UTF-8 kódolású ékezetes betűket a fordítás során repülő ékezetekre konvertálja. Azonban alapesetben a repülő ékezetes betűket a L^AT_EX két karakternek kezeli – egyik az alapbetű, a másik a rátett ékezet –, ami néhány problémát fog okozni:


- Ékezetes betűket tartalmazó szótagok után nem tud elválasztani a sor végén.
- Az elkészült pdf-ben nem lehet rákeresni ékezetes betűket tartalmazó szavakra.
- Ha a pdf fájból ékezetes betűket tartalmazó szöveget másol ki, akkor az ékezetes betűk rosszul fognak megjelenni.

Mindezek kiküszöbölésére szükségünk lesz a **fontenc** csomagra **T1** opcióval, amely az úgynevezett T1 belső kódolást tölti be (bővebben lásd a 22.2. szakaszban). Ezt írja a 3. sorba:

```
1 \documentclass[a4paper,12pt]{article}
2 \usepackage[utf8]{inputenc}
3 \usepackage[T1]{fontenc}
4 \begin{document}
5 Hello Eszterházy Károly University!
6 \end{document}
```

Ez még mindig nem elég a helyes elválasztás beállításához, hiszen a \LaTeX nem tudja kitalálni magától, hogy milyen nyelven írjuk a dokumentumot. Jelen esetben angolul, amit a **babel** csomag **english** opciójával kell a forrásban közölni (lásd 4. sorban):

```
\documentclass[a4paper,12pt]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\begin{document}
Hello Eszterházy Károly University!
\end{document}
```

Ezzel tetszőleges angol nyelvű szöveg kiszedhető, melyben az angol elválasztási szabályok és egyéb angol tipográfiai elemek érvényesülnek. Az így elkészült forrást mentheti sablonként is , aminek az az előnye, hogy bármikor visszatölthető, nem kell ezeket a sorokat újból beírni.

Alakítsa át a forrást magyar nyelvre. Az **english** opciót javítsa **magyar** opcióra és írjon be magyarul valamilyen szöveget:

```
\documentclass[a4paper,12pt]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[magyar]{babel}
\begin{document}
Magyar nyelvű szöveg.
\end{document}
```

A **babel** csomag **magyar** opciója betölti a **magyar.ldf** fájlt, amely a magyar tipográfia megvalósításáért felelős. A **magyar.ldf** első verzióját BÍRÓ ÁRPÁD és BÉRCES JÓZSEF készítették. A ma használatos jóval nagyobb tudású verziót SZABÓ PÉTER írta, ami úgy van beállítva kompatibilitási okok miatt, hogy alapesetben a régivel legyen egyenértékű. (Leírást itt talál róla: [klikk ide.](#)) Ahhoz, hogy az új elemek is érvényesülhessenek, a **babel** betöltése előtt át kell állítani a **magyar.ldf** alapbeállításait:

```
\documentclass[a4paper,12pt]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\begin{document}
Magyar nyelvű szöveg.
\end{document}
```

Próbaképpen fordítsa le a forráskódot. Ezzel a kóddal tetszőleges magyar nyelvű szöveg kiszedhető. Ezt ismét elmentheti sablonként.

Ezen a ponton érdemes kipróbálni a hibakezelést. Például a `\begin{document}` parancsot írja át rosszra, mondjuk így: `\Begin{document}`. Ezután fordítsa le a forráskódot. Ekkor a \LaTeX -fordító egy hibaüzenetet küld, miszerint a `\Begin` parancs nincs definiálva:

Undefined control sequence. \Begin

Ezt a hibaüzenetet a TeXstudio is kiírja és a hibás sorra ugrik. Ezután a hibás kódot javítsa vissza jóra. Ismét lefordítva már nem kap hibaüzenetet.

Mielőtt bezárná a TeXstudiot, még egy feladatot el kell végezni. A munka elején megnyitott mappában a tex és pdf fájlokon kívül néhány munkafájl is létrejött. Többek között egy log kiterjesztésű naplófájl is, ami az esetlegesen rosszul begépelte forráskódból származó hibákat is rögzíti. A munka végeztével ezeket érdemes törölni, amit TeXstudioból könnyen megtehet: **Eszközök > Segédfájlok törlése**.



Videó: Az első \LaTeX -dokumentum készítése

A TeXstudio rengeteg kényelmi szolgáltatást biztosít, melyek segítségével sokkal gyorsabban állíthatja elő a \LaTeX -forrást. Ezeket ebben a jegyzetben nem tárgyaljuk, hiszen a TeXstudio újabb verzióinak kiadásával megváltozhatnak. Ezért ezeket a funkciókat célszerű önállóan felfedezni és megtanulni a használatukat.

3. fejezet

A dokumentum nyelve

3.1. A babel csomag

Ahogy korábban már volt róla szó, a dokumentum nyelvét a `babel` csomaggal kell beállítani, amely többek között a következő nyelvek tipográfiáját ismeri: `bulgarian`, `croatian`, `czech`, `danish`, `dutch`, `english`, `esperanto`, `estonian`, `finnish`, `french`, `ngerman`, `greek`, `hebrew`, `magyar`, `icelandic`, `irish`, `italian`, `latin`, `polish`, `portuges`, `romanian`, `russian`, `scottish`, `serbian`, `slovak`, `slovene`, `spanish`, `swedish`, `turkish`, `ukrainian`, `welsh`. Például, ha angolul írunk, akkor a következő kód megfelelő:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\begin{document}
English text
\end{document}
```

Egy dokumentumon belül több nyelven is írhatunk. A következő kódban az alapnyelv a német, amelybe beszúrhatunk angol nyelvű részteket is:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[english,ngerman]{babel}
\begin{document}
Deutsch Text
{\selectlanguage{english} English text}
\end{document}
```

Azt is láttuk, hogy magyar nyelv esetén a `magyar.ldf` alapbeállításait át kell állítani a `babel` csomag betöltése előtt a

```
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
```

paranccsal. Ehelyett használható a

```
\def\magyarOptions{defaults=hu-min}
```

parancs is, de szintén csak a `babel` előtt. A kettő között annyi a különbség, hogy a `\PassOptionsToPackage` parancsot többször kiadva, mindegyik opció érvényesülni fog, míg a második megoldást többször alkalmazva, csak az utolsó érvényesül.

3.2. A szavak elválasztása

A \LaTeX alapról sorkizártan szedi a szöveget, így a sorvégi szavak elválasztása hosszabb szövegek esetén elkerülhetetlen. Amikor beállítottuk a nyelvet, akkor a szavak nagy részét helyesen el fogja tudni választani, de teljesen ezt nem lehet automatizálni. Például a „karóra” szó esetében kétféle szótagolás is lehetséges, aszerint, hogy mit jelent: kar-óra vagy ka-ró-ra.

Fontos, hogy az ebben a szakaszban leírtakat ne az összes forrásban leírt szóra alkalmazza, mert egyrészt felesleges, másrészt a forrást olvashatatlaná tenné. Csak a dokumentum megírásának legvégén nézze meg a sorvégi elválasztásokat, és a helytelen eseteknél lépjen közbe!

Ha azt tapasztalja, hogy egy adott szó rosszul lett elválasztva, akkor alkalmazhatja az `\-` ún. puha elválasztójelet. Például

■ Már nem volt a szarkánál a `kar\-\ó\`-ra, mikor felrepült a `ka\-\ró\`-ra.

Ebben az esetben az adott szót csak a `\-` módon megjelölt helyeken lehet elválasztani. Ha a „karóra” összetett szóként szerepel a szövegben, azaz a szótagolása `kar-ó-ra`, akkor `kar\-\ó\`-ra helyett ez is írható:

■ `kar`_óra`

Ekkor a ``_` jel mutatja, hogy hol van a szóösszetétel határa, így a \LaTeX helyesen tudja elválasztani.

Amennyiben a dokumentumban van egy többször is használatos szó, amit a \LaTeX rosszul választ el, akkor célszerű még a preambulumban beállítani a helyes elválasztást, nem mindig az adott helyen megadni puha elválasztójelekkel. Például, ha szükség van magyar nyelvű környezetben a „significance” angol szó elválasztására, akkor ezt a magyar szabályok szerint `sig-ni-fi-can-ce` módon kellene megtenni, azonban az angol szabályok szerint `sig-nif-i-cance` a helyes. Ilyenkor a preambulumban megadhatjuk ennek a szónak a helyes szótagolását:

■ `\hyphenation{sig-nif-i-cance}`

Ezután már ezt a szót minden esetben helyesen választja el. A `\hyphenation` parancsba több szó is beírható, melyeket szóközzel kell elválasztani. Például

■ `\hyphenation{sig-nif-i-cance tél-a-pó}`

A magyarban még további gond is van. Gondoljon a „mennyi” szó elválasztására: meny-nyi. Másrészt például a „magánnyomozó” szó elválasztása: ma-gán-nyo-mo-zó. Vagyis az `nny` jelenthet kettőzött többjegyű betűt és `n+ny` kapcsolatot is. A \LaTeX alapesetben a szavakat nem tudja elválasztani kettőzött többjegyű betűnél. Ha a bekezdés törése viszont optimálisabb lenne így elválasztva a szót, akkor írjon ezen kettőzött többjegyű betű elé egy fordított aposztrófjelet a **AltGr** + **7** gombokkal:

■ `me`nnyi`

Ekkor, ha a magyar nyelv aktív, a \LaTeX tudni fogja, hogy a ``` jelet követő `nny` elválasztható `ny-ny` módon, ha arra szükség van.

Ha egy szóban kettőzött többjegyű betű van, akkor a `\hyphenation` parancsban nem adható meg annak az elválasztása, azaz például helytelen a következő kód:

■ `\hyphenation{i-dő-hosz-szab-bí-tás} % HELYTELEN!`

Ugyanis ebben az esetben a helytelenül írt „időhoszszabbítás” szó elválasztását adtuk meg. A helyes megoldás a következő:

```
\hyphenation{i-dő-hosszab-bí-tás} % HELYES!
```

Ezután pedig időho`sszabbítás módon beírva mindenképpen jó elválasztást kapunk.

Ha egy szóban kötőjel van, akkor azt a L^AT_EX csak a kötőjelnél tudja elválasztani. Ha ezt felül akarja bíráltni, és a magyar nyelv aktív, akkor a kötőjel elé rakjon fordított aposztrófjelet:

```
egyszer`-kétszer
```

Ekkor a kötőjelnél és minden szótagnál el tud választani. Ha kötőjelnél választ el, akkor a kötőjelet a következő sor elején nem ismétli meg. Ha mégis szükség van erre, mert ki akarjuk hangsúlyozni a kötőjel szerepét, akkor használjuk a `| kódot, ami az ún. fontos kötőjelet jelenti. De ez csak akkor fog működni, ha a magyar nyelv aktív. Például

```
nátrium`|klorid
```

Néha szükség lehet egy adott szó elválasztásának a tiltására is. Ekkor az adott szót tegye az `\mbox` parancs argumentumába. Például

```
\mbox{Fazekas} Mihály
```

Ha a teljes dokumentumban tiltani akarja az elválasztást, akkor a preambulumba írja a következőket:

```
\hyphenpenalty10000
\tolerance10000
```

Ha egy mód van rá, ezt a megoldást kerülje, hiszen így a sorkizárás miatt nem lehet optimálisan tördelni a bekezdéseket!

3.3. Sorvégi túlcsondolás

Ha a L^AT_EX nem tudja megoldani sorvégén egy szó elválasztását, akkor ún. túlcsondolás jöhet létre. Ezeket a helyeket célszerű bejelölni a végeredményben. Ehhez gépelje a következőt a preambulumba:

```
\setlength{\overfullrule}{5pt}
```

Az így észlelt hibákat azután javíthatja a forrásban.

Túlcsondolás akkor is létrejöhet, ha a sorvégi szót el tudja választani a L^AT_EX, de egyetlen megoldás esetén sem lesznek a sorban a szóközök optimálisak. Ilyen esetben a legjobb megoldás a szöveg átfogalmazása. Kényelmesebbnek tűnő lehetőség a `\sloppy` parancs használata, ami után a túlcsondolások úgy szűnnek meg, hogy az adott sorban a szóközök túl nagyok lesznek. Ez tipográfia hiba, ezért ezt a megoldást csak legvégső esetben alkalmazza, vagy inkább még akkor sem. A `\sloppy` hatása a `\fussy` paranccsal szüntethető meg.

3.4. A magyar.lda aktív karakterei

Láttuk, hogy amennyiben a magyar nyelv aktív, akkor a fordított aposztrófjelnek parancs szerepe van bizonyos esetekben. Ezekon kívül még akkor is aktívvá válik, amikor angol nyitó idézőjelet akarunk írni. Később látni fogjuk, hogy angol nyelv esetében `` módon kell nyitó idézőjelet írni. Viszont ha a magyar nyelv aktív, akkor a `magyar.lda` ezt átalakítja magyar nyitó idézőjellé:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[english,magyar]{babel}
\begin{document}
``idézet'' {\selectlanguage{english} ``idézet''}
\end{document}
```

„idézet” “idézet”

Magyar nyelv esetén is írhat közvetlenül (azaz az angol nyelv aktívva tétele nélkül) angol nyitó idézőjelet `` az **AltGr** + **7** majd **Shift** + **1** gombokkal:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\begin{document}
`'idézet''
\end{document}
```

“idézet”

Egy adott helyen ki is lehet kapcsolni a fordított aposztrófjel aktív szerepét úgy, hogy elé kell tenni a `\string` parancsot. Ezzel a megoldással is lehet magyar nyelv esetén közvetlenül angol nyitó idézőjelet írni:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\begin{document}
\string`\string`idézet''
\end{document}
```

“idézet”

Bizonyos csomagokkal nem kompatibilis a fordított aposztrófjel aktív szerepe (pl. a kot-taírásra alkalmas `musixtex` esetén). Ilyenkor a `magyar.ldf` `active=onlycs` opciójával ezt ki lehet kapcsolni:

```
\PassOptionsToPackage{defaults=hu-min,active=onlycs}{magyar.ldf}
```

Ekkor a fordított aposztrófjel már nem működik a korábban leírtak szerint, helyette használja a `\shu`` parancsot. Például

```
időho\shu`sszabbítás
```

A magyar tipográfia megköveteli, hogy kettőspont, pontosvessző, kérdőjel és felkiáltójel előtt legyen egy rövid szóköz. Ezt a `magyar.ldf` ezen jelek aktívva tételével valósítja meg. Sajnos ez néha kompatibilitási problémákhoz vezethet. Ekkor használja a `magyar.ldf` `activespace=none` opcióját:

```
\PassOptionsToPackage{defaults=hu-min,activespace=none}{magyar.ldf}
```

Ebben az esetben az előző tipográfiai követelmény nem teljesül, ezért ezt a megoldást csak végső esetben alkalmazza.

4. fejezet

Alapvető formai elemek

4.1. Karakterek

4.1.1. Foglalt karakterek

Vannak olyan billentyűzetről beírható karakterek, melyek közvetlenül nem jeleníthetők meg, mert a forrásállományban speciális jelentésük van:

<code>\</code>	(backslash) parancsok ezzel kezdődnek
<code>%</code>	kommentek ezzel kezdődnek
<code>{ }</code>	blokkok, illetve parancsok argumentumainak határai
<code>\$</code>	matematikai mód határolójele
<code>&</code>	táblázatoknál kell
<code>#</code>	(hash mark) változót tartalmazó parancs definiálásához kell
<code>_</code>	alsó index
<code>^</code>	felső index
<code>~</code>	törhetetlen szóköz

Ha ezeket meg akarja a pdf-ben jeleníteni, akkor a következő parancsokat használhatja:

<code>\</code>	<code>\textbackslash</code> vagy <code>\char\string`\`</code> (<code>`</code> a fordított aposztrófjel AltGr + 7)
<code>%</code>	<code>\%</code>
<code>{ }</code>	<code>\{ }</code> <code>\}</code>
<code>\$</code>	<code>\\$</code>
<code>&</code>	<code>\&</code>
<code>#</code>	<code>\#</code>
<code>_</code>	<code>_</code>
<code>^</code>	<code>\textasciicircum</code> vagy <code>\char\string`\^</code>
<code>~</code>	<code>\textasciitilde</code> vagy <code>\char\string`\~</code>

4.1.2. Ékezetes betűk

Korábban már volt róla szó, hogy a forrásállomány kódolásának beállítása után az ékezetes betűk közvetlenül a billentyűzetről is bevihetők. De mi van akkor, ha olyan stílusfájlt írunk, melyben a kódlap kiválasztását a felhasználóra bizzuk, vagy olyan ékezetes betűre van szükségünk, amely nincs a billentyűzeten? Ilyenkor használhatja a repülő ékezeteket.

ó <code>\'{o}</code>	ò <code>\`{o}</code>	ō <code>\={o}</code>	ö <code>\v{o}</code>	ø <code>\k{o}</code>	ôo <code>\t{oo}</code>
õ <code>\H{o}</code>	ô <code>\^{o}</code>	ô <code>\.{o}</code>	ö <code>\r{o}</code>	ø <code>\d{o}</code>	
ö <code>\" {o}</code>	õ <code>\~{o}</code>	ö <code>\u{o}</code>	ø <code>\c{o}</code>	ø <code>\b{o}</code>	

Természetesen az o betű bármire kicserélhető, kivéve a két ékezetes angol betűt: i és j. Ezekre nem szabad másik ékezetet rakni, mert pl. `\H{i}` eredménye í. Ezért az i és j betűknek van ékezet nélküli verziója is, amiket `\i` és `\j` parancsokkal érhetünk el: i, j. Ezzel már le tudja írni az í betűt `\H{\i}` módon. Ez alól a két gyakrabban előforduló í és ï betűk kivételt képeznek, ezek így is írhatók: `\'{i}`, `\" {i}`.

A `\k` parancs csak T1 belső kódkészlet esetén érhető el. Ezen belső kódkészlet esetén a `\v` parancs az L, l, d, t betűk esetében más ékezetet jelent: Ľ ľ đ ť. A TeXstudio-ban minden repülő ékezet elérhető innen: **Side Panel > Symbols > Special**.

4.1.3. Speciális betűk

Œ <code>\OE</code>	Æ <code>\AE</code>	ß <code>\ss</code>	ø <code>\o</code>	ſ <code>\j</code>	ı <code>\l</code>
œ <code>\oe</code>	æ <code>\ae</code>	Ø <code>\O</code>	ı <code>\i</code>	Ł <code>\L</code>	

További, csak T1 belső kódkészlet esetén használható speciális betűk:

Đ <code>\DH</code>	Đ <code>\DJ</code>	Đ <code>\NG</code>	Þ <code>\TH</code>
đ <code>\dh</code>	đ <code>\dj</code>	đ <code>\ng</code>	þ <code>\th</code>

4.1.4. Ligatúrák

Ligatúrán a betűknek a szokásosnál szorosabb összekötését értik. A legismertebbek a következő ún. f-ligatúrák:

■ ff fi fl ffi ffl

ff fi fl ffi ffl

A L^AT_EX alapból kezeli a ligatúrákat, létrejöttükért külön nem kell parancsot kiadni. Ha le akar tiltani egy helyen egy ligatúrát, akkor a betűk közé tegyen `{}` jelet:

■ f{}f f{}i f{}l f{}f{}i f{}f{}l

ff fi fl ffi ffl

4.1.5. Különleges karakterek

Itt felsorolunk néhány érdekes karaktert. Bővebben erről a `symbols-a4.pdf`-ben olvashat, amit a

`texdoc symbols-a4`

parancssorba írásával találhat meg, vagy TeXstudio-ban **Súgó > Súgó csomagok**, és itt be kell írni a `symbols-a4` szót.

€	<code>\euro</code> ∈ <code>eurosym</code>	◦	<code>\textopenbullet</code> ∈ <code>textcomp</code>
£	<code>\pounds</code>	⌵	<code>\textvisiblespace</code>
¢	<code>\textcent</code> ∈ <code>textcomp</code>	¡	<code>!`</code>
Ⓟ	<code>\textcircledP</code>	¿	<code>?`</code>
®	<code>\textregistered</code> ∈ <code>textcomp</code>	‰	<code>\textperthousand</code> ∈ <code>textcomp</code>
©	<code>\textcopyright</code> ∈ <code>textcomp</code>	§	<code>\S</code>
Ⓒ	<code>\textcopyrightleft</code> ∈ <code>textcomp</code>	¶	<code>\P</code>
†	<code>\dag</code>	№	<code>\textnumero</code> ∈ <code>textcomp</code>
‡	<code>\ddag</code>	※	<code>\textreferencemark</code> ∈ <code>textcomp</code>
*	<code>\textasteriskcentered</code>	5˙	<code>5\.</code> {}
•	<code>\textbullet</code>		

A `textcomp` mellett a `wasysym` csomag is sok érdekes szimbólumot tartalmaz. Ezek elérhetők a TeXstudióból is: **Side Panel** > **Symbols** > **Misc. Text** és **Side Panel** > **Symbols** > **wasysym**.

Úgynevezett PostScript jeleket is kiíráthat a

`\ding{<kódszám>}` ∈ `pifont`

parancssal. A `<kódszám>` helyére 33-tól 254-ig írhatunk számokat, melyeknek a hatása a következő táblázatban látható:

✂	33	✚	60	✱	87	◻	114	④	175	❶	202
✂	34	†	61	✱	88	▲	115	⑤	176	❷	203
✂	35	‡	62	✱	89	▼	116	⑥	177	❸	204
✂	36	‡	63	✱	90	◆	117	⑦	178	❹	205
☞	37	⌘	64	✱	91	❖	118	⑧	179	❺	206
🕒	38	☆	65	✱	92	◐	119	⑨	180	❻	207
🕒	39	✚	66	✱	93		120	⑩	181	❼	208
✈	40	✚	67	✱	94	l	121	❶	182	❽	209
✉	41	♣	68	✱	95	■	122	❷	183	❾	210
☞	42	✚	69	✱	96	‘	123	❸	184	❿	211
☞	43	◆	70	✱	97	,	124	❹	185	➔	212
✌	44	◇	71	✱	98	“	125	❺	186	➔	213
☞	45	★	72	✱	99	”	126	❻	187	↔	214
☞	46	☆	73	✱	100	♯	161	❼	188	↕	215
☞	47	⊛	74	✱	101	?	162	❽	189	↘	216
☞	48	☆	75	✱	102	?	163	❾	190	➔	217
👁	49	☆	76	✱	103	♥	164	❿	191	↘	218
🔑	50	★	77	✱	104	♣	165	❶	192	➔	219
✓	51	☆	78	✱	105	♣	166	❷	193	➔	220
✓	52	☆	79	✱	106	♣	167	❸	194	➔	221
✕	53	☆	80	✱	107	♣	168	❹	195	➔	222
✕	54	✱	81	●	108	◆	169	❺	196	➔	223
✕	55	✱	82	○	109	♥	170	❻	197	➔	224
✕	56	✱	83	■	110	♠	171	❼	198	➔	225
⊕	57	✱	84	◻	111	❶	172	❽	199	➔	226
⊕	58	✱	85	◻	112	❷	173	❾	200	➔	227
⊕	59	✱	86	◻	113	❸	174	❿	201	➔	228

➡ 229	↩ 234	⇨ 239	➡ 245	➡ 250	⇒ 254
➡ 230	↩ 235	⇨ 241	➡ 246	➡ 251	
➡ 231	↩ 236	⇨ 242	➡ 247	➡ 252	
➡ 232	↩ 237	➡ 243	➡ 248	➡ 253	
⇨ 233	⇨ 238	➡ 244	➡ 249		

4.2. Szóközök

Forrásállományban egy szóközt a **Space** billentyű lenyomásával tehet. Több szóköz egymás után a forrásállományban, csak egy szóközt jelent a végeredményben, viszont a sor elején található szóköz a végeredményben nem jelenik meg. Szintén szóköznek számít a végeredményben, ha a forrásállományban sortörés van. Ez csak akkor nem igaz, ha a sor végén egy % jel van úgy, hogy közvetlenül előtte nincs szóköz. Például

```
Egy, kettő,      három,
né%
gy, öt, %
hat.
```

Egy, kettő, három, négy, öt, hat.

Ha egy parancsnak nincs argumentuma, akkor általában az utána található szóközt nem jeleníti meg. Például

```
\LaTeX kézikönyv
```

\LaTeX kézikönyv

Ha ez nem kívánatos eredményt ad, akkor vagy lezárjuk kapcsos zárójelekkel a parancs hatását, vagy `_` paranccsal kikényszerítjük a szóközt (a `_` jel a szóközt jelenti):

```
\LaTeX{} kézikönyv, {\LaTeX} kézikönyv, \LaTeX\ kézikönyv.
```

\LaTeX kézikönyv, \LaTeX kézikönyv, \LaTeX kézikönyv.

Van olyan eset is, amikor egy szóköz után nem szabad sort törni. Például ha azt írjuk, hogy IV. Béla, akkor a pont után nem lehet sortörés. Ennek érdekében a pont után ún. törhetetlen szóközt kell rakni. Forrásban `~` a törhetetlen szóköz jele:

```
IV.~Béla
```

Ezt érdemes megtenni minden olyan pont után, amikor az nem mondat végét jelzi. Így az ilyen pontok nem kerülhetnek a sor végére. Vigyázat, ha már valahová tett törhetetlen szóközt, akkor utána ne tegyen még egy szóközt, mert az két szóközt eredményez, és a törhetetlenség is megszűnik:

```
IV.~ Béla (Így helytelen!)
```

IV. Béla (Így helytelen!)

A törhetetlen szóköznek van egy olyan változata is, ami a normál szóköz méretének a fele. Ezt mértékszám és mértékegység között, illetve számok ezres csoportosításánál szoktuk használni. Forrásban `\,`, a törhetetlen feles szóköz jele:

```
5\,cm, 14\,216\,123
```

5 cm, 14 216 123

4.3. Központozás

. , :: ? !

Ezek elé ne, de utána tegyen szóközt! Kivétel, ha utána záró idézőjel vagy) jel van.

Angol nyelvű szövegben a mondat végi pont után nagyobb térköz kell, mint két szó között. Ezt a \LaTeX megoldja, ha a `babel` csomag `english` opciója van bekapcsolva. Viszont, ha egy mondat nagybetűre végződik, akkor azt rövidítésnek tekinti, így az azt követő pont után nem hagy ki nagyobb térközt. Ennek az a megoldása, hogy az ilyen pont elé tegye a `\@` parancsot. Például

```
Catch your HBO favorites whenever you want, wherever you are --
it's every episode of every season of the best of HBO\@.
More channels to watch. All available in HD\@.
```

Catch your HBO favorites whenever you want, wherever you are – it's every episode of every season of the best of HBO. More channels to watch. All available in HD.

Kötőjel

A kötőjel forrásállományban - jellel adható meg. Például

```
levegő-mintavétel; elő- vagy utótag; betűtípus és -méret;
egy-két ember; 5-6 éves lehet; tudod-e;
```

levegő-mintavétel; elő- vagy utótag; betűtípus és -méret; egy-két ember; 5-6 éves lehet; tudod-e;

Nagykötőjel

A nagykötőjel forrásállományban -- jellel adható meg. Például

```
lásd 15--21.~oldalakon; kelet--nyugati; az orosz TU--154 repülő;
brazil--magyar meccs;
```

lásd 15-21. oldalakon; kelet-nyugati; az orosz TU-154 repülő; brazil-magyar meccs;

A szerzőpárok neveit is nagykötőjellel kötjük össze, de ebben az esetben a nagykötőjel elé és után is törhetetlen feles szóközt kell rakni. Például

```
Bolzano\,--\,Weierstrass-tétel
```

A magyar nyelv használata esetén a `\,--\,` helyett használható a ``--` kód is, azaz az előző kód így is írható:

```
Bolzano`--Weierstrass-tétel
```

Bolzano – Weierstrass-tétel

Gondolatjel

A gondolatjel forrásállományban `--` jellel adható meg. Gondolatjel előtt és után is szóköz áll, kivéve, ha írásjel követi. Például

Ilyen korán `--` legalábbis hétvégén `--` nem szokott felkelni.

Ilyen korán – legalábbis hétvégén – nem szokott felkelni.

Sokszor vitatkoztak `--` legtöbbször semmiségekért `--`, de szerették egymást.

Sokszor vitatkoztak – legtöbbször semmiségekért –, de szerették egymást.

Kvirtmínusz

Gondolatjelként az angolban a kvirtmínusz (`---`) jel is használható, de ez előtt és után nem szabad szóközt rakni. A magyarban ez a megoldás tilos. A kvirtmínusz forrásban `---` módon írandó.

Zárójelek

Itt pontosan az a szabály, mint a gondolatjelnél.

Hármaspont

A hármaspont forrásállományban a `\dots` paranccsal adható meg. Ehelyett soha ne használjon három darab pontot egymás után írva. Például

A `\dots\` jó, de a `...` nem. (`\dots` várom a párom `\dots\` üres a polc`\dots`)

A ... jó, de a ... nem. (... várom a párom ... üres a polc...)

Idézőjel

Idézőjelként soha ne használja a forrásban a " `Shift` + `2` jelet! Ez tipográfiai hiba. Az idézőjel és a belső idézőjel nyelvenként változó. Belső idézőjel akkor kell, ha idézet van az idézetben belül. Magyar szöveg esetén a következőt kell tenni:

`„szöveg >>szöveg<< szöveg' ' vagy`
`\textqq{szöveg \textqq{szöveg} szöveg} ∈ [magyar]babel`

„szöveg »szöveg« szöveg” vagy „szöveg »szöveg« szöveg”

Tehát a nyitó külső idézőjel a forrásban két vessző, míg a záró külső idézőjel a forrásban két aposztrófjel `Shift` + `1`. A nyitó belső idézőjel `>>` és a záró belső idézőjel `<<` a forrásban. A `\textqq{szöveg}` parancs esetén arra kell ügyelni, hogy a `szöveg` nem állhat több bekezdésből.

Angol szövegben a brit szabályok szerint ezt kell tenni:

``text ``text' ' text'`

‘text “text” text’

Tehát a nyitó külső idézőjel a forrásban egy fordított aposztrófjel **AltGr** + **7**, míg a záró külső idézőjel a forrásban egy aposztrófjel **Shift** + **1**. A nyitó belső idézőjel a forrásban két fordított aposztrófjel, míg a záró belső idézőjel a forrásban két aposztrófjel.

Az amerikai szabályok szerint fordítva van a sorrend. Azaz a nyitó külső idézőjel a forrásban két fordított aposztrófjel, míg a záró külső idézőjel a forrásban két aposztrófjel. A nyitó belső idézőjel a forrásban egy fordított aposztrófjel, míg a záró belső idézőjel a forrásban egy aposztrófjel:

■ ``text `text' text'`

“text ‘text’ text”

Az előzőeken kívül létezik egy univerzális megoldás is. Töltse be a `csquotes` csomagot `autostyle` opcióval. Ez a csomag a következő nyelvek idézőjeleit ismeri: `croatian`, `danish`, `dutch`, `english`, `finnish`, `french`, `german`, `greek`, `italian`, `norwegian`, `portuguese`, `russian`, `spanish`, `swedish`. A magyart nem ismeri, így ebben az esetben a `csquotes` csomag betöltése után a preambulumba gépelje a következőt:

■ `\DeclareQuoteStyle{magyar}{,}{'}{>}{<} \in [autostyle]csquotes`

Ezután az

■ `\enquote{<szöveg> \enquote{<szöveg>} <szöveg>} \in [autostyle]csquotes`

kód az érvényben lévő nyelvnek megfelelően használja az idézőjelet (külsőt és a belsőt is). Ha közvetlenül belső idézőjelet akar megjeleníteni, akkor használja az `\enquote*` parancsot. Az `\enquote` és `\enquote*` parancsok argumentumában használható több bekezdésből álló szöveg is.

4.4. Betűváltozatok

4.4.1. Osztályozás

A betűváltozatokat családjuk, testességük és alakjuk szerint osztályozhatjuk.

Család (family)

Antikva (roman): `\textrm{<szöveg>}, {\rmfamily <szöveg>}`
 Góteszk (sans serif): `\textsf{<szöveg>}, {\sffamily <szöveg>}`
 Írógép (typewriter): `\texttt{<szöveg>}, {\ttfamily <szöveg>}`

A családok jellemzői:

	talpas	vonaltastagság	betűszélesség
antikva	igen	változó	változó
góteszk	nem	állandó	változó
írógép	igen	állandó	állandó

Testesség (series)

Normál (medium): `\textmd{<szöveg>}, {\mdseries <szöveg>}`
 Félkövé (boldface): `\textbf{<szöveg>}, {\bfseries <szöveg>}`

Alak (shape)

Álló (upright): `\textup{<szöveg>}, {\upshape <szöveg>}`
 Döntött (slanted): `\textsl{<szöveg>}, {\slshape <szöveg>}`
 Dőlt (italics): `\textit{<szöveg>}, {\itshape <szöveg>}`
 KISKAPITÁLIS (SMALL CAPS): `\textsc{<szöveg>}, {\scshape <szöveg>}`

A család, testesség és alak keverhetőek. Például

```
\textit{\textbf{\textsf{szöveg}}}
```

szöveg

A `\text..` parancsokat több bekezdésre nem lehet alkalmazni (ahol `..` = `up`, `sl`, `it`, `sc`, `md`, `bf`, `rm`, `sf`, `tt`).

Alapesetben a betű álló, normál és antikva. Amikor nem alap betűváltozatot használ, de ideiglenesen vissza akar arra térni, akkor használja a

```
\textnormal{<szöveg>}, {\normalfont <szöveg>}
```

parancsokat. (Az első több bekezdésre nem használható.)

Az `\upshape`, `\slshape`, `\itshape` stb. ún. deklarációs parancsok. Ezek használhatók környezetként is. Például

```
{\bfseries <szöveg>} = \begin{bfseries}<szöveg>\end{bfseries}
```

A fonttípusok beállításáról a 22. fejezetben olvashat részletesebben.

4.4.2. Kurzív kiegyenlítés

Ha egy dőlt vagy döntött betűs szöveget egy álló betűs szöveg követ, akkor közéjük kicsivel nagyobb szóközt kell tenni, különben a ferdén álló betű nagyon rádőlne az állóra. Ezt nevezik *kurzív kiegyenlítésnek*. Ennek illusztrálására a következő mondatot először kurzív kiegyenlítés nélkül, majd pedig kurzív kiegyenlítéssel szedtük ki:

„Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.”

„Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.”

A `\textit` és `\textsl` parancsok a kurzív kiegyenlítést automatikusan elvégzik, így a következő két megoldás helyes eredményt ad:

```
Éhes \textit{zsiráf} fogyasztja épp ízes uzsonnáját.\\
Éhes \textsl{zsiráf} fogyasztja épp ízes uzsonnáját.
```

Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.

Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.

Azonban ezek deklarációs párjai, az `\itshape` és az `\slshape` parancsok, illetve ezek környezetes verziói nem kezelik ezt a problémát. Így ezt a felhasználónak kell megoldani a `\` parancssal:

```
Éhes {\itshape zsiráf\} fogyasztja épp ízes uzsonnáját.\\
Éhes {\slshape zsiráf\} fogyasztja épp ízes uzsonnáját.
```

Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.

Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.

4.4.3. Kiemelés

Amikor egy szót, vagy gondolatot ki akar emelni, használja az

`\emph{<szöveg>}`, `{\em <szöveg>}`, `\begin{em}<szöveg>\end{em}`

parancsokat illetve környezetet. (Az első megoldás több bekezdésre nem használható.) Standard dokumentumosztályok esetén ezek figyelik az aktuális betűválozatot, és aszerint emelnek ki. Álló alak esetén dőlt, nem álló alak esetén álló alakra vált. Az `\emph` a kurzív kiegyenlítést automatikusan elvégzi, de az `\em` parancs, illetve az `em` környezet nem. Ekkor ezt a `\` parancssal nekünk kell megoldani. Például:

```
Éhes \emph{zsiráf} fogyasztja épp ízes uzsonnáját.\\
Éhes {\em zsiráf\} fogyasztja épp ízes uzsonnáját.
```

Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.
 Éhes *zsiráf* fogyasztja épp ízes uzsonnáját.

Kiemelésre lehetőleg ne használja a félkövér típust, mert az a címekre van fenntartva. Az írógépek korában betűritkítással emeltek ki. Ez \LaTeX -ben is megoldható:

`\so{<szöveg>} \in soulutf8`

Például

```
\so{Ritkított szöveg, ami állhat akár több bekezdésből is.}
```

Ritkított szöveg, ami állhat több sorból, vagy akár több bekezdésből is.

További kiemelési lehetőségek alá- illetve áthúzással:

szöveg	<code>\underline{szöveg}</code>	szöveg	<code>\xout{szöveg} \in ulem</code>
szöveg	<code>\uline{szöveg} \in ulem</code>	szöveg	<code>\cancel{szöveg} \in cancel</code>
szöveg	<code>\uuline{szöveg} \in ulem</code>	szöveg	<code>\bcancel{szöveg} \in cancel</code>
szöveg	<code>\uwave{szöveg} \in ulem</code>	szöveg	<code>\xcancel{szöveg} \in cancel</code>
szöveg	<code>\sout{szöveg} \in ulem</code>		

Az `ulem` csomag használata esetén az `\emph` parancs aláhúzással fog kiemelni. Ha ezt nem akarja, akkor használja az `ulem` csomag `normalem` opcióját.

Szavak, kifejezések kiemelésére alkalmas lehet csupa nagybetűvel, vagy nagybetűs szövegben csupa kisbetűvel szedésük.

`\MakeUppercase{<szöveg>}` a szöveget csupa nagybetűvel szedi ki.

`\MakeLowercase{<szöveg>}` a szöveget csupa kisbetűvel szedi ki.

`\MakeTextUppercase{<szöveg>} \in textcase` a szöveget csupa nagybetűvel szedi ki, de a matematikai képletek betűin nem változtat.

`\MakeTextLowercase{<szöveg>} \in textcase` a szöveget csupa kisbetűvel szedi ki, de a matematikai képletek betűin nem változtat.

`\NoCaseChange{<szöveg>} \in textcase` nem változtat a betűkön.

Ha színes háttérrel vagy színes aláhúzással akar kiemelni, akkor olvassa el a 4.12.4. és a 4.12.5. alszakaszokat.

4.5. Betűméretek

Ha a T1 belső kódkészletet használja, akkor alapból a *European Computer Modern* fontkészlet töltődik be, amelyben a betűméret csak a következő értékeket veheti fel pt-ben mérve: 5, 6, 7, 8, 9, 10, 10.95, 12, 14.4, 17.28, 20.74, 24.88, 29.86, 35.83. Ha más értéket szeretne, akkor a lehetséges méretek közül a hozzá legközelebbi töltődik be. Ez a korlátozás feloldható az `anyfontsize` csomaggal.

A European Computer Modern fontkészlet helyett használhat mást is. Ehhez töltsse be például az `lmodern`, `pxfonts`, `txfonts`, `newtxtext`, `times`, `lfonts`, `bera`, `cyklop` csomagok valamelyikét. Az ezekben található fontok minden méretben használhatók. Új fontok betöltéséről bővebben a 22. fejezetben olvashat.

4.5.1. Alapbetűméret

Az alapbetűméret a dokumentumosztály opcióinál állítható be. A standard `article`, `report` és `book` osztályok esetén három méret adható meg: 10pt, 11pt és 12pt. Ha ettől különböző méretet szeretne, például 15pt, akkor a dokumentumosztály betöltése után írja a preambulumba a következőt:

```
\usepackage[fontsize=15pt]{scrextend}
```

A dokumentum tetszőleges pontján is át lehet állítani az alapbetűméretet:

```
\KOMAOPTIONS{fontsize=<betűméret>} ∈ scrextend
```

Egy másik lehetőség tetszőleges alapbetűméret beállítására az 5.2. szakaszban tárgyalt `geometry` csomag `mag` opciója.

4.5.2. Relatív betűméretek

Az alapbetűmérethez viszonyított relatív betűméretek (az arány 1: 1,2):

szöveg	<code>\tiny szöveg</code>	szöveg	<code>\Large szöveg</code>
szöveg	<code>\scriptsize szöveg</code>	szöveg	<code>\LARGE szöveg</code>
szöveg	<code>\footnotesize szöveg</code>	szöveg	<code>\huge szöveg</code>
szöveg	<code>\small szöveg</code>	szöveg	<code>\Huge szöveg</code>
szöveg	<code>\normalsize szöveg</code>		
szöveg	<code>\large szöveg</code>		

Ezek deklarációs parancsok, így használhatók környezetként is. Például

```
\large <szöveg> = \begin{large}<szöveg>\end{large}
```

Tetszőleges relatív betűméret is beállítható:

```
\scalefont{<arányszám>} ∈ scalefnt
```

ahol az `<arányszám>` azt adja meg, hogy az alapbetűméretnek hány-szorosát szeretné. Például

```
\scalefont{2.5}szöveg
```

esetén a szöveg az alapbetűméret 2,5-szeresével jelenik meg.

4.5.3. Abszolút betűméretek

Abszolút betűméretet a következő paranccsal érhet el:

```
\fontsize{<betűméret>}{<sortávolság>}\selectfont
```

Például 25 pontos szöveget 12 pontos sortávolsággal így lehet írni:

```
\fontsize{25}{30}\selectfont
Ez egy hosszú mondat, hogy ne férjen ki egy sorban!
```

Ez egy hosszú mondat, hogy ne férjen ki egy sorban!

Ha a sortávolságot meg akarja hagyni alaphelyzeten, akkor *<sortávolság>* helyére

```
\the\baselineskip
```

parancsot írja.

4.6. Térközök

A \LaTeX minden nyomdászati használatos mértékegységet ismer. Most csak néhányat sorolunk fel:

pt pont
mm milliméter
cm centiméter
in inch, $1\text{ in} = 25,4\text{ mm} = 72,27\text{ pt}$
ex aktuális betűalakzatban az x betű magassága
em aktuális betűalakzat mérete

4.6.1. Fix méretű vízszintes térközök

Vízszintes helykihagyás méretét a következő paranccsal adhatja meg:

```
\hspace{<térköz mérete>}
```

A *<térköz mérete>* lehet negatív is. Például

```
AAA\hspace{1cm}BBB 0000\hspace{-5mm}oooo
```

AAA BBB OOOOoo

Ez a parancs egy sor elejére vagy végére kerülve nem fejt ki a hatását. A

```
\hspace*{<térköz mérete>}
```

parancs ugyanazt tudja, mint a `\hspace`, de a sor elején és végén is kifejtődik. Ha azt akarja, hogy az adott helykihagyásnál ne lehessen sort törni (törhetetlen köz), akkor használja a következő parancsot:

```
\kern<térköz mérete>
```

További vízszintes méretű helykihagyások:

```
\_                    = \hspace{0.33333em}
```



```

\enskip      = \hspace{0.5em}
\quad       = \hspace{1em}
\qquad      = \hspace{2em}
\negthinspace = \kern-0.16667em
\,,         = \kern0.16667em
~           = \kern0.33333em
\enspace    = \kern0.5em

```

4.6.2. Rugalmas méretű vízszintes térközök

Rugalmas térköz lehet például egy „rugó”, melynek erejét a következő paranccsal adhatja meg.

```
\stretch{<rugó erő>}
```

Ennek működése a következő példán érthetővé válik:

```
A\hspace{\stretch{1}}B\hspace{\stretch{2}}C
```



Ekkor az A és B betűk közötti távolság aránya a B és C betűk közötti távolsághoz 1 : 2. A *<rugó erő>* lehet törtszám is. További parancsok:

```

\fill = \stretch{1}
\hfill = \hspace{\fill}

```

A következő négy parancs hatása megegyezik a `\hfill` hatásával, de a térközt kitölti az alábbi módokon:

```

A\hrulefill B
C\dotfill D
E\rightrightarrowfill F
G\leftarrowfill H

```



Rugalmas térköz a következő módon is megadható:

```
A\hspace{12pt plus 4pt minus 2pt}B
```



Ekkor az A és B betűk távolsága 12 pt, ha az adott sor tördelése megengedi, de ha az optimális tördelés azt megkívánja, ez a méret változhat $12 - 2 = 10$ -tól $12 + 4 = 16$ pontig.

4.6.3. Fix méretű függőleges térközök

Függőleges helykihagyás méretét a következő paranccsal adhatja meg:

```
\vspace{<térköz mérete>}
```

Ekkor a függőleges helykihagyás mérete a *<térköz mérete>* + az aktuális sortávolság. A *<térköz mérete>* lehet negatív is. Ez a parancs csak akkor működik, ha a \TeX függőleges módban van. Ez elérhető pl., ha a szöveg és a `\vspace` között legalább egy üres sor van. A térköz az oldal tetején és alján elnyelődik. A

■ `\vspace*{<térköz mérete>}`

parancs ugyanazt tudja, mint a `\vspace`, de az oldal tetején és alján is kifejtődik. További parancsok:

■ `xxx\lower0.5ex\hbox{xxx}`
 ■ `17h`

xxx_{xxx} 17^h

4.6.4. Rugalmas méretű függőleges térközök

A rugalmas méret pontosan úgy adható meg itt is, mint vízszintes esetben, csak `\vspace` parancsban. Például

AAA
`\vspace{\stretch{1}}` BBB
`\vspace{\stretch{2}}` CCC
`\vspace{12pt plus 4pt minus 2pt}` DDD

További parancsok:

`\smallskip` = `\vspace{3pt plus 1pt minus 1pt}`
`\medskip` = `\vspace{6pt plus 2pt minus 2pt}`
`\bigskip` = `\vspace{12pt plus 4pt minus 4pt}`
`\vfill` = `\vspace{\fill}`

4.6.5. Sortávolság

A sortávolság automatikusan lesz beállítva, de ha ezen változtatni akar, akkor használja a

■ `\linespread{<szorzó>}`

parancsot, ami az alapértelmezett sortávolságot megszorozza a `<szorzó>` értékével. Az írógépeknél használt másfeles illetve kettes sorközhöz tartozó szorzó függ az alap betűmérettől:

	10 pt	11 pt	12 pt
másfeles	1.25	1.21	1.24
kettes	1.67	1.62	1.66

Ha betölti a `setspace` csomagot, akkor másfeles sorköz a `\onehalfspacing` paranccsal, illetve kettes sorköz a `\doublespacing` paranccsal állítható be. De pl. hármas sorköz is megadható a `\setstretch{3}` paranccsal. A `setspace` és `hyperref` csomagok együttes használatánál a `setspace` előbb legyen betöltve.

4.7. Törések

4.7.1. Sortörések

A \LaTeX automatikusan végzi a sortöréseket, de adott esetben ki is kényszerítheti azt:

`\` új sort kezd sorkizárás nélkül.
`\[2mm]` ugyanaz mint a `\` de a következő sor távolsága 2 mm-rel megnő.
`*` ugyanaz mint a `\` de nem enged meg oldaltörést.
`*[2mm]` ugyanaz mint `\[2mm]` de nem enged meg oldaltörést.
`\linebreak` új sort kezd sorkizárással.
`\nolinebreak` a sortörést letiltja az adott helyen.

4.7.2. Oldaltörések

A \LaTeX maga végzi az oldaltöréseket. Ha azt akarja, hogy a telített oldalak alja egymáshoz igazított legyen, akkor használja a `\flushbottom` parancsot. Ennek hatása a `\raggedbottom` paranccsal szüntethető meg. Az oldaltörést adott esetben ki is kényszerítheti:

`\newpage` új oldalt (illetve kéthasábos szedésnél új hasábot) kezd. Az utolsó sort vízszintesen, azután pedig az oldalt (vagy hasábot) függőlegesen feltölti térközzel.
`\clearpage` az előzőtől annyiban különbözik, hogy kéthasábos szedésnél is új oldalt kezd, másrészt az új oldal kezdése előtt megjeleníti az ún. úszó objektumokat (lásd a 11. fejezetben).
`\cleardoublepage` ugyanaz mint a `\clearpage`, de kétoldalas szedésnél a dokumentum megjelenítését csak a következő páratlan oldalon folytatja.
`\pagebreak` oldalt tör oldalkitöltéssel.
`\nopagebreak` letiltja az oldaltörést.
`\enlargethispage{3mm}` az aktuális oldal függőleges méretét 3 mm-rel megnöveli, de az élőláb helyzetét nem igazítja hozzá.
`\enlargethispage*{3mm}` ugyanaz mint az előbbi, de az extra térközök elhagyásával maximalizálja az adott oldalra írható szövegmennyiséget.



Videó: Betűtípusok és -méretek, térközök, törések

4.8. Bekezdések

Új bekezdés esetén a forrásállományban hagyni kell egy üres sort, vagy ki kell adni a `\par` parancsot. (Gyakori hiba, hogy új bekezdés helyett sortörést alkalmaznak. Ez tipográfiai hiba, kerülje!)

Minden bekezdés behúzással kezdődik, kivéve az ún. fejezetnyitó bekezdést. Ha ezeket is behúzással szeretné kezdeni, akkor töltsse be az `indentfirst` csomagot, vagy a magyar.ldf `afterindent=force-yes` opcióját:

```
\PassOptionsToPackage{defaults=hu-min,afterindent=force-yes}{magyar.ldf}
```

Alaphelyzetben a bekezdések sorkizártak, azaz a sorok a bal margónál kezdődnek és a jobb margónál végződnek, kivéve az első sor elejét és az utolsó sor végét.

`\indent` kikényszeríti az adott bekezdés elején a behúzást.

`\noindent` letiltja az adott bekezdés elején a behúzást.

`\setlength{\parindent}{5pt}` a bekezdés behúzásának mértékét átállítja 5 pontra.

`\setlength{\parskip}{5pt}` két bekezdés közötti térközt megnöveli 5 ponttal.

4.8.1. Bekezdések balra zárása

Ilyenkor a bekezdést kezdő sor is a bal margónál kezdődik és nincs a jobb oldalon kiegyenlítés, így szóelválasztások sincsenek. Megvalósítása:

```
\begin{flushleft}
<szöveg>
\end{flushleft}
```

vagy

```
{\raggedright <szöveg>\par}
```

A két megoldás között az a különbség, hogy a `flushleft` környezet függőleges térközöket helyez a szöveg elejére és végére.

4.8.2. Bekezdések jobbra zárása

Képzeld el egy balra zárt szöveget, de most minden sort toljon el úgy, hogy a sorvégek a jobb margóhoz kerüljenek. Ez a jobbra zárás. Megvalósítása:

```
\begin{flushright}
<szöveg>
\end{flushright}
```

vagy

```
{\raggedleft <szöveg>\par}
```

A két megoldás között az a különbség, hogy a `flushright` környezet függőleges térközöket helyez a szöveg elejére és végére.

4.8.3. Bekezdések középre zárása

Képzeld el egy balra zárt szöveget, de most minden sort toljon el középre. Ez a középre zárás. Megvalósítása:

```
\begin{center}
<szöveg>
\end{center}
```

vagy

```
{\centering <szöveg>\par}
```

A két megoldás között az a különbség, hogy a `centering` környezet függőleges térközöket helyez a szöveg elejére és végére.

```
\begin{center}
Ez egy hosszabb szöveg, ami középre van zárva,
így szóelválasztások sincsenek benne.
De a sortörések pontjait mi is meg tudjuk adni:\\
```

```
Ez külön sorba kerül.\\ Ez is külön sorba kerül.\\
\\end{center}
```

Ez egy hosszabb szöveg, ami középre van zárva, így szóelválasztások sincsenek benne. De a sortörések pontjait mi is meg tudjuk adni:

Ez külön sorba kerül.
Ez is külön sorba kerül.

4.8.4. Többsoros idézetek

Ha többsoros idézetet akar kiemelni, akkor használja a `quotation` környezetet:

```
\\begin{quotation}
,,Örökös vigyora nemegyszer tévedésbe ejtette azokat, akik kissé
könnyelműen, a külsejük után ítélik meg embertársaikat, és ezért
a vigyorgó Jimmyt felületesen kezelték, vagy kicsúfolták. Az ilyen
emberek, felépülésük után, sokat gondolkodtak a látszat megtévesztő
benyomásairól, és elhatározták, hogy a jövőben senkiről sem vonnak
le következtetéseket alapos tájékozódás híján.'\\
\\hspace*{\\fill}(Rejtő Jenő)
\\end{quotation}
```

„Örökös vigyora nemegyszer tévedésbe ejtette azokat, akik kissé könnyelműen, a külsejük után ítélik meg embertársaikat, és ezért a vigyorgó Jimmyt felületesen kezelték, vagy kicsúfolták. Az ilyen emberek, felépülésük után, sokat gondolkodtak a látszat megtévesztő benyomásairól, és elhatározták, hogy a jövőben senkiről sem vonnak le következtetéseket alapos tájékozódás híján.”

(Rejtő Jenő)

4.8.5. Versek

Versszakokat a `verse` környezettel formázhatunk:

```
\\begin{verse}
\\textbf{Szabó Lőrinc: Szél hozott, szél visz el} (részlet)

Köd előtttem, köd mögöttem,\\
isten tudja, honnan jöttem,\\
szél hozott, szél visz el,\\
minek kérdejem: mért visz el?

Sose néztem, merre jártam,\\
a felhőknek kiabáltam,\\
erdő jött: jaj, be szép!\\
-- megcibáltam üstökét.
\\end{verse}
```

Szabó Lőrinc: Szél hozott, szél visz el (részlet)

Köd előtttem, köd mögöttem,
 isten tudja, honnan jöttem,
 szél hozott, szél visz el,
 minek kérdejem: mért visz el?

Sose néztem, merre jártam,
 a felhőknek kiabáltam,
 erdő jött: jaj, be szép!
 – megcibáltam üstökét.

4.8.6. Párbeszédek

Egy szereplő által mondott szöveget új bekezdésben, gondolatjellel kezdje. A gondolatjel után a szokásosnál nagyobb, rugalmatlan és törhetetlen szóközt kell hagyni. Ezt valósítja meg a

■ `\mond ∈ [magyar]babel`

parancs. A kimondott szövegbe gondolatjelek közt leírást is ékelhet, melyet ponttal kell lezárni. A kimondott szöveg végére szükség esetén ki kell tenni a kérdőjelet vagy a felkiáltójelet, de a pontot tilos. Például

```
Egy deszkán találta magát, amely a tenger hullámain zötykölődött.
\mond Hol a Titanic? -- kérdezte, de nem kapott választ.
\mond Ez nem lehet -- szólalt meg ismét. -- Öt perce még a kabinomban
voltam.
```

Egy deszkán találta magát, amely a tenger hullámain zötykölődött.
 – Hol a Titanic? – kérdezte, de nem kapott választ.
 – Ez nem lehet – szólalt meg ismét. – Öt perce még a kabinomban voltam.

A `magyar.ldf defaults=hu-min` opciója aktiválja a `mond=yes` opciót is, amely definiálja a `\mond` parancsot. Enélkül `\mond` helyett a következő írható:

■ `\par--\enspace`

4.9. Tabulálás

Szöveg tabulálása a `tabbing` környezettel és abban a következő parancsok használatával oldható meg:

■ `\= \\ \> \kill \+ \- \``

Ezek használata a következő példákon érthetővé válik:

```
\begin{tabbing}
0000000000000 \= 111111111111\\
                \> 11111111 \= 22222222\\
                \>                \> 222222 \\\
00000000 \=\ \\
                \> 111111                \> 222222
\end{tabbing}
```

```
000000000000 111111111111
                1111111 222222222
                  222222
00000000
                111111      222222
```

```
\begin{tabbing}
0000 \= 1111 \= 2222\kill
0    \> 1    \> 2\\
00   \> 11   \> 22\\
000  \> 111  \> 222\\
0000 \> 1111 \> 2222
\end{tabbing}
```

```
0    1    2
00   11   22
000  111  222
0000 1111 2222
```

```
\begin{tabbing}
0000 \= 1111 \= 2222 \= 3333\+\+\
                2222222222\\
                222222\-\
                111111111111\\
                111111\-\
0000000 \` Ez a sor végére kerül!
\end{tabbing}
```

```
0000 1111 2222 3333
                2222222222
                222222
                111111111111
                111111
0000000 Ez a sor végére kerül!
```

4.10. Lábjegyzetek

Ahová lábjegyzetet szeretne írni, ott adja ki a

```
\footnote{<lábjegyzet szövege>}
```

parancsot. Ez eggyel megnöveli a lábjegyzet sorszámát. Ha a

```
\footnote[<szám>]{<lábjegyzet szövege>}
```

parancsot használja, akkor a lábjegyzet száma nem nő, hanem az a szám íródik ki, amit a *<szám>* opcióban adott meg.

A `\footnote` előtt nem lehet szóköz. Ha a jegyzet egy adott szóra vonatkozik, akkor a parancsot közvetlenül a szó után írjuk, ha egy mondatra vagy mondatrészre, akkor az azt lezáró írásjel után. A lábjegyzet teljes mondatokból áll. Így nagybetűvel kell kezdeni és mondatzáró írásjellel befejezni.

A `magyar.ldf` fájl `defaults=hu-min` opciója a lábjegyzetek fölé nem tesz vízszintes vonalat. Ha mégis szeretne tenni, akkor írja be a következőt:

```
\footnotestyle{rule=fourth} ∈ [magyar]babel
```

Az `article` osztályban a lábjegyzet sorszámozása folyamatos, míg `report` és `book` esetén fejezetenként 1-től kezdődik. Ha azt akarja, hogy oldalanként előlről kezdődjön a számozás, akkor használja a következő parancsot a preambulumban:

```
\MakePerPage{footnote} ∈ perpage
```

Elvileg ugyanezt valósítja meg a `\footnotestyle{reset=page} ∈ [magyar]babel` parancs is, de nem ajánlom a használatát, mert valamikor hibás számozást eredményez.

A lábjegyzetek számozását átállíthatja csillagosra a következő paranccsal:

```
\footnotestyle{mark=stars-max} ∈ [magyar]babel
```

Visszaállítani arab számozásra így lehet:

```
\footnotestyle{mark=arabic} ∈ [magyar]babel
```

Ha a szerkesztő szeretne a műhöz megjegyzéseket írni lábjegyzetben, akkor használja a következő parancsot:

```
\editorfootnote{<szerkesztő megjegyzése>} ∈ [magyar]babel
```

Ez csillagos számozást használ és oldalanként újra indul.

Szintek (rész, fejezet, szakasz stb. lásd később) címében tipográfiailag helytelen lábjegyzetet használni. Ha mégis szükség van rá, akkor nem használható a `\footnote` parancs, mert a fejléc és tartalomjegyzék hibás lesz. Ehelyett használja a

```
\headingfootnote{<lábjegyzet szövege>} ∈ [magyar]babel
```

parancsot. További parancsok:

`\footnotemark` megnöveli egyel a lábjegyzet számát és az adott helyre kiteszi a lábjegyzet jelét.

`\footnotemark[<szám>]` a lábjegyzet számát változtatlanul hagyja és az adott helyre kiteszi a lábjegyzet jelét, amit a `<szám>` értéke ad meg.

`\value{footnote}` a lábjegyzet aktuális számát adja meg, ami beírható az előző parancsba a `<szám>` helyére.

`\footnotetext{<lábjegyzet szövege>}` szöveget ír a lábjegyzetbe, de nem változtatja meg a lábjegyzet számát és az adott helyre nem teszi ki a lábjegyzet jelét.

`\footnotetext[<szám>]{<lábjegyzet szövege>}` szöveget ír a lábjegyzetbe az opcióban megadott `<szám>` alatt, de nem változtatja meg a lábjegyzet számát és az adott helyre nem teszi ki a lábjegyzet jelét.

A lábjegyzet az oldalnak csak bizonyos százalékát foglalhatja el, így lehetséges, hogy egy hosszabb lábjegyzet több oldalon jelenik meg. Ha ezt a megoldást le akarja tiltani, akkor ki kell adni az

```
\interfootnotelinepenalty=10000
```

parancsot.

4.11. Széljegyzetek

Széljegyzeteket a

`\marginpar{<széljegyzet>}`

paranccsal írhat. A széljegyzetek alapértelmezésben a lapok bekötésének oldalával ellentétes ún. külső margóra kerülnek. Kétoldalas szedésnél a páros oldalakon a külső margó bal oldalra esik, páratlanakon pedig jobb oldalra. Egyoldalas szedésnél a külső margó mindig jobb oldalon van.

Ha azt akarja, hogy a külső margóval ellentétes ún. belső margóra kerüljön a széljegyzet, akkor adja ki a

`\reversemarginpar`

parancsot. Alapértelmezésre visszatérni a

`\normalmarginpar`

paranccsal lehet.

Kétoldalas szedés esetén a széljegyzetek hol bal, hol jobb oldalon lesznek. Ha azt akarja, hogy a bal oldalra kerülve a széljegyzet másképpen nézzen ki, mint jobb oldalon, használhatja a következőt:

`\marginpar[<széljegyzet bal oldalon>]{<széljegyzet jobb oldalon>}`

Például, ha azt akarja, hogy a széljegyzet szövege bal oldalon jobbra legyen igazítva, akkor használja a következő kódot:

`\marginpar[\raggedleft széljegyzet]{széljegyzet}`

Ha egy bekezdés elejére ír széljegyzetet, akkor a `\marginpar` parancs elé kell tenni egy `\mbox{}` parancsot, különben a széljegyzet és a bekezdés első sora között szintkülönbség lép fel. Ez azért van így, mert a `\marginpar` nem kezd új bekezdést.

4.12. Színek kezelése

4.12.1. Színmodellek és paraméterek

Színek kezelésére az `xcolor` csomag használható. Ez sok színmodellt ismer, itt csak néhányat említünk:

RGB használatakor három paramétert kell megadni vesszővel elválasztva, mindhárom 0 és 255 közötti egész szám. Az első a vörös, a második a zöld, a harmadik a kék mennyiségét jelenti.

rgb használatakor három paramétert kell megadni vesszővel elválasztva, mindhárom 0 és 1 közötti törtszám. Az első a vörös, a második a zöld, a harmadik a kék mennyiségét jelenti.

cmk használatakor négy paramétert kell megadni vesszővel elválasztva, mindegyik 0 és 1 közötti törtszám. Az első a cián, a második a magenta, a harmadik a sárga, a negyedik a fekete mennyiségét jelenti.




















HTML paramétere a szín hatjegyű hexadecimális kódja. (Lásd például itt: [klikk ide.](#))

gray a szürke skálát jelenti. Itt egy paramétert kell megadni, mely 0 és 1 közötti tört szám (0 = fekete, 1 = fehér).

wave esetén a paraméter a szín hullámhossza nanométerben. A hullámhossz értéke 363 és 814 közötti törtszám.

4.12.2. Színnevek

Az `xcolor` csomagban vannak előre definiált színek is, pontosabban, bizonyos paraméterű színekre adott néven is hivatkozhat. Ezek a következők:

 <code>black</code>	 <code>gray</code>	 <code>olive</code>	 <code>teal</code>
 <code>blue</code>	 <code>green</code>	 <code>orange</code>	 <code>violet</code>
 <code>brown</code>	 <code>lightgray</code>	 <code>pink</code>	 <code>white</code>
 <code>cyan</code>	 <code>lime</code>	 <code>purple</code>	 <code>yellow</code>
 <code>darkgray</code>	 <code>magenta</code>	 <code>red</code>	

Mi is megadhatunk színneveket a következő paranccsal:

```
\definecolor{<színnév>}{<modell>}{<színparaméter>} ∈ xcolor
```

Például

```
\definecolor{halvanyszurke}{gray}{0.8}
\definecolor{macibarna}{RGB}{128,64,0}
```

Arra is lehetőség van, hogy két adott nevű szín összekeveréséből adjon meg újabb színnevet:

```
<színnév1>!<szám>!<színnév2>
```

azt jelenti, hogy *<szám>* százalék *<színnév1>* színhez $(100 - \textit{<szám>})$ százalék *<színnév2>* színt keverünk. Például

```
green!30!yellow
```

esetén 30% zöldhöz kevertünk 70% sárgát.

Ha fehérrel akar keverni más színt, akkor egyszerűbb a kód:

```
<színnév1>!<szám> = <színnév1>!<szám>!white
```

Például

```
green!30
```

esetén 30% zöldhöz kevertünk 70% fehéret.

Színnevet definiálhat korábban definiált színnévvel is:

```
\colorlet{<új színnév>}{<régi színnév>} ∈ xcolor
```

Például

```
\colorlet{piros}{red!80}
\colorlet{fekete}{black}
```

4.12.3. Színes szöveg

Szövegek színezéséhez a következő parancsokat használhatja:

```
\textcolor[<modell>]{<színparaméter>}{<bekezdés>} ∈ xcolor
\textcolor{<színnév>}{<egy bekezdés>} ∈ xcolor
{\color[<modell>]{<színparaméter>}<több bekezdés>} ∈ xcolor
{\color{<színnév>}<több bekezdés>} ∈ xcolor
```

Például

```
\colorlet{piros}{red!80}
\textcolor{piros}{Piros szöveg.}
\textcolor[RGB]{0,255,0}{Zöld szöveg.}
{\color{black!50} Szürke szöveg.}
```

Piros szöveg. Zöld szöveg. Szürke szöveg.

4.12.4. Szöveg kiemelése színes háttérrel

Ehhez az `xcolor` csomag mellett használja a `soulutf8` csomagot is. Ekkor a következő parancsokat használhatja:

```
\sethlcolor{<színnév>} ∈ soulutf8
\hl{<szöveg>} ∈ soulutf8
```

A `\sethlcolor` parancs megadja a kiemelés színét. Alapértelmezése `yellow`. A `\hl` színezi ki a `<szöveg>` háttérét, ami akár több sorból, vagy akár több bekezdésből is állhat. Például

```
\hl{Ez egy fontos szöveg, azért van kiemelve!}
```

Ez egy fontos szöveg, azért van kiemelve!

4.12.5. Szöveg kiemelése színes aláhúzással

Ehhez az `xcolor` csomag mellett használja a `soulutf8` csomagot is. Ekkor a következő parancsokat használhatja:

```
\setul{<mélység>}{<vonaltagság>}
\setulcolor{<színnév>} ∈ soulutf8
\ul{<szöveg>} ∈ soulutf8
```

A `\setul` parancs `<mélység>` paramétere beállítja, hogy az aláhúzás mennyivel legyen az alapvonal alatt, a `<vonaltagság>` pedig, hogy milyen vastag legyen a vonal. A `\setulcolor` parancs `<színnév>` paramétere megadja az aláhúzás színét. Alapértelmezése `black`. A `\hl` parancs húzza alá a `<szöveg>` részt, ami akár több sorból, vagy akár több bekezdésből is állhat. Például a

```
\setul{1pt}{1pt}
\setulcolor{blue}
```

preambulumba írása után

```
\ul{Ez egy fontos szöveg, azért van aláhúzva kékkel!}
```

Ez egy fontos szöveg, azért van aláhúzva kékkel!

4.12.6. Színes lapok

A lap háttérszíne így adható meg:

```
\pagecolor[<modell>]{<színparaméter>} ∈ xcolor
\pagecolor{<színnév>} ∈ xcolor
```

4.13. Dátumtípusok

Tegyük fel, hogy a dokumentum fordításának dátuma 2018. december 7. Ekkor

2018	<code>\number\year</code>
12	<code>\number\month</code>
7	<code>\number\day</code>
2018. december 7.	<code>\today</code> (ha a magyar nyelv aktív)
December 7, 2018	<code>\today</code> (ha az angol nyelv aktív)
2018-12-07	<code>\emitdate{a}{\today} ∈ [magyar]babel</code>
2018. december 7.	<code>\emitdate{b}{\today} ∈ [magyar]babel</code>
2018. dec. 7.	<code>\emitdate{c}{\today} ∈ [magyar]babel</code>
2018. XII. 7.	<code>\emitdate{d}{\today} ∈ [magyar]babel</code>
2018. 12. 07.	<code>\emitdate{e}{\today} ∈ [magyar]babel</code>
2018. december	<code>\emitdate{f}{\today} ∈ [magyar]babel</code>
2018. december 7	<code>\emitdate{g}{\today} ∈ [magyar]babel</code>
2018 december	<code>\emitdate{h}{\today} ∈ [magyar]babel</code>

Rögzített dátumok esetén:

1848-03-15	<code>\emitdate{a}{1848-3-15} ∈ [magyar]babel</code>
1848. március 15.	<code>\emitdate{b}{1848-3-15} ∈ [magyar]babel</code>
1848. márc. 15.	<code>\emitdate{c}{1848-3-15} ∈ [magyar]babel</code>
1848. III. 15.	<code>\emitdate{d}{1848-3-15} ∈ [magyar]babel</code>
1848. 03. 15.	<code>\emitdate{e}{1848-3-15} ∈ [magyar]babel</code>
1848. március	<code>\emitdate{f}{1848-3-15} ∈ [magyar]babel</code>
1848. március 15	<code>\emitdate{g}{1848-3-15} ∈ [magyar]babel</code>
1848 március	<code>\emitdate{h}{1848-3-15} ∈ [magyar]babel</code>

`\weekday ∈ eukdate`

Kiírja, hogy a fordítás időpontja a hét melyik napjára esik. Csak angol verziója van. Ha magyarul akarja használni, akkor írja be a következőt a preambulumba az `eukdate` csomag betöltése után:

```
\makeatletter
\renewcommand\weekday{%
\ifcase\theuk@date Szombat\or Vas\{'a}rnap\or H\{'e}tf\H{o}\or
Kedd\or Szerda\or Cs\{"u}t\{"o}rt\{"o}k\or P\{'e}ntek\fi
\makeatother
```

`\DayAfter[⟨nap⟩] ∈ advdate`

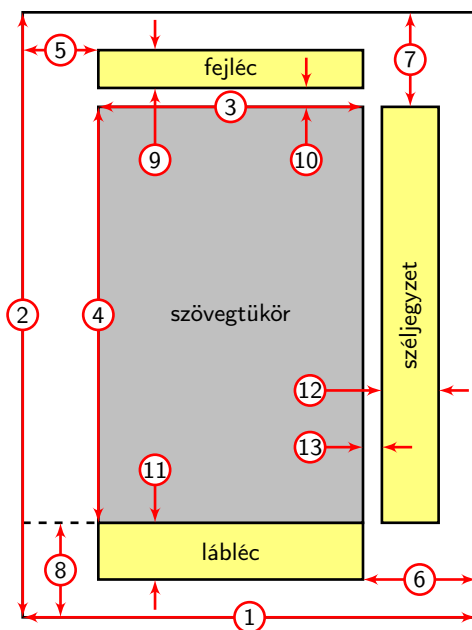
Kiírja, hogy a fordítás dátumához képest `⟨nap⟩` múlva mi a dátum. A `⟨nap⟩` alapértéke 1.

5. fejezet

Oldalak kinézete

5.1. Oldalak szerkezete és méretei

Egy oldal szerkezete a következő ábrán látható. Főbb részei: szövegtükör, margók, fejléc, lábléc, széljegyzet.



Az ábrán számokkal jelölt méreteket a `geometry` csomaggal állíthatja be, melyek a következők:

1. `paperwidth` Oldal szélessége.
2. `paperheight` Oldal magassága.
3. `textwidth` Szövegtükör szélessége.
4. `textheight` Szövegtükör magassága.
5. `inner` Belső margó szélessége. A belső margó a lapok kötése felőli margó. Egyoldalas dokumentum esetén ez a bal margót, míg kétoldalas dokumentum esetén páratlan oldalon a bal, illetve páros oldalon a jobb margót jelenti.
6. `outer` Külső margó (belső margóval ellentétes oldalon) szélessége.
7. `top` Felső margó magassága.
8. `bottom` Alsó margó magassága.
9. `headheight` Fejléc magassága.

10. `headsep` Fejléc és szövegtükör távolsága.
11. `footskip` Lábléc magassága.
12. `marginparwidth` Széljegyzet területének szélessége.
13. `marginparsep` Széljegyzet és szövegtükör távolsága.

Ha szabványos méretet akar (A0–A6, B0–B6), akkor az `a0paper`, ..., `a6paper`, `b0paper`, ..., `b6paper` opciók valamelyikét kell betölteni. Például

```
\usepackage[b5paper]{geometry}
```

Ha ugyanezt a méretet szeretné, de 90 fokkal elforgatva, akkor használja a `landscape` opciót is:

```
\usepackage[b5paper,landscape]{geometry}
```

Ha egyedi méreteket akar, akkor például a következőt kell tenni:

```
\usepackage[paperwidth=105mm,paperheight=75mm]{geometry}
```

Fontos, hogy ezek fizikailag is beállítják a lap méretét, nem úgy, mint a standard dokumentumosztályok lapméretre vonatkozó opciói, melyek csak a margókra vannak hatással. A `geometry` csomag opcióit parancsban is meg lehet adni:

```
\geometry{<opciók>} ∈ geometry
```

Például

```
\geometry{paperwidth=105mm,paperheight=75mm}
```

Ha egy dokumentumon belül az oldal geometriáját néhány oldal erejéig át akarja állítani, akkor használja a `geometry` csomag `\newgeometry` parancsát:

```
\newgeometry{<opciók>} ∈ geometry
```

Fontos, hogy ezzel a paranccsal a lap méretét nem lehet átállítani, csak az azon belüli méreteket (margók, lábléc, stb.). Például

```
\newgeometry{inner=20mm,outer=10mm}
```

Az alapgeometria visszaállítása:

```
\restoregeometry ∈ geometry
```

5.2. Oldalak nagyítása/kicsinyítése

A `geometry` csomag

```
\mag=<nagyítás>
```

opciójával a dokumentumot nagyítani/kicsinyíteni is tudja, ahol a `<nagyítás>` értéke ezrelékben megadott egész szám. Ez az opció a később tárgyalt `hyperref` csomag használatakor csak akkor működik jól, ha a `hyperref` előbb van betöltve, mint a `geometry`.

Például a `mag=1500` opcióval másfélszeres nagyítást érhet el, illetve a `mag=500` felére kicsinyít. Ilyenkor a fontok mérete és bármilyen mértékegységgel megadott hosszmeret is megváltozik $\frac{\text{mag}}{1000}$ -szeresére.

Ha valamely mértékegységgel megadott hosszmeretet nem akarja, hogy a nagyítás során megváltozzon, akkor a mértékegység elé tegye a `true` szót (`truemm`, `truecm`, `truept`). Ha nagyításnál a beállított szabványos oldalméretet nem akarja, hogy változzon, akkor használja a `truedimen` opciót. Például

```
\usepackage[mag=500,truedimen,a4paper]{geometry}
```

Ezzel be lehet állítani tetszőleges alapbetűméretet, hiszen, ha például az előző esetben az alap betűméret 12 pt volt, akkor a végeredmény alap betűmérete 6 pt lesz, miközben az oldal maradt A4-es méretű.

A következő példában a dokumentum szélessége 150 mm, magassága 250 mm, minden margó 20 mm, az alapbetűméret 13 pt. A beírt két sor, az aktuális sortávolságnál 2 cm-rel nagyobb távolságra vannak egymástól.

```
\documentclass{article}
\usepackage[mag=1300,paperwidth=150truemm,paperheight=250truemm,
  inner=20truemm,outer=20truemm,top=20truemm,bottom=20truemm]{geometry}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\begin{document}
Első sor.

\vspace{2truecm}
Második sor.
\end{document}
```

5.3. Többhasábos szedés

Kéthasábos szedés a dokumentumosztály `twocolumn` opciójával is lehetséges, de az eredmény több szempontból is kifogásolható, melyeket most nem részletezünk. Helyette a `multicol` csomag `multicols` környezetét használja:

```
\begin{multicols}{\langle hasábszám \rangle} \in multicol
\langle szöveg \rangle
\end{multicols}
```

A `\langle hasábszám \rangle` maximum 10 lehet. A hasábok közötti távolság 10 pt. Ennek átállítása például 1 cm-re:

```
\setlength{\columnsep}{1cm} \in multicol
```

A hasábok közötti vonalvastagság 0 pt. Ennek átállítása például 1 pt-ra:

```
\setlength{\columnseprule}{1pt} \in multicol
```

5.4. Oldal elforgatása

Ha egy oldal tartalma megkívánja (például egy széles táblázat), szükség lehet az álló tájolású oldal tartalmának elforgatására. Erre szolgál az `lscape` csomag `landscape` környezete. Hatása:

- új oldalt nyit;
- a szövegtükör és széljegyzet tartalmát elforgatja 90 fokkal, de a fejléc és a láblécet nem;
- a végén visszavált normál módra, de előtte új oldalt nyit.

Ez a megoldás a pdf nézőben az oldalt fizikailag nem forgatja el, csak a szövegtükör és a széljegyzet tartalmát. Ha az `lscape` helyett a `pdfscape` csomag `landscape` környezetét használja, akkor ugyanazt az eredményt kapja, de a pdf nézőben az oldal fizikailag is el lesz forgatva, aminek a hatására a szövegtükör és a széljegyzet vízszintesen fog elhelyezkedni a pdf nézőben. Ekkor azonban a fej- és lábléc helyezkedik el függőlegesen.

Ha a szövegtükör és széljegyzet tartalmával együtt a fej- és lábléc tartalmát is el akarja forgatni, továbbá azt akarja, hogy a pdf nézőben az oldal fizikailag is legyen elforgatva, akkor továbbra is használja `landscape` környezetet, de az `lscape` vagy `pdfscape` csomagok betöltése helyett írja a következőket a preambulumba:

```
\makeatletter
\def\rotatepage{
\clearpage
\pdfpagewidth=\paperheight
\pdfpageheight=\paperwidth
\textwidth=\dimexpr\paperheight-\paperwidth+\textwidth\relax
\textheight=\dimexpr\paperwidth-\paperheight+\textheight\relax
\paperwidth=\pdfpagewidth
\paperheight=\pdfpageheight
\hsize=\textwidth
\global\vsizer=\textheight
\global\@colht=\textheight
\global\@colroom=\textheight
\columnwidth=\dimexpr(\textwidth-\columnsep)/2\relax
\if@twocolumn\hsize=\columnwidth\fi
\@ifundefined{headwidth}{\headwidth=\textwidth}
\linewidth=\textwidth}
\makeatother
\newenvironment{landscape}{\rotatepage}{\rotatepage}
```

5.5. Méretek ellenőrzése

A szerkesztés folyamata alatt szükség lehet az oldal méreteinek ellenőrzésére. Erre több csomag is lehetőséget ad, melyek közül talán az `fgruler` a legpraktikusabb. Ha ezt betölti, akkor minden oldal előterében megjelenik egy vízszintes és egy függőleges vonalzó, melynek a kezdőpontja a lap bal felső sarkában lesz. Ha csak egy adott oldal adott pozíciójába akar ilyet helyezni, akkor akkor tegye a következőt:

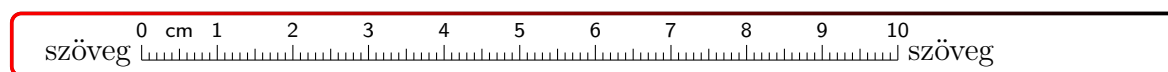
```
\documentclass{article}
\usepackage[type=none]{fgruler}
\begin{document}
\fgruler{upperleft}{\langle jobbra \rangle}{\langle lefelé \rangle}
\end{document}
```

A `\langle jobbra \rangle` helyére azt a távolságot kell beírni, amennyivel el akarja tolni jobbra a vonalzót a bal felső sarokhoz viszonyítva. A `\langle lefelé \rangle` helyére azt a távolságot kell beírni, amennyivel el akarja tolni lefelé a vonalzót a bal felső sarokhoz viszonyítva. Például ezen az oldalon a következő szerepel:

```
\fgruler{upperleft}{0cm}{0cm}
```

Ezzel a csomaggal szövegbe is helyezhet vonalzót. Például:

■ `szöveg \ruler{rightup}{10cm} szöveg`



Arra is lehetőség van, hogy ellenőrzés céljából az oldal elemeit (szövegtükör, széljegyzet, lábléc, fejléc) láthatóvá tegye vonalakkal. Ehhez használja a `showframe` opcióját a csomagnak. Az `fgruler` csomag rengeteg további lehetőséget ad vonalzó megjelenítésére, többek között például angol vonalzót is tud rajzolni, amelyben inch a mértékegység centiméter helyett. Ezek áttekintésére olvassa el a csomag dokumentációját.

További lehetőségek:

- Az `eso-pic` csomag `grid` opcióval. Ekkor minden oldal 5 mm-es közőkkel rácsvonatosan jelenik meg.
- A `geometry` csomag `showframe` opciója, vagy a `showframe` csomag, amely hasonló feladatot lát el, mint az `fgruler` csomag `showframe` opciója.
- A `layout` csomag `\layout` parancsa, amely megadja az adott dokumentum minden méretét.

6. fejezet


Kereszthivatkozások

Egy dokumentumban sok olyan elem lehet, amit számozunk. Ha többoldalas a dokumentum, akkor az oldalakat célszerű számozni. De a fejezeteket, szakaszokat is számozzuk. Ez például a 6. fejezet, amely az 52. oldalon kezdődik. További számozott elemek: listák, ábrák, táblázatok, matematikai képletek és tételek, irodalomjegyzék elemei, stb.

Az ilyen számozott elemekre nagyon sok esetben hivatkozunk. Ezek az ún. kereszthivatkozások. Természetesen ezeket nem érdemes a forrásban konkrétan beírni, hiszen egy ilyen szám a szerkesztés során még változhat, így állandóan változtatnunk kellene, ami egy idő után sok hibát eredményezne. Erre az a megoldás, hogy a \LaTeX -re bízunk a számozott elemeknél és a kereszthivatkozásoknál a megfelelő számok beírását.

6.1. Címkék

Ha egy számozott elemről kiderül, hogy hivatkoznunk kell rá, akkor először ezt az elemet meg kell címkézni a


```
 \label{<címké>}
```

paranccsal. A $\langle\text{címké}\rangle$ tetszőleges lehet, de azért érdemes néhány tanácsot megfogadni. Célszerű először arra utalni, hogy milyen típusú elemre hivatkozunk (fejezet, szakasz, ábra, táblázat, stb.). Ezzel a későbbi keresés a címkék között jóval könnyebb lesz. Ezután érdemes valamilyen írásjelet tenni. Az általános ajánlás erre a kettőspont, de látni fogjuk, hogy a magyarban ez nem feltétlenül a legjobb megoldás. Végül a címkében az elem tartalmára kell utalni, és semmiképpen sem a számára, mert ezzel pont az automatikus kereszthivatkozás lényegét sértenénk. Sok érthetetlen hibát megelőzhet, ha a címkében nem használ ékezetes betűket, szóközt és aktív karaktereket (magyarban ilyen a fordított aposztróf jel, kettőspont, kérdőjel, felkiáltójel és a pontosvessző).

Például, a későbbiekben látni fogjuk, hogy egy számozott listát a `enumerate` környezettel hozhat létre, melyben minden listaelemet `\item` paranccsal indítunk:

```
1 \begin{enumerate}
2 \item Ez egy listaelem.
3 \item Ez egy másik listaelem.
4 \end{enumerate}
```

Ha a 2. listaelemre akar hivatkozni, akkor a kódban a 3. sort így módosítsa:

```
 \item\label{lista-proba} Ez egy másik listaelem.
```

1. Ez egy listaelem.
2. Ez egy másik listaelem.

A címkében a prefix a `lista`, ami arra utal hogy ez egy listaelemre vonatkozik. Azután nem az általánosan tanácsolt kettőspont került, mert a magyarban ez aktív karakter, ami bizonyos esetekben gondokat okozhat. A kötőjel megfelel kettőspont helyett. Ezután jön maga a név, ami most `proba`.

6.2. Hivatkozás címkézett elemekre

Már csak az a kérdés, hogyan hivatkozunk a címkével ellátott elemre. Alapesetben a

```
\ref{<címke>}
```

paranccsal tudja ezt megtenni. Az előző példát folytatva:

```
Lásd \ref{lista-proba}.~listaelemet.
```

Lásd 2. listaelemet.

Sokkal szebb lenne a mondat, ha a sorszám elé határozott névelőt raknánk: „az 1.”, „a 2.”, stb. Amint látjuk a magyarban a névelő függ a sorszámtól. Ezt a problémát is megoldja a `magyar.ldf`. Ilyenkor használja az

```
\aref{<címke>} ∈ [magyar]babel
\Aref{<címke>} ∈ [magyar]babel
```

vagy az ezzel egyenértékű

```
\az{\ref{<címke>}} ∈ [magyar]babel
\Az{\ref{<címke>}} ∈ [magyar]babel
```

parancsokat, attól függően, hogy a sorszám előtti névelőt kis vagy nagy kezdőbetűvel szeretné:

```
Lásd \aref{lista-proba}.~listaelemet.\
\Aref{lista-proba}.~listaelemben olvasható.
```

Lásd a 2. listaelemet.

A 2. listaelemben olvasható.

Amikor címkézünk egy elemet, akkor nem csak az adott sorszámot tudja a \LaTeX , hanem azt is, hogy az adott elem melyik oldalon található. Adott címkéhez tartozó oldalszámot a

```
\pageref{<címke>}
```

paranccsal írathatja ki. Ennek is vannak névelős verziói:

```
\apageref{<címke>} ∈ [magyar]babel
\Apageref{<címke>} ∈ [magyar]babel
```

vagy az ezzel egyenértékű

```
\az{\pageref{<címke>}} ∈ [magyar]babel
\Az{\pageref{<címke>}} ∈ [magyar]babel
```

Például

`\Aref{lista-proba}.`~listaelemet `\apageref{lista-proba}.`~oldalon találjuk.

A 2. listaelemet az 53. oldalon találjuk.

A TeXstudio a címkéket a L^AT_EX-től függetlenül is tudja kezelni, azaz még a L^AT_EX-fordítás előtt meg tudja mondani, hogy létezik-e olyan címke, amire hivatkozunk, vagy esetleg egy címkét két külön elemhez is hozzárendeltünk. További segítség a címkék használatához a TeXstudióban, hogy a már meglévő címkéket kilistázza, amiből könnyen kiválaszthatjuk, melyikre akarunk hivatkozni.

A TeXstudio alapból nem ismeri az `\aref`, `\Aref`, `\apageref` és `\Apageref` parancsokat, így ezekben az esetekben nem élvezhetjük a TeXstudio adta kényelmi szolgáltatást. Erre két megoldás is van. Az egyik, hogy „megtanítja” a TeXstudiot ezekre a parancsokra. Ennek módját nem részletezzük, a TeXstudio leírásában megtalálhatja a menetét az Olvasó. A másik pedig, hogy `\aref{...}` helyett `\az{\ref{...}}`, `\Aref{...}` helyett `\Az{\ref{...}}`, `\apageref{...}` helyett `\az{\pageref{...}}` illetve `\Apageref{...}` helyett `\Az{\pageref{...}}` parancsokat ír a forrásba.

A `\pageref{<címke>}` kifejtése az az oldalszám, ahol a `\label{<címke>}` parancs ki lett adva, míg a `\ref{<címke>}` parancs kifejtése a `\label{<címke>}` kiadásakor aktuális `\@currentlabel` tartalma, ami alapesetben az adott elem sorszáma. Ezt át is lehet definiálni. Például

```
\section{Nagy számok törvénye}
\makeatletter
\def\@currentlabel{,,Nagy számok törvénye''}
\makeatother
\label{sec-nszt}
\Aref{sec-nszt} című szakaszban
```

1. Nagy számok törvénye

A „Nagy számok törvénye” című szakaszban

Létezik még ezeken kívül is hivatkozási forma (egyenlet, irodalomjegyzék), de ezeket majd az adott fejezetekben tárgyaljuk.



Videó: Bekezdések, lábjegyzetek, színek,
kereszthivatkozások

7. fejezet

Listák

7.1. Számozatlan listák

Számozatlan listákra az `itemize` környezet használható. Minden listaelemet `\item` parancs vezet be.

```
\begin{itemize}
  \item <listaelem>
  \item <listaelem>
\end{itemize}
```

E környezetek négy szint mélységig ágyazhatók egymásba. Például:

```
Lista előtti szöveg.
\begin{itemize}
  \item Listaelem az első szinten.
  \begin{itemize}
    \item Listaelem a második szinten.
    \item Újabb listaelem a második szinten.
  \end{itemize}
  \item Egy másik listaelem az első szinten.
\end{itemize}
Lista utáni szöveg.
```

Lista előtti szöveg.

- Listaelem az első szinten.
 - Listaelem a második szinten.
 - Újabb listaelem a második szinten.
- Egy másik listaelem az első szinten.

Lista utáni szöveg.

7.1.1. Felsorolásjelek megváltoztatása

Ha csak egy adott listaelem jelét akarja megváltoztatni, akkor azt az `\item` parancs opciójában teheti meg:

`\item[⟨jel⟩] ⟨listaelem⟩`

Például

```
\begin{itemize}
  \item[\textasteriskcentered] Listaelem.
  \item[\textbullet] Listaelem.
  \item Listaelem.
\end{itemize}
```

- * Listaelem.
- Listaelem.
- Listaelem.

Ha egy adott lista adott szintjének a jelét akarja megváltoztatni, akkor használja a következőt:

```
\begin{itemize}[⟨jel⟩] ∈ paralist
  \item ⟨listaelem⟩
  \item ⟨listaelem⟩
\end{itemize}
```

Például

```
\begin{itemize}[\textasteriskcentered]
  \item Listaelem.
  \item Listaelem.
\end{itemize}
```

- * Listaelem.
- * Listaelem.

Ha a felsorolás alapértelmezett jeleit szeretné megváltoztatni, akkor a következőket írja be:

```
\renewcommand{\labelitemi}{⟨1. szintjel⟩}
\renewcommand{\labelitemii}{⟨2. szintjel⟩}
\renewcommand{\labelitemiii}{⟨3. szintjel⟩}
\renewcommand{\labelitemiv}{⟨4. szintjel⟩}
```

vagy

```
\setdefaultitem{⟨1. szintjel⟩}{⟨2. szintjel⟩}{⟨3. szintjel⟩}{⟨4. szintjel⟩} ∈ paralist
```

Utóbbi esetben, ha egy szint jelét nem akarja átdefiniálni, akkor annak helyét hagyja üresen. Például, ha a `pifont` csomag betöltése után azt írja be, hogy

```
\renewcommand{\labelitemi}{\ding{42}}
\renewcommand{\labelitemii}{\ding{43}}
```

vagy

```
\setdefaultitem{\ding{42}}{\ding{43}}{}{}
```

akkor az utána következő

```
\begin{itemize}
  \item Listaelem.
```

```

\begin{itemize}
  \item Listaelem.
  \begin{itemize}
    \item Listaelem.
    \begin{itemize}
      \item Listaelem.
    \end{itemize}
  \end{itemize}
\end{itemize}
\end{itemize}

```

kód eredménye

```

• Listaelem.
  • Listaelem.
    ◦ Listaelem.
      * Listaelem.

```

7.1.2. Számozatlan listák extra térközök nélkül

Az `itemize` környezet minden listaelem között hagy egy extra függőleges térközt. Ha ezt nem akarja, akkor használja a `paralist` csomag `compactitem` környezetét. Ezt pontosan úgy kell használni, mint az előzőekben ismertetett `itemize` környezetet.

```

\begin{compactitem}[\langle jel \rangle] \in paralist
  \item[\langle jel \rangle] \langle listaelem \rangle
  \item[\langle jel \rangle] \langle listaelem \rangle
\end{compactitem}

```

Például

```

Lista előtti szöveg.
\begin{compactitem}
  \item Listaelem az első szinten.
  \begin{compactitem}
    \item Listaelem a második szinten.
    \item Újabb listaelem a második szinten.
  \end{compactitem}
  \item Egy másik listaelem az első szinten.
\end{compactitem}
Lista utáni szöveg.

```

```

Lista előtti szöveg.
– Listaelem az első szinten.
  • Listaelem a második szinten.
  • Újabb listaelem a második szinten.
– Egy másik listaelem az első szinten.
Lista utáni szöveg.

```


7.3. Számozott listák

Számozott listákra az `enumerate` környezet való. Minden listaelemet `\item` parancs előzmeg.

```
\begin{enumerate}
  \item <listaelem>
  \item <listaelem>
\end{enumerate}
```

E környezetek négy szint mélységig ágyazhatók egymásba. A szintek számozása a `magyar.ldf` fájl `defaults=hu-min` opciója esetén: arab számok, latin ábécé kisbetűi, görög ábécé kisbetűi, latin ábécé nagybetűi. Például:

```
Lista előtti szöveg.
\begin{enumerate}
  \item Listaelem az első szinten.
  \begin{enumerate}
    \item Listaelem a második szinten.
    \item Újabb listaelem a második szinten.
  \end{enumerate}
  \item Egy másik listaelem az első szinten.
\end{enumerate}
Lista utáni szöveg.
```

Lista előtti szöveg.

1. Listaelem az első szinten.
 - a) Listaelem a második szinten.
 - b) Újabb listaelem a második szinten.
2. Egy másik listaelem az első szinten.

Lista utáni szöveg.

Ha a `magyar.ldf` fájl `defaults=hu-min` opciója mellett használja a `labelenums=hu-A` opciót is

```
\PassOptionsToPackage{defaults=hu-min,labelenums=hu-A}{magyar.ldf}
```

módon, akkor a szintek számozása: nagy római számok, arab számok, latin ábécé nagybetűi, latin ábécé kisbetűi.

7.3.1. Számozott listák számozási stílusának megváltoztatása

Ha számozott listában egy adott listaelemet nem számozni, hanem csak egy jellel szeretné ellátni, akkor ezt az `\item` parancs opciójában adhatja meg:

```
\item[<jel>] <listaelem>
```

Például

```
\begin{enumerate}
  \item Listaelem.
  \item[---] Listaelem.
```

```
\item Listaelem.
\item[---] Listaelem.
\item Listaelem.
\end{enumerate}
```

1. Listaelem.
- Listaelem.
2. Listaelem.
- Listaelem.
3. Listaelem.

A `\item[<jel>]` esetén a számozás nem növekszik.

Ha csak egy adott lista adott szintjének számozását akarja megváltoztatni, akkor használja a következőt:

```
\begin{enumerate}[<címke>] \in paralist
\item <listaelem>
\item <listaelem>
\end{enumerate}
```

A *<címke>* bármilyen karaktert tartalmazhat, de ötnél a számozási stílus beállítása a feladata:

- 1 arab számozás
- i kis római számozás
- I nagy római számozás
- a latin ábécé kisbetűi szerinti számozás (alfanumerikus)
- A latin ábécé nagybetűi szerinti számozás (alfanumerikus)

Ezekből a betűkből a *<címke>* csak egyet tartalmazhat. Ha ezen öt karakter valamelyikét nem számozási stílus jeleként, hanem tényleges betűként akarja bevinni, akkor tegye kapcsos zárójelek közé. Például

```
\begin{enumerate}[\bfseries I. {axióma}.]
\item Listaelem.
\item Listaelem.
\end{enumerate}
```

- I. axióma.** Listaelem.
- II. axióma.** Listaelem.

```
\begin{enumerate}[\itshape(i)]
\item Listaelem.
\item Listaelem.
\end{enumerate}
```

- (i) Listaelem.
- (ii) Listaelem.

A négy szint mindegyike rendelkezik egy ún. számlálóval:

```
enumi    1. szint számlálójának a neve
enumii   2. szint számlálójának a neve
enumiii  3. szint számlálójának a neve
enumiv   4. szint számlálójának a neve
```

Egy számláló megjelenítése többféleképpen lehetséges:

```
\arabic{<számláló>} számláló kiírása arab számmal (1, 2, 3, ...)
\roman{<számláló>}  számláló kiírása kis római számmal (i, ii, iii, ...)
\Roman{<számláló>}  számláló kiírása nagy római számmal (I, II, III, ...)
\alph{<számláló>}   számláló kiírása latin ábécé kisbetűivel (a, b, c, ...)
\Alph{<számláló>}   számláló kiírása latin ábécé nagybetűivel (A, B, C, ...)
```

Tehát például

```
\Roman{enumii}
```

kiírja az adott lista éppen aktuális második szintjének a számát nagy római számmal.

Ha az első szint alapértelmezett számozását át akarja állítani nagy római számozásra, akkor ezt a következő kóddal teheti meg:

```
\renewcommand{\theenumi}{\Roman{enumi}}
```

Még azt is be lehet állítani, hogy ez a szám hogyan jelenjen meg. Például, ha azt akarja, hogy félkövér legyen és utána álljon egy pont, akkor tegye ezt:

```
\renewcommand{\labelenumi}{\bfseries\theenumi.}
```

Próbálja ki a következő kódot:

```
\renewcommand{\theenumi}{\arabic{enumi}}
\renewcommand{\theenumii}{\alph{enumii}}
\renewcommand{\theenumiii}{\roman{enumiii}}
\renewcommand{\theenumiv}{\Alph{enumiv}}
\renewcommand{\labelenumi}{\theenumi.}
\renewcommand{\labelenumii}{\itshape\theenumii}}
\renewcommand{\labelenumiii}{\theenumiii.}
\renewcommand{\labelenumiv}{\theenumiv.}
```

Ezután írjon egy számozott listát:

```
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\end{enumerate}
\end{enumerate}
\end{enumerate}
\end{enumerate}
```

1. Listaelem.

(a) Listaelem.

i. Listaelem.

A. Listaelem.

Egy másik példa:

```
\renewcommand{\theenumi}{\arabic{enumi}}
\renewcommand{\theenumii}{\arabic{enumii}}
\renewcommand{\theenumiii}{\arabic{enumiii}}
\renewcommand{\labelenumi}{\theenumi.}
\renewcommand{\labelenumii}{\theenumi.\theenumii.}
\renewcommand{\labelenumiii}{\theenumi.\theenumii.\theenumiii.}
```

Ezután írjon egy számozott listát:

```
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\end{enumerate}
\end{enumerate}
\end{enumerate}
```

1. Listaelem.

1.1. Listaelem.

1.1.1. Listaelem.

Ehhez hasonló eredmény érhető el, ha betölti a `paralist` csomagot `pointedenum` opcióval.

A számlálókat megjelenítő parancsok sora újakkal is bővíthető. Például a `pifont` csomag betöltése után a következő kód

```
\newcommand{\dingI}[1]
{\ifcase\value{#1}\or\ding{172}\or\ding{173}\or\ding{174}\or\ding{175}%
\or\ding{176}\or\ding{177}\or\ding{178}\or\ding{179}\or\ding{180}\or%
\ding{181}\else\ding{109}\fi}
```

egy `\dingI{<számláló>}` parancsot definiál, amely a számlálót a következő módon jeleníti meg: ①, ②, ..., ⑩, ○, ○, Egy másik példa, szintén a `pifont` csomag betöltése után:

```
\newcommand{\dingII}[1]
{\ifcase\value{#1}\or\ding{182}\or\ding{183}\or\ding{184}\or\ding{185}%
\or\ding{186}\or\ding{187}\or\ding{188}\or\ding{189}\or\ding{190}\or%
\ding{191}\else\ding{108}\fi}
```

egy `\dingII{<számláló>}` parancsot definiál, amely a számlálót a következő módon jeleníti meg: ①, ②, ..., ⑩, ●, ●, Ezután az első két szintet állítsa így át:

```
\renewcommand{\theenumi}{\dingI{enumi}}
\renewcommand{\theenumii}{\dingII{enumii}}
```

```
\renewcommand{\labelenumi}{\theenumi}
\renewcommand{\labelenumii}{\theenumii}
```

Most próbáljon ki egy számozott listát:

```
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\item Listaelem.
\end{enumerate}
\item Listaelem.
\end{enumerate}
```

- ① Listaelem.
 - ❶ Listaelem.
 - ❷ Listaelem.
 - ② Listaelem.

Van egy másik lehetőség is a szintek számozási stílusának globális átalakítására:

```
\setdefaultenum{<1. szintjel>}{<2. szintjel>}{<3. szintjel>}{<4. szintjel>} ∈ paralist
```

Ha egy szint számozási stílusát nem akarja átdefiniálni, akkor annak helyét hagyja üresen. Az *<n. szintjel>* bármilyen karaktert tartalmazhat, de ötnek a számozási stílus beállítása a feladata:

- 1 arab számozás
- i kis római számozás
- I nagy római számozás
- a latin ábécé kisbetűi szerinti számozás (alfanumerikus)
- A latin ábécé nagybetűi szerinti számozás (alfanumerikus)

Ezekből a betűkből az *<n. szintjel>* csak egyet tartalmazhat. Ha ezen öt karakter valamelyikét nem számozási stílus jeleként, hanem tényleges betűként akarja bevinni, akkor tegye kapcsos zárójelek közé. Például

```
\setdefaultenum{\bfseries I. {axióma.}}{\itshape (a)}{ }{ }
```

esetén a

```
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\begin{enumerate}
\item Listaelem.
\end{enumerate}
\end{enumerate}
\end{enumerate}
\end{enumerate}
```

kód eredménye

I. axióma. Listaelem.

(a) Listaelem.

α) Listaelem.

A) Listaelem.

7.3.2. Hivatkozás számozott listaelemre

A kereszthivatkozásoknál láttuk, hogy hogyan lehet hivatkozni egy listaelemre.

```
\begin{enumerate}
  \item Ez egy listaelem.
  \item\label{lista-masik} Ez egy másik listaelem.
\end{enumerate}
\Aref{lista-masik}.~listaelem miatt \dots
```

1. Ez egy listaelem.

2. Ez egy másik listaelem.

A 2. listaelem miatt ...

Mi történik akkor, ha a második szint egy elemére akarunk hivatkozni?

```
\begin{enumerate}
  \item Ez egy listaelem.
  \begin{enumerate}
    \item\label{lista-2.szint} Ez egy listaelem a 2. szinten.
  \end{enumerate}
\end{enumerate}
\Aref{lista-2.szint}.~listaelem miatt \dots
```

1. Ez egy listaelem.

a) Ez egy listaelem a 2. szinten.

Az 1a. listaelem miatt ...

Látjuk, hogy nem csak az adott listaelem száma jelenik meg, hanem előtte az is, hogy melyik listaelem alatt helyezkedik el. Ez a listaelem ún. prefixe. Ezek a prefixek átláthatók. Például

```
\makeatletter
\renewcommand{\p@enumii}{\theenumi} % 2. szint prefixe
\renewcommand{\p@enumiii}{\p@enumii(\theenumii)} % 3. szint prefixe
\renewcommand{\p@enumiv}{\p@enumiii\theenumiii-} % 4. szint prefixe
\makeatother
```

hatására a hivatkozások alakja (magyar nyelv esetén alapesetben):

1. szinten: 1

2. szinten: 1a

- 3. szinten: 1(a) α
- 4. szinten: 1(a) α -A

7.3.3. Számozott listák extra térközök nélkül

Ha nem akarja, hogy a listaelemek között legyen extra függőleges térköz, akkor az `enumerate` környezet helyett használja a `paralist` csomag `compactenum` környezetét. Használata pontosan megegyezik az `enumerate` környezettel.

```
\begin{compactenum} ∈ paralist
  \item <listaelem>
  \item <listaelem>
\end{compactenum}
```

Például

```
Lista előtti szöveg.
\begin{compactenum}
  \item Listaelem az első szinten.
  \begin{compactenum}
    \item Listaelem a második szinten.
    \item Újabb listaelem a második szinten.
  \end{compactenum}
  \item Egy másik listaelem az első szinten.
\end{compactenum}
Lista utáni szöveg.
```

```
Lista előtti szöveg.
1. Listaelem az első szinten.
  a) Listaelem a második szinten.
  b) Újabb listaelem a második szinten.
2. Egy másik listaelem az első szinten.
Lista utáni szöveg.
```

7.3.4. Sorfolytonos számozott listák

Erre a `paralist` csomag `inparaenum` környezete használható:

```
\begin{inparaenum}[<címke>] ∈ paralist
  \item <listaelem>
  \item <listaelem>
\end{inparaenum}
```

A `<címke>` pontosan úgy állítható be, mint az `enumerate` környezet leírásánál (alapbeállítás: 1.). Például

```
Szöveg
\begin{inparaenum}
  \item szöveg
  \item szöveg
  \item szöveg
\end{inparaenum}
```

Szöveg 1. szöveg 2. szöveg 3. szöveg

Szöveg

```
\begin{inparaenum}[\itshape (a)]
```

```
\item szöveg
```

```
\item szöveg
```

```
\item szöveg
```

```
\end{inparaenum}
```

Szöveg (a) szöveg (b) szöveg (c) szöveg



Videó: Listák

8. fejezet

Képek

Képek beillesztése esetén használja a `graphicx` csomagot. A képeket helyezze a forrásállományt tartalmazó mappába, vagy ami még praktikusabb, annak egy almappjába. A dokumentum forrásának hordozhatósága miatt célszerű a képeknek és az almappáknak is olyan nevet adni, amiben nincs ékezetes betű és szóköz.

Ha a forrásállomány fordítását `latex.exe` végzi, azaz dvi a cél, akkor eps képeket kell használni. Ha `pdflatex.exe` a fordító (TeXstudióban ez az alapbeállítás), akkor jpg, png vagy pdf képeket használjon. Ha mindkét fordítás szóba jöhet, akkor egy képet mindkét formátumban (azaz például eps és jpg) tegye be a megfelelő mappába.

8.1. Kép beillesztése

Amikor a forrásállomány azon pontjához ér, ahol meg kell jeleníteni a képet, használja a következő parancsot:

```
\includegraphics[<opciók>]{<képfájl>} ∈ graphicx
```

A *<képfájl>* megadásakor a kiterjesztést nem kell megadni. Azaz például, ha az `abra.jpg` képet kell beilleszteni, akkor

```
\includegraphics{abra}
```

Kiterjesztést azért nem kell megadni, mert `pdflatex.exe` használata esetén pdf, jpg vagy png kiterjesztést fog keresni, míg `latex.exe` esetén eps kiterjesztést. Így, ha mindkét formátumban megadtuk a képet, akkor bármely fordítót használhatjuk a forrás változtatása nélkül. Fontos, hogy ebben az esetben az `abra.jpg` fájlnak az aktuális mappában kell elhelyezkednie. Ha az aktuális mappa egy almappjában van a kép, például a `grafikonok` nevű almappában, akkor a következő kódot lehet használni:

```
\includegraphics{grafikonok/abra}
```

Gyakori hiba, hogy a teljes elérési utat megadják. Például

```
\includegraphics{C:/minta/grafikonok/abra.jpg} % ÍGY SOHA!
```

Ez rossz megoldás, hiszen ekkor a forrás csak ezen az útvonalon fog lefordulni, azaz nem lesz hordozható.

Arra is van lehetőség, hogy az almappa nevét nem minden egyes `\includegraphics` parancsban adjuk meg. Ekkor használjuk a

```
\graphicspath{{<almappa>/}} ∈ graphicx
```

parancsot. Például, ha az almappa neve `grafikonok`, akkor

```
\graphicspath{{grafikonok/}}
```

Ebben az esetben a `grafikonok` almappában található `abra.jpg` fájl a következő kóddal is megjeleníthető:

```
\includegraphics{abra}
```

Ezzel a forrás nagyon rugalmassá válik, hiszen az almappa esetleges átnevezésekor a forrásban csak egy helyen kell javítani. Ha több almappába is tett képeket, például az előzőn kívül a `geometria` nevűbe is, akkor ezt kell beírni:

```
\graphicspath{{grafikonok/}{geometria/}}
```

Értelemszerű változtatással további almappákat is beírhat. A különböző almappákba ne tegyen azonos nevű fájlokat.

Sajnos ennek a megoldásnak van egy veszélye. Ugyanis a fordító a képet először az aktuális mappában keresi, majd a telepített csomagok között, és csak utolsónak a `\graphicspath`-ban megadott almappá(k)ban. Például a `notes` nevű csomag tartalmaz egy `info.pdf` nevű képfájlt. Így, ha van egy saját `info.pdf` képfájlja a `grafikonok` nevű almappában, akkor a

```
\graphicspath{{grafikonok/}}
\includegraphics{info}
```

kód a `notes` csomagban található képet fogja betölteni, nem a felhasználóét. Ez csak egy módon védhető ki biztosan, ha minden képnek ad egy olyan egyedi prefixet, ami biztosan nem lehet a telepített csomagok között. Ez lehet egy számsor vagy egy név is, például `tomacs-info.pdf`. A magam részéről nem kedvelem ezt a megoldást, ezért nem használom a `\graphicspath` parancsot.

Az `\includegraphics` parancsnak a következő opciói vannak:

`width=<szélesség>` A kép szélessége (például `width=5cm`).

`height=<magasság>` A kép magassága (például `height=5cm`). A `width` és a `height` együttes megadásával a képet torzíthatjuk is.

`scale=<arányyszám>` Nagyítás/kicsinyítés mértéke (például `scale=2`).

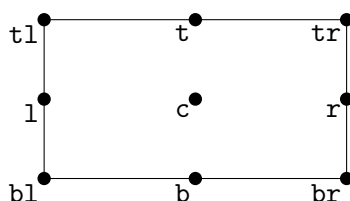
`trim=<bal> <lent> <jobb> <fent>` A képen meghatároz egy keretet. Ezután a kép pozicionálása a keret levágása után maradó képre történik, de az egész kép meg fog jelenni, azaz tényleges vágás nem történik. Például, ha azt írjuk be, hogy `trim=10mm 11mm 12mm 13mm`, akkor a keret bal oldalon 10, alul 11, jobb oldalon 12 és felül 13 mm széles lesz.

`clip` Ha a `trim` mellé ezt is betöltjük, akkor tényleges vágás történik.

`page=<oldal>` Többoldalas pdf fájl esetén az oldal kiválasztása (például `page=5`).

`angle=<fok>` Kép forgatásának szöge fokban. A pozitív érték az óra járásával ellentétes irány.

`origin=<origó>` Forgatás középpontja. Az `<origó>` értékei a következők lehetnek: `t1`, `t`, `tr`, `l`, `c`, `r`, `bl`, `b`, `br` (alapérték: `bl`). Ezek magyarázata a következő ábrán látható:



A következő példában a kép szélességét 3 cm-re állítjuk és elforgatjuk 90 fokkal az óra járásával megegyező irányban a középpontja körül:

```
\includegraphics[width=3cm,angle=-90,origin=c]{abra}
```

Körbenyírjuk a képet 10 mm-rel majd lekicsinyítjük a felére:

```
\includegraphics[trim=10mm 10mm 10mm 10mm,clip,scale=0.5]{abra}
```

A többoldalas abra.pdf fájlból az 5. oldalt jeleníti meg képként 10 cm magasan:

```
\includegraphics[height=10cm,page=5]{abra}
```

8.2. Hát- illetve előtérbe illesztés

Háttérképet a `graphicx` után betöltött `eso-pic` csomaggal tud beilleszteni.

```
\AddToShipoutPictureBG*{<háttér>} ∈ eso-pic
```

Például

```
\AddToShipoutPictureBG*{\setlength{\unitlength}{1mm}\put(10,20)
{\includegraphics[width=15cm]{hatter}}}
```

a `hatter` nevű 15 cm széles képet az adott oldal háttereként helyezi el úgy, hogy a kép bal alsó sarka az oldal bal alsó sarkához, mint origóhoz viszonyított (10, 20) koordinátájú pontban van, ahol egy egység 1mm.

Ugyanez a kód `*` nélkül az adott képet minden oldalon megjeleníti háttérként. Így lehet például vízjelet készíteni. Ennek hatása a

```
\ClearShipoutPictureBG ∈ eso-pic
```

paranccsal szüntethető meg. Természetesen kép helyett bármilyen szöveg is hasonlóan beilleszthető háttérként.

Ha az előtérbe akar helyezni valamit (például egy pecsétet), akkor az előző parancsokban a BG (background) betűk helyére írjon FG (foreground) betűket.

8.3. Külső pdf oldalak beszúrása

Ha a dokumentumba néhány oldalt be akar tenni egy külső pdf fájlból, akkor használja a `pdfpages` csomag

```
\includepdf[<opciók>]{<pdf fájl>} ∈ pdfpages
```

parancsát. Például

```
\includepdf[pages={3,8-11,15}]{doc.pdf}
```

a `doc.pdf` 3, 8, 9, 10, 11, 15 sorszámú oldalait szúrja be. A következő

```
\includepdf[pages=-]{doc.pdf}
```

a `doc.pdf` minden oldalát beszúrja. A következő

```
\includepdf[pages=last-1]{doc.pdf}
```

a `doc.pdf` minden oldalát beszúrja fordított sorrendben.

Az előzőekben ismertetett módon beszúrt pdf oldalakon meg fog jelenni az adott dokumentumstílusnak megfelelő fej- és lábléc is. Amennyiben ezt nem szeretné, azaz a

beszúrt oldalakon pontosan azt akarja látni, amit az eredeti pdf-ben is, akkor használja a `pagecommand={\thispagestyle{empty}}` opciót. Például

```
■ \includepdf[pagecommand={\thispagestyle{empty}},pages=-]{doc.pdf}
```

9. fejezet

Ábrák készítése

Az itt látható példák működéséhez töltsse be a `pict2e` és `xcolor` csomagokat. Ha bonyolultabb képelemeket tartalmazó rajzokat szeretne készíteni \LaTeX -hel, akkor nézze át a `curves`, `curve2e`, `xpicture`, `tikz` csomagok valamelyikének a leírását. A legnagyobb tudású csomag ezek közül a `tikz`.

9.1. Koordináta-rendszer, referenciapont és vonalvastagság

A rajzot egy képzeletbeli koordináta-rendszerben készítjük el.

```
\setlength{\unitlength}{\langle hossz \rangle}
```

Ezzel adjuk meg a koordináta-rendszerben az egység hosszát. Alapértéke 1pt.

```
\begin{picture}(\langle x \rangle,\langle y \rangle)
\end{picture}
```

Egy $\langle x \rangle$ egység széles és $\langle y \rangle$ egység magas doboz jön létre, melynek a bal alsó sarkában található az origó, azaz a $(0, 0)$ koordinátájú pont.

```
\begin{picture}(\langle x \rangle,\langle y \rangle)(\langle p \rangle,\langle q \rangle)
\end{picture}
```

Egy $\langle x \rangle$ egység széles és $\langle y \rangle$ egység magas doboz jön létre, melynek a bal alsó sarkában található a $(\langle p \rangle, \langle q \rangle)$ koordinátájú pont.

```
\put(\langle x \rangle,\langle y \rangle){\langle képelem \rangle}
```

A `picture` környezet által létrehozott dobozban a $\langle képelem \rangle$ úgy kerül megrajzolásra, hogy annak referenciapontja az $(\langle x \rangle, \langle y \rangle)$ koordinátájú pontba kerüljön. A $\langle képelem \rangle$ akár szöveg is lehet.

```
\multiput(\langle x \rangle,\langle y \rangle)(\langle dx \rangle,\langle dy \rangle){\langle szám \rangle}{\langle képelem \rangle}
```

A `picture` környezet által létrehozott dobozban a $\langle képelem \rangle$ $\langle szám \rangle$ darab példánya úgy kerül megrajzolásra, hogy az első referenciapontja az $(\langle x \rangle, \langle y \rangle)$ koordinátájú pontba kerüljön, minden további pedig az előzőhöz képest $(\langle dx \rangle, \langle dy \rangle)$ vektorral legyen eltolva.

■ `\linethickness{<vastagság>}`

A megrajzolt vonalak vastagsága.

9.2. Szakaszok, törött vonalak és vektorok

■ `\line(<x>,<y>){<v>}`

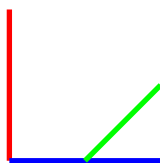
Rajzol egy szakaszt, melynek a referenciapontja a kezdőpontja, irányvektora $(\langle x \rangle, \langle y \rangle)$ és a vízszintes vetületének hossza $\langle v \rangle$. Ha a szakasz függőleges, akkor $\langle v \rangle$ a szakasz hosszát jelenti.

```

1 \setlength{\unitlength}{1cm}
2 \begin{picture}(2,2)
3   \linethickness{2pt}
4   \put(0,0){\color{red}\line(0,1){2}}
5   \put(0,0){\color{blue}\line(1,0){2}}
6   \put(1,0){\color{green}\line(1,1){1}}
7 \end{picture}

```

- Az 1. sorban a koordináta-rendszerben az egységet 1 cm-re állítottuk be.
- A 2. sorban a vonalak vastagságát állítottuk 2 pt-ra.
- A 3. sorban létrehoztunk egy 2 egység széles és 2 egység magas rajzfelületet, aminek a bal alsó sarkában lesz az origó.
- A 4. sorban húztunk egy piros szakaszt, amelynek a kezdő- azaz a referenciapontja a (0,0) pont, irányvektora (0,1), azaz függőleges, és a hossza 2 egység.
- Az 5. sorban húztunk egy kék szakaszt, amelynek a kezdőpontja a (0,0) pont, irányvektora (1,0), azaz vízszintes, és a hossza 2 egység.
- A 6. sorban húztunk egy zöld szakaszt, amelynek a kezdőpontja az (1,0) pont, irányvektora (1,1), és a vízszintes vetületének a hossza 1 egység.

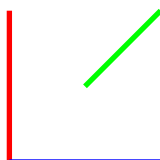


A következő rajzban a doboz bal alsó sarkának a koordinátája $(-1, -1)$:

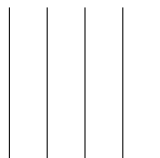
```

\setlength{\unitlength}{1cm}
\begin{picture}(2,2)(-1,-1)
  \linethickness{2pt}
  \put(-1,-1){\color{red}\line(0,1){2}}
  \put(-1,-1){\color{blue}\line(1,0){2}}
  \put(0,0){\color{green}\line(1,1){1}}
\end{picture}

```



```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \multiput(0,0)(0.5,0){5}{\line(0,1){2}}
\end{picture}
```



■ `\polyline($\langle x1 \rangle, \langle y1 \rangle$)($\langle x2 \rangle, \langle y2 \rangle$)...($\langle xn \rangle, \langle yn \rangle$)`

Rajzol egy törött vonalat, mely az adott koordinátájú pontokon halad át.

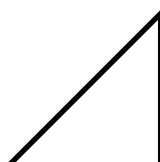
```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \linethickness{2pt}
  \polyline(0,0)(2,0)(2,2)
\end{picture}
```



■ `\polygon($\langle x1 \rangle, \langle y1 \rangle$)($\langle x2 \rangle, \langle y2 \rangle$)...($\langle xn \rangle, \langle yn \rangle$)`

Rajzol egy sokszöget, melynek csúcspontjai az adott koordinátájú pontok.

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \linethickness{2pt}
  \polygon(0,0)(2,0)(2,2)
\end{picture}
```





■ `\polygon*($\langle x1 \rangle, \langle y1 \rangle$)($\langle x2 \rangle, \langle y2 \rangle$)...($\langle xn \rangle, \langle yn \rangle$)`

Rajzol egy teli sokszöget, melynek csúcspontjai az adott koordinátájú pontok.

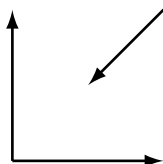
```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  {\color{red}\polygon*(0,0)(2,0)(2,2)}
\end{picture}
```



■ `\vector{⟨x⟩,⟨y⟩}{⟨v⟩}`

Rajzol egy vektort, melynek a referenciapontja a kezdőpontja, irányvektora $(\langle x \rangle, \langle y \rangle)$ és a vízszintes vetületének hossza $\langle v \rangle$. Ha a vektor függőleges, akkor $\langle v \rangle$ a vektor hosszát jelenti. Alapesetben a vektornyíl alakja , de ha a `pict2e` csomagot `pstarrows` opcióval tölti be, akkor .

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \linethickness{1pt}
  \put(0,0){\vector(0,1){2}}
  \put(0,0){\vector(1,0){2}}
  \put(2,2){\vector(-1,-1){1}}
\end{picture}
```

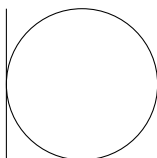


9.3. Körvonalak

■ `\circle{⟨átmérő⟩}`

Rajzol egy $\langle \text{átmérő} \rangle$ egység átmérőjű körvonalat, melynek a referenciapontja a középpontja.

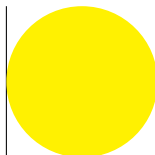
```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(2,0)(0,0)(0,2)
  \put(1,1){\circle{2}}
\end{picture}
```



■ `\circle*{⟨átmérő⟩}`

Rajzol egy *⟨átmérő⟩* egység átmérőjű körlapot, melynek a referenciapontja a középpontja.

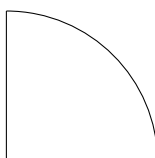
```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(2,0)(0,0)(0,2)
  \put(1,1){\color{yellow}\circle*{2}}
\end{picture}
```



■ `\arc[⟨szög1⟩,⟨szög2⟩]{⟨sugár⟩}`

Rajzol egy *⟨sugár⟩* egység sugarú körívet *⟨szög1⟩*-től *⟨szög2⟩*-ig, melynek a referenciapontja a középpontja. A *⟨szög1⟩* és *⟨szög2⟩* fokokban van megadva, melyek alapértékei 0 illetve 360.

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(2,0)(0,0)(0,2)
  \put(0,0){\arc[0,90]{2}}
\end{picture}
```



■ `\arc*[⟨szög1⟩,⟨szög2⟩]{⟨sugár⟩}`

Rajzol egy *⟨sugár⟩* egység sugarú telített körcíkkel *⟨szög1⟩*-től *⟨szög2⟩*-ig, melynek a referenciapontja a középpontja. A *⟨szög1⟩* és *⟨szög2⟩* fokokban van megadva, melyek alapértékei 0 illetve 360.

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \put(0,0){\color{red}\arc*[45,90]{2}}
\end{picture}
```



9.4. Lekerekített sarkú téglalapok

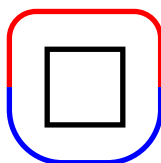
■ `\oval[$\langle sugár \rangle$]($\langle x \rangle$, $\langle y \rangle$)[$\langle rész \rangle$]`

Rajzol egy $\langle x \rangle$ egység széles és $\langle y \rangle$ egység magas lekerekített sarkú téglalapot, melynek sarkai negyed körök. A referenciapontja a középpont. A $\langle rész \rangle$ opcióval lehet megadni, hogy a téglalap melyik része kerüljön megrajzolásra. Lehetséges értékek: **t**: felső fele; **b**: alsó fele; **l**: bal fele; **r**: jobb fele; **tl**: bal felső negyede; **tr**: jobb felső negyede; **br**: jobb alsó negyede; **bl**: bal alsó negyede. A sarkokat jelentő negyed köröknek a sugara a lehetséges legnagyobb olyan érték, amely kisebb vagy egyenlő a $\langle sugár \rangle$ -nál, melynek alapértéke 20pt. A $\langle sugár \rangle$ lehet egy szám, amikor is az értéke $\langle sugár \rangle$ egység, és lehet egy konkrét hossz is. Ha a $\langle sugár \rangle$ értéke 0, akkor normál téglalapot kapunk. A $\langle sugár \rangle$ nem csak opcióban adható meg, hanem a

■ `\renewcommand{\maxovalrad}{ $\langle sugár \rangle$ }`

paranccsal is.

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)(-1,-1)
  \linethickness{2pt}
  \put(0,0){\color{red}\oval[10pt](2,2)[t]}
  \put(0,0){\color{blue}\oval(2,2)[b]}
  \put(0,0){\oval[0](1,1)}
\end{picture}
```

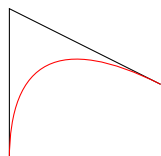


9.5. Bézier-görbék

■ `\qbezier[$\langle n \rangle$]($\langle x1 \rangle$, $\langle y1 \rangle$)($\langle x2 \rangle$, $\langle y2 \rangle$)($\langle x3 \rangle$, $\langle y3 \rangle$)`

Rajzol egy másodfokú Bézier-görbét az adott koordinátájú kontrollpontokkal. Ha $\langle n \rangle$ értéke 0, vagy nincs megadva, akkor folytonos vonalat húz, ellenkező esetben csak $\langle n \rangle$ darab pontot ábrázol.

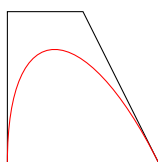
```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(0,0)(0,2)(2,1)
  \color{red}
  \qbezier(0,0)(0,2)(2,1)
\end{picture}
```



■ `\cbezier[$\langle n \rangle$]($\langle x1 \rangle, \langle y1 \rangle$)($\langle x2 \rangle, \langle y2 \rangle$)($\langle x3 \rangle, \langle y3 \rangle$)($\langle x4 \rangle, \langle y4 \rangle$)`

Rajzol egy harmadfokú Bézier-görbét az adott koordinátájú kontrollpontokkal. Ha $\langle n \rangle$ értéke 0, vagy nincs megadva, akkor folytonos vonalat húz, ellenkező esetben csak $\langle n \rangle$ darab pontot ábrázol.

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(0,0)(0,2)(1,2)(2,0)
  \color{red}
  \cbezier(0,0)(0,2)(1,2)(2,0)
\end{picture}
```



9.6. Útvonalak

Olyan útvonalakat is megadhatunk és megrajzolhatunk, amelyek szakaszokból, kör-ívekből és másodfokú Bézier-görbékből áll.

■ `\moveto($\langle x \rangle, \langle y \rangle$)`

Az első útvonalelemnek az $(\langle x \rangle, \langle y \rangle)$ koordinátájú pont lesz a referenciapontja.

■ `\lineto($\langle x \rangle, \langle y \rangle$)`

A referenciapontból az $(\langle x \rangle, \langle y \rangle)$ koordinátájú pontba húz egy szakaszt. A következő útvonalelemnek az $(\langle x \rangle, \langle y \rangle)$ koordinátájú pont lesz a referenciapontja.

■ `\curveto($\langle x2 \rangle, \langle y2 \rangle$)($\langle x3 \rangle, \langle y3 \rangle$)($\langle x4 \rangle, \langle y4 \rangle$)`

A referenciapont és az adott három pont, mint kontrollpontok segítségével húz egy harmadfokú Bézier-görbét. A következő útvonalelemnek az $(\langle x4 \rangle, \langle y4 \rangle)$ koordinátájú pont lesz a referenciapontja.

■ `\circlearc[$\langle n \rangle$]{ $\langle x \rangle$ }{ $\langle y \rangle$ }{ $\langle sugár \rangle$ }{ $\langle szög1 \rangle$ }{ $\langle szög2 \rangle$ }`

Ha $\langle n \rangle$ értéke 0 (ez az alapérték), akkor húz egy $(\langle x \rangle, \langle y \rangle)$ középpontú $\langle sugár \rangle$ egység sugarú körívet $\langle szög1 \rangle$ -től $\langle szög2 \rangle$ -ig, majd a referenciapontot és a körív kezdőpontját összeköti egy szakasszal. A következő útvonalelemnek a referenciapontja a körív végpontja lesz.

Ha az útvonalnak ez az első eleme, akkor $\langle n \rangle$ helyére írja az 1 számot. Ebben az esetben a `\moveto` parancs elhagyható. Ekkor az útvonal kezdőpontja a körív kezdőpontja, míg a következő útvonalelemnek a referenciapontja a körív végpontja lesz.

Ha $\langle n \rangle$ értéke 2 akkor úgy módosul az útvonal, hogy a csatlakozási pontban ne legyen törés.

■ `\closepath`

Az útvonal kezdő és végpontját összeköti egy szakasszal.

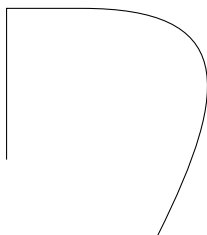
■ `\strokepath`

Megrajzolja a korábban meghatározott útvonalat.

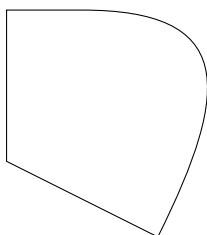
■ `\fillpath`

Kitölti az adott útvonallal határolt síkidomot.

```
\setlength{\unitlength}{1cm}
\begin{picture}(3,3)
  \moveto(0,1)
  \lineto(0,3)
  \lineto(1,3)
  \curveto(3,3)(3,2)(2,0)
  \strokepath
\end{picture}
```



```
\setlength{\unitlength}{1cm}
\begin{picture}(3,3)
  \moveto(0,1)
  \lineto(0,3)
  \lineto(1,3)
  \curveto(3,3)(3,2)(2,0)
  \closepath
  \strokepath
\end{picture}
```



```

\setlength{\unitlength}{1cm}
\begin{picture}(3,3)
  \color{red}
  \moveto(0,1)
  \lineto(0,3)
  \lineto(1,3)
  \curveto(3,3)(3,2)(2,0)
  \fillpath
\end{picture}

```



```

\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \circlearc[1]{0}{0}{2}{0}{90}
  \fillpath
\end{picture}

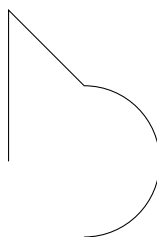
```



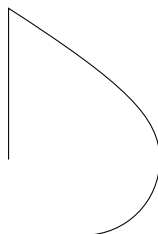
```

\setlength{\unitlength}{1cm}
\begin{picture}(3,3)
  \moveto(0,1)
  \lineto(0,3)
  \circlearc{1}{1}{1}{90}{-90}
  \strokepath
\end{picture}

```



```
\setlength{\unitlength}{1cm}
\begin{picture}(3,3)
  \moveto(0,1)
  \lineto(0,3)
  \circlearc[2]{1}{1}{1}{90}{-90}
  \strokepath
\end{picture}
```



9.7. Vonalak végeinek és útvonalak csatlakozási pontjainak stílusa

`\buttcap`

Alapértelmezett végpontstílus.

```
\begin{picture}(100,0)
  \linethickness{10pt}
  \put(0,0){\line(1,0){100}}
  \linethickness{.4pt}
  \put(0,0){\color{red}\line(1,0){100}}
\end{picture}
```



`\roundcap`

A végponthoz egy félkört illeszt.

```
\begin{picture}(100,0)
  \linethickness{10pt}
  \put(0,0){\roundcap\line(1,0){100}}
  \linethickness{.4pt}
  \put(0,0){\buttcap\color{red}\line(1,0){100}}
\end{picture}
```



`\squarecap`

A végponthoz egy fél négyzetet illeszt.

```

\begin{picture}(100,0)
  \linethickness{10pt}
  \put(0,0){\squarecap\line(1,0){100}}
  \linethickness{.4pt}
  \put(0,0){\buttcap\color{red}\line(1,0){100}}
\end{picture}

```



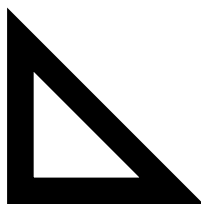
`\miterjoin`

Alapértelmezett csatlakozás.

```

\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \linethickness{10pt}
  \moveto(0,0)
  \lineto(0,2)
  \lineto(2,0)
  \closepath
  \strokepath
\end{picture}

```



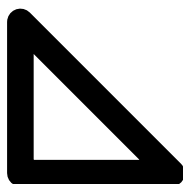
`\roundjoin`

Lekerekített csatlakozás.

```

\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \linethickness{10pt}
  \roundjoin
  \moveto(0,0)
  \lineto(0,2)
  \lineto(2,0)
  \closepath
  \strokepath
\end{picture}

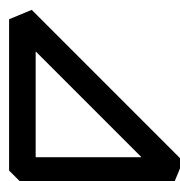
```



■ `\beveljoin`

Tompaszögű csatlakozás.

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \linethickness{10pt}
  \beveljoin
  \moveto(0,0)
  \lineto(0,2)
  \lineto(2,0)
  \closepath
  \strokepath
\end{picture}
```

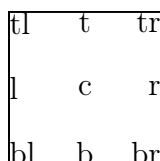


9.8. Betűk elhelyezése ábrában

■ `\framebox(<x>,<y>)[<pozíció>]{<szöveg>}`

Egy $\langle x \rangle$ egység széles és $\langle y \rangle$ egység magas bekeretezett doboz jön létre, melynek a bal alsó sarkában található a referenciapont. A $\langle szöveg \rangle$ ebben jelenik meg úgy pozicionálva, ahogy azt a $\langle pozíció \rangle$ opció megadja. A $\langle pozíció \rangle$ lehetséges értékei: **c** (alapérték), **t**, **b**, **l**, **r**, **tl**, **tr**, **br**, **bl**, melyek jelentését következő példán szemléltetjük:

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \put(0,0){\framebox(2,2)[t]{t}}
  \put(0,0){\framebox(2,2)[b]{b}}
  \put(0,0){\framebox(2,2)[l]{l}}
  \put(0,0){\framebox(2,2)[r]{r}}
  \put(0,0){\framebox(2,2)[tl]{tl}}
  \put(0,0){\framebox(2,2)[tr]{tr}}
  \put(0,0){\framebox(2,2)[br]{br}}
  \put(0,0){\framebox(2,2)[bl]{bl}}
  \put(0,0){\framebox(2,2){c}}
\end{picture}
```



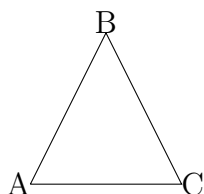
■ `\makebox(<x>,<y>)[<pozíció>]{<szöveg>}`

Pontosan úgy működik, mint a `\framebox` parancs, csak a doboz nincs bekeretezve.

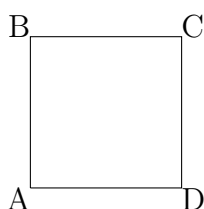
```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \put(0,0){\makebox(2,2)[t]{t}}
  \put(0,0){\makebox(2,2)[b]{b}}
  \put(0,0){\makebox(2,2)[l]{l}}
  \put(0,0){\makebox(2,2)[r]{r}}
  \put(0,0){\makebox(2,2)[tl]{tl}}
  \put(0,0){\makebox(2,2)[tr]{tr}}
  \put(0,0){\makebox(2,2)[br]{br}}
  \put(0,0){\makebox(2,2)[bl]{bl}}
  \put(0,0){\makebox(2,2){c}}
\end{picture}
```

tl	t	tr
l	c	r
bl	b	br

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(0,0)(1,2)(2,0)(0,0)
  \put(0,0){\makebox(0,0)[r]{A}}
  \put(1,2){\makebox(0,0)[b]{B}}
  \put(2,0){\makebox(0,0)[l]{C}}
\end{picture}
```



```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(0,0)(0,2)(2,2)(2,0)(0,0)
  \put(0,0){\makebox(0,0)[tr]{A}}
  \put(0,2){\makebox(0,0)[br]{B}}
  \put(2,2){\makebox(0,0)[bl]{C}}
  \put(2,0){\makebox(0,0)[tl]{D}}
\end{picture}
```

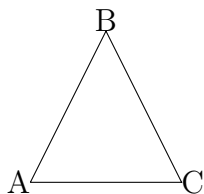


9.9. Koordináta megadása hosszmérettel

Az eddigiekben láthattuk, hogy először megadtuk a koordináta-rendszerünkben az egység hosszát, majd minden koordináta ebben az egységben volt megadva. Ha valamiért szeretne konkrét hosszt is beírni, akkor a `pict2e` és `xcolor` csomagok után még töltsse be a `picture` csomagot is.

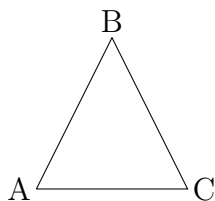
A következő esetben például a betűk túl közel vannak a háromszög csúcsaihoz:

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(0,0)(1,2)(2,0)(0,0)
  \put(0,0){\makebox(0,0)[r]{A}}
  \put(1,2){\makebox(0,0)[b]{B}}
  \put(2,0){\makebox(0,0)[l]{C}}
\end{picture}
```



Ilyenkor módosítsa következőképpen a kódot:

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
  \polyline(0,0)(1,2)(2,0)(0,0)
  \put(-2pt,0){\makebox(0,0)[r]{A}}
  \put(1,2\unitlength+2pt){\makebox(0,0)[b]{B}}
  \put(2\unitlength+2pt,0){\makebox(0,0)[l]{C}}
\end{picture}
```



10. fejezet

Táblázatok

A táblázatok elkészítése az egyik legbonyolultabb feladat a \LaTeX -ben. Nem tárgyaljuk általánosan az ide vonatkozó parancsokat, csak példákon keresztül tekintjük át a lehetőségeket a teljesség igénye nélkül.

A TeXstudio táblázatvarázslója illetve a [LaTeX Tables Generator](#) weblap sokat segít a táblázatok vizuális szerkesztésében.

10.1. Példatáblázatok

Kezdjük egy egyszerű példával:

```
\begin{tabular}{lrrr}
Budapest & 7:00 & 9:30 & 13:15\\
Dömsöd & 7:58 & 10:40 & 14:38\\
\end{tabular}
```

Budapest	7:00	9:30	13:15
Dömsöd	7:58	10:40	14:38

Tehát táblázat a `tabular` környezettel készíthető. Ennek paraméterében kell megadni, hogy hány oszlop van, és a tartalmuk hogyan legyen igazítva. Az előző példában az `lrrr` azt jelenti, hogy 4 oszlop van, az első balra (`l` mint left), a többi 3 pedig jobbra (`r` mint right) legyen igazítva. Ha egy oszlopot középre akar igazítani, akkor azt a `c` (mint center) betűvel jelezze. A `&` az ún. tabulátor jel, ami két oszlop elválasztását jelzi. A `\\` sortörést jelöl. A táblázatba vonalakat is húzhat:

```
\begin{tabular}{|l|rrr|}
\hline
Budapest & 7:00 & 9:30 & 13:15\\
\cline{2-4}
Dömsöd & 7:58 & 10:40 & 14:38\\
\hline
\end{tabular}
```

Budapest	7:00	9:30	13:15
Dömsöd	7:58	10:40	14:38

Ahol függőleges vonalat akar húzni, oda a `tabular` környezet paraméterében rakjon `|` jelet az **AltGr** + **W** gombokkal. Ahová vízszintes vonalat akar húzni, oda a `tabular`

környezetben tegyen `\hline` parancsot. Ha egy vízszintes vonalat nem akar teljesen meghúzni, csak mondjuk a 2. oszloptól a 4. oszlopig, akkor `\hline` helyett használjon `\cline{2-4}` parancsot. Több `\cline` is írható egymásután:

```
\begin{tabular}{|c|c|c|c|}
\hline
1 & 2 & 3 & 4\\
\cline{1-1}
\cline{3-4}
5 & 6 & 7 & 8\\
\hline
\end{tabular}
```

1	2	3	4
5	6	7	8

A következő példában azt mutatjuk meg, hogyan lehet szabályozni, hogy mi történjen két oszlop között:

```
\begin{tabular}{|@{\ 1\,}l@{ = }r@{,}l@{\,mm }|}
\hline
pont & 0 & 35\\
pica & 4 & 22\\
inch & 25 & 4\\
\hline
\end{tabular}
```

1 pont =	0,35 mm
1 pica =	4,22 mm
1 inch =	25,4 mm

A `tabular` környezet `@{...}` opciójában lehet megadni, hogy egy cella elé vagy után mi kerüljön. A `@{}` azt eredményezi, hogy nincs semmi, még térköz sem:

```
\begin{tabular}{@{}lrrr@{}}
\hline
Budapest & 7:00 & 9:30 & 13:15\\
Dömsöd & 7:58 & 10:40 & 14:38\\
\hline
\end{tabular}
```

Budapest	7:00	9:30	13:15
Dömsöd	7:58	10:40	14:38

```
\begin{tabular}{@{}r@{}r@{}}
&12345\\
& 1234\\
+& 123\\
\hline
&13702\\
\end{tabular}
```

12345
1234
+ 123

13702

Az `array` csomag definiál egy `>{\dots}` opciót is, mellyel az oszlop formázásának lehetőségeit bővíti:

```
\begin{tabular}{c>{\bfseries}cc}
egy & kettő & három\\
egy & kettő & három\\
egy & kettő & három\\
\end{tabular}
```

egy	kettő	három
egy	kettő	három
egy	kettő	három

Cellákat vízszintesen is összevonhat a

`\multicolumn{<cellaszám>}{<cellaforma>}{<szöveg>}`

paranccsal. A `<cellaszám>` az összevont cellák számát jelenti. A `<cellaforma>` az adott összevont cellára vonatkozó formázás, amely pontosan úgy történik, mint a `tabular` környezet paraméterében. Ez a parancs akkor is célravezető, ha nem összevonni akar, csak az adott cellának a formázását akarja megváltoztatni, az általánosan megadotthoz képest. Ilyenkor a `<cellaszám>` értelemszerűen 1. Például

```
\begin{tabular}{|l|rr|}
\cline{2-3}
\multicolumn{1}{|l|}{&}\multicolumn{2}{c|}{Év}\\
\cline{2-3}
\multicolumn{1}{|l|}{&}\multicolumn{1}{c}{2002}&\multicolumn{1}{c}{2003}\\
\hline
Jövedelem (Ft) & 994\,000 & 1\,231\,500\\
Adó (Ft) & 165\,000 & 194\,950\\
\hline
\end{tabular}
```

	Év	
	2002	2003
Jövedelem (Ft)	994 000	1 231 500
Adó (Ft)	165 000	194 950

Vegyük észre, hogy az „Év”-re ráhúzódik a vonal. A szöveg feletti térköz például 2 pt-tal megnövelhető az `array` csomag `\extrarowheight` parancsával a következőképpen:

`\setlength{\extrarowheight}{2pt} \in array`

Ezt írja az előző kód elé, és megkapja a következő javított táblázatot.

	Év	
	2002	2003
Jövedelem (Ft)	994 000	1 231 500
Adó (Ft)	165 000	194 950

Cellák függőleges összevonását a következő paranccsal teheti meg:

```
\multirow{<cellaszám>}*{<szöveg>} ∈ multirow
\multirow{<cellaszám>}{<szélesség>}{<szöveg>} ∈ multirow
```

Például

```
\begin{tabular}{|l|c|}
\hline
\multirow{2}*{Egysoros szöveg} & 1\\
& 2\\
\hline
\multirow{3}{3cm}{3\,cm széles szöveg törve} & 3\\
\cline{2-2}
& 4\\
\cline{2-2}
& 5\\
\hline
\end{tabular}
```

Egysoros szöveg	1
	2
3 cm széles szöveg törve	3
	4
	5

A következő példában azt mutatjuk meg, hogyan lehet beállítani az egyes oszlopok szélességét.

```
\begin{tabular}{|p{2cm}|p{2cm}|p{2cm}|p{2cm}|}
\hline
Ez egy kis tábla, jó lesz vigyázni.\rightskip\fill &
Ez egy kis tábla, jó lesz vigyázni.\leftskip\fill &
Ez egy kis tábla, jó lesz vigyázni.\leftskip\fill\rightskip\fill &
Ez egy kis tábla, jó lesz vigyázni.\\
\hline
\end{tabular}
```

A következő kód az előzővel azonos hatású az `array` csomag betöltésével.

```
\begin{tabular}{|>\raggedright\arraybackslashp{2cm}
|>\raggedleft\arraybackslashp{2cm}
|>\centering\arraybackslashp{2cm}
|p{2cm}|}
\hline
Ez egy kis tábla, jó lesz vigyázni.&
Ez egy kis tábla, jó lesz vigyázni.&
Ez egy kis tábla, jó lesz vigyázni.&
Ez egy kis tábla, jó lesz vigyázni.\\
```

```
\hline
\end{tabular}
```

Ez egy kis tábla, jó lesz vigyázni.	Ez egy kis tábla, jó lesz vigyázni.	Ez egy kis tábla, jó lesz vigyázni.	Ez egy kis tábla, jó lesz vigyázni.
--	--	--	--

A következő kód a cella tartalmának függőleges középre igazítására egy példa. A kód az `array` csomag betöltése után működik.

```
\begin{tabular}{|>{\raggedright\arraybackslash}m{16mm}
                |>{\raggedleft\arraybackslash}m{16mm}
                |>{\centering\arraybackslash}m{16mm}
                |m{22mm}|}
\hline
szöveg szöveg szöveg &
szöveg szöveg &
szöveg &
sz ö v e g szöveg szöveg\\
\hline
\end{tabular}
```

szöveg szöveg szöveg	szöveg szöveg	szöveg	sz ö v e g szöveg szö- veg
----------------------------	------------------	--------	----------------------------------

Az előző kódban az oszloptípusokat előre definiálhatja a `\newcolumntype` \in `array` paranccsal például így:

```
\newcolumntype{L}{>{\raggedright\arraybackslash}m{16mm}}
\newcolumntype{R}{>{\raggedleft\arraybackslash}m{16mm}}
\newcolumntype{C}{>{\centering\arraybackslash}m{16mm}}
\newcolumntype{M}{m{22mm}}
```

Ezután az előző táblázat már így is kiszedhető:

```
\begin{tabular}{|L|R|C|M|}
\hline
szöveg szöveg szöveg &
szöveg szöveg &
szöveg &
sz ö v e g szöveg szöveg\\
\hline
\end{tabular}
```

A következő példában a táblázat teljes szélessége van megadva (5 cm).

```
\begin{tabular*}{5cm}{|l@{ -- }l@{\extracolsep{\fill}}r|}
\hline
FTC & MTK & 1:1\\
Vasas & ETO & 0:0\\
\hline
\end{tabular*}
```

FTC – MTK	1:1
Vasas – ETO	0:0

Itt a `@{\extracolsep{\fill}}` az utolsó oszlopot kinyomja az 5 cm széles táblázat széléig.

A táblázatok vonalai alapesetben 0,4 pt vastagok. Ezt átállíthatja az `array` csomag betöltése után a következő kóddal például 1 pt-ra:

```
\setlength{\arrayrulewidth}{1pt}
\begin{tabular}{|c|c|}
\hline
A & B\\
\hline
C & D\\
\hline
\end{tabular}
```

A	B
C	D

Az `array` csomaggal egyetlen függőleges vonalnak a vastagságát is átállíthatja:

```
\begin{tabular}{|c!{\vrule width 2pt}c|}
A & B\\
C & D
\end{tabular}
```

A	B
C	D

Oszlopokat színezzhet a `colortbl` csomaggal:

```
\begin{tabular}{>{\columncolor{cyan}}c
>{\color{red}\columncolor{green}}c
>{\columncolor{yellow}}c}
egy & kettő & három\\
egy & kettő & három\\
egy & kettő & három\\
\end{tabular}
```

egy	kettő	három
egy	kettő	három
egy	kettő	három

A `colortbl` csomaggal sorokat színezzhet:

```
\begin{tabular}{ccc}
\rowcolor{cyan}    egy & kettő          & három\\
\rowcolor{green}   egy & \color{red}kettő & három\\
\rowcolor{yellow}  egy & kettő          & három\\
\end{tabular}
```


egy	kettő	három
egy	kettő	három
egy	kettő	három

A `colortbl` csomaggal megadhatja egy cella háttérszínét:

```
\begin{tabular}{|c|c|c|}
\hline
egy & kettő & három\\
\hline
egy & kettő & \cellcolor{red} három\\
\hline
\end{tabular}
```

egy	kettő	három
egy	kettő	három

Egy táblázatot váltott színű sorokkal jeleníthet meg, ha az `xcolor` csomagot `table` opcióval tölti be.

```
\rowcolors{1}{gray!30}{gray!50}
\begin{tabular}{ccc}
egy & kettő & három\\
egy & kettő & három\\
egy & kettő & három\\
\end{tabular}
```

egy	kettő	három
egy	kettő	három
egy	kettő	három

A `\rowcolors` első argumentuma azt adja meg, hogy hányadik sortól kezdje a színezést, a másik két argumentum pedig a színeket adja meg.

10.2. Hosszú táblázatok

Ha a táblázat olyan hosszú, hogy nem fér ki egy oldalon, akkor használja a `longtable` csomagot.

```
\begin{longtable}[<pozíció>]{<oszlopok>} ∈ longtable
\endfirsthead ∈ longtable
\endhead ∈ longtable
\endfoot ∈ longtable
\endlastfoot ∈ longtable
```

A `<pozíció>` lehet `r`, `l`, `c` (alapérték `c`). Ezek rendre jobbra, balra illetve középre helyezik a táblázatot. Az `<oszlopok>` a szokásos oszlopformázó utasításokat tartalmazzák. Például

```
\begin{longtable}{ll}
\caption{A táblázat címe} % táblázat címe
\label{longtable-minta} \\ % kereszthivatkozás esetén
AAA & BBB \\ \hline % fejléc
\endfirsthead
```

```

CCC & DDD \\ \hline % táblázattörés utáni fejléc
\endhead
\hline\multicolumn{2}{r}{folyt. a köv. oldalon} % törésnél információ
\endfoot
\hline
\endlastfoot
% ide jönnek a táblázat sorai
\end{longtable}

```

10.3. Kiadói minőségű táblázatok

Az előzőekben tárgyalt táblázatok hagyományos szerkezetűek voltak. Viszont a kiadói szintű táblázatok tipográfiája egy kicsit más. A legfontosabb különbségek:

- A táblázat tetejére és aljára vastagabb vonal kell, mint a köztesek.
- A táblázat két szélén ne legyenek extra térközök, melyek a formátumvezérlő két szélére írt egy-egy @{} paranccsal megoldható.
- Nincsenek függőleges vonalak.

Mindezek a `booktabs` csomaggal oldhatók meg. Erre nézzünk most egy példát.

```

\begin{tabular}{@{}lrr@{}}
\toprule
& \multicolumn{2}{c}{Év} \\
\cmidrule{2-3}
& \multicolumn{1}{c}{2002} & \multicolumn{1}{c}{2003} \\
\midrule
Jövedelem (Ft) & 775\,000 & 1\,166\,500 \\
Adó (Ft) & 165\,000 & 194\,950 \\
\bottomrule
\end{tabular}

```

	Év	
	2002	2003
Jövedelem (Ft)	775 000	1 166 500
Adó (Ft)	165 000	194 950

Az előző kódban a `\cmidrule{2-3}` sor helyett `\cmidrule(lr){2-3}` beírva:

	Év	
	2002	2003
Jövedelem (Ft)	775 000	1 166 500
Adó (Ft)	165 000	194 950

10.4. Táblázatok alapvonalhoz igazítása

A `tabular` és `tabular*` környezeteknek nem csak paramétereik, hanem opcióik is vannak. Ezekben lehet megadni az alapvonalhoz viszonyított pozíciójukat:

```
\begin{tabular}[<opció>]{<paraméterek>}
\begin{tabular*}{<szélesség>}[<opció>]{<paraméterek>}
```

Opció nélkül (pontosabban alapopcióval) az igazítás középre történik:

```
szöveg
\begin{tabular}{|cc|}
\hline X&X\\X&X\\X&X\\X&X\\\hline
\end{tabular}
szöveg
```

szöveg	X X	szöveg
	X X	
	X X	
	X X	

Ha az `<opció>` `t` (mint top) akkor a táblázat teteje kerül az alapvonalhoz:

```
szöveg
\begin{tabular}[t]{|cc|}
\hline X&X\\X&X\\X&X\\X&X\\\hline
\end{tabular}
szöveg
```

szöveg	X X	szöveg
	X X	
	X X	
	X X	

Ha az `<opció>` `b` (mint bottom) akkor a táblázat alja kerül az alapvonalhoz:

```
szöveg
\begin{tabular}[b]{|cc|}
\hline X&X\\X&X\\X&X\\X&X\\\hline
\end{tabular}
szöveg
```

szöveg	X X	szöveg
	X X	
	X X	
	X X	

Az utóbbi két illesztésnél zavaró lehet, hogy a szöveg alapvonala nem esik egybe a táblázat utolsó illetve első sorának alapvonalával. Ezen lehet segíteni az `array` csomag `\firsthline` és `\lasthline` parancsaival:

```
\begin{tabular}[t]{|cc|}
\firsthline X&X\\X&X\\X&X\\X&X\\\hline
\end{tabular}
```

```

szöveg
\begin{tabular}[b]{|cc|}
\hline X&X\\X&X\\X&X\\X&X\\\lasthline
\end{tabular}

```

X	X
X	X
X	X
X	X

szöveg

X	X
X	X
X	X
X	X

11. fejezet

Objektumok úsztatása

A táblázatok, képek beillesztését már az eddigiek alapján is el tudjuk végezni. De előfordulhat, hogy az adott oldalon már nem fér el, és a következő oldalra való áthelyezésével az oldal alja telítetlen marad. Ennek megoldására született az úgynevezett „úsztatás”. Ez azt jelenti, hogy a problémás objektumot áthelyezi egy általunk megadott helyre (az aktuális oldal aljára, tetejére, vagy külön oldalra), az oldalt pedig telíti a soron következő szöveggel.

11.1. Képek és táblázatok úsztatása

Képek úsztatására a `figure`, míg táblázatok úsztatására a `table` környezet használható. Ezen környezetek opciói:

- h** Maradjon helyben, ha lehetséges.
- t** Az aktuális oldal tetejére kerüljön.
- b** Az aktuális oldal aljára kerüljön.
- p** Külön oldalra kerüljön.
- !** Ekkor megszűnnek bizonyos korlátozások, így az objektum nagyobb eséllyel kerül arra helyre, ahová szeretnénk.

Opciónak ezen betűk bármilyen kombinációja használható. A betűk sorrendje mindegy, ugyanis az objektum a legelső olyan helyre kerül, amelyet az opció megenged. Ez alól csak a **h** kivétel, aminek mindennel szemben elsőbbsége van.

Nézzünk néhány példát.

```
\begin{figure}
\centering
\includegraphics{fig}
\end{figure}
```

Mivel itt nem adtunk meg opciót, így az alapérték érvényesül, mely `tbp`. Ez azt jelenti, hogy ebben az esetben a képet először megpróbálja az oldal tetejére, ha oda nem kerülhet, akkor az oldal aljára, ha oda sem, akkor külön lapra tenni.

```
\begin{figure}[th]
\centering
\includegraphics{fig}
\end{figure}
```

Ebben az esetben a képet először megpróbálja helybenhagyni, de ha oda nem kerülhet, akkor az oldal tetejére teszi.

```
\begin{figure}[ht!]  
\centering  
\includegraphics{fig}  
\end{figure}
```

A képet bizonyos korlátozások feloldása mellett próbálja helyben tartani, de ha oda nem kerülhet, akkor az oldal tetejére teszi. Az esetek nagy részében a saját dokumentumaimban ezt az opciót szoktam alkalmazni. Ilyen esetekben célszerűnek tűnik a `tbp` alapopciót átállítani `ht!` értékre. Ezt például `figure` környezet esetén így lehet megtenni:

```
\makeatletter\def\fps@figure{ht!}\makeatother
```

vagy

```
\floatplacement{figure}{ht!} ∈ float
```

Előfordulhat, hogy egy úszó objektum a lap tetejére kerülve az előző téma sorai közé kerül, ami nem szerencsés. Ilyenkor használja a

```
\suppressfloats[⟨opció⟩]
```

parancsot. Az ezután következő úszó objektum nem jelenhet meg az oldal *⟨opció⟩* szerinti helyén, amely `t` vagy `b` lehet. Ha az opció nincs megadva, akkor egyik helyen sem jelenhet meg úszó objektum. A parancs hatása csak egy oldalra korlátozódik és csak a forráskódban következő úszó objektumra vonatkozik.

Ha azt akarja, hogy egy adott pontig az addig elindított úsztatások befejeződjenek, akkor ott használja a

```
\FloatBarrier ∈ placeins
```

parancsot. Később ismertetjük a hosszabb művek szakaszokkal (section) való tagolását. Ekkor szerencsés lenne, ha a szakaszokon belül minden úsztatás lezárulna. Ezt valósítja meg a `placeins` csomag `section` opciója. A fejezetek (chapter) esetén ez nem gond, mert minden fejezet `\clearpage` paranccsal zárul, ami megjeleníti az addig még függőben maradt úsztatásokat.

11.2. Úsztatott objektumok címkézése

Sokszor előfordul, hogy a képekre, táblázatokra hivatkozni szeretnénk. Ilyenkor célszerű az objektumnak automatikus sorszámot és címet adni. Másrészt ha ezen objektumokból nagyon sok van, akkor az áttekinthetőség miatt célszerű ezen címeket táblázat- illetve ábrajegyzékben szerepeltetni oldalszám feltüntetésével, hasonlóan a tartalomjegyzékhez. Ezen feladatok elvégzésére szolgál a

```
\caption[⟨jegyzékbe kerülő cím⟩]{⟨cím⟩}
```

parancs. Az opció alapértéke megegyezik a *⟨cím⟩*-mel. Például

```
\begin{figure}[ht!]  
\centering  
\includegraphics[width=3cm]{lion}  
\caption{A \TeX\ szimbóluma (tervezte Duane Bibby)}\label{fig-lion}  
\end{figure}
```

■ `\Aref{fig-lion}.`~ábrán látható `\dots`

Ennek hatására a képet megjeleníti középen és felcímkézi. A címkébe aszerint kerül „ábra” vagy „táblázat” felirat, hogy `figure` vagy `table` környezetbe raktuk a `\caption` parancsot. A sorszám automatikus. A megadott cím bekerül a megfelelő jegyzékbe.



1. ábra. A TeX szimbóluma (tervezte Duane Bibby)

Az 1. ábrán látható ...

Ha nem akar számozást, csak címet, akkor használja a következő parancsot:

■ `\caption*{<cím>}` ∈ `caption`

Ha nem akar címet adni, elég a számozás, akkor a `caption` csomag használata mellett tegye ezt:

```
\begin{figure}[ht!]
\centering
\includegraphics[width=3cm]{lion}
\caption{}\label{fig-lion}
\end{figure}
```



1. ábra

Ha azt akarja, hogy a jegyzékbe más cím kerüljön mint a címkébe, akkor a jegyzékbe kerülő címet adja meg a `\caption` parancs opciójaként. Például

■ `\caption[A \TeX\ szimbóluma]{A \TeX\ szimbóluma (tervezte Duane Bibby)}`

A cím az előző példában azért jelent meg a kép alatt, mert a `\caption` parancsot a kép betöltése után hívtuk meg. Ha elé íránk, akkor a kép felett lenne a cím. Ha a `\caption` parancs kiadásának helyétől függetlenül például a táblázatok esetében mindig a táblázatok felett szeretné a címkét, akkor használja a

■ `\floatstyle{plaintop}\restylefloat{table}` ∈ `float`

kódot. Ha a címkék stílusát szeretné átalakítani, akkor használja a `caption` csomagot. Ennek részleteit nem írjuk le, a csomag dokumentációjában minden megtalálható. Ha ezeket a változtatásokat magyar nyelvű dokumentumban szeretné érvényesíteni, akkor a `magyar.ldf` fájl `defaults=hu-min` opciója mellett, fel kell venni a `longcaption=unchanged` opciót is:

■ `\PassOptionsToPackage{defaults=hu-min,longcaption=unchanged}{magyar.ldf}`



Videó: Képek és táblázatok

11.3. Saját úsztatott objektumok létrehozása

Alaphelyzetben a táblázatokat és az ábrákat tudjuk úsztatni saját címkével és jegyzékkel. De saját úsztató környezetet is definiálhatunk. Például szeretnénk grafikonokat készíteni. A környezet neve legyen `graf`, a címke legyen „grafikon” és a jegyzék címe legyen „Grafikonok jegyzéke”. Ekkor a következőt írja a preambulumba:

```
\DeclareCaptionType{graf}[grafikon][Grafikonok jegyzéke] ∈ caption
```

Ezután pontosan úgy használhatja a `graf` környezetet, mint a `table` vagy `figure` környezeteket.

11.4. Úsztatás mellőzése

Ha egy objektumot nem akarunk úsztatni, hanem mi szeretnénk a helyét „kisakkozni”, akkor az úsztató környezetnek használja a `H` opcióját, amely a `float` csomag betöltésével válik elérhetővé. Például

```
\begin{figure}[H]
\includegraphics{fig}
\caption{2002-es statisztika}\label{fig-2002stat}
\end{figure}
```

Ilyenkor az objektum biztosan ott jelenik meg, ahol a kód szerint kell lennie. De így az oldalak telítettsége nem feltétlenül lesz megfelelő, ezért ez a megoldás sok kísérletezést igényel, vagyis nem kényelmes. Felmerül a kérdés, hogy ha valamit nem akarunk úsztatni, akkor miért rakjuk úsztató környezetbe. A válasz az, hogy a `\caption` parancs az úsztató környezetből tudja, hogy milyen címkét és sorszámot kell adnia. Ennek megoldására egy másik lehetőség a

```
\captionof{<környezet>}[<cím jegyzékben>]{<cím>} ∈ caption
```

használata, amit nem kell úsztatott környezetbe rakni, mert a címke típusát és számát a `<környezet>` megadása miatt tudja. Például az előző kóddal azonos hatású a következő:

```
\begin{center}
\includegraphics{fig}
\captionof{figure}{2002-es statisztika}\label{fig-2002stat}
\end{center}
```

11.5. Objektumok körbefuttatása szöveggel

Ezt képek esetében a `floatflt` csomag `floatingfigure` környezetével lehet megtenni. Ez ábrákra lett kitalálva, de a `\captionof ∈ caption` paranccsal táblázatokra, vagy bármely saját úsztatott objektumra is alkalmazható. Lássunk egy példát:

```
\begin{floatingfigure}[r]{4cm}
\centering
```



```
\includegraphics[width=3cm]{lion}
\caption{A \TeX\ szimbóluma}\label{fig-lion}
\end{floatingfigure}
```

Opciók:

- r** jobbra helyezi az objektumot,
- l** balra helyezi az objektumot,
- p** (alapopció) páratlan oldalon jobbra, páros oldalon pedig balra, azaz a külső margóhoz helyezi az objektumot.

Ha azt akarja, hogy az alapopció **r** legyen, akkor a `floatflt` csomagot `rflt` opcióval töltsse be. Ha azt akarja, hogy az alapopció **l** legyen, akkor a `floatflt` csomagot `lflt` opcióval töltsse be. A belső margóhoz való helyezéshez nem rendeltek opciót, de a következő kóddal ez is megoldható:

```
\begin{floatingfigure}{4cm}
\ifodd\value{page}\global\oddpagesfalse\else\global\oddpagetrue\fi
\centering
\includegraphics[width=3cm]{lion}
\caption{A \TeX\ szimbóluma}\label{fig-lion}
\end{floatingfigure}
```

A `floatingfigure` környezet paraméterének megadott `4cm` egy olyan doboz szélessége, melybe a `\centering` parancs miatt a képet középre teszi. A környező szöveg ettől a doboztól oldalról 12 pt távolságra lesz. Ha ezt át akarja állítani például 5 mm-re, akkor adja ki a

```
\setlength{\figgutter}{5mm} ∈ floatflt
```

parancsot. A `floatingfigure` környezet csak akkor működik, ha ír utána szöveget, hiszen ezzel lesz az objektum körbefuttatva. Ez a szöveg új bekezdésnek számít. Ha ezt nem akarja, akkor írjon elé `\noindent` parancsot.

A `floatflt` csomagnak létezik három hibája, amire érdemes odafigyelni. Bizonyos esetekben a `floatingfigure` környezetbe zárt objektum vízszintesen nem jól pozicionál. Ilyenkor a `floatingfigure` környezet elé írja be a következő kódot:

```
\par\mbox{}\vspace{-\baselineskip}
```

A másik hiba, hogy ha a `floatingfigure` környezet után nincs szöveg, vagy az objektum nem fér ki az oldal alján, akkor az objektum nem jelenik meg a dokumentumban. Ez súlyos hibája a csomagnak, ezért erre különösen figyeljen.

A harmadik hiba a `floatflt` csomag `floatingtable` környezetével kapcsolatos. Ezzel táblázatokat tudunk körbefuttatni szöveggel, a szélesség megadása nélkül, mert azt a táblázat méretéből veszi át. Azonban sok esetben rosszul pozicionál a táblázat, mely egy egyszerű kóddal általánosan nem orvosolható. Így ennek használatát tanácsos kerülni.

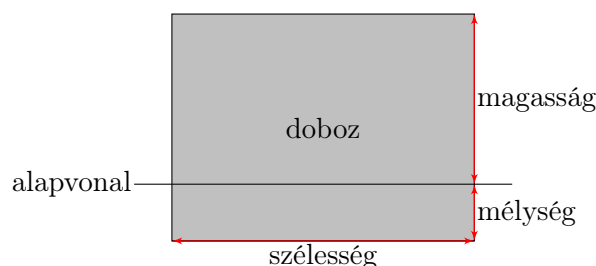
12. fejezet

Dobozok

A doboz a dokumentum olyan része, melynek a tartalma nem törhető el a sor végén vagy a lap alján, azaz sem függőlegesen, sem vízszintesen. Ilyenek az úszó objektumok, de doboz például egy betű vagy egy vonal is. Háromféle dobozt ismertetünk:

- Egysoros doboz: egysoros, balról jobbra feltöltődő doboz.
- Bekezdésdoboz: akár több sorból álló doboz.
- Vonaldoboz: állítható méretű vonal.

A doboz méreteire a következő szóhasználatot vezetjük be:



A magasság és mélység összegét teljes magasságnak nevezzük.

12.1. Egysoros dobozok

Egysoros doboz készítéséhez a következő parancs használható:

```
■ \makebox[⟨doboz szélessége⟩][⟨szöveg pozíciója⟩]{⟨szöveg⟩}
```

Ha a *⟨szöveg pozíciója⟩* **c**, akkor középre helyezi a szöveget a dobozban (alapopció), ha **l**, akkor balra, ha **r**, akkor jobbra és **s** esetén széthúzza/összenyomja a teljes dobozszélességre. Ha a szélességet és pozíciót nem adja meg, akkor a doboz szélessége a szöveg szélességével fog megegyezni:

```
■ \makebox{szöveg}
```

szöveg

Az így kapott „szöveg” szó nem elválasztható, hiszen a L^AT_EX dobozként kezeli. Egy másik példa:

```
\makebox[5cm][s]{szöveg szöveg}\\
\makebox[5cm][s]{s z ö v e g}
```

szöveg	szöveg
s z ö v e g	

Egysoros doboz be is keretezhető a következő paranccsal:

```
\framebox[⟨doboz szélessége⟩][⟨szöveg pozíciója⟩]{⟨szöveg⟩}
```

Ezt pontosan úgy kell használni, mint a `\makebox` parancsot. Például

```
\framebox{szöveg}\\
\framebox[5cm][s]{szöveg szöveg}\\
\framebox[5cm][s]{s z ö v e g}
```

szöveg
szöveg szöveg
s z ö v e g

A keret vonalvastagsága, mely alapesetben 0,4 pt, a következő paranccsal állítható be például 1 pt-ra:

```
\setlength{\fboxrule}{1pt}
```

A keret és a szöveg távolsága, mely alapesetben 3 pt, a következő paranccsal állítható be például 2 pt-ra:

```
\setlength{\fboxsep}{2pt}
```

Ha a `\makebox` parancsot opciók nélkül használja, akkor elég csak `\mbox` parancsot írni. Hasonlóan, ha a `\framebox` parancsot opciók nélkül használjuk, akkor csak `\fbox` parancsot kell írni:

```
\mbox{⟨szöveg⟩} = \makebox{⟨szöveg⟩}
\fbox{⟨szöveg⟩} = \framebox{⟨szöveg⟩}
```

Színes egysoros dobozok is előállíthatók. Ezeket néhány példán mutatjuk meg:

```
\colorbox{red}{szöveg} ∈ xcolor
```

szöveg

```
\colorbox[RGB]{128,0,128}{szöveg} ∈ xcolor
```

szöveg

```
\fcolorbox{red}{yellow}{szöveg} ∈ xcolor
```

szöveg

```
\fcolorbox[RGB]{0,64,128}{192,192,192}{szöveg} ∈ xcolor
```

szöveg

Ezeknél a keretet pontosan úgy lehet beállítani, mint a `\framebox` esetén.

A következő parancs egy olyan egysoros dobozt készít, amely az alapvonalától magasabban/alacsonyabban helyezkedik el:

```
\raisebox{⟨emelés⟩}{⟨szöveg⟩}
```

Például

```
AAA\raisebox{4pt}{BBB}CCC\raisebox{-4pt}{DDD}
```



Az `⟨emelés⟩`-ben használhatók még a `\width`, `\height`, `\depth` és `\totalheight` hosszúságparancsok is, melyek a `⟨szöveg⟩` által létrehozott doboz szélességét, magasságát, mélységét és teljes magasságát jelentik. Például

```
AAA\raisebox{0.5\height}{BBB}CCC\raisebox{-\height}{DDD}
```



12.2. Bekezdésdobozok

Bekezdésdobozokba akár többsoros vagy több bekezdésnyi szöveget is rakhat a következő paranccsal illetve környezettel:

```
\parbox[⟨pozíció⟩][⟨magasság⟩][⟨szöveg pozíció⟩]{⟨szélesség⟩}{⟨szöveg⟩}
```

vagy

```
\begin{minipage}[⟨pozíció⟩][⟨magasság⟩][⟨szöveg pozíció⟩]{⟨szélesség⟩}
  ⟨szöveg⟩
\end{minipage}
```

`⟨pozíció⟩` azt szabályozza, hogy a doboz hogyan helyezkedjen el a környezet alapvonalához képest. Alapértéken a doboz közepe az illeszkedési pont, **t** esetén a doboz felső sorának alapvonala, illetve **b** esetén az alsó sor alapvonala.

`⟨magasság⟩` a doboz teljes magassága.

`⟨szöveg pozíció⟩` akkor használható, ha a magasság is meg van adva. Azt adja meg, hogy a szöveg a dobozban függőlegesen hogyan helyezkedjen el. Értékei **t**, **b**, **c**, **s**, melyek rendre a szöveget a doboz tetejéhez, aljához, függőlegesen középre rakja, illetve széthúzza a doboz teljes magasságában. Az **s** opció csak akkor működik, ha a szövegbe rugalmas függőleges térközöket rakunk (például `\medskip`).

`⟨szélesség⟩` a doboz szélessége.

`⟨szöveg⟩` a doboz tartalma.

Például

```
SZÖVEG
\begin{minipage}[t][2cm][s]{5cm}
  szöveg szöveg szöveg szöveg szöveg szöveg
\par\medskip szöveg szöveg szöveg szöveg
\end{minipage}
```

SZÖVEG szöveg szöveg szöveg szöveg
szöveg szöveg

szöveg szöveg szöveg szöveg

Egy bekezdésdobozt be is lehet keretezni, amihez nincs szükség újabb parancsra, hiszen a bekezdésdoboz berakható egy egysoros keretezett dobozba, mivel az már egy egységnek, doboznak számít:

SZÖVEG
\fbox{\begin{minipage}[t][2cm][s]{5cm}
szöveg szöveg szöveg szöveg szöveg szöveg
\par\medskip szöveg szöveg szöveg szöveg
\end{minipage}}

SZÖVEG szöveg szöveg szöveg szöveg
szöveg szöveg

szöveg szöveg szöveg szöveg

A `\parbox` parancsnak illetve a `minipage` környezetnek van egy kellemetlen tulajdonsága, amit az alábbi példán illusztrálunk:

\fbox{\begin{minipage}{6cm}
szöveg szöveg szöveg szöveg szöveg szöveg
\end{minipage}}

szöveg szöveg szöveg szöveg szö-
veg szöveg

Itt az adott betűtípus és -méret miatt a 6 cm szélesség nem optimális, így az első sorban a szóközök mérete túl nagy. Ennek a problémának egy lehetséges megoldása a `varwidth` környezet:

\begin{varwidth}[\langle pozíció \rangle][\langle magasság \rangle][\langle szöveg pozíció \rangle]{\langle szélesség \rangle} \in varwidth
\langle szöveg \rangle
\end{varwidth}

Ez pontosan úgy működik, mint a `minipage` környezet, de a doboz szélessége azt a `\langle szélesség \rangle` értékénél nem nagyobb maximális értéket vesz fel, amely esetén még optimális a tördelés. Az előző kódot például nézzük meg `varwidth` környezettel:

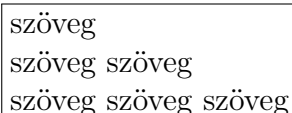
\fbox{\begin{varwidth}{6cm}
szöveg szöveg szöveg szöveg szöveg szöveg
\end{varwidth}}

szöveg szöveg szöveg szöveg szö-
veg szöveg

Itt már az első sorban megfelelő méretűek a szóközök, de ennek érdekében a doboz szélességét 6 cm-ről csökkenteni kellett egy kicsit.

A `varwidth` környezet akkor is használható, ha a töréspontokat mi adjuk meg, így a doboz szélessége nem ismert. Ekkor a $\langle széllesség \rangle$ helyére írja a `\textwidth` parancsot. Például

```
\fbox{\begin{varwidth}{\textwidth}
szöveg\\
szöveg szöveg\\
szöveg szöveg szöveg
\end{varwidth}}
```



12.3. Vonaldobozok

Vonaldobozokat a következő paranccsal készíthet:

```
\rule[\emelés]{\széllesség}{\magasság}
```

Ez egy $\langle széllesség \rangle$ szélességű és $\langle magasság \rangle$ magasságú téglalapot rajzol, melynek az alja az alapvonalától az $\langle emelés \rangle$ mértékével lesz feljebb. Például

```
xxxxx\rule[1ex]{2cm}{2mm}
```



```
x\rule[0.5ex]{3cm}{1pt}x
```



```
x\rule[-0.5ex]{3cm}{1pt}x
```



12.4. Dobozok egymásra helyezése

A következő kód a $\langle doboz1 \rangle$ és $\langle doboz2 \rangle$ dobozokat egymásra helyezi úgy, hogy a jobb széleik ugyanott lesznek.

```
\langle doboz1 \rangle \llap{\langle doboz2 \rangle}
```

Például

```
\mbox{xxxxxxxxxxxxxxxx}\llap{\rule[0.5ex]{2cm}{0.4pt}}
```



A következő kód a $\langle doboz1 \rangle$ és $\langle doboz2 \rangle$ dobozokat egymásra helyezi úgy, hogy a bal széleik ugyanott lesznek.

```
\mbox{} \rlap{\langle doboz1 \rangle}{\langle doboz2 \rangle}
```

Például

```
\mbox{}\rlap{\mbox{xxxxxxxxxxxxxxxx}}{\rule[0.5ex]{2cm}{0.4pt}}
```

```
xxxxxxxxxxxxxxxx
```

12.5. Dobozok forgatása

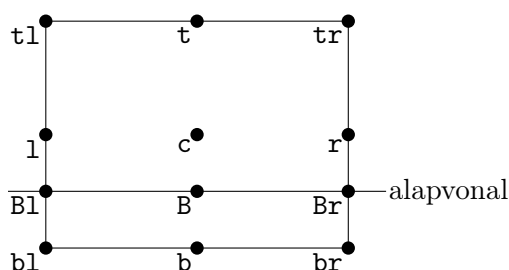
Dobozokat a következő paranccsal forgathat:

```
\rotatebox[origin=<centrum>]{<szög>}{<doboz>} ∈ graphicx
```

<doboz> az elforgatandó doboz. Ennek helyére egyszerű szöveg is kerülhet, amit ekkor dobozként kezel.

<szög> a forgatás szöge fokban. Pozitív érték esetén az óra járásával ellentétes irányban forgat.

<centrum> a forgatás középpontja, ami a **tl**, **t**, **tr**, **l**, **c**, **r**, **Bl**, **B**, **Br**, **bl**, **b**, **br** értékeket veheti fel (alapérték **Bl**). Ezek magyarázata a következő ábrán található:



Például

```
szöveg
\rotatebox[origin=c]{90}{\fbox{szöveg}}
szöveg
\rotatebox{90}{\fbox{szöveg}}
szöveg
\rotatebox[origin=bl]{60}{\fbox{szöveg}}
szöveg
\rotatebox[origin=Br]{-60}{szöveg}
szöveg
```

12.6. Dobozok nyújtása, tükrözése

Dobozok nyújtása a következő paranccsal oldható meg:

```
\scalebox{x}{y}{<doboz>} ∈ graphicx
```

<doboz> a nyújtandó doboz. Ennek helyére egyszerű szöveg is kerülhet, amit ekkor dobozként kezel.

$\langle x \rangle$ a vízszintes nyújtás szorzó (lehet negatív is).

$\langle y \rangle$ a függőleges nyújtás szorzó (lehet negatív is), melynek alapértéke $\langle x \rangle$.

Például

```
szöveg
\scalebox{1.5}{\fbox{szöveg}}
\scalebox{1.5}[1]{\fbox{szöveg}}
\scalebox{-1}[1]{szöveg}
\scalebox{1}[-1]{szöveg}
\scalebox{-1}[-1]{szöveg}
```

Amint látjuk ezzel tükrözni is tudunk. A függőleges tengelyű tükrözésre külön parancs is létezik:

`\reflectbox{<doboz>}` \in `graphicx`

amely egyenértékű a `\scalebox{-1}[1]{<doboz>}` paranccsal.

12.7. Dobozok átméretezése

`\resizebox{<szélesség>}{<magasság>}{<doboz>}` \in `graphicx`
`\resizebox*{<szélesség>}{<magasság>}{<doboz>}` \in `graphicx`

$\langle doboz \rangle$ az átméretezett doboz. Ennek helyére egyszerű szöveg is kerülhet, amit ekkor dobozként kezel.

$\langle szélesség \rangle$ az átméretezett doboz szélessége.

$\langle magasság \rangle$ az átméretezett doboz magassága. Ez a `\resizebox` esetén az alapvonaltól mért magasságot, míg `\resizebox*` esetén a teljes magasságot jelenti.

Ha $\langle szélesség \rangle$ vagy $\langle magasság \rangle$ helyén `!` jel van, akkor azt a méretet a másikhöz arányosan állítja be. Például

```
szöveg
\resizebox{!}{0.5cm}{szöveg}
\resizebox*{!}{0.5cm}{szöveg}
\resizebox{3cm}{0.5cm}{szöveg}
```

12.8. Doboz méreteinek nullázása

`\smash{<szöveg>}` \in `amsmath` a létrehozott dobozt megjeleníti, de annak teljes magasságát 0 pt-nak tekinti.

`\smash[t]{<szöveg>}` \in `amsmath` a létrehozott dobozt megjeleníti, de úgy kezeli, mintha annak magassága 0 pt lenne.

`\smash[b]{<szöveg>}` \in `amsmath` a létrehozott dobozt megjeleníti, de úgy kezeli, mintha annak mélysége 0 pt lenne.

`\vphantom{<szöveg>}` létrehoz egy `<szöveg>` teljes magasságával megegyező teljes magasságú 0 pt szélességű láthatatlan ún. gyámfát.

Elemezzük a következő kód hatását!

```
\setlength{\fboxsep}{0pt}
\fbox{\Huge g}
\fbox{\smash{\Huge g}}
\fbox{\smash[t]{\Huge g}}
\fbox{\smash[b]{\Huge g}}
\fbox{\vphantom{\Huge g}}
```

g g g g |

12.9. Láthatatlan dobozok

A következő parancs által létrehozott doboz úgy viselkedik, mintha láthatatlan betűkkel íródott volna:

■ `\phantom{<szöveg>}`

Például

```
\noindent Ez most látszik,\
\phantom{Ez most látszik,} de most nem.
```

Ez most látszik,
de most nem.

13. fejezet

Verbatim, programkód, URL

13.1. Verbatim

A verbatim olyan része a forrásállománynak, melynek egyik része sem értelmeződik, nem fordítódik le, hanem úgy jelenik meg a dokumentumban, mint a forrásállományban. Ha a verbatim szöveg nem hosszabb egy input sornál, akkor használja a

```
\verb|verbatim szöveg|  
\verb*|verbatim szöveg|
```

parancsokat. A | határolójel lehet bármely más, szóköztől, *-tól és betűtől különböző jel, ami nem szerepel a verbatim szövegben. Például

```
\verb|\LaTeX\ könyv|\|  
\verb+\LaTeX\ kód+
```

```
\LaTeX\ könyv  
\LaTeX\ kód
```

\verb helyett \verb* írva, az eredményben a szóközők helyén □ jelenik meg. Például

```
\verb*|\LaTeX\ könyv|\|  
\verb*+\LaTeX\ kód+
```

```
\LaTeX\□könyv  
\LaTeX\□kód
```

A \verb illetve \verb* parancsok nem tehetők más parancsok argumentumába.

Ha egy input sornál többet akar beírni verbatimként, akkor **verbatim** vagy **verbatim*** környezetet használjon. Például

```
\begin{verbatim}  
\LaTeX\ könyv  
\LaTeX\ kód  
\end{verbatim}  
\begin{verbatim*}  
\LaTeX\ könyv  
\LaTeX\ kód  
\end{verbatim*}
```

```
\LaTeX\ könyv
\LaTeX\ kód

\LaTeX\_\könyv
\LaTeX\_\kód
```

Ezek a környezetek nem tehetők parancsok argumentumába.

Mindezeket még rugalmasabban tehetjük meg a **fancyvrb** csomaggal. A csomag használatát nem részletezzük, a dokumentációjában mindent megtalál az Olvasó. Csak egy példán illusztráljuk a tudását:

```
\begin{Verbatim}[formatcom={\color{cyan}\footnotesize},
showspaces,frame=single,rulecolor=\color{red},numbers=left]
\LaTeX\ könyv
\LaTeX\ kód
\end{Verbatim}
```

```
1 \LaTeX\_\könyv
2 \LaTeX\_\kód
```

Az előző kódban színeket használtunk, ezért ehhez még be kell tölteni az **xcolor** csomagot is. A **Verbatim** környezet nem tehető parancs argumentumába. Sem a **verbatim** sem a **Verbatim** környezetek nem ágyazható egymásba. Például az alábbi kód hibás:

```
\begin{verbatim}
\begin{verbatim}
...
\end{verbatim}
\end{verbatim}
```

De a **verbatim** beágyazható a **Verbatim** környezetbe vagy fordítva:

```
\begin{Verbatim}
\begin{verbatim}
...
\end{verbatim}
\end{Verbatim}
```

Ha valamilyen **verbatim** parancsot más parancs argumentumába kell tenni, akkor erre a **fancyvrb** csomag ad megoldást a következők használatával:

```
\SaveVerb{<név>}|<verbatim szöveg>| ∈ fancyvrb
\UseVerb[<opciók>]{<név>} ∈ fancyvrb
```

Itt a **|** határolóra hasonló a szabály, mint a **\verb** parancsnál. Az **<opciók>** ugyanazok lehetnek, mint a **Verbatim** környezetnél. Például, ha széljegyzetbe akarunk **verbatim** szöveget tenni, akkor a következőt tehetjük:

```
\SaveVerb{latex}|\LaTeX\ könyv|
\marginpar{\UseVerb[formatcom={\tiny},showspaces]{latex}}
```

Lábjegyzetre is jó az előző megoldás, csak ekkor **\marginpar** helyett a **\footnote** parancsot kell beírni. De lábjegyzet esetére a **fancyvrb** csomag egyszerűbb megoldást is ad. Ha beírja a

```
\VerbatimFootnotes ∈ fancyvrb
```

parancsot, akkor utána verbatim használható `\footnote` parancs argumentumában, azaz például ez a kód is működik:

```
\footnote{\verb|\LaTeX\ könyv|}
```

13.2. Verbatim szöveg kiírása fájlba

Néha szükség lehet rá, hogy fordítás közben egy \LaTeX -kód ne értelmeződjön, hanem egy fájlba legyen elmentve. Erre valók a `newfile` csomag következő parancsai:

```
\newoutputstream{<streamnév>} ∈ newfile
\openoutputfile{<fájlnév>}{<streamnév>} ∈ newfile
\begin{writeverbatim}{<streamnév>} ∈ newfile
\closeoutputstream{<streamnév>} ∈ newfile
```

A `writeverbatim` környezet nem tehető parancs argumentumába. Ezen parancsok használatát a következő kóddal szemléltethetjük:

```
\newoutputstream{proba}
\openoutputfile{minta.tex}{proba}
AAA
\begin{writeverbatim}{proba}
Ezt verbatimként kimentí a \texttt{minta.tex}-be,
\end{writeverbatim}
BBB
\begin{writeverbatim}{proba}
majd ezt hozzáfűzi.
\end{writeverbatim}
\closeoutputstream{proba}
CCC
```

Ebben a mentés folyamatának `proba` nevet adtunk, amely a `minta.tex` fájlba menti el verbatim szöveggént azon `writeverbatim` környezetek tartalmát, melyek argumentumában `proba` szerepel. Ha több ilyen környezet is van, akkor azok tartalmát összefűzi. Ha korábban már létezett a `minta.tex` fájl, akkor annak tartalmát először törli az `\openoutputfile{minta.tex}{proba}`. A mentés folyamatát lezárhatjuk a `\closeoutputstream{proba}` paranccsal. Így tehát ezt a kódot lefordítva, az eredmény

AAA BBB CCC

lesz, továbbá létrejön egy `minta.tex` fájl a dokumentum mappájában, melynek tartalma

```
Ezt verbatimként kimentí a \texttt{minta.tex}-be,
majd ezt hozzáfűzi.
```

Az előbb leírtak jól használhatók például a következő esetben. Az a feladatunk, hogy írjunk egy példatárat úgy, hogy a megoldások külön kötetben szerepeljenek. A feladatok számozását automatikusra kell állítani (lásd később), hiszen előfordulhat, hogy már begépelte két feladat közé kell beékelni egy harmadikat. Ilyenkor a számozások elcsúsznának.

A probléma az, hogy ilyen beékelések esetén a megoldásoknál is meg kell keresni a beszúrási pontot. Ez gyakorlatilag átláthatatlan káoszt okozna egy idő után. A megoldás az, hogy egy feladat begépelése után ugyanazon forrásállományba kell gépelni a

megoldást is `writeverbatim` környezetbe. Fordítás után csak a feladatok jelennek meg, míg a megoldások forráskódja helyes sorrendben egy külön fájlban lesznek, melyből a megoldáskötet is elkészíthető.

Hasonló feladatra az `answers` csomag is használható:

```
\Opensolutionfile{<streamnév>}[<fájlnév>] ∈ answers
\Closesolutionfile{<streamnév>} ∈ answers
\begin{Filesave}{<streamnév>} ∈ answers
\Newassociation{<verbatim környezet>}{<környezet>}{<streamnév>} ∈ answers
```

Ha a `<fájlnév>` nincs megadva opcióként, akkor az megegyezik a `<streamnév>`-vel. Például az előző kóddal azonos hatást érünk el, ha az `answers` csomagot betöltve, a következő kódot használja:

```
\Opensolutionfile{minta}
AAA
\begin{Filesave}{minta}
Ezt verbatimként kimentti a \texttt{minta.tex}-be,
\end{Filesave}
BBB
\begin{Filesave}{minta}
majd ezt hozzáfűzi.
\end{Filesave}
\Closesolutionfile{minta}
CCC
```

Ha a `minta.tex` fájlt például a `sections` almappába akarja menteni, akkor módosítsa így az előző kód első sorát:

```
\Opensolutionfile{minta}[sections/minta]
```

A `\Newassociation ∈ answers` parancssal további lehetőségek is vannak. Például:

```
\Newassociation{solution}{megoldas}{megold}
\def\megoldaslabel{\textbf{Megoldás.}}
\Opensolutionfile{megold}
AAA
\begin{solution}
BBB
\end{solution}
\Closesolutionfile{megold}
\input{megold}
```

Ennek hatására létrejön egy `solution` és egy `megoldas` nevű környezet. A `solution` környezetbe rakott kód verbatimként kiíródik a `megold.tex` fájlba, de ennek tartalmát `megoldas` környezetbe rakja. Tehát az előző kód hatására a `megold.tex` tartalma a következő lesz:

```
\begin{megoldas}{}
BBB
\end{megoldas}
```

Fordítás után az eredmény:

```
AAA
Megoldás. BBB
```

Az előző kódban, ha a `megoldas` környezet már korábban definiált volt, akkor azt nem definiálja felül, de ekkor a `\megoldasparams` parancsot hatástalanítani kell. Például

```
\newtheorem{megoldas}{Megoldás}
\Newassociation{solution}{megoldas}{megold}
\def\megoldasparams{}
\Opensolutionfile{megold}
AAA
\begin{solution}
BBB
\end{solution}
\Closesolutionfile{megold}
\input{megold}
```

Ennek eredménye:

```
AAA
1. Megoldás. BBB
```

13.3. Programkódok

Különböző programnyelvek kódjainak megjelenítésére alkalmas a `listings` csomag.

```
\lstinline[opciók]|kód| ∈ listings
\begin{lstlisting}[opciók] kód \end{lstlisting} ∈ listings
\lstinputlisting[opciók]{kódot tartalmazó fájl} ∈ listings
```

Az `\lstinline` sorközi kód esetén alkalmazható. Az opciók a következő parancsban is megadhatók:

```
\lstset{opciók} ∈ listings
```

Vannak olyan opciók, melyek értékében szerepelhetnek a `[` illetve `]` jelek. Például `language=[Sharp]C`. Ez az `\lstset` parancsba rakható minden gond nélkül

```
\lstset{language=[Sharp]C}
```

de az `\lstinline`, `\lstinputlisting` parancsok illetve `lstlisting` környezet opciói közé már nem. Ebben az esetben az értéket kapcsos zárójelek közé kell tenni. Például

```
\lstinputlisting[language={ [Sharp]C }]{code.pas}
```

Az `lstlisting` környezet és `\lstinline`, `\lstinputlisting` parancsok nem tehetők parancs argumentumába.

Az `\lstinputlisting` parancs használatakor a programkódot tartalmazó fájl legyen ugyanolyan kódolású, mint a tex forrásállomány.

A `listings` csomag 1 bájtos kódolást tud kezelni. Így ha Latin-2 kódolással dolgozunk, akkor a programkódban található ékezetes betűk jól fognak megjelenni. De UTF-8 esetén, ha a programkódban ékezetes betűket vannak, akkor a fordítás hibás lesz. Ekkor `listings` helyett használja a `listingsutf8` csomagot. A csomag betöltése után írja be a következő kódot:

```
\lstset{inputencoding=utf8/latin2} ∈ listingsutf8
```

Ezután a listingsutf8 csomag az `\lstinputlisting` parancs használatakor pontosan úgy működik, mint a listings, csak először az UTF-8 kódolású karaktereket Latin-2-re konvertálja. Sajnos a listingsutf8 nem működik `\lstinline` parancs illetve `lstlisting` környezet esetén. Ebben az esetben inkább használja a következő kódot:

```
\lstset{iterate={ö}{\o}}1{ü}{\u}}1{ó}{\o}}1{ő}{\H o}}1{ú}{\u}}1
{ű}{\H u}}1{é}{\e}}1{á}{\a}}1{i}{\i}}1{ö}{\O}}1{Ü}{\U}}1
{Ó}{\O}}1{Ő}{\H O}}1{Ú}{\U}}1{Ű}{\H U}}1{É}{\E}}1{Á}{\A}}1
{Í}{\I}}1}
```

Ez a magyar ékezetes betűket repülő ékezetekre konvertálja, ami megoldja a problémát.

Opciók

Tekintsük át az előbbi parancsok opcióit. Az értékekben szereplő színekre vonatkozó kódok az `xcolor` csomag betöltésével működnek.

`basicstyle=<stílus>` Kód fontjai (például `basicstyle=\small\ttfamily`).

`columns=<érték>` Ha a kód fontjai változó szélességűek, akkor is van lehetőség a kód oszlopos elrendezésére. Ekkor az `<érték>` legyen `fixed` (alapérték). Ha azt akarjuk, hogy minden karakter a természetes szélességében jelenjen meg, akkor az `<érték>` legyen `fullflexible`. Mindkét esetben a szóközök számát és méretét rugalmasan kezeli.

`keepspaces` Az előző opcióban láttuk, hogy a szóközök száma a végeredményben nem biztosan annyi, mint a forrásban. Ha ez nem kívánatos eredményt ad, akkor használjuk ezt az opciót. Ekkor pontosan annyi szóköz lesz, amennyit a forrásba tettünk és a tabulátorok helyére is szóközöket rak.

`breaklines` Hosszú sorok törése (soft wrap).

`postbreak=\hbox{<jel>}` Hosszú sorok törése utáni jel (például jobbra mutató piros nyíl `postbreak=\hbox{\textcolor{red}{\rightarrowfill}}`).

`prebreak=\hbox{<jel>}` Ugyanaz, mint előbb, csak a sorok törése elé tesz egy jelet.

`breakindent=<hossz>` Hosszú sorok törése után, a következő sor behúzásának mértéke (például `breakindent=10pt`).

`gobble=<szám>` A kód sorainak első `<szám>` darab karakterét nem veszi figyelembe. Az `\lstinline` parancsban hatástalan.

`backgroundcolor=<szín>` Háttérszín (például `backgroundcolor=\color{red}`).

`xleftmargin=<hossz>` Szövegtükör bal széle és a kód bal széle közötti távolság (például `xleftmargin=1cm`).

`xrightmargin=<hossz>` Szövegtükör jobb széle és a kód jobb széle közötti távolság (például `xrightmargin=1cm`).

`linewidth=<hossz>` Szövegtükör bal széle és a kód jobb széle közötti távolság (például `linewidth=12cm`).

`showspaces` Szóköz □ módon jelölve.

`showtabs` Tabulátort jelöli.

`tabsize=<szóközök száma>` Tabulátor mérete `<szóközök száma>` darab szóköz (például `tabsize=2`).

`tab=<jel>` Tabulátor jele (például `tab=\rightarrowfill`).

`numbers=<típus>` Kód sorainak számozása. Ha a `<típus>` `none` (alapértelmezés), akkor nincs számozás, ha `left`, akkor bal oldalon van számozás, ha `right`, akkor jobb oldalon van számozás.

`numberstyle=<stílus>` A sorszámok fontjainak beállítása (például `numberstyle=\tiny`).

A számlálóját `lstnumber`.

`numbersep=<hossz>` A sorszám és a kód távolsága (például `numbersep=10pt`).

`stepnumber=<egész szám>` Például `stepnumber=2` esetén csak minden második sorszám jelenik meg.

`firstnumber=<egész szám>` Például `firstnumber=100` esetén a kód első sorának száma 100. Az `<egész szám>` helyére `last` írva, kezdéskor nem nullázódik a számláló, így ilyenkor az előző kód számozását folytatja.

`frame=<érték>` Keretvonalak rajzolása. Az érték a `trblTRBL` bármilyen részhalmaza lehet. `t`: fent, `b`: lent, `r`: jobbra, `l`: balra (a nagybetűk jelentése hasonló, de dupla vonalat húznak). Lehet még `shadowbox` és `none` is az érték. Például, ha fent és bal oldalon akarunk vonalat húzni, akkor `frame=tl`.

`framround=<érték>` Keretsarkok stílusa. Az érték a `ttttffff` bármilyen négyelemű részhalmaza lehet. `t`: kerekített sarok, `f`: derékszögű sarok. Sorrend: jobb felső saroktól negatív forgási irányban. Például `framround=tftf`.

`framerule=<hossz>` Keret vonalának vastagsága (például `framerule=0.4pt`).

`framesep=<hossz>` Keret és kód közötti távolság (például `framesep=5pt`).

`rulesep=<hossz>` Keret dupla vonalai közötti távolság (például `rulesep=2pt`).

`rulecolor=<szín>` Keret vonalának színe (például `rulecolor=\color{red}`).

`rulesepcolor=<szín>` A dupla keretvonalak közötti területnek a színét ezzel állíthatjuk be (például `rulesepcolor=\color{red}`).

`fillcolor=<szín>` Keret és kód közötti szín (például `fillcolor=\color{red}`).

`literate={<mit>}{<mire>}{<szám>}` A programkódban található `<mit>` helyére a `<mire>` \LaTeX -kód kifejtését teszi úgy, hogy az eredményben `<szám>` karakternyi helyet foglal el. Például, ha a kódban található `<=` helyére `\leq`, illetve `>=` helyére `\geq` jeleket akarunk tenni, akkor írjuk ezt: `literate={<=}&{{\leq}}1{>=}&{{\geq}}1`

`escapeinside={<innen>}{<eddig>}` Például `escapeinside={(*}{*)}` esetén a kódban szereplő `(\pounds)` helyén a végeredményben `£` lesz, azaz `(` és `)` jelek közötti \LaTeX -parancs a kódban kifejtődik.

`language=<programnyelv>` Programnyelv kulcsszavainak, megjegyzéseinek a kiemelését tölti be. Az előre definiált nyelvek listája megtalálható a csomag leírásában. Például `language=Delphi`.

Saját nyelvet így lehet definiálni: `\lstdefinlanguage{<név>}{<opciók>}` (`<opciók>` lásd később.) Vannak olyan előre definiált nyelvek, melyeknek több dialektusa van. Például `C#` esetén így kell betölteni: `language=[Sharp]C`. Az előre definiált nyelvek dialektusainak listája megtalálható a csomag leírásában.

Ha korábban betöltött nyelvek kiemelését törölni akarjuk, akkor ezt használja:

`language={}`

Ha a későbbiekben ismertetett opciókkal magunk is beállítunk kiemelést, akkor azt a `language` opció után tegyük, különben a `language` felülbírálhatja.

`keywords=[<osztály>]{<lista>}` Az `<osztály>` számú osztályba tartozó kiemelendő kulcsszavak listája, mely a `<lista>`-ban van felsorolva, vesszővel elválasztva. Az `<osztály>` egy pozitív egész szám. Az `[1]` elhagyható. Például `keywords={begin,end}` vagy `keywords=[2]{procedure,function}`.

`morekeywords=[<osztály>]{<lista>}` Az `<osztály>` számú osztály kulcsszavainak listáját ezzel lehet bővíteni. Az `[1]` elhagyható.

`keywordstyle=[<osztály>]{<stílus>}` Az `<osztály>` számú osztály kulcsszavainak stílusa. Az `[1]` elhagyható. Például `keywordstyle=[2]\bfseries`

`comment=[s][<stílus>]{<ettől>}{<eddig>}` Megjegyzés kiemelése. A korábban beállított megjegyzés stílusok törlődnek. Ha `comment` helyett `morecomment` opciót használunk, akkor a korábbi megjegyzés beállítások megmaradnak.

Például `comment=[s][\itshape\color{red}]{/*}{*/}` esetén a kódban található `/*...*/` részt az adott stílusban jeleníti meg, beleértve a `/*` és `*/` határolójeleket is.

`comment=[n][<stílus>]{<ettől>}{<eddig>}` Ugyanaz, mint az előbb, de itt a megjegyzések egymásba ágyazhatók.

`comment=[l][<stílus>]{<ettől>}` Egysoros megjegyzések kiemelése. A korábban beállított megjegyzés stílusok törlődnek. Ha `comment` helyett `morecomment` opciót használunk, akkor a korábbi megjegyzés beállítások megmaradnak.

Például `comment=[l][\itshape\color{red}]{//}` esetén a `//` jeltől az adott sor az adott stílusban jelenik meg, beleértve a `//` jelet is.

`commentstyle=<stílus>` Például, ha `comment=[l]{//}` módon definiáltunk megjegyzést, akkor alapból dőlt betűvel fog megjelenni. Ezt a stílust utólag ezzel az opcióval módosíthatjuk (például `commentstyle=\itshape\color{green}`).

`delim=[s][<stílus>]{<ettől>}{<eddig>}` A határolójelek közötti rész kiemelése. A korábban beállított `delim` stílusok törlődnek. Ha `delim` helyett `moredelim` opciót használunk, akkor a korábbi `delim` beállítások megmaradnak.

Például `delim=[s][\color{red}]{"}{"` esetén a kódban található `"..."` részt az adott stílusban jeleníti meg, beleértve a `"` határolójeleket is.

`delim=[is][<stílus>]{<ettől>}{<eddig>}` Az előzőtől annyiban különbözik, hogy ekkor a határolójelek nem jelennek meg.

`alsoletter={<karaktersorozat>}` Például, ha a `\chapter` és `\section` szavakat kulcsszóként akarjuk definiálni, akkor ehhez először a `\` karaktert betűre kell állítani az `alsoletter={\}` opcióval. Ezután a `morekeywords={\chapter,\section}` opcióval definiálhatjuk a kulcsszavakat.

`style=<stílusnév>` Előre definiált stílust hív meg. Stílus definiálása a következő paranccsal történik: `\lstdefinestyle{<stílusnév>}{<opciók>}`

`title={<kódcím>}` Kód címe sorszám nélkül. Ez nem kerül be a kódok jegyzékébe.

`caption={<kódcím>}` Kód címe sorszámmal, címkével. Ha címkének például a „kód” szót szeretné, akkor használja ezt a parancsot: `\def\lstlistingname{kód}`. Ha magyar nyelvű dokumentumot ír, akkor még töltsse be a `caption` csomagot is. Ezután a cím így jelenik meg: „1. kód. ...” vagy „1.1. kód. ...”. Ennek a számlálója `lstlisting`. A cím bekerül a kódok jegyzékébe.

`nolol` Számozott kód ne kerüljön be a kódok jegyzékébe.

`numberbychapter=false` A kódok számozása ne a fejezetszámmal együtt történjen.

`label={<címke>}` Kereszthivatkozás címkéje. Ezt a `\label` parancs helyett kell használni. A parancs azért nem használható, mert azt a \LaTeX már a programkód részének tekintené.

A `magyar.ldf` és a `listings` csomag egy esetben összeakad. Nevezetesen, ha aktív karakter (`;!?'``) van egy címben (`\title`, `\author`, `\chapter`, `\section`, stb.) vagy egy `\label`-ben, akkor kód betöltésénél hibával leállhat a fordítás. Egy lehetséges megoldás a `magyar.ldf` betöltésekor az aktív karakterek kikapcsolása az `active=onlycs` illetve az `activespace=none` opciók beírásával:

```
\PassOptionsToPackage{defaults=hu-min,active=onlycs,activespace=none}
{magyar.ldf}
```

Ez nem a legszerencsésebb megoldás, mert ezzel a magyar tipográfiában kötelező kis szóköz a `;;!?` jelek előtt nem fog megjelenni. Szerencsésebb (bár macerásabb) megoldás, hogy a címben szereplő `;;!?` karakterek elé `\kern.1em\string` parancsot, míg ``` elé `\string` parancsot teszünk.

Nézzünk néhány példát:

```
\begin{lstlisting}[language=Delphi,basicstyle=\footnotesize]
function Trim(s:string):string;
var i:integer;
begin
  result:='';
  for i:=1 to length(s) do if s[i]<>' '
  then result:=result+s[i];
end;
\end{lstlisting}
```

```
function Trim(s:string):string;
var i:integer;
begin
  result:='';
  for i:=1 to length(s) do if s[i]<>' '
  then result:=result+s[i];
end;
```

```
\def\lstlistingname{kód}
\lstset{language=Delphi,basicstyle=\footnotesize,
keywordstyle=\bfseries\color{blue},numbers=left,
frame=tRBl,framround=tftt}

\begin{lstlisting}[caption={Trim függvény},label={kod-trim}]
function Trim(s:string):string;
var i:integer;
begin
  result:='';
  for i:=1 to length(s) do if s[i]<>' '
  then result:=result+s[i];
end;
\end{lstlisting}
\Aref{kod-trim}~kódban \dots
```

1. kód. Trim függvény

```
1 function Trim(s:string):string;
2 var i:integer;
3 begin
4   result:='';
5   for i:=1 to length(s) do if s[i]<>' '
6   then result:=result+s[i];
7   end;
```

Az 1. kódban ...

13.4. URL címek megadása

Az internetcímek is verbatimnak tekinthetők bizonyos értelemben. Ezeket

```
\url{URL cím} ∈ url
```

módon lehet beírni. Ez annyiban különbözik a `\verb` használatától, hogy meg tudja törni a sor végén az URL címet. Ha elektronikus publikációt készít, akkor `url` helyett a később ismertetésre kerülő `hyperref` csomagot kell használni, melynek szintén van `\url` parancsa. Például

```
\url{http://www.tug.org}
```

```
http://www.tug.org
```

A `hyperref` csomag `latex.exe` fordító esetén nem töri meg a linkeket. Ekkor töltsse még be a `hyperref` után a `breakurl` csomagot is.

Az URL cím betűtípusát az `\UrlFont` parancs tárolja. Például, ha azt akarja, hogy antikva betűkkel jelenjenek meg az URL címek, akkor írja be a következőt:

```
\def\UrlFont{\rmfamily}
```

Az URL címek lehetséges töréspontjai is megadhatók az `\UrlBreaks` paranccsal. Például

```
\def\UrlBreaks{\do\.\do\@\do\\\do\/\do\!\do\_ \do\|\do\;\do\>\do\]\do\~}
```

14. fejezet

Képletek

14.1. Matematikai mód

Ha matematikai képletet akar szerkeszteni, akkor használja az `amsmath` és `amssymb` csomagokat. Ha egy parancs csak ezen két csomag valamelyikének betöltésével érhető el, akkor azt a továbbiakban már nem fogjuk külön jelezni.

Képletben konstansokat és változókat más betűvel kell szedni, mint folyószöveget. Ennek magyarázataként figyelje meg a következő két mondatot.

„Ha az a pozitív és z negatív, akkor az az negatív.”

„Ha az a pozitív és z negatív, akkor az az negatív.”

De nem csak erre kell odafigyelni. A képletek szerkesztése, az egyik legösszetettebb szedői munka. Ezért a \LaTeX -hel tudatnunk kell, hogy képlet következik.

Ha egy képlet kb. akkora mint egy szó, akkor azt a szövegbe illesztjük, mint például a $\sin^2 \alpha + \cos^2 \alpha = 1$ esetén. Ez az ún. *szövegszerű matematikai mód*. Ha a képlet nagyobb, bonyolultabb, vagy fontossága miatt ki kell emelnünk, akkor külön sorba kell szedni, mint például az

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \quad \forall x \in \mathbb{R}$$

esetén. Ez az ún. *kiemelt matematikai mód*.

Szövegszerű matematikai mód megadása a következő három sor bármelyikével lehetséges:

```
$\langle képlet \rangle$  
\(\langle képlet \rangle\)  
\begin{math}\langle képlet \rangle\end{math}
```

Például

Bármit is teszünk, $\$2+2=4\$$.

Bármit is teszünk, $2 + 2 = 4$.

Kiemelt matematikai mód megadása a következő három sor bármelyikével lehetséges:

```
\[\langle képlet \rangle\]  
\begin{displaymath}\langle képlet \rangle\end{displaymath}  
\begin{equation*}\langle képlet \rangle\end{equation*}
```

Például

```
■ Bármít is teszünk, \[2+2=4.\]
```

Bármít is teszünk,

$$2 + 2 = 4.$$

A kiemelt matematikai képletek alaphelyzetben középre igazítva jelennek meg. Azonban az `amsmath` csomag `fleqn` opciójával elérhető, hogy balra legyen igazítva úgy, hogy a képlet a bal margótól 2,5 cm távolságra kezdődjön. Ez az érték átállítható például 2 cm-re, a következő paranccsal:

```
■ \setlength{\mathindent}{2cm}
```

A következő parancs makrók írásánál hasznos:

```
■ \ensuremath{\langle képlet \rangle}
```

Ez függetlenül attól, hogy az `\ensuremath` matematikai vagy szöveg módban lett aktiválva, az argumentumában található képlet mindenképpen matematikai módban lesz. Például

```
■ \newcommand{\kp}{\ensuremath{2\pi}}
```

A koszinusz `\kp` szerint periodikus, így $\cos(x + \kp) = \cos x$.

A koszinusz 2π szerint periodikus, így $\cos(x + 2\pi) = \cos x$.

Bizonyos esetekben előfordulhat, hogy egy képletben magyarázó vagy összekötő szöveget kell beiktatni. Ilyenkor ideiglenesen ki kell lépni a matematikai módból a

```
■ \text{\langle szöveg \rangle}
```

paranccsal. Például

```
■ \[1+1=2 \text{ és } 2+2=4\]
```

$$1 + 1 = 2 \text{ és } 2 + 2 = 4$$

Amint látjuk a képletben rosszul jelent meg a szöveg, pedig a forrásban volt szóköz a szöveg előtt és után. Ennek az a magyarázata, hogy a matematikai módban begépett szóközöket a \LaTeX felülbírálja. A nagyon speciális tipográfia miatt, nem bízva a szerzőre. Ilyenkor az a megoldás, hogy a szóközöket szöveg módban adjuk ki:

```
■ \[1+1=2 \text{ és } 2+2=4\]
```

$$1 + 1 = 2 \text{ és } 2 + 2 = 4$$

De ez még mindig nem tökéletes. Ugyanis a képletekben maguktól megjelenő térközök miatt nem különül el jól a szöveg. Ilyenkor lehet használni a `\quad` parancsot:

```
■ \[1+1=2\quad\text{és}\quad2+2=4\]
```

$$1 + 1 = 2 \quad \text{és} \quad 2 + 2 = 4$$

A szövegközi és a kiemelt matematikai mód között, nem csak elrendezésbeli különbség van. Például

```
■ $\frac{31}{54}$
■ \[\frac{31}{54}\]
```

$$\frac{31}{54}$$

$$\frac{31}{54}$$

Amint látjuk a méret sem egyforma. Ha kiemelt matematikai módban olyan betűmérettel és stílusban szeretnénk valamit megjeleníteni, mintha az szövegszerű matematikai módban lenne, vagy fordítva, akkor használja a következő parancsokat:

`\textstyle` szövegszerű stílusra vált

`\displaystyle` kiemelt stílusra vált

Például

`\frac{31}{54}`

`\[\textstyle\frac{31}{54}\]`

$$\frac{31}{54}$$

$$\frac{31}{54}$$

Az indexek mindkét matematikai módban ugyanakkorák, hasonlóképpen az index indexe is. Erre a stílusra is átválthat:

`\scriptstyle` index stílusra vált

`\scriptscriptstyle` index indexe stílusra vált

Például

`$2^x\scriptstyle 2^x$`

$$2^x_{2^x}$$

A betűméretet változtató deklarációs parancsok matematikai módban kiadva hatástalanok, de szövegmódban kiadva, az utánuk következő képletek méretét is megváltoztatják. Például

Pitagorasz-tétel: `$a^2+b^2=c^2$`

`\Large` Pitagorasz-tétel: `$a^2+b^2=c^2$`

$$\text{Pitagorasz-tétel: } a^2 + b^2 = c^2$$

$$\text{Pitagorasz-tétel: } a^2 + b^2 = c^2$$

Az `amsmath` csomag gondoskodik arról, hogy ez a megoldás akkor is működjön, ha a képlet nagy operátorjeleket (szumma, produktum stb.) is tartalmaz. Azonban, ha az `lmodern` fontváltó csomagot használja, akkor a nagy operátorjelek nem lesznek átméretezhetőek még az `amsmath` csomaggal együtt sem. Ezen a gondon segít a `fixcmex` csomag, melyet az `lmodern` után kell betölteni.

14.2. Matematikai betűváltozatok

Matematikai módban a betűk közötti távolság és a szóközök kezelése másképpen történik, mint szövegmódban. Ezért a betűtípusokat nem a `\textit`, `\textrm` stb. parancsokkal választjuk ki, hanem

```
\mathit{<karakterek>}
\mathrm{<karakterek>}
\mathbf{<karakterek>}
\mathsf{<karakterek>}
\mathtt{<karakterek>}
\mathnormal{<karakterek>}
```

Például a numerikus konstansokat álló betűvel kell szedni:

```
$\mathrm{e}^{\mathrm{i}\pi}+1=0$
```

$$e^{i\pi} + 1 = 0$$

A `\mathbf` parancs nem feltétlenül ad jó eredményt. Ha például címben mindent félkövéren akar szedni, mint a következő esetben:

```
\section*{A $\mathbf{\sum\frac{1}{n}}$ sor tulajdonságai}
```

A $\sum \frac{1}{n}$ sor tulajdonságai

Hibák: Az **n** nem dőlt, a szummajel és a törtvonal nem félkövér. Ilyenkor használja a

```
\pmb{<karakterek>}
```

parancsot. Például

```
\section*{A $\pmb{\sum\frac{1}{n}}$ sor tulajdonságai}
```

A $\sum \frac{1}{n}$ sor tulajdonságai

A `\pmb` (poor man's boldface) az argumentumát többször egymás közelébe nyomtatja, így érve el a félkövér hatást. Ennek a megoldásnak a gyengéje nagyítva tűnik fel. Például `$\pmb{\alpha}$`

α

Azonban sok matematikai szimbólumnak és betűnek van félkövér verziója, ami a

```
\boldsymbol{<karakterek>}
```

parancssal jelenik meg. Például `$\boldsymbol{\alpha}$` kinagyítva

α

Sajnos néhány jelre (mint a szumma vagy integrál) nincs félkövér verzió, ilyenkor háttástan a `\boldsymbol`. Ekkor csak a `\pmb` használható.

14.3. Kalligrafikus, dupla szárú betűk és fraktúrák

```
\mathcal{<karakterek>}
\mathscr{<karakterek>} \in mathrsfs
\mathbb{<karakterek>}
\mathds{<karakterek>} \in dsfont
\mathds{<karakterek>} \in [sans]dsfont
\mathfrak{<karakterek>}
```

Például

```
\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\
\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\
\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\
\mathds{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\
\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
```

ABCDEFGHIJKLMNOPQRSTUVWXYZ
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Ha a dsfont csomagot sans opcióval töltötte be, akkor

```
\mathds{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
```

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

14.4. Görög betűk

α	<code>\alpha</code>	π	<code>\pi</code>	ϱ	<code>\varrho</code>	Γ	<code>\varGamma</code>
β	<code>\beta</code>	ρ	<code>\rho</code>	ς	<code>\varsigma</code>	Δ	<code>\varDelta</code>
γ	<code>\gamma</code>	σ	<code>\sigma</code>	φ	<code>\varphi</code>	Θ	<code>\varTheta</code>
δ	<code>\delta</code>	τ	<code>\tau</code>	Γ	<code>\Gamma</code>	Λ	<code>\varLambda</code>
ϵ	<code>\epsilon</code>	υ	<code>\upsilon</code>	Δ	<code>\Delta</code>	Ξ	<code>\varXi</code>
ζ	<code>\zeta</code>	ϕ	<code>\phi</code>	Θ	<code>\Theta</code>	Π	<code>\varPi</code>
η	<code>\eta</code>	χ	<code>\chi</code>	Λ	<code>\Lambda</code>	Σ	<code>\varSigma</code>
θ	<code>\theta</code>	ψ	<code>\psi</code>	Ξ	<code>\Xi</code>	Υ	<code>\varUpsilon</code>
ι	<code>\iota</code>	ω	<code>\omega</code>	Π	<code>\Pi</code>	Φ	<code>\varPhi</code>
κ	<code>\kappa</code>	F	<code>\digamma</code>	Σ	<code>\Sigma</code>	Ψ	<code>\varPsi</code>
λ	<code>\lambda</code>	ε	<code>\varepsilon</code>	Υ	<code>\Upsilon</code>	Ω	<code>\varOmega</code>
μ	<code>\mu</code>	ϑ	<code>\vartheta</code>	Φ	<code>\Phi</code>		
ν	<code>\nu</code>	\varkappa	<code>\varkappa</code>	Ψ	<code>\Psi</code>		
ξ	<code>\xi</code>	ϖ	<code>\varpi</code>	Ω	<code>\Omega</code>		

14.5. Matematikai ékezetek

\hat{a}	<code>\hat{a}</code>	\acute{a}	<code>\acute{a}</code>	\dot{a}	<code>\dot{a}</code>	\ddot{a}	<code>\ddot{a}</code>
\tilde{a}	<code>\tilde{a}</code>	\grave{a}	<code>\grave{a}</code>	\ddot{a}	<code>\ddot{a}</code>	\mathring{a}	<code>\mathring{a}</code>
\bar{a}	<code>\bar{a}</code>	\breve{a}	<code>\breve{a}</code>	\ddot{a}	<code>\ddot{a}</code>		
\vec{a}	<code>\vec{a}</code>	\check{a}	<code>\check{a}</code>				

Ha az ι és j jelekre akar matematikai módban ékezetet tenni, akkor ne a korábban megismert `\i` és `\j`, hanem az `\imath` és `\jmath` parancsokat használja. Például

```
\check{\imath}\ddot{\jmath}
```


\ddot{ij}

14.6. Műveleti jelek

$+$	$+$	\times	<code>\times</code>	\vee	<code>\vee</code>	\ominus	<code>\ominus</code>
$-$	$-$	\div	<code>\div</code>	\star	<code>\star</code>	\odot	<code>\odot</code>
$/$	$/$	\setminus	<code>\setminus</code>	$*$	<code>*</code>	\oslash	<code>\oslash</code>
\pm	<code>\pm</code>	\cap	<code>\cap</code>	\circ	<code>\circ</code>	\otimes	<code>\otimes</code>
\mp	<code>\mp</code>	\cup	<code>\cup</code>	\bullet	<code>\bullet</code>		
\cdot	<code>\cdot</code>	\wedge	<code>\wedge</code>	\oplus	<code>\oplus</code>		

Például

```
$1+1$
```

 $1 + 1$

esetén vegyük észre, hogy a kódban nincs szóköz, de az eredményben igen. Ugyanis az a szabály, hogy a műveleti jelek elé és után is térköz kell. Ezt a \LaTeX tudja, így helyettünk is cselekszik. Azonban ehhez tudnia kell, hogy mi számít műveleti jelnek. Az előbbieket automatikusan annak tekinti, de bármit annak tekint, ha a

```
\mathbin{<karakterek>}
```

parancsba írjuk. Például

```
$a\dag ab$
```

 $a\dag ab$

de

```
$a\mathbin{\dag}ab$
```

 $a \dag ab$

14.7. Relációjelek

$=$	$=$	\approx	<code>\approx</code>	\geq	<code>\geqslant</code>	\subseteq	<code>\subseteq</code>
$:=$	<code>:=</code>	\cong	<code>\cong</code>	\ll	<code>\ll</code>	\supseteq	<code>\supseteq</code>
\doteq	<code>\doteq</code>	$<$	<code><</code>	\gg	<code>\gg</code>	$:$	<code>:</code> (arány)
\doteq	<code>\doteq</code>	$>$	<code>></code>	\in	<code>\in</code>	$ $	<code>\mid</code>
\equiv	<code>\equiv</code>	\leq	<code>\leq</code>	\ni	<code>\ni</code>	\parallel	<code>\parallel</code>
\sim	<code>\sim</code>	\geq	<code>\geq</code>	\subset	<code>\subset</code>	\perp	<code>\perp</code>
\simeq	<code>\simeq</code>	\leq	<code>\leqslant</code>	\supset	<code>\supset</code>		

Ekvivalencia reláció esetén még a modulus jelölése is kell:

```
$a \bmod m$\\
$a \equiv b \pmod{m}$\\
$a \equiv b \mod{m}$\\
$a \equiv b \pod{m}$
```

$$a \bmod m$$

$$a \equiv b \pmod{m}$$

$$a \equiv b \bmod m$$

$$a \equiv b \pmod{m}$$

A magyarban tipográfiai szabály, hogy az $=$ jel, ha a sor végére kerül, akkor az a következő sor elején megismétlődjön. Ezt a `magyar.ldf` automatikusan meg is oldja. Viszont, ha $:=$ kerül a sor végére, akkor a következő sor elején csak az $=$ jel ismétlődik meg. Az oka, hogy a sor eleji ismétlést végző `\MathBrk \in [magyar]babel` parancs csak egyetlen jelre működik, míg a $:=$ jelet kettőnek tekinti. A megoldás az, hogy először egyetlen jelként kell definiálni, majd azt kell a `\MathBrk` parancsba írni. Hasonló a gond a $=:$ jellel. Ezeket oldja meg a következő kód a preambulumba írva:

```
\DeclareSymbolFont{symbolsC}{U}{pxsyc}{m}{n}
\DeclareMathSymbol{\Coloneq}{\mathrel}{symbolsC}{66}
\DeclareMathSymbol{\Eqcolon}{\mathrel}{symbolsC}{67}
\def\coloneq{\MathBrk{\Coloneq}}
\def\eqcolon{\MathBrk{\Eqcolon}}
```

Ezután $:=$ helyett használjuk a `\coloneq`, míg $=:$ helyett az `\eqcolon` parancsot.

A nyilak is a relációjelek közé tartoznak.

\leftarrow	<code>\leftarrow</code>	\Leftrightarrow	<code>\Leftrightarrow</code>	\Uparrow	<code>\updownarrow</code>
\longleftarrow	<code>\longleftarrow</code>	\Longleftrightarrow	<code>\Longleftrightarrow</code>	\Uparrow	<code>\Uparrow</code>
\rightarrow	<code>\rightarrow</code>	\mapsto	<code>\mapsto</code>	\Downarrow	<code>\Downarrow</code>
\longrightarrow	<code>\longrightarrow</code>	\longmapsto	<code>\longmapsto</code>	\Updownarrow	<code>\Updownarrow</code>
\leftrightharpoonup	<code>\leftrightharpoonup</code>	\leftharpoonup	<code>\leftharpoonup</code>	\nearrow	<code>\nearrow</code>
\longleftrightharpoonup	<code>\longleftrightharpoonup</code>	\leftharpoonup	<code>\leftharpoonup</code>	\searrow	<code>\searrow</code>
\Leftarrow	<code>\Leftarrow</code>	\rightharpoonup	<code>\rightharpoonup</code>	\swarrow	<code>\swarrow</code>
\Longleftarrow	<code>\Longleftarrow</code>	\rightharpoonup	<code>\rightharpoonup</code>	\nwarrow	<code>\nwarrow</code>
\Rightarrow	<code>\Rightarrow</code>	\uparrow	<code>\uparrow</code>		
\Longrightarrow	<code>\Longrightarrow</code>	\downarrow	<code>\downarrow</code>		

Relációjeleket negálni (áthúzni) a `\not` parancssal lehet. Például

```
$a\not=b$
```

$$a \neq b$$

Néhány esetben ez nem ad megfelelő eredményt:

```
$a\not\mid b$, $a\not\parallel b$
```

$$a \nmid b, a \not\parallel b$$

Ezek helyett külön tervezésű negált reláció jelet kell használni:

```
$a\nmid b$, $a\nparallel b$
```

$$a \nmid b, a \nparallel b$$

A relációjelek körüli térközökre ugyanaz a szabály, mint a műveleti jelekre:

```
$a=b$
```

$$a = b$$

A \LaTeX bármit relációjelnek tekint, amit a

`\mathrel{<karakterek>}`

parancsba írunk. Például

`$a|b$`

$$a|b$$

de

`$a\mathrel{||}b$`

$$a \mid b$$

Jelek egymásra helyezésével is készíthet relációjelet a

`\stackrel{<felül>}{<alul>}`

paranccsal. Például

`$A\stackrel{f}{\longrightarrow}B$`

$$A \xrightarrow{f} B$$

14.8. Közöséges matematikai jelek

$\%$ <code>\%</code>	\Re <code>\Re</code>	\triangle <code>\triangle</code>	\sharp <code>\sharp</code>
\perp <code>\bot</code>	\Im <code>\Im</code>	\square <code>\square</code>	\natural <code>\natural</code>
\top <code>\top</code>	∇ <code>\nabla</code>	\blacksquare <code>\blacksquare</code>	$\#$ <code>\#</code>
\neg <code>\neg</code>	∂ <code>\partial</code>	\angle <code>\angle</code>	1° <code>1^\circ</code>
\forall <code>\forall</code>	\eth <code>\eth</code>	\measuredangle <code>\measuredangle</code>	$1'$ <code>1'\prime</code>
\exists <code>\exists</code>	\emptyset <code>\emptyset</code>	\sphericalangle <code>\sphericalangle</code>	$1''$ <code>1''\prime\prime</code>
\nexists <code>\nexists</code>	∞ <code>\infty</code>	\flat <code>\flat</code>	

14.9. Három pont

x_1, \dots, x_n	<code>x_1,\ldots,x_n</code>
$x_1 + \dots + x_n$	<code>x_1+\cdots+x_n</code>
$x_1 x_2 \cdots x_n$	<code>x_1x_2\cdots x_n</code>
$\int \cdots \int$	<code>\int\cdots\int</code>
\vdots	<code>\vdots</code>
\ddots	<code>\ddots</code>

Az `\ldots` az alapvonalra, míg a `\cdots` függőlegesen középre teszi a három pontot. Több esetben ez automatizálható a `\dots` paranccsal. Például

x_1, \dots, x_n	<code>x_1,\dots,x_n</code>
$x_1 + \dots + x_n$	<code>x_1+\dots+x_n</code>

ugyanazt az eredményt adják, de

$x_1 x_2 \dots x_n$ (rossz!)	<code>x_1x_2\dots x_n</code>
$\int \dots \int$ (rossz!)	<code>\int\dots\int</code>

esetén nem jó helyen lesznek a pontok.

14.10. Matematikai zárójelek

<code>(</code>	<code>[</code>	<code>\lfloor</code>	<code>]</code>	<code>\rfloor</code>	<code>\lceil</code>	<code>\rceil</code>	<code>\urcorner</code>
<code>[</code>	<code>\lceil</code> vagy <code>\lbrack</code>	<code>\ulcorner</code>	<code>}</code>	<code>\}</code> vagy <code>\rbrace</code>	<code>\lrcorner</code>		
<code>{</code>	<code>\{</code> vagy <code>\brace</code>	<code>\llcorner</code>	<code>\rangle</code>	<code>\rangle</code>	<code>\ </code> vagy <code>\Vert</code>		
<code>\langle</code>	<code>\langle</code>	<code> </code> vagy <code>\vert</code>	<code>\rceil</code>				
<code>\lceil</code>	<code>\lceil</code>	<code>\rfloor</code>	<code>\rfloor</code>				

A matematikai zárójeleket övező térközök nagyon speciálisan viselkednek, továbbá az sem mindegy, hogy nyitó vagy csukó zárójelről van szó. Ezt közölni kell a forráskódban. A nyitó zárójel elé `\left` míg a csukó zárójel elé `\right` parancsot kell írni. Példaként nézzük meg a következő két sor eredménye közötti különbséget:

```
$a(b+c)d$\n$
$a\left(b+c\right)d$
```

$$a(b+c)d$$

$$a(b+c)d$$

Van olyan eset, amikor a `\left` és `\right` parancsok elhagyása teljesen rossz eredményt ad. Például

```
$|-7|\n$
$\left|-7\right|$
```

$$|-7|$$

$$|-7|$$

Az első eset azért rossz, mert a program azt hiszi, hogy a `|` jelből kivonjuk a 7-et. Így a `-` jel körül térközöket hagy. Felmerül a kérdés, hogy a \LaTeX miért nem tudja, hogy például a `]` jel egy csukó zárójel? Hiszen ekkor nem kellene elé rakni a `\right` parancsot. Ez azért van, mert a matematikában egyáltalán nem biztos, hogy `]` valóban csukó zárójel. Például gondoljunk a $[0, 1]$ félig nyílt, félig zárt intervallumra. Az sem biztos, hogy például a `{` jellel zárójelet akarunk kifejezni. Gondoljunk az esetek szétválasztására:

$$f(x) = \begin{cases} 1, & \text{ha } x < 1, \\ 0, & \text{ha } x \geq 1. \end{cases}$$

Vagyis a zárójelek nem automatizálhatók a képletekben. A felhasználónak kell megmondani, hogy mi számít nyitó és mi csukó zárójelnek.

A `\left` és `\right` parancsok nem csak a térközöket, hanem a zárójelek nagyságát is beállítják. Például

```
$\left(1+\left(x+y\right)^2\right)^3$
```

$$(1 + (x + y)^2)^3$$

Ha egyetlen zárójelre van szükség, melynek méretben igazodni kell a képlethez, míg a zárójel párját nem akarja megjeleníteni, akkor tudatni kell, hogy hol van a képlet másik határa, különben nem tudna a méret mihez igazodni. Ezt a határt egy láthatatlan zárójellel adjuk meg `\left.` illetve `\right.` parancsokkal. Például

```
$\left.\left(1+x^2\right)\right|_{x=1}=2$
```

$$(1 + x^2)'|_{x=1} = 2$$

Ha automatikus méretű zárójelben van egy formula, ami csak több sorban fér el, továbbá az első sorban magasabbak a képletek mint a másodikban, akkor a csukó zárójel nem lesz megfelelő méretű. Például

```
\dotfill$\left(\frac{1+\frac{1}{2}},1,2,\ldots,\right.$\\
$\left.n-1,n\right)$
```

$$\dots\dots\dots\left(\frac{1}{1+\frac{1}{2}},1,2,\dots,\right.\\n-1,n)$$

Megoldás

```
\dotfill$\left(\frac{1+\frac{1}{2}},1,2,\ldots,\right.$\\
$\left.n-1,n\right\phantom{\frac{1}{1+\frac{1}{2}}}\right)$
```

$$\dots\dots\dots\left(\frac{1}{1+\frac{1}{2}},1,2,\dots,\right.\\n-1,n)$$

Néhány esetben nem ad megfelelő eredményt a zárójelek automatikus méretezése. Például

```
$\left\{\left\{\left\{a,b\right\}\right\},\left\{c,d\right\}\right\}$
```

$$\{\{a,b\},\{c,d\}\}$$

A külső zárójeleknek egy picit nagyobbaknak kellene lenniük, de ezt a közbezárt képlet nem generálja. Ilyenkor rögzített méreteket is használhat. A `\left` helyett

```
\bigl \Bigl \biggl \Biggl
```

illetve `\right` helyett

```
\bigr \Bigr \biggr \Biggr
```

Ezek hatása:

```
$\Biggl(\biggl(\Bigl(\bigl(\left(\cdot\right)\bigr)\Bigr)\biggr)\Biggr)$
```

$$\left(\left(\left(\left(\left(\cdot\right)\right)\right)\right)\right)$$

Például az előző képlet a következő módon oldható meg helyesen:

```
$\bigr\{\left\{a,b\right\},\left\{c,d\right\}\bigr\}$
```

$$\{ \{a, b\}, \{c, d\} \}$$

Ha egy rögzített méretű zárójelet közönséges matematikai jelként akar használni, akkor alkalmazza a következő parancsokat:

`\big \Big \bigg \Bigg`

Például

`\left(1+x^2\right)'\Big|_{x=1}=2`

$$(1+x^2)' \Big|_{x=1} = 2$$

Ha egy rögzített méretű zárójelet relációjelként akar használni, akkor alkalmazza a következő parancsokat:

`\bigm \Bigm \biggm \Biggm`

Például

`\left\{\frac{2^{n^2}}{n^2} \mid n \text{ egész} \right\}`

$$\left\{ \frac{2^{n^2}}{n^2} \mid n \text{ egész} \right\}$$

Az eddigiekben ismertetett rögzített méretű zárójelek az `amsmath` csomagban következő séma szerint vannak definiálva:

```
\makeatletter
\def\langle név \rangle{\bBigg@{\langle méret \rangle}}
\def\langle név \rangle m{\mathrel\langle név \rangle}
\def\langle név \rangle l{\mathopen\langle név \rangle}
\def\langle név \rangle r{\mathclose\langle név \rangle}
\makeatother
```

ahol

$\langle név \rangle$	$\langle méret \rangle$
<code>big</code>	1
<code>Big</code>	1.5
<code>bigg</code>	2
<code>Bigg</code>	2.5

Amint látjuk, a 3 illetve 3.5 méretek már nincsenek definiálva, de ezt megteheti a következő kóddal, az `amsmath` csomag betöltése után:

```
\makeatletter
\def\biggg{\bBigg@{3}}
\def\bigggm{\mathrel\biggg}
\def\biggggl{\mathopen\biggg}
\def\biggggr{\mathclose\biggg}
\def\Biggg{\bBigg@{3.5}}
\def\Bigggm{\mathrel\Biggg}
\def\Biggggl{\mathopen\Biggg}
```

```
\def\Bigggr{\mathclose\Biggg}
\makeatother
```

Ezután a `\biggg`, `\bigggm`, `\biggg1`, `\biggggr`, `\Biggg`, `\Bigggm`, `\Biggg1`, `\Biggggr` parancsok hasonlóan használhatók, ahogyan azt a korábbiakban ismertettük. A 4, 4.5, stb. méretek hasonlóan definiálhatók.

14.11. Esetek szétválasztása

A korábbiakban szóba került az esetek szétválasztása, amikor egy zárójel nem zárójelként funkcionál. Erre a `cases` környezet használható. Például

```
\[f(x)=
\begin{cases}
0 & \text{ha } x \text{ racionális,} \\
1 & \text{ha } x \text{ irracionális.} \\
\end{cases}\]
```

$$f(x) = \begin{cases} 0 & \text{ha } x \text{ racionális,} \\ 1 & \text{ha } x \text{ irracionális.} \end{cases}$$

Látható, hogy a `cases` környezet úgy működik, mint egy két oszlopból álló táblázat.

14.12. Matematikai jelek több szerepben

Vannak olyan matematikai jelek, amelyeknek többféle szerepe is lehet. Ezeket a következő táblázatban foglaljuk össze:

	közönséges mat. jel	műveleti jel	relációjel	írásjel	zárójel
\	<code>\backslash</code>	<code>\setminus</code>			
:			:	<code>\colon</code>	
			<code>\mid</code>		<code>\left </code> <code>\right </code>
	<code>\ </code>		<code>\parallel</code>		<code>\left\ </code> <code>\right\ </code>
<			<		<code>\left\langle</code> <code>\rangle</code>
>			>		<code>\left\rangle</code> <code>\rangle</code>
⊥	<code>\bot</code>		<code>\perp</code>		
†	<code>\dag</code>	<code>\dagger</code>			
‡	<code>\ddag</code>	<code>\ddagger</code>			

Például

```
$f\colon A\rightarrow B$ (helyes)\\
$f:A\rightarrow B$ (helytelen)
```

$f: A \rightarrow B$ (helyes)
 $f: A \rightarrow B$ (helytelen)

A második megoldás azért rossz, mert ott az szerepel, hogy f aránylik az A -hoz.

Ha a magyar.ldf fájlt defaults=hu-min opcióval töltötte be, akkor a vessző matematikai üzemmódban két szám között tizedesvesszőként értelmezett, de egyéb esetben megmarad az eredeti szerepe. Például

```
$2,5\cdot2=5$\\
$a,b,c$
```

```
2,5 · 2 = 5
a, b, c
```

Ha két szám között a vesszőt nem tizedesvesszőként használja, akkor a vessző után tegyen egy szóközt:

```
$1, 2, 3,\dots$
```

```
1, 2, 3, ...
```

Ha nem a magyar nyelv van beállítva, akkor a vesszőnek nincs kettős szerepköre. Ha magyar nyelv esetén sem akarja ezt a kettős szerepkört, akkor töltsé még be a `mathhucomma=unchanged` opciót is a defaults=hu-min után. Ekkor tizedesvesszőt így kell írni:

```
$2{,}5\cdot2=5$
```

```
2,5 · 2 = 5
```

14.13. Változó hosszúságú vízszintes jelek

\widehat{xz}	<code>\widehat{xz}</code>	$A \xleftarrow{f \circ g} B$	<code>\$A\xleftarrow{f\circ g}B\$</code>
\widetilde{xz}	<code>\widetilde{xz}</code>	$A \xleftarrow[f \circ g]{} B$	<code>\$A\xleftarrow[f\circ g]{}B\$</code>
\overline{xz}	<code>\overline{xz}</code>	$A \xrightarrow{f \circ g} B$	<code>\$A\xrightarrow{f\circ g}B\$</code>
\underline{xz}	<code>\underline{xz}</code>	$A \xrightarrow[f \circ g]{} B$	<code>\$A\xrightarrow[f\circ g]{}B\$</code>
\overleftarrow{xz}	<code>\overleftarrow{xz}</code>	\overbrace{xxxzzz}_n	<code>\overbrace{xxxzzz}</code>
\overrightarrow{xz}	<code>\overrightarrow{xz}</code>	\overbrace{xxxzzz}^n	<code>\overbrace{xxxzzz}^{\sim n}</code>
\underrightarrow{xz}	<code>\underrightarrow{xz}</code>	\underbrace{xxxzzz}	<code>\underbrace{xxxzzz}</code>
$\overleftrightharpoonup{xz}$	<code>\overleftrightharpoonup{xz}</code>	\underbrace{xxxzzz}_n	<code>\underbrace{xxxzzz}_{\sim n}</code>
$\underleftrightharpoonup{xz}$	<code>\underleftrightharpoonup{xz}</code>		

14.14. Gyökvonás

```
\sqrt{2}      \sqrt{2}
\sqrt{2+\sqrt{2}}  \sqrt{2+\sqrt{2}}
\sqrt[n]{2}     \sqrt[n]{2}
```

Az utolsó képletben lehetőség van az n finom igazítására.

```
\uproot{<fel>}
\leftroot{<balra>}
```


$\langle \textit{fel} \rangle$ egész szám, hatására n felcsúszik $\frac{\langle \textit{fel} \rangle}{18}$ em-mel.

$\langle \textit{balra} \rangle$ egész szám, hatására n balra csúszik $\frac{\langle \textit{balra} \rangle}{18}$ em-mel.

Például

```
\sqrt[\uproot{1}\leftroot{1}n]{2}$
```

$$\sqrt[n]{2}$$

A következő kód nem ad tökéletes megoldást.

```
\sqrt{x}+\sqrt{y}$
```

$$\sqrt{x} + \sqrt{y}$$

Az y mélysége pozitív, míg az x -nek 0. Így a két gyökjel függőleges mérete nem egyezik meg. Ezt a következő kóddal lehet megoldani:

```
\sqrt{x}+\sqrt{\smash[b]{y}}$
```

$$\sqrt{x} + \sqrt{y}$$

Az utóbbiban a `\smash[b]` parancs az y mélységét 0-nak veszi, így a két gyökjel mérete egyforma lesz.

Amennyiben a kézíráshoz hasonlóan a gyökjelet lezárt véggel, azaz $\sqrt{\quad}$ alakban szeretné használni, akkor írja a preambulumba az `amsmath` csomag betöltése után a következőket:

```
\usepackage{letltxmacro}
\makeatletter
\let\oldr@@t\r@@t
\def\r@@t#1#2{%
\setbox0=\hbox{$\oldr@@t#1{#2\,,}$}\dimen0=\ht0
\advance\dimen0-0.2\ht0
\setbox2=\hbox{\vrule height\ht0 depth -\dimen0}%
{\box0\lower0.04em\box2}}
\LetLtxMacro{\oldsqrt}{\sqrt}
\renewcommand*{\sqrt}[2][\oldsqrt]{#1}{#2}
\makeatother
```

14.15. Mátrixok

$\begin{matrix} a & b \\ c & d \end{matrix}$ `\begin{matrix} a&b\\ c&d\\ \end{matrix}`

$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ `\begin{pmatrix} a&b\\ c&d\\ \end{pmatrix}`

$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ `\begin{bmatrix} a&b\\ c&d\\ \end{bmatrix}`

$\begin{Bmatrix} a & b \\ c & d \end{Bmatrix}$ `\begin{Bmatrix} a&b\\ c&d\\ \end{Bmatrix}`

$\begin{vmatrix} a & b \\ c & d \end{vmatrix}$ `\begin{vmatrix} a&b\\ c&d\\ \end{vmatrix}`

$\begin{vmatrix} a & b \\ c & d \end{vmatrix}$ `\begin{Vmatrix} a&b\\ c&d\\ \end{Vmatrix}`

Szöveg közben az $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ helyett szebb az $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ mátrix. Ennek kódja:

```
\left(
\begin{smallmatrix}
a&b\\
c&d\\
\end{smallmatrix}
\right)
```

Három pont helyett hosszabb pontsorozatok is kiírhatók a

```
\hdotsfor[⟨sűrűség⟩]{⟨oszlopok⟩}
```

parancssal, ahol $\langle sűrűség \rangle$ a pontsor sűrűsége (alapérték 1) és $\langle oszlopok \rangle$ a keresztezett oszlopok száma. Például

```
\[ \begin{pmatrix}
1&2&3&\hdotsfor[2]{3}&n\\
2&3&\hdotsfor{2}&n&n+1\\
3&\hdotsfor{2}&n&n+1&n+2
\end{pmatrix} \]
```

$$\begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 2 & 3 & \dots & n & n+1 \\ 3 & \dots & n & n+1 & n+2 \end{pmatrix}$$

```
\[ \begin{pmatrix}
1&2&3&\ldots&n\\
0&1&2&\ldots&n-1\\
\hdotsfor[0.5]{5}\\
0&0&0&\ldots&1
\end{pmatrix} \]
```

$$\begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 0 & 1 & 2 & \dots & n-1 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

A `\hdotsfor` parancs nem ad jó eredményt, ha a `colortbl` csomagot is használja. Ebben az esetben az `amsmath` csomag betöltése után a preambulumba írja a következőket:

```
\makeatletter
\def\hdotsfor#1#2{\multicolumn{#2}c%
{\m@th\dotsspace@1.5mu\mkern-#1\dotsspace@
\xleaders\hbox{$\m@th\mkern#1\dotsspace@.\mkern#1\dotsspace@$}%
\hskip\z@\@plus 1filll
\mkern-#1\dotsspace@}%
}
\makeatother
```

14.16. Matematikai jelek egymásra helyezése

Ezekre már láttunk korábban lehetőségeket a változó hosszúságú vízszintes jeleknél és a relációjeleknél. Az ott leírt `\stackrel`, `\xleftarrow`, `\xrightarrow` parancsok mindegyike reláció típust eredményez. Most két másik parancsot ismertetünk:

```
\overset{⟨amit⟩}{⟨amire⟩}
\underset{⟨amit⟩}{⟨ami alá⟩}
```

Mindkettő típusa meg fog egyezni a második argumentumba írt jel típusával. Például

```
$a\overset{*}{+}b\underset{\mathrm{d}}{=}{c}$
```

$$a \overset{*}{+} b \underset{d}{=} c$$

Ha közönséges matematikai jelet szeretne építeni `\scriptstyle` stílusban, akár kettőnél több sorban, akkor használja a

```
\substack{⟨sor1⟩\\ ⟨sor2⟩\\ ...}
```

parancsot, vagy a

```
\begin{subarray}{⟨igazítás⟩}
⟨sor1⟩\\ ⟨sor2⟩\\ ...
\end{subarray}
```

környezetet. Az `⟨igazítás⟩` lehet `c` (középre) `l` (balra) és `r` (jobbra). Például

```
\[\sum_{\substack{i=1,\ldots\\ j\in\mathbb{Z}\\ k=j,\ldots}}a_{ijk}\]
```

$$\sum_{\substack{i=1,\dots\\ j\in\mathbb{Z}\\ k=j,\dots}} a_{ijk}$$

```
\[\sum_{\begin{subarray}{l} i=1,\ldots\\ j\in\mathbb{Z}\\ k=j,\ldots \end{subarray}}a_{ijk}\]
```

$$\sum_{\begin{subarray}{l} i=1,\dots\\ j\in\mathbb{Z}\\ k=j,\dots \end{subarray}} a_{ijk}$$

14.17. Matematikai indexek

x_n `x_n` vagy `x\sb n`
 x_{n+1} `x_{n+1}` vagy `x\sb{n+1}`
 x^n `x^n` vagy `x\sp n`
 x^{n+1} `x^{n+1}` vagy `x\sp{n+1}`

A később tárgyalt ún. operátorok mind a négy sarkába, vagy alá és fölé is tehet indexet.

$$\prod_{a=2}^b \prod_c^d \quad \backslash \text{sideset}_{_a^b}_{_c^d} \backslash \text{prod}$$

$$\prod_1^2 \quad \backslash \text{prod} \backslash \text{limits}_1^2$$

Ha ugyanezt nem operátorral szeretné csinálni, akkor az indexelendő jelet átmenetileg operátorrá kell tenni a `\mathop` paranccsal:

`\$ \backslash \text{sideset}_{_a^b}_{_c^d} \backslash \text{mathop}\{X\} \$` és `\$ \backslash \text{mathop}\{X\} \backslash \text{limits}_1^2 \$`

$${}_a^b X_c^d \text{ és } \overset{2}{X}_1$$

14.18. Törtek, binomiális együtthatók

`\frac{⟨számláló⟩}{⟨nevező⟩}`
`\dfrac{⟨számláló⟩}{⟨nevező⟩} = \displaystyle \frac{⟨számláló⟩}{⟨nevező⟩}`
`\tfrac{⟨számláló⟩}{⟨nevező⟩} = \textstyle \frac{⟨számláló⟩}{⟨nevező⟩}`

Például

`\$ \backslash \text{frac}\{x^2\}\{x+1\} \$`

$$\frac{x^2}{x+1}$$

`\binom{⟨fent⟩}{⟨lent⟩}`
`\dbinom{⟨fent⟩}{⟨lent⟩} = \displaystyle \binom{⟨fent⟩}{⟨lent⟩}`
`\tbinom{⟨fent⟩}{⟨lent⟩} = \textstyle \binom{⟨fent⟩}{⟨lent⟩}`

Például

`\$ \backslash \text{binom}\{n+1\}\{m+1\} \$`

$$\binom{n+1}{m+1}$$

Saját stílusú törteket is létrehozhatunk:

`\genfrac{⟨bal⟩}{⟨jobb⟩}{⟨vastagság⟩}{⟨stílus⟩}{⟨fent⟩}{⟨lent⟩}`

`⟨bal⟩` bal oldali zárójel,

`⟨jobb⟩` jobb oldali zárójel,

`⟨vastagság⟩` törtvonal vastagsága (ha üres: 0.4pt),

`⟨stílus⟩` 0: `\displaystyle`, 1: `\textstyle`, 2: `\scriptstyle`, 3: `\scriptscriptstyle`,
 ha üresen hagyja, akkor a környezethez alkalmazkodik.

Például

`\[`
`\genfrac{\{}{\}}{1pt}{}{n+1}{m+1}`
`\genfrac{[]{}{0pt}{1}{n+1}{m+1}`
`\]`

$$\left\{ \frac{n+1}{m+1} \right\}^{n+1}_{m+1}$$

Lánc törtek a következő paranccsal írhatók:

■ `\cfrac[⟨igazítás⟩]{⟨számláló⟩}{⟨nevező⟩}`

ahol az *⟨igazítás⟩* lehet *l* (balra), *r* (jobbra) és *c* (középre, alapérték). Például

■ `\[\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}\]`

$$\frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \dots}}}$$

■ `\[\cfrac[1]{1}{1+\cfrac[1]{1}{1+\cfrac[1]{1}{1+\cdots}}}\]`

$$\frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \dots}}}$$

14.19. Operátorok, függvények

14.19.1. Nagy operátorok

\sum <code>\sum</code>	\bigsqcup <code>\bigsqcup</code>	\bigcup <code>\bigcup</code>	\bigwedge <code>\bigwedge</code>
\prod <code>\prod</code>	\int <code>\int</code>	\bigodot <code>\bigodot</code>	\bigvee <code>\bigvee</code>
\coprod <code>\coprod</code>	\iint <code>\iint</code>	\oint <code>\oint</code>	\bigotimes <code>\bigotimes</code>
\bigoplus <code>\bigoplus</code>	\bigcap <code>\bigcap</code>	\iiint <code>\iiint</code>	\biguplus <code>\biguplus</code>
\bigtimes <code>\varprod</code> ∈ pxfonts			

A `pxfonts` csomag az alap fontkészletet is átállítja. Ezen csomag használata nélkül úgy definiálhatja a `\varprod` operátort, ha a preambulumba beírja a következőket:

■ `\DeclareSymbolFont{largesymbolsA}{U}{pxexa}{m}{n}`
 ■ `\DeclareMathSymbol{\varprod}{\mathop}{largesymbolsA}{16}`

A nagy operátorok más méretben jelennek meg szövegszerű illetve kiemelt matematikai módban. Például

■ `\sum`
 ■ `\[\sum\]`

 Σ
 Σ

14.19.2. „Nolimits” függvények

A „nolimits” függvények indexei mindig a függvény neve mellett jelennek meg. Ilyenek a következők:

<code>arccos</code>	<code>\arccos</code>	<code>coth</code>	<code>\coth</code>	<code>lg</code>	<code>\lg</code>	<code>tanh</code>	<code>\tanh</code>
<code>arcsin</code>	<code>\arcsin</code>	<code>csc</code>	<code>\csc</code>	<code>ln</code>	<code>\ln</code>	\lim	<code>\varliminf</code>
<code>arctan</code>	<code>\arctan</code>	<code>deg</code>	<code>\deg</code>	<code>log</code>	<code>\log</code>	\lim	<code>\varlimsup</code>
<code>arg</code>	<code>\arg</code>	<code>dim</code>	<code>\dim</code>	<code>sec</code>	<code>\sec</code>	\lim	<code>\varinjlim</code>
<code>cos</code>	<code>\cos</code>	<code>exp</code>	<code>\exp</code>	<code>sin</code>	<code>\sin</code>	\lim	<code>\varprojlim</code>
<code>cosh</code>	<code>\cosh</code>	<code>hom</code>	<code>\hom</code>	<code>sinh</code>	<code>\sinh</code>	\int	<code>\int</code>
<code>cot</code>	<code>\cot</code>	<code>ker</code>	<code>\ker</code>	<code>tan</code>	<code>\tan</code>		

Például

```
$\log_2x$  
\[\log_2x\]
```

$$\log_2 x \qquad \log_2 x$$

```
$\int_a^b$  
\[\int_a^b\]
```

$$\int_a^b \qquad \int_a^b$$

14.19.3. „Limits” függvények

A nagy operátorok (az integráljel kivételével) és a „limits” függvények indexei szövegszerű matematikai módban mellette jelennek meg, de kiemelt matematikai módban alatta és fölötte. A „limits” függvények a következők:

<code>det</code>	<code>\det</code>	<code>inj lim</code>	<code>\injlim</code>	<code>lim sup</code>	<code>\limsup</code>	<code>proj lim</code>	<code>\projlim</code>
<code>gcd</code>	<code>\gcd</code>	<code>lim</code>	<code>\lim</code>	<code>max</code>	<code>\max</code>	<code>Pr</code>	<code>\Pr</code>
<code>inf</code>	<code>\inf</code>	<code>lim inf</code>	<code>\liminf</code>	<code>min</code>	<code>\min</code>	<code>sup</code>	<code>\sup</code>

Például

```
$\sum_{n=1}^{\infty} a_n$  
\[\sum_{n=1}^{\infty} a_n\]
```

$$\sum_{n=1}^{\infty} a_n \qquad \sum_{n=1}^{\infty} a_n$$

```
$\lim_{n\rightarrow\infty}a_n$  
\[\lim_{n\rightarrow\infty}a_n\]
```

$$\lim_{n \rightarrow \infty} a_n$$

$$\lim_{n \rightarrow \infty} a_n$$

Ha ezen egy adott helyen változtatni akar, akkor a `\limits` és `\nolimits` parancsokkal teheti meg. Például

```
\sum\limits_{n=1}^{\infty} a_n$
\[\sum\nolimits_{n=1}^{\infty} a_n\]
```

$$\sum_{n=1}^{\infty} a_n$$

$$\sum_{n=1}^{\infty} a_n$$

A `\limits` parancs nincs hatással a „nolimits” függvényekre, kivéve az integráljelet:

```
\log\limits_2x$
\[\log\limits_2x\]
```

$$\log_2 x$$

$$\log_2 x$$

```
\int\limits_a^b$
\[\int\limits_a^b\]
```

$$\int_a^b$$

$$\int_a^b$$

A integráljel „limits” függvénné tehető az `amsmath` csomag `intlimits` opciójával. Ezután már az integrál is pontosan úgy viselkedik, mint bármelyik más nagy operátorjel.

14.19.4. Új függvények definiálása

Előfordulhat, hogy olyan függvényre van szükség, amely alapból nem áll rendelkezésre. Például a magyarban a tangens jele tg, amelynek csak az angol verziója (tan) definiált. Ilyenkor magunk is gyárthatunk újakat. A „limits” függvények a következő parancsokkal definiálhatók:

```
\newcommand{\langle parancs \rangle}{\mathop{\mathrm{\langle jel \rangle}}}
```

vagy

```
\DeclareMathOperator*{\langle parancs \rangle}{\langle jel \rangle} % Ez csak preambulumba írható!
```

Például

```
\newcommand{\Min}{\mathop{\mathrm{Min}}}
```

vagy

```
\DeclareMathOperator*{\Min}{Min}
```

után

```
$\Min_{k\in\mathbb{N}}$  
\[\Min_{k\in\mathbb{N}}\]
```

$$\text{Min}_{k \in \mathbb{N}}$$

$$\text{Min}_{k \in \mathbb{N}}$$

Egy már létező „limits” függvényt át is definiálhatók.

```
\renewcommand{\langle parancs \rangle}{\mathop{\mathrm{\langle jel \rangle}}}
```

Például

```
\renewcommand{\min}{\mathop{\mathrm{Min}}}
```

után

```
$\min_{k\in\mathbb{N}}$  
\[\min_{k\in\mathbb{N}}\]
```

$$\text{Min}_{k \in \mathbb{N}}$$

$$\text{Min}_{k \in \mathbb{N}}$$

Új „nolimits” függvényt a következő parancsokkal definiálhatók:

```
\newcommand{\langle parancs \rangle}{\mathop{\mathrm{\langle jel \rangle}}\nolimits}
```

vagy

```
\DeclareMathOperator{\langle parancs \rangle}{\langle jel \rangle} % Ez csak preambulumba írható!
```

Például

```
\newcommand{\tg}{\mathop{\mathrm{tg}}\nolimits}
```

vagy

```
\DeclareMathOperator{\tg}{tg}
```

után

```
$\tg^2x$  
\[\tg^2x\]
```

$$\text{tg}^2 x$$

$$\text{tg}^2 x$$

Egy már létező „nolimits” függvényt át is definiálhatók.

```
\renewcommand{\langle parancs \rangle}{\mathop{\mathrm{\langle jel \rangle}}\nolimits}
```

Például

```
\renewcommand{\tan}{\mathop{\mathrm{tg}}\nolimits}
```

után

```
$\tan^2x$  
\[\tan^2x\]
```

$$\text{tg}^2 x$$

$$\text{tg}^2 x$$

14.19.5. Differenciál operátor, differenciálás

$f'(x), f''(x)$ `f'(x), f''(x)` (' az aposztrófjel **Shift** + **1**)

$\frac{\partial f(x,y)}{\partial y}$ `\frac{\partial f(x,y)}{\partial y}`

Az integrálásnál és deriválásnál szokásos differencia operátor jelet nekünk kell definiálni a preambulumban:

`\DeclareMathOperator{\diff}{d\!}`

Ezután például

```
\[
\int f(x)\diff x
\quad\text{és}\quad
\frac{\diff f(x)}{\diff x}
\]
```

$$\int f(x) dx \quad \text{és} \quad \frac{df(x)}{dx}$$

14.20. Képletek bekeretezése

Képletek bekeretezésére ugyanúgy használható az `\fcolorbox`, `\framebox` és az `\fbox` parancsok, mint a hagyományos szövegre. Például

```
\colorbox{red}{\sum_{n=1}^{\infty}}
\fbox{\sum_{n=1}^{\infty}}
\[
\colorbox{red}{\displaystyle\sum_{n=1}^{\infty}}
\quad\text{és}\quad
\fbox{\displaystyle\sum_{n=1}^{\infty}}
\]
```

$$\sum_{n=1}^{\infty}$$

$$\sum_{n=1}^{\infty}$$

és

$$\sum_{n=1}^{\infty}$$

Létezik egy kifejezetten képlet bekeretezésére alkalmas `\boxed` parancs, melynek a belsejében matematikai mód van `\displaystyle` stílusban. A keret vastagsága és a képlettől való távolsága ugyanúgy állítható, mint a `\framebox` esetén. Például

```
\boxed{\sum_{n=1}^{\infty}}
\boxed{\textstyle\sum_{n=1}^{\infty}}
\[\boxed{\sum_{n=1}^{\infty}}\]
```

$$\sum_{n=1}^{\infty} \sum_{n=1}^{\infty}$$

$$\sum_{n=1}^{\infty}$$

Hasonló megoldás színes dobozra nincs, de magunk definiálhatunk. Például:

```
\newcommand{\colorboxed}[2]{%
\colorbox{#1}{\ensuremath{\displaystyle #2}}} \in xcolor
```

után

```
\colorboxed{red}{\sum_{n=1}^{\infty}}
\colorboxed{red}{\textstyle\sum_{n=1}^{\infty}}
\[\colorboxed{red}{\sum_{n=1}^{\infty}}\]
```

$$\sum_{n=1}^{\infty} \sum_{n=1}^{\infty}$$

$$\sum_{n=1}^{\infty}$$

14.21. Kommutatív diagramok

Az alábbi példa az `amscd` csomag `CD` környezetével készült.

```
\[
\begin{CD}
A @>>> B @<<< C \\
@VVV @AAA @| \\
D @>f>l> E @= F \\
@VbVjV @AbAjA \\
G @<f<l< H
\end{CD}
\]
```

$$\begin{array}{ccccc}
A & \longrightarrow & B & \longleftarrow & C \\
\downarrow & & \uparrow & & \parallel \\
D & \xrightarrow[l]{f} & E & \xlongequal{\quad} & F \\
b \downarrow j & & b \uparrow j & & \\
G & \xleftarrow[l]{f} & H & &
\end{array}$$

Ettől többet tud az `xy` csomag, amit itt nem részletezünk.

14.22. Kiemelt képletek sorszámozása

A kiemelt képletek sorszámozására használja az `equation` környezetet. Hivatkozás esetén `\ref` helyett az `\eqref` parancs használható:

```
\begin{equation}\label{<címké>}
<képlet>
\end{equation}
\eqref{<címké>}
```

Vigyázat, a magyar.ldf-ben az `\eqref`-nek nincs névelős `\Aeqref` illetve `\aeqref` verziója. Ehelyett a következőket kell használni:

```
\Az{\eqref{<címké>}}
\az{\eqref{<címké>}}
```

A magyar.ldf szerzője az előbbi helyett az `\aref({<címké>})` illetve `\Aref({<címké>})` megoldást javasolja, de én ezzel nem értek egyet. Ugyanis az `\eqref` parancs eredménye mindig álló betű lesz, még dőlt betűs környezetben is. Ezt viszont az `\aref({...})` és `\Aref({...})` parancsok nem teljesítik.

Például

```
\begin{equation}\label{egyenlet-masodfoku}
x^2+2x-3=0
\end{equation}
\Az{\eqref{egyenlet-masodfoku}} miatt \dots
```

$$x^2 + 2x - 3 = 0 \quad (1)$$

Az (1) miatt ...

Ha a számozást bal oldalra szeretné, akkor az `amsmath` csomagot `leqno` opcióval töltsé be.

Az előbbi számozást `article` osztályban kapjuk. Ekkor az egész dokumentumban folytonos a számozás, azaz új szakasz nyitáskor nem kezdődik ismét 1-től.

Ha `report` vagy `book` osztályt használ, akkor a képletszámhoz társul az aktuális fejezet sorszáma is. Például az 1. fejezet 2. képlete (1.2) számozást kapja. Másrészt ekkor a képletszám új fejezet nyitáskor újra indul. Tehát például a 2. fejezet 1. képlete a (2.1) számozást kapja.

Ha az `article` osztályban ugyanezt a hatást akarja elérni (csak szakasszal fejezet helyett), akkor használja a következő kódot:

```
\numberwithin{equation}{section}
```

Ha menet közben kiderül, hogy az adott képletnek mégsem kell számozás, akkor csak annyit kell tenni, hogy `equation` helyett `equation*` környezetet használ.

Ha egy dokumentumban kevés olyan kiemelt képlet van, amelyre hivatkozik, akkor számok helyett más egyéni jeleket is használhat. Erre való a `\tag` és `\tag*` parancsok. Például

```
\begin{equation}\label{egyenlet-masodfoku}
x^2+2x-3=0\tag{A}
\end{equation}
```

$$x^2 + 2x - 3 = 0 \quad (A)$$

```
\begin{equation}\label{egyenlet-masodfoku}
x^2+2x-3=0\tag*{\fbox{A}}
\end{equation}
```

$$x^2 + 2x - 3 = 0$$

A

Ha `\tag*` paranccsal számozott, akkor arra ne az `\eqref` paranccsal hivatkozzon, mert az zárójelbe teszi a képletszámot. Helyette a `\ref` parancsot vagy valamelyik névelős verzióját alkalmazza.

14.23. Képletek eltörése

Ha egy képlet nem fér ki egy sorban, akkor meg is lehet törni a `multline` környezettel.

```
\begin{multline}\label{<címk>}
<képlet 1. sora>\\
<képlet 2. sora>\\
...
<képlet n. sora>
\end{multline}
```

Ebben a környezetben az első sor balra, az utolsó jobbra, a többi pedig középre lesz igazítva, továbbá a számozás az utolsó sorban jobb oldalon lesz.

Ha az `amsmath` csomagot `fleqn` opcióval töltötte be, hogy a kiemelt képletek balra legyenek igazítva, akkor a középre igazított sorok a bal oldalra igazodnak.

Ha az `amsmath` csomagot `leqno` opcióval töltötte be, hogy a számozás a bal oldalon legyen, akkor a számozás az első sor bal oldalán lesz.

Ha egy sort a bal oldalra akar igazítani, akkor tegye a `\shoveleft{<képlet sora>}` parancsba. Ha jobb oldalra akarja tenni, akkor használja a `\shoveright{<képlet sora>}` parancsot.

Egyéni képletjelölésre itt is használhatóak a `\tag` illetve `\tag*` parancsok. Ha nem akar képletszámozást, akkor a `multline*` környezetet használja. Például

```
\begin{multline}\label{egyenlet-pelda}
1+8+27+64=\\
=1+3+5+7+{\}\\
+9+11+13+{\}\\
+15+17+19
\end{multline}
```

$$1 + 8 + 27 + 64 =$$

$$= 1 + 3 + 5 + 7 +$$

$$+ 9 + 11 + 13 +$$

$$+ 15 + 17 + 19 \quad (1)$$

```
\begin{multline}\label{egyenlet-pelda}
1+8+27+64=\\
\shoveleft{=1+3+5+7+{\}}\\
+9+11+13+{\}\\
+15+17+19
```


$$\begin{aligned}
 100 &= 1 + 8 + 27 + 64 = \\
 &= 1 + 3 + 5 + 7 + 9 + \\
 &\quad + 11 + 13 + 15 + 17 + 19
 \end{aligned}
 \tag{1}$$

14.24. Több képlet egymás alatt

Ha több kiemelt képletet ír egymás alá, akkor nem ad jó eredményt a `\[...\]`, a `displaymath`, az `equation*` vagy az `equation` környezetek egymás utáni alkalmazása, mert túl nagy lesz közöttük a függőleges térköz. Ilyenkor használja a `gather` környezetet.

```

\begin{gather}
\langle 1. \text{ képlet} \rangle \backslash label{\langle címke 1 \rangle} \\
\langle 2. \text{ képlet} \rangle \backslash label{\langle címke 2 \rangle} \\
\ldots \\
\langle n. \text{ képlet} \rangle \backslash label{\langle címke n \rangle}
\end{gather}

```

Egyéni képletjelölésre itt is használhatóak a `\tag` illetve `\tag*` parancsok. Ha nem akar képletszámozást, akkor a `gather*` környezetet használja. Ha csak egy sort nem akar számozni, akkor annak végére tegye a `\notag` parancsot. Például

```

\begin{gather}
x+y \quad \backslash label{egyenlet-pelda-a} \\
x^2+xy+y^2 \backslash label{egyenlet-pelda-b}
\end{gather}

```

$$\begin{aligned}
 &x + y && (1) \\
 &x^2 + xy + y^2 && (2)
 \end{aligned}$$

```

\begin{gather}
x+y \quad \backslash notag \\
x^2+xy+y^2 \backslash label{egyenlet-pelda}
\end{gather}

```

$$\begin{aligned}
 &x + y \\
 &x^2 + xy + y^2 && (1)
 \end{aligned}$$

A `gather*` környezet ún. részformulaképző változata a `gathered` környezet. Ez azt jelenti, hogy úgy működik mint a `gather*`, de szövegközi matematikai módba, `equation` vagy `equation*` környezetbe kell rakni. A `gathered` környezetnek opciója is van, aminek az értéke `c` (alapérték), `t` vagy `b` lehet, attól függően, hogy az alapvonalat középre, fentre vagy alulra akarja igazítani. Nézzünk néhány példát:

```

\[\left. \begin{gathered}
x+y \\
x^2+xy+y^2
\end{gathered} \right\} \backslash]

```

$$\left. \begin{array}{l} x + y \\ x^2 + xy + y^2 \end{array} \right\}$$

```
\begin{equation}\label{egyenlet-pelda}
\left.\begin{gathered}
x+y\\
x^2+xy+y^2
\end{gathered}\right\}
\end{equation}
```

$$\left. \begin{array}{l} x + y \\ x^2 + xy + y^2 \end{array} \right\} \quad (1)$$

```
\[\left.\begin{gathered}
x+y\\
x^2+xy+y^2
\end{gathered}\right\}
\quad\text{és}\quad
\left.\begin{gathered}
2x+y\\
x^2+3xy+y^2
\end{gathered}\right\}
\]
```

$$\left. \begin{array}{l} x + y \\ x^2 + xy + y^2 \end{array} \right\} \quad \text{és} \quad \left. \begin{array}{l} 2x + y \\ x^2 + 3xy + y^2 \end{array} \right\}$$

```
szöveg
$\begin{gathered}
x+y\\
x^2+xy+y^2
\end{gathered}$
szöveg
```

$$\text{szöveg} \left. \begin{array}{l} x + y \\ x^2 + xy + y^2 \end{array} \right\} \text{szöveg}$$

```
szöveg
$\begin{gathered}[t]
x+y\\
x^2+xy+y^2
\end{gathered}$
szöveg
```

$$\text{szöveg} \quad x + y \quad \text{szöveg} \\ x^2 + xy + y^2$$

```
szöveg
```



```
\end{align*}
```

$$\begin{array}{ll} x = y + z & \text{a definícióból} \\ = bd + bc & \text{mivel } ac = b \\ = 1000 & \text{behelyettesítve} \end{array}$$

```
\begin{align*}
x&=y+z & \&\text{a definícióból}\\
&=bd+bc & \&\text{mivel }ac=b\\
&=1000 & \&\text{behelyettesítve}
\end{align*}
```

$$\begin{array}{ll} x = y + z & \text{a definícióból} \\ = bd + bc & \text{mivel } ac = b \\ = 1000 & \text{behelyettesítve} \end{array}$$

Az `align*` környezet ún. részformulaképző változata az `aligned` környezet. Ez azt jelenti, hogy úgy működik mint az `align*`, de szövegek közötti matematikai módra, `equation` vagy `equation*` környezetbe kell rakni. A `aligned` környezetnek opciója is van, aminek az értéke `c` (alapérték), `t` vagy `b` lehet, attól függően, hogy az alapvonalat középre, fentre vagy alulra akarja igazítani. Nézzünk néhány példát:

```
szöveg
$\begin{aligned}
x&=y+z & y&=bd & z&=bc\\
b&=10 & 2c&=56 & d&=44
\end{aligned}$
szöveg
```

$$\begin{array}{lll} \text{szöveg} & x = y + z & y = bd \quad z = bc \\ & b = 10 & 2c = 56 \quad d = 44 \end{array} \text{szöveg}$$

```
szöveg
$\begin{aligned}[t]
x&=y+z & y&=bd & z&=bc\\
b&=10 & 2c&=56 & d&=44
\end{aligned}$
szöveg
```

$$\begin{array}{lll} \text{szöveg} & x = y + z & y = bd \quad z = bc \\ & b = 10 & 2c = 56 \quad d = 44 \end{array} \text{szöveg}$$

```
szöveg
$\begin{aligned}[b]
x&=y+z & y&=bd & z&=bc\\
b&=10 & 2c&=56 & d&=44
\end{aligned}$
szöveg
```

$$\begin{array}{lcl} x = y + z & y = bd & z = bc \\ \text{szöveg } b = 10 & 2c = 56 & d = 44 \text{ szöveg} \end{array}$$

```
\begin{equation}\label{egyenlet-pelda}
\left.\begin{aligned}
x&=y+z & y&=bd & z&=bc\\
b&=10 & 2c&=56 & d&=44
\end{aligned}\right\}
\end{equation}
```

$$\left.\begin{array}{lcl} x = y + z & y = bd & z = bc \\ b = 10 & 2c = 56 & d = 44 \end{array}\right\} \quad (1)$$

```
\left.\begin{aligned}
x&=y+z\\
b&=10
\end{aligned}\right\}
\quad\text{és}\quad
\left.\begin{aligned}
y&=bd\\
2c&=56
\end{aligned}\right\}
```

$$\left.\begin{array}{l} x = y + z \\ b = 10 \end{array}\right\} \quad \text{és} \quad \left.\begin{array}{l} y = bd \\ 2c = 56 \end{array}\right\}$$

A `flalign` és `flalign*` környezetek pontosan azt teszik, mint az `align` és `align*` környezetek, de az első oszlop előtti és az utolsó oszlop utáni térköz szélessége 0 pt. Például

```
\begin{flalign*}
x&=y+z & y&=bd & z&=bc\\
b&=10 & 2c&=56 & d&=44
\end{flalign*}
```

$$\begin{array}{lll} x = y + z & y = bd & z = bc \\ b = 10 & 2c = 56 & d = 44 \end{array}$$

Az `alignat` környezetben is `r@{}l r@{}l ...` az illesztés, de itt csak az első oszlop előtti és utolsó oszlop utáni térközök oszlanak meg egyenletesen, másrészt azt is meg kell adni paraméterként, hogy hány `r@{}l` oszloppár van. A `\tag`, `\tag*`, `\notag` parancsok itt is ugyanúgy használhatók, mint a `gather` környezetben. Az `alignat*` környezet pontosan azt csinálja, mint az `alignat`, de nem tesz ki képletszámokat. Például

```
\begin{alignat}{3}
1&=1 & \quad 2&=2 & \quad 2&=1+1 & \quad \label{egyenlet-pelda-a}\\
3&=3 & \quad 3&=1+2 & \quad 3&=1+1+1 & \quad \label{egyenlet-pelda-b}
\end{alignat}
```

$$\begin{array}{lll} 1 = 1 & 2 = 2 & 2 = 1 + 1 \end{array} \quad (1)$$

$$\begin{array}{lll} 3 = 3 & 3 = 1 + 2 & 3 = 1 + 1 + 1 \end{array} \quad (2)$$

Ezzel lehet megvalósítani a lineáris egyenletrendszerek felírását is.

```
\begin{alignat*}{3}
13&x+{} & 4&y & & & && =9\\
3&x-{} & 12&y+{} & 23&z&=14
\end{alignat*}
```

$$\begin{array}{rcl} 13x + 4y & = & 9 \\ 3x - 12y + 23z & = & 14 \end{array}$$

```
\begin{alignat*}{4}
13&x+{} & 4&y & & & {} & 9\\
3&x-{} & 12&y+{} & & 23&z&{} & 14
\end{alignat*}
```

$$\begin{array}{rcl} 13x + 4y & = & 9 \\ 3x - 12y + 23z & = & 14 \end{array}$$

Az `alignat*` környezet ún. részformulaképző változata az `alignedat` környezet. Ez azt jelenti, hogy úgy működik mint az `alignat*`, de szövegeközi matematikai módba, `equation` vagy `equation*` környezetbe kell rakni. A `alignedat` környezetnek opciója is van, aminek az értéke `c` (alapérték), `t` vagy `b` lehet, attól függően, hogy az alapvonalat középre, fentre vagy alulra akarja igazítani. Nézzünk néhány példát:

```
\begin{equation}\label{egyenlet-pelda}
\left.\begin{alignedat}{2}
11&x-\{\} \ \& \ 4y\&=7\\
&x-\{\} \ \& \ y\&=0
\end{alignedat}\right\}
\end{equation}
```

$$\left. \begin{array}{l} 11x - 4y = 7 \\ x - y = 0 \end{array} \right\} \quad (1)$$

$$\begin{aligned} 11x - y &= 7 \\ x &= 0 \end{aligned} \quad \Rightarrow$$

$$\left. \begin{array}{l} 11x - 4y = 7 \\ x - y = 0 \end{array} \right\} \Rightarrow x = y = 1$$

```
szöveg

$$\begin{alignedat}{2} 11x-{} & 4y=7\end{alignedat}$$

```

```
&x-{} & y&=0
\end{alignedat}$
szöveg
```

$$\begin{array}{l} 11x - 4y = 7 \\ x - y = 0 \end{array}$$

```
szöveg
$\begin{alignedat}[t]{2}
11&x-{} & 4y&=7\\
&x-{} & y&=0
\end{alignedat}$
szöveg
```

$$\begin{array}{l} 11x - 4y = 7 \\ x - y = 0 \end{array}$$

```
szöveg
$\begin{alignedat}[b]{2}
11&x-{} & 4y&=7\\
&x-{} & y&=0
\end{alignedat}$
szöveg
```

$$\begin{array}{l} 11x - 4y = 7 \\ x - y = 0 \end{array}$$

Az `eqnarray` környezetben három oszlop van, az első jobbra, a második középre, a harmadik balra zárt. Az oszlopok közötti távolság felét a `\arraycolsep` hosszúságparancs tárolja. Itt a `\tag` és `\tag*` nem használható, továbbá `\notag` helyett a `\nonumber` működik. Az `eqnarray*` környezet pontosan azt csinálja, mint az `eqnarray`, de nem tesz ki képletszámokat.

```
\begin{eqnarray}
2x=2y & \Rightarrow & x=y & \label{egyenlet-pelda}\\
6z=600 & \Rightarrow & z=100 & \nonumber
\end{eqnarray}
```

$$\begin{array}{l} 2x = 2y \Rightarrow x = y \\ 6z = 600 \Rightarrow z = 100 \end{array} \quad (1)$$

Ezt a környezetet gyakran használják tévesen a következő esetben:

```
\begin{eqnarray*}
1+3 & =& 4\\
1+3+5 & =& 9
\end{eqnarray*}
```

$$\begin{array}{l} 1 + 3 = 4 \\ 1 + 3 + 5 = 9 \end{array} \quad \begin{array}{l} \text{Ez rossz példa!} \\ \text{Így soha!} \end{array}$$

Amint látható, itt az egyenlőségjel nem relációjelként van kezelve, hanem csak berakja középre, körülötte túl nagy térközzel. Ez a környezet nem az ilyen feladatokra lett kitalálva. A helyes megoldása:

```
\begin{align*}
1+3 &=4\\
1+3+5 &=9
\end{align*}
```

$$1 + 3 = 4$$

$$1 + 3 + 5 = 9$$

Szöveget is elhelyezhet illesztett képletek sorai között. Erre a `\intertext{<szöveg>}` parancs használható, amely a következő környezetekben működik: `align`, `align*`, `flalign`, `flalign*`, `alignat`, `alignat*`. Például

```
\begin{align*}
f(x) &= \int 4x \ln x \, dx = \\
\intertext{parciális integrálás után}
&= 2x^2 \ln x - x^2 + C = \\
&= x^2 (2 \ln x - 1) + C
\end{align*}
```

$$f(x) = \int 4x \ln x \, dx =$$

parciális integrálás után

$$= 2x^2 \ln x - x^2 + C =$$

$$= x^2 (2 \ln x - 1) + C$$

14.26. Részformulák számozása

Ha az illesztett képletekben részformulák vannak és azokat szeretné számozni, akkor használhatja a `subequations` környezetet, amelybe a következő környezetek ágyazhatók: `gather`, `align`, `flalign`, `alignat`, `eqnarray`. Például

```
\begin{subequations}
\begin{gather}
x=ac+bc \quad \text{\label{reszformula-pelda-a}}\\
y>dc \quad \text{\label{reszformula-pelda-b}}
\end{gather}
\end{subequations}
```

$$x = ac + bc \tag{1a}$$

$$y > dc \tag{1b}$$

Ha a részformulák számozásának stílusát át akarja állítani például (1/a) alakúra, akkor az `amsmath` csomag betöltése után másolja be a következő kódot:

```
\let\Subequations\subequations
\renewenvironment{subequations}
{\begin{Subequations}
\renewcommand{\theequation}{\theparentequation/\alph{equation}}}
{\end{Subequations}}
```

14.27. Oldaltörés többsoros képletekben

Alapértelmezésben a többsoros képletek közben nem megengedett az oldaltörés. Ezt a preambulumba írt `\allowdisplaybreaks` paranccsal oldhatja fel. Ekkor az oldaltörés automatikusan történik.

Adott helyen úgy kényszeríthet ki oldaltörést, ha `\\` helyett `\displaybreak\\` parancsot ír.

Adott helyen letilthatja az oldaltörést, ha `\\` helyett `*` parancsot ír.

14.28. Táblázat matematikai módban

Erre az `array` környezet a megoldás. Ezt pontosan úgy kell használni, mint a `tabular` környezetet. (Matematikai módban ne használjon `tabular` környezetet!) Például

```
\[
A\to
\begin{array}{|c|c|}
\hline
a_{11} & a_{12}\\
\hline
a_{21} & a_{22}\\
\hline
\end{array}
\]
```

$$A \rightarrow \begin{array}{|c|c|} \hline a_{11} & a_{12} \\ \hline a_{21} & a_{22} \\ \hline \end{array}$$

```
\[
\begin{array}[t]{|ccc|}
\hline
1 & 2 & 3\\
4 & 5 & 6\\
7 & 8 & 9\\
\hline
\end{array}
\begin{array}[b]{|cc|}
\hline
\alpha & \beta\\
\gamma & \delta\\
\hline
\end{array}
\]
```

 \]

			α	β
			γ	δ
1	2	3		
4	5	6		
7	8	9		

15. fejezet

További formai elemek

15.1. Cirill betűk

Latin betűs szövegben néha szükség lehet cirill betűkre is. Ehhez a T1 belső kódkészlet elé töltse be az OT2-t is:

```
\usepackage[OT2,T1]{fontenc}
```

Ekkor a T1 lesz az alapértelmezett. Az OT2 cirill betűi:

А а	А а	К к	К к	Ц ц	Ц ц
Б б	Б б	Ќ ѓ	\{'K} \{'k}	Ч ч	Ч ч
В в	В в	Л л	Л л	Ш ш	Ш ш
Г г	Г г	Љ љ	Љ љ	Щ щ	Щ щ
Ѓ ѓ	\{'G} \{'g}	М м	М м	Ъ ъ	Ъ ъ
Д д	Д д	Н н	Н н	Ы ы	Ы ы
Ђ ђ	Ђ ђ	О о	О о	Ь ь	Ь ь
Е е	Е е	П п	П п	Э э	Э э
Ё ё	Ё ё	Т т	Т т	Ю ю	Ю ю
Є є	Є є	Ћ ћ	Ћ ћ	Љ љ	\{'\CYRYU}
Ж ж	Ж ж	С с	С с	ј ј	\{'\cyryu}
З з	З з	Т т	Т т	Я я	Я я
С с	С с	Ќ ѓ	\{'C} \{'c}	Љ љ	\{'\CYRYA}
И и	И и	У у	У у	ђ ђ	\{'\cyrya}
І і	І і	Ў ў	Ў ў	ѐ ё	\CYRIZH \cyrizh
Ї ї	\{"\CYRII} \{"\i}	Ф ф	Ф ф	Ђ ђ	\CYRYAT \cyryat
Њ њ	Њ њ	Х х	Х х	Ѳ ѳ	\CYRFITA \cyrfita
Ј ј	Ј ј				

Az OT2 cirill betűit az alábbi paranccsal jelenítheti meg:

```
\fontencoding{OT2}\selectfont <cirill betűk>
```

Nem csak OT2, hanem T2C belső kódkészlettel is lehet cirill betűket írni:

А	\CYRA	Д	\CYRD	З	\CYRZ	Л	\CYRL
Б	\CYRB	Е	\CYRE	И	\CYRI	М	\CYRM
В	\CYRV	Ё	\CYRYO	Њ	\CYRISHRT	Н	\CYRN
Г	\CYRG	Ж	\CYRZH	К	\CYRK	О	\CYRO

П	\CYRP	Ж	\cyrzh	Ю	\cyryu	І	\CYRpalochka
Р	\CYRR	З	\cyrz	Я	\cyrja	Ѣ	\cyrabhch
С	\CYRS	И	\cyri	Ѥ	\CYRABHCH	Ѧ	\cyrabhchdsc
Т	\CYRT	Й	\cyrishrt	Ѧ	\CYRABHCHDSC	Ѣ	\cyrabhdze
У	\CYRU	К	\cyrk	Ѣ	\CYRABHDZE	Ѧ	\cyrkhcrs
Ф	\CYRF	Л	\cyr l	Ѧ	\CYRKHCRS	Ѧ	\cyrkdsc
Х	\CYRH	М	\cym	Ѧ	\CYRKDSC	Ѧ	\cyrmdsc
Ц	\CYRC	Н	\cyrn	Ѧ	\CYRMDSC	Ѧ	\cyrndsc
Ч	\CYRCH	О	\cyro	Ѧ	\CYRNDSC	Ѧ	\cyrotld
Ш	\CYRSH	П	\cyrp	Ѧ	\CYROTLD	Ѧ	\cyrphk
Щ	\CYRSHCH	Р	\cyr r	Ѧ	\CYRPHK	Ѧ	\cyr rtick
Ъ	\CYRHRDSN	С	\cyrs	Ѧ	\CYRRTICK	Ѧ	\cyrtdsc
Ы	\CYRERY	Т	\cyrt	Ѧ	\CYRTDSC	Ѧ	\cyrshha
Ь	\CYRSFTSN	У	\cyru	Ѧ	\CYRSHHA	Ѧ	\cyrghk
Э	\CYREREV	Ф	\cyrf	Ѧ	\CYRGHK	Ѧ	\cyrhdsc
Ю	\CYRYU	Х	\cyrh	Ѧ	\CYRHDSC	Ѧ	\cyrdzhe
Я	\CYRYA	Ц	\cyr c	Ѧ	\CYRDZHE	Ѧ	\cyrdze
а	\cyra	Ч	\cyrch	Ѧ	\CYRDZE	Ѧ	\cyr tetse
б	\cyrb	Ш	\cyrsh	Ѧ	\CYRTETSE	Ѧ	\cyrchr dsc
в	\cyrv	Щ	\cyrshch	Ѧ	\CYRCHRDSC	Ѧ	\cyrse misftsn
г	\cyrg	Ъ	\cyrhrdsn	Ѧ	\CYRSEMISFTSN	Ѧ	\cyr schwa
д	\cyrd	Ы	\cyrery	Ѧ	\CYRSCHWA	Ѧ	\cyrii
е	\cyre	Ь	\cyr sftsn	Ѧ	\CYRII	Ѧ	\cyrje
ё	\cyryo	Э	\cyrerev	Ѧ	\CYRJE	Ѧ	\cyrabhha

Az T2C cirill betűit az alábbi paranccsal jelenítheti meg:

```
\fontencoding{T2C}\selectfont <cirill betűk>
```

Ezek elérhetők TeXstudioból is: **Side Panel > Symbols > Cyrillic**.

15.2. Gótikus írás

A latin és cirill betűkön kívül még sokféle áll rendelkezésre. Például gótikus az **yfonts** csomaggal írható:

```
{\frakfamily Und da er ihn fand, sprach er zu ihm:
Glaubest du an den Sohn Gottes? 36. Er antwortete und sprach:
Herr, welcher ist's ... Du hast ihn gesehen, und der mit dir
redet, der ist's. 38. Er aber sprach: Herr, ich glaube; und
betete ihn an.}
```

Und da er ihn fand, sprach er zu ihm: Glaubest du an den Sohn Gottes? 36. Er antwortete und sprach: Herr, welcher ist's ... Du hast ihn gesehen, und der mit dir redet, der ist's. 38. Er aber sprach: Herr, ich glaube; und betete ihn an.

15.3. Görög betűk

Legtöbbször görög betűkre képletek írásakor van szükség. Ezt az esetet a 14.4. szakaszban tárgyaljuk. Ha latin betűs környezetben görög szöveget ír, akkor a `babel` csomagnak töltsse be a `greek` opcióját a főnyelv elé, majd használja a `begingreek` csomagot:

```
\begin{greek}[\langle\text{görög fonttípus}\rangle] \in begingreek
\langle\text{görög szöveg}\rangle
\end{greek}
\greektxt[\langle\text{görög fonttípus}\rangle]{\langle\text{görög szöveg}\rangle} \in begingreek
```

A `\langle\text{görög fonttípus}\rangle` lehetséges értékei: `artemisla`, `gfsbaskerville`, `bodoni`, `complutum`, `udidot`, `neohellenic`, `porson`, `solomos`, `txr`, `mak`, `llcmss`, `lmr`. Ezek használhatóak a `begingreek` csomag opciójaként is. Az alapérték `lmr`. Például

```
\greektxt{abcdefghijklmnopqrstuxyz}\\
\greektxt[gfsbaskerville]{abcdefghijklmnopqrstuxyz}\\
\greektxt[solomos]{abcdefghijklmnopqrstuxyz}
```

αβγδεφγηηθκλμνοπρστυξψζ
 αβγδεφγηηθκλμνοπρστυξψζ
 αβγδεφγηηθκλμνοπρστυξψζ

A `babel` csomag `greek` opciója nélkül, a `textgreek` csomaggal lehet görög betűket írni:

α <code>\textalpha</code>	ξ <code>\textxi</code>	Δ <code>\textDelta</code>	Σ <code>\textSigma</code>
β <code>\textbeta</code>	ο <code>\textomikron</code>	Ε <code>\textEpsilon</code>	Τ <code>\textTau</code>
γ <code>\textgamma</code>	π <code>\textpi</code>	Ζ <code>\textZeta</code>	Υ <code>\textUpsilon</code>
δ <code>\textdelta</code>	ρ <code>\textrho</code>	Η <code>\textEta</code>	Φ <code>\textPhi</code>
ε <code>\textepsilon</code>	σ <code>\textsigma</code>	Θ <code>\textTheta</code>	Χ <code>\textChi</code>
ζ <code>\textzeta</code>	τ <code>\texttau</code>	Ι <code>\textIota</code>	Ψ <code>\textPsi</code>
η <code>\texteta</code>	υ <code>\textupsilon</code>	Κ <code>\textKappa</code>	Ω <code>\textOmega</code>
θ <code>\texttheta</code>	φ <code>\textphi</code>	Λ <code>\textLambda</code>	ς <code>\textvarsigma</code>
ι <code>\textiota</code>	χ <code>\textchi</code>	Μ <code>\textMu</code>	φ <code>\straightphi</code>
κ <code>\textkappa</code>	ψ <code>\textpsi</code>	Ν <code>\textNu</code>	ϑ <code>\scripttheta</code>
λ <code>\textlambda</code>	ω <code>\textomega</code>	Ξ <code>\textXi</code>	θ <code>\straighttheta</code>
μ <code>\textmu</code>	Α <code>\textAlpha</code>	Ο <code>\textOmikron</code>	ε <code>\straightepsilon</code>
μ <code>\textmugreek</code>	Β <code>\textBeta</code>	Π <code>\textPi</code>	
ν <code>\textnu</code>	Γ <code>\textGamma</code>	Ρ <code>\textRho</code>	

15.4. Iniciálék

15.4.1. Latin iniciálék

Írja a következőket a preambulumba

```
\usepackage{anyfontsize,lettrine}
\setcounter{DefaultLines}{4}
```

majd a dokumentumtestbe

```
\lettrine{K}{ezdetben} teremte Isten az eget és a földet. A föld pedig
kietlen és pusztá vala, és setétség vala a mélység színén, és az Isten
```

Lelke lebeg vala a vizek felett. És monda Isten: Legyen világosság: és lőn világosság. És látá Isten, hogy jó a világosság; és elválasztá Isten a világosságot a setétségtől. És nevezé Isten a világosságot nappalnak, és a setétséget nevezé éjszakának: és lőn este és lőn reggel, első nap.

KEZDETBEN teremté Isten az eget és a földet. A föld pedig kietlen és pusztta vala, és setétség vala a mélység színén, és az Isten Lelke lebeg vala a vizek felett. És monda Isten: Legyen világosság: és lőn világosság. És látá Isten, hogy jó a világosság; és elválasztá Isten a világosságot a setétségtől. És nevezé Isten a világosságot nappalnak, és a setétséget nevezé éjszakának: és lőn este és lőn reggel, első nap.

Ha ékezetes betűt használ iniciálénak, akkor azt repülő ékezetként írja be. Például

```
\lettrine{'E}{s} monda Isten:
```

15.4.2. Díszes latin iniciálé

Írja a következőket a preambulumba

```
\input Zallman.fd
\usepackage{anyfontsize,lettrine}
\setcounter{DefaultLines}{4}
\renewcommand{\LettrineFontHook}{\usefont{U}{Zallman}{xl}{n}}
```

majd a dokumentumtestbe

```
\lettrine{K}{ezdetben} teremté Isten az eget és a földet. A föld pedig
kietlen és pusztta vala, és setétség vala a mélység színén, és az Isten
Lelke lebeg vala a vizek felett. És monda Isten: Legyen világosság: és
lőn világosság. És látá Isten, hogy jó a világosság; és elválasztá Isten
a világosságot a setétségtől. És nevezé Isten a világosságot nappalnak,
és a setétséget nevezé éjszakának: és lőn este és lőn reggel, első nap.
```

KEZDETBEN teremté Isten az eget és a földet. A föld pedig kietlen és pusztta vala, és setétség vala a mélység színén, és az Isten Lelke lebeg vala a vizek felett. És monda Isten: Legyen világosság: és lőn világosság. És látá Isten, hogy jó a világosság; és elválasztá Isten a világosságot a setétségtől. És nevezé Isten a világosságot nappalnak, és a setétséget nevezé éjszakának: és lőn este és lőn reggel, első nap.

Ha ékezetes betűt használ iniciálénak, akkor azt repülő ékezetként írja be. Például

```
\Initial{'E}{s} monda Isten:
```

A [Zallman](#) mintázat helyére a következő mintázatok is beírhatók: [Acorn](#), [AnnSton](#), [ArtNouv](#), [ArtNouv](#), [Carrickc](#), [Eichenla](#), [Eileen](#), [EileenBl](#), [Elzevier](#), [GotIn](#), [GoudyIn](#), [Kinigcap](#), [Konanur](#), [Kramer](#), [MorrisIn](#), [Nouveaud](#), [Romantik](#), [Rothdn](#), [RoyalIn](#), [Sanremo](#), [Starburst](#), [Typocaps](#).

15.4.3. Gótikus iniciálé

Írja a következőket a preambulumba

```
\usepackage{yfonts,lettrine}
\setcounter{DefaultLines}{4}
\renewcommand{\LettrineTextFont}{}
```

majd a dokumentumtestbe

```
{\frakfamily\fraklines\lettrine{U}nd da er ihn fand, sprach er zu ihm:
Glaubest du an den Sohn Gottes? 36. Er antwortete und sprach:
Herr, welcher ist's ... Du hast ihn gesehen, und der mit dir
redet, der ist's. 38. Er aber sprach: Herr, ich glaube; und
betete ihn an. Und Jesus sprach: Ich bin zum Gerichte auf diese
Welt kommen, auf da\ss, die da nicht sehen, sehend werden, und
die da sehen, blind werden. Und solches höreten etliche der Pharis\{a}er,
die bei ihm waren, und sprachen zu ihm: Sind wir denn auch blind?\par}
```

Und da er ihn fand, sprach er zu ihm: Glaubest du an den Sohn Gottes? 36. Er antwortete und sprach: Herr, welcher ist's ... Du hast ihn gesehen, und der mit dir redet, der ist's. 38. Er aber sprach: Herr, ich glaube; und betete ihn an. Und Jesus sprach: Ich bin zum Gerichte auf diese Welt kommen, auf daß, die da nicht sehen, sehend werden, und die da sehen, blind werden. Und solches höreten etliche der Pharisäer, die bei ihm waren, und sprachen zu ihm: Sind wir denn auch blind?

A `\par` parancs nélkül az iniciálé alá nem folyik be a szöveg.

15.4.4. Díszes gótikus iniciálé

Írja a következőket a preambulumba

```
\usepackage{yfonts}
\def\initdefault{yinit}
```

majd a dokumentumtestbe

```
{\frakfamily\fraklines\yinipar{U}nd da er ihn fand, sprach er zu ihm:
Glaubest du an den Sohn Gottes? 36. Er antwortete und sprach:
Herr, welcher ist's ... Du hast ihn gesehen, und der mit dir
redet, der ist's. 38. Er aber sprach: Herr, ich glaube; und
betete ihn an. Und Jesus sprach: Ich bin zum Gerichte auf diese
Welt kommen, auf da\ss, die da nicht sehen, sehend werden, und
die da sehen, blind werden. Und solches höreten etliche der Pharis\{a}er,
die bei ihm waren, und sprachen zu ihm: Sind wir denn auch blind?\par}
```

Und da er ihn fand, sprach er zu ihm: Glaubest du an den Sohn Gottes? 36. Er antwortete und sprach: Herr, welcher ist's ... Du hast ihn gesehen, und der mit dir redet, der ist's. 38. Er aber sprach: Herr, ich glaube; und betete ihn an. Und Jesus sprach: Ich bin zum Gerichte auf diese Welt kommen, auf daß, die da nicht sehen, sehend werden, und die da sehen, blind werden. Und solches höreten etliche der Pharisäer, die bei ihm waren, und sprachen zu ihm: Sind wir denn auch blind?

Az előző kódban azért írtuk be a `\def\initdefault{yinit}` sort, mert különben az `\yinipar` parancs az `yinit`-as fontokat tölti be, melynek a licence nem tisztázott, így a TeX Live nem tartalmazza. Ha mégis ezt a fontot akarja használni, akkor töltsse le a CTAN-ről az [yinit-as.zip](#) fájlt ([klikk ide](#)), csomagolja ki, majd a benne található `yinitas.mf` fájlt másolja be az aktuális dokumentum könyvtárába. Másik megoldás, hogy a kicsomagolt `yinit-as.zip` mappa tartalmát bemásolja a következő helyekre:

```
C:\texlive\texmf-local\fonts\tfm\yinit-as
C:\texlive\texmf-local\fonts\source\yinit-as
C:\texlive\texmf-local\doc\yinit-as
C:\texlive\texmf-local\tex\latex\yinit-as
```

Majd írja parancssorba, hogy

```
mktextlsr
```

végül . Ezután a kódban már ne szerepeljen a `\def\initdefault{yinit}` sor, különben hiába telepítette az yinit-as fontokat.

15.5. Betűk kontúrozása és árnyékolása

A betűk kontúrozására a `contour` csomag használható `outline` opcióval. Ez automatikusan betölti a `color` csomagot is, ami az `xcolor`-nak egy kisebb tudású verziója. Ha ki akarja használni az `xcolor` lehetőségeit, akkor azt is töltsse be. A

```
\contourlength{<méret>} ∈ contour
```

parancssal a kontúr vastagságát állíthatja be, ahol a `<méret>` alapértéke 0.03em. A kontúrozás parancsa:

```
\contour{<színnév>}{<szöveg>} ∈ contour
```

Például

```
\contourlength{1pt}
\contour{blue}{\Huge\bfseries\color{white}SZÖVEG}
```

SZÖVEG

Szöveg árnyékolásához használja a `shadowtext` csomagot:

```
\shadowtext{<szöveg>} ∈ shadowtext
```

Például

```
\shadowtext{Árnyékolt szöveg}
```

Árnyékolt szöveg

A korábban ismertetett `xcolor` csomaggal az árnyék színe is beállítható:

```
\shadowcolor{<szín>} ∈ shadowtext
```

Például

```
\shadowcolor{blue!40!white}
\shadowtext{Árnyékolt szöveg}
```

Árnyékolt szöveg

Az árnyék távolsága a következő parancsokkal állítható be:

```
\shadowoffset{<távolság>} ∈ shadowtext
\shadowoffsetx{<távolság>} ∈ shadowtext
\shadowoffsety{<távolság>} ∈ shadowtext
```

Például

```
\shadowoffset{2pt}
\shadowtext{Árnyékolt szöveg}
```

Árnyékolt szöveg

```
\shadowoffsetx{4pt}
\shadowoffsety{2pt}
\shadowtext{Árnyékolt szöveg}
```

Árnyékolt szöveg

15.6. Alá- és fölhúzás egyszerre

```
\overunderline{Egy érdekes kiemelés}
```

Egy érdekes kiemelés

Az `\overunderline` parancs alaphól nincs definiálva. A használatához írja a preambulumba a következőket:

```
\usepackage{calc}
\usepackage[outline]{contour}
\newlength{\ruleht}
\newlength{\rulesep}
\newcommand{\overunderline}[1]{%
  \leavevmode
  \begin{group}
    \setbox1=\hbox{#1}%
    \setbox2=\hbox{m}%
    \contourlength{1pt}%      kontúrvastagság
    \setlength{\ruleht}{0.4pt}% vonalvastagság
    \setlength{\rulesep}{1.2pt}% a vonalak és az "m" betű távolsága
    \rlap{\rlap{\rule[-\rulesep-\ruleht]{\wd1}{\ruleht}}%
      \rule[\ht2+\rulesep]{\wd1}{\ruleht}}%
    \contour{white}{\copy1}%
  \endgroup
}
```

15.7. Dátumtípusok automatikus toldalékolása

Tegyük fel, hogy a dokumentum fordításának dátuma 2018. december 7. Ekkor

2018. december 7-én	<code>\ontoday ∈ [magyar]babel</code>
2018. december 7-én	<code>\ondate magyar ∈ [magyar]babel</code>
2018. december 7-én	<code>\emitdate[a+an]{g}{\today} ∈ [magyar]babel</code>
2018. december 7-e	<code>\emitdate[e]{g}{\today} ∈ [magyar]babel</code>

Rögzített dátumok esetén:

1848. március 15-én `\emitdate[a+an]{g}{1848-3-15} ∈ [magyar]babel`
 1848. március 15-e `\emitdate[e]{g}{1848-3-15} ∈ [magyar]babel`

15.8. Számok automatikus toldalékolása

Ehhez használja a `\told ∈ [magyar]babel` parancsot. Ha automatikus névelőt is akar elé tenni, akkor pedig az `\atold ∈ [magyar]babel` illetve `\Atold ∈ [magyar]babel` parancsokat. A lehetséges toldalékok: `a as ad adik an at on nal ul val hoz ban nak ba ra tol rol szor`. Például

```
\atold\ref{sec-a}+at{}\
\told\ref{sec-a}+as{}\
\told\ref{sec-a}+ad+szor{}\
\told(\ref{eq-c})+at
```

a 3-at
 3-as
 3-adszor
 (1)-et

ahol `\ref{sec-a}` kifejtése 3 és `\ref{eq-c}` kifejtése 1. Az utolsó sorban nem használható a `\told\eqref{eq-c}+at` parancs!

15.9. Lorem ipsum

Az XVI. században egy ismeretlen nyomdász egy latint utánzó összefüggő értelmetlen szöveget kreált a különböző nyomdai elrendezések bemutatására, amit azért alkalmazott, mert az ember önkéntelenül elkezd olvasni a számára értelmes szöveget, így nem tudva elvonatkoztatni attól és az elrendezésre koncentrálni. Ezt a nyomdászatban és az informatikában a mai napig is használják a betűtípusok, a tipográfia és az elrendezés bemutatására. Nagyszerűsége abban rejlik, hogy ebben a szövegben található betűk és betűközök kombinációjában láthatóak a legszebben a betűtípusok fontosabb jellemzői. Az angoléhoz és a magyaréhoz hasonló betűelosztása van, amely szintén segít abban, hogy az emberek ne a tartalmat figyeljék.

A szöveget CICERO *De finibus bonorum et malorum* („A legfőbb jóról és rosszról”) című műve néhány bekezdésének véletlenszerűen összevágott szavaiból alakították ki. Így tehát nincs értelmes jelentése, sokszor még a szavaknak sem.

Ez a szöveg az ún. *lorem ipsum* (röviden: *lipsum*), amely L^AT_EX-ben egyszerűen generálható a `lipsum` csomag segítségével:

```
\lipsum[<szám1>-<szám2>] ∈ lipsum
```

Ekkor a lorem ipsum szövege jelenik meg a `<szám1>`-edik bekezdéstől a `<szám2>`-edik bekezdésig. Az utóbbi maximális értéke 150 lehet. A `\lipsum` parancs opciójának alapértéke: 1–7.

```
\lipsum[<szám>] ∈ lipsum
```

Ekkor a lorem ipsum `<szám>`-edik bekezdése jelenik meg.

A *lorem ipsum* számos változata ismert különböző nyelveken. A magyar verziót 2016-ban Nagy Viktor és Takács Dávid dolgozták ki, melynek a *Lórum ipse* címet

adták (lásd <http://www.lorumipse.hu/>). Ennek a szövegét a `hulipsum` csomaggal jeleníthetjük meg \LaTeX -ben. A csomag betöltése után a `\hulipsum` parancs ugyanúgy használható, mint az előbb ismertetett `\lipsum`.

Az angol verzió a `kantlipsum` csomaggal érhető el. Ekkor a `\kant` parancs használható hasonló opciókkal, mint a `\lipsum`.

15.10. \TeX -hel kapcsolatos logók

```
\TeX, \LaTeX, \LaTeXe
\AmS ∈ amsmath
\MF, \MP ∈ mflogo
\XeTeX, \XeLaTeX, \LuaTeX, \LuaLaTeX ∈ metalogo
\amslogo, \bibtexlogo, \metafontlogo{ }{ } ∈ texlogos
```

A METAFONT logót az `mflogo` illetve `texlogos` csomagok nélkül a következőképpen lehet definiálni:

```
\font\mffont=manfnt
\def\MF{{\mffont METAFONT}}
```

A `metalogo` nélkül a következőképpen lehet definiálni \XeLaTeX logót a `graphicx` csomag betöltése után:

```
\DeclareRobustCommand{\XeLaTeX}
{X\kern-.11em\lower.5ex\hbox{\scalebox{-1}[1]{E}}\kern-.11em\LaTeX}
```

15.11. Prímszámok kiírása

A következő kód Donald Ervin Knuth-tól származik (lásd [2]):

```
\newif\ifprime \newif\ifunknown
\newcount\n \newcount\p \newcount\d \newcount\a
\def\primes#1{2,~3\n=#1 \advance\n by-2 \p=5
\loop\ifnum\n>0 \printifprime\advance\p by2 \repeat}
\def\printp{\, \number\p \advance\n by -1 }
\def\printifprime{\testprimality \ifprime\printp\fi}
\def\testprimality{{\d=3 \global\primetrue
\loop\trialdivision \ifunknown\advance\d by2 \repeat}}
\def\trialdivision{\a=\p \divide\a by\d
\ifnum\a>\d \unknowntrue\else\unknownfalse\fi
\multiply\a by\d
\ifnum\a=\p \global\primefalse\unknownfalse\fi}
```

Ez definiálja a `\primes{<szám>}` parancsot, amely kiírja az első *<szám>* darab prímszámot, ahol a *<szám>* értéke legalább 2. Például az első 50 prím így listázható ki:

```
\primes{50}
```

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229

15.12. Vonalazott lapok

Ha vonalazott lapot szeretne előállítani, akkor írja preambulumba a következőket:

```
\usepackage{picture,xcolor,atbegshi}
\AtBeginShipout{%
  \AtBeginShipoutUpperLeft{%
    {\color{blue}%
      \put(\dimexpr 1in+\oddsidemargin,
        -\dimexpr 1in+\topmargin+\headheight+\headsep+\topskip)%
      {%
        \vtop to\dimexpr\vsizet+\baselineskip{
          \hrule
          \leaders\vbox to\baselineskip{\hrule width\hsize\vfill}\vfill
        }%
      }%
    }%
  }%
}
```

15.13. Négyzetrácsos lapok

Ha négyzetrácsos lapot szeretne előállítani, akkor írja preambulumba a következőket:

```
\usepackage{tikz,eso-pic}
\AddToShipoutPicture{%
\begin{tikzpicture}[remember picture,overlay]
\tikzset{normal lines/.style={black!20,very thin}}
\node at ([yshift=2mm] current page.south west){
\begin{tikzpicture}[remember picture,overlay]
\draw[style=normal lines,step=5mm](0,0)grid(\paperwidth,\paperheight);
\end{tikzpicture}};
\end{tikzpicture}}
```

15.14. Az oldal két pontjának összekötése vonallal

Ehhez töltsse be a preambulumba a következőket:

```
\usepackage{tikz}
\newcommand{\Node}[2]{\tikz[remember picture,inner sep=0pt,outer sep=0pt,
baseline=(#1.base)]\node(#1){#2};}
\newcommand{\Draw}[4][\tikz[remember picture,overlay]
\draw[#1](#3)#2(#4);\ignorespaces}
```

Ezután például

```
Először \Node{A}{innen} húzunk egy piros nyilat a következő egyenlet
egyenlőségjeléhez.
\[5x^2+2x\Node{B}{\{=\}=\{=\}5.}\]
Most \Node{C}{\fbox{innen}} húzunk egy kék nyilat az előző „innen”
szóhoz. Végül pedig \Node{D}{ettől} a ponttól húzunk egy rózsaszínű
nyilat az egyenlőségjelhez, illetve egy zöld nyilat a következő
táblázat első sorának második oszlopához.
```

```

\begin{center}
\begin{tabular}{|c|c|}
\hline
1 & \Node{E}{ide} \\
\hline
2 & 3 \\
\hline
\end{tabular}
\end{center}
\Draw[->,color=red]{to[out=-30,in=130]}{A}{B}
\Draw[->.>,>=stealth,color=blue]{to[bend left]}{C}{A}
\Draw[<.->,color=green,>=latex,line width=4pt,opacity=.5]{to}{D}{E}
\Draw[->,color=pink,line width=2pt]{--+(0mm,3mm)-|}{D}{B}

```

Először innen húzunk egy piros nyilat a következő egyenlet egyenlőségjeléhez.

$$5x^2 + 2x = 5.$$

Most innen húzunk egy kék nyilat az előző „innen” szóhoz. Végül pedig ettől a ponttól húzunk egy rózsaszínű nyilat az egyenlőségjelhez, illetve egy zöld nyilat a következő táblázat első sorának második oszlopához.

1	ide
2	3

Tehát a következő két parancsot használhatja:

```

\Node{<név>}{<szöveg>}
\Draw[<nyíl típusa>]{<vonala alakja>}{<név1>}{<név2>}

```

A `\Node` kijelöli a pontokat, a `\Draw` pedig összeköti őket. A `<vonala alakja>` azt adja meg, hogy az összekötő vonal milyen alakú legyen:

`to` Egyenes vonal.

`to[bend left]` Íves vonal, amely balra kanyarodva indul.

`to[bend right]` Íves vonal, amely jobbra kanyarodva indul.

`to[out=<szög>,in=<szög2>]` Íves vonal, amely `<szög1>` fokos szögben indul és `<szög2>` fokos szögben érkezik.

`--+(<koord1>mm,<koord2>mm)-|` Törött vonal, amelynek kezdő pontja össze van kötve a hozzá relatív `(<koord1>mm,<koord2>mm)` koordinátájú ponttal, amit egy vízszintes, majd egy függőleges vonal követ.

A `<nyíl típusa>` opciók (alapértelmezésben nincs nyíl, csak vonal):

`->` A nyíl `<név2>` felé mutat.

`<-` A nyíl `<név1>` felé mutat.

`<->` A nyíl `<név1>` és `<név2>` felé is mutat.

`color=<szín>` A vonal színe.

`>=<nyílvég>` A nyílvég alakja. (Pl. `>=latex`, `>=stealth`, stb. Bővebben lásd a `tikz` csomag leírásában.)

`line width=<vastagság>` A vonal vastagsága. Alapértéke 0.4pt.

`opacity=<szám>` Átlátszóság értéke. A `<szám>` egy 0 és 1 közötti érték. Minél kisebb az érték, annál átlátszóbb. Alapértéke 1.

15.15. Nem vízszintes alapvonalú szöveg szedése

Írja a preambulumba a következőket:

```
\usepackage{tikz}
\usetikzlibrary{decorations.text}
```

Ezután a dokumentumtestbe ezt írja:

```
% ZÖLD SZÖVEG
\begin{tikzpicture}[baseline=-3pt]
\path [decorate,
        decoration={text effects along path,
                    text={SZ{Ö}VEG},
                    text effects/.cd,
                    scale text to path},
        text effects={text=green,
                    character widths={inner xsep=1pt}}]
(0,0) -- (3,2);
\end{tikzpicture}
% KÉK SZÖVEG
\begin{tikzpicture}[baseline]
\path [decorate,
        decoration={text effects along path,
                    text={SZ{Ö}VEG},
                    text effects/.cd,
                    text along path,
                    scale text to path},
        text effects={text=blue,
                    character widths={inner xsep=0pt}}]
(0,0) -- (3,2);
\end{tikzpicture}
% SÁRGA SZÖVEG
\begin{tikzpicture}[baseline]
\path [decorate,
        decoration={text effects along path,
                    text={SZ{Ö}VEG},
                    text effects/.cd,
                    text along path,
                    scale text to path,
                    characters={font=\color{yellow},
                                xslant=0.6666}},
        text effects={character widths={inner xsep=0pt}}]
(0,0) -- (3,2);
\end{tikzpicture}
```

The image shows three examples of the text "SZÖVEG" rendered using TikZ's text effects. Each example is contained within a red rectangular border. The first example is green, the second is blue, and the third is yellow. All three examples are slanted and rotated, demonstrating the 'text effects' library's capabilities.

A következő példához írja be a preambulumba a következőket:

```
\usepackage{tikz}
\usetikzlibrary{decorations.text}
\usepackage[outline]{contour}
```

Ezután a dokumentumtestbe ezt írja:

```
\begin{tikzpicture}[baseline]
\def\mycontour#1{\contour{red}{#1}}
\path[decorate,
  decoration={text effects along path,
    text={SZ{Ö}VEG},
    text effects/.cd,
    text along path,
    scale text to path,
    characters={font=\color{white},
      character command=\mycontour,
      xslant=0.6666}},
  text effects={character widths={inner xsep=0pt}}]
(0,0) -- (3,2);
\end{tikzpicture}
```



A következő példához írja be a preambulumba a következőket:

```
\usepackage{tikz}
\usetikzlibrary{decorations.text}
```

Ezután a dokumentumtestbe ezt írja:

```
\begin{tikzpicture}[baseline]
\path [decorate,
  decoration={text effects along path,
    text={SZ{Ö}VEG SZ{Ö}VEG SZ{Ö}VEG SZ{Ö}VEG SZ{Ö}VEG
      SZ{Ö}VEG},
    text effects/.cd,
    text along path,
    scale text to path},
  text effects={text=magenta,
    character widths={inner xsep=0pt}}]
(0,0)..controls (3,3) and (6,-3)..(9,0);
\end{tikzpicture}
```



Ügyeljen arra, hogy az előző példákban az ékezetes betűket kapcsos zárójelek közé kell tenni. Pl.: SZ{Ö}VEG.

15.16. Vízjel

Írja a preambulumba a következőket:

```
\usepackage{eso-pic,tikz,graphicx}
\AddToShipoutPictureBG{%
\begin{tikzpicture}[overlay]
\node[rotate=<szög>,color=<színnév>] at (current page.center)
{\resizebox{!}{<magasság>}{<szöveg>}};
\end{tikzpicture}}
```

Ezután a dokumentum minden oldalának közepén, a háttérben megjelenik a *<szöveg>*, amely *<magasság>* magas, *<színnév>* színű és el van forgatva *<szög>* fokkal. Például a következő kód hatására a dokumentum minden oldalán azt fogjuk látni vízjelként, amit most ezen az oldalon:

```
\AddToShipoutPictureBG{%
\begin{tikzpicture}[overlay]
\node[rotate=30,color=blue!30] at (current page.center)
{\resizebox{!}{3cm}{\sffamily\bfseries VÍZJEL}};
\end{tikzpicture}}
```

Hasonló hatás érhető el a következők preambulumba írásával is:

```
\usepackage{eso-pic,xcolor,graphicx}
\newsavebox{\mybox}
\newlength{\myboxheight}
\newlength{\myboxwidth}
\sbox{\mybox}{\rotatebox{<szög>}{\resizebox{!}{<magasság>}{%
\color{<színnév>}<szöveg>}}}
\settowidth{\myboxwidth}{\usebox{\mybox}}
\settoheight{\myboxheight}{\usebox{\mybox}}
\AddToShipoutPictureBG{\AtPageCenter{%
\hspace{-.5\myboxwidth}\lower.5\myboxheight\hbox{\usebox{\mybox}}}}
```

15.17. A pdf készítésének ideje óra percben

Ennek megjelenítéséhez először írja a következőket a preambulumba:

```
\newcount\hour \newcount\minute
\hour=\time \divide \hour by 60
\minute=\time
```

```
\loop \ifnum \minute > 59 \advance \minute by -60 \repeat
\def\hourminute{\number\hour:{\ifnum\minute<10 0\fi}\number\minute}
```

Tegyük fel, hogy a dokumentum fordításának időpontja 609 perc, azaz 10 óra 9 perc. Ekkor

```
609    \number\time
10     \number\hour
9      \number\minute
10:09  \hourminute
```

15.18. QR-kód

QR-kód könnyen generálható a `qrcode` csomaggal:

```
\qrcode[height=<magasság>]{<URL-cím>} ∈ qrcode
```

Például

```
\qrcode[height=25mm]{https://uni-eszterhazy.hu/}
```



15.19. Vonalkód

Vonalkód generálásához használhatja a `GS1` csomagot. Egyelőre ezzel még csak EAN-8 és EAN-13 szabványú vonalkódok generálhatók, de a későbbiekben várható ezeknek a kiterjesztése. Használata a következő:

```
\EANBarcode[module_height=<magasság>]{<szám>}
```

Például

```
\EANBarcode[module_height=2cm]{ISBN 978-615-5297-19-9}
```



Több szabványt ismer a `pst-barcode` csomag, de ez csak `latex` és `xelatex` fordítókkal működik alapesetben. Ha `pdflatex` vagy `lualatex` fordítót használ, akkor azt tegye a `-shell-escape` kapcsolóval, továbbá még az `auto-pst-pdf` csomagot is töltsse be `pdfcrop={- hires}` opcióval. Ekkor vonalkód a következő módon állítható elő:

```
\begin{pspicture}(<szélesség>in,<magasság>in)
\psbarcode{<szám>}{includetext width=<szélesség> height=<magasság>}{<típus>}
\end{pspicture}
```

A *szélesség* illetve a *magasság*, a vonalkód szélességének illetve magasságának mértékszáma inch-ben mérve. Például

```
\begin{pspicture}(1.5in,1in)
\psbarcode{1787-6117}{includetext width=1.5 height=1}{issn}
\end{pspicture}
\hspace{1cm}
\begin{pspicture}(1in,1in)
\psbarcode{01335583}{includetext width=1 height=1}{ean8}
\end{pspicture}
```



15.20. Különleges bekezdések

Érdekes bekezdések készíthetők a **shapepar** csomaggal. Egyik parancsa

\diamondpar{<szöveg>} ∈ shapepar

Például

```
\diamondpar{a á b c d e é f g h i í j k l m n
           o ó ö ő p q r s t u ú ü ű v z x y}
```



A **shapepar** csomag segítségével más formájú bekezdések is készíthetők, sőt magunk is tervezhetünk újakat.

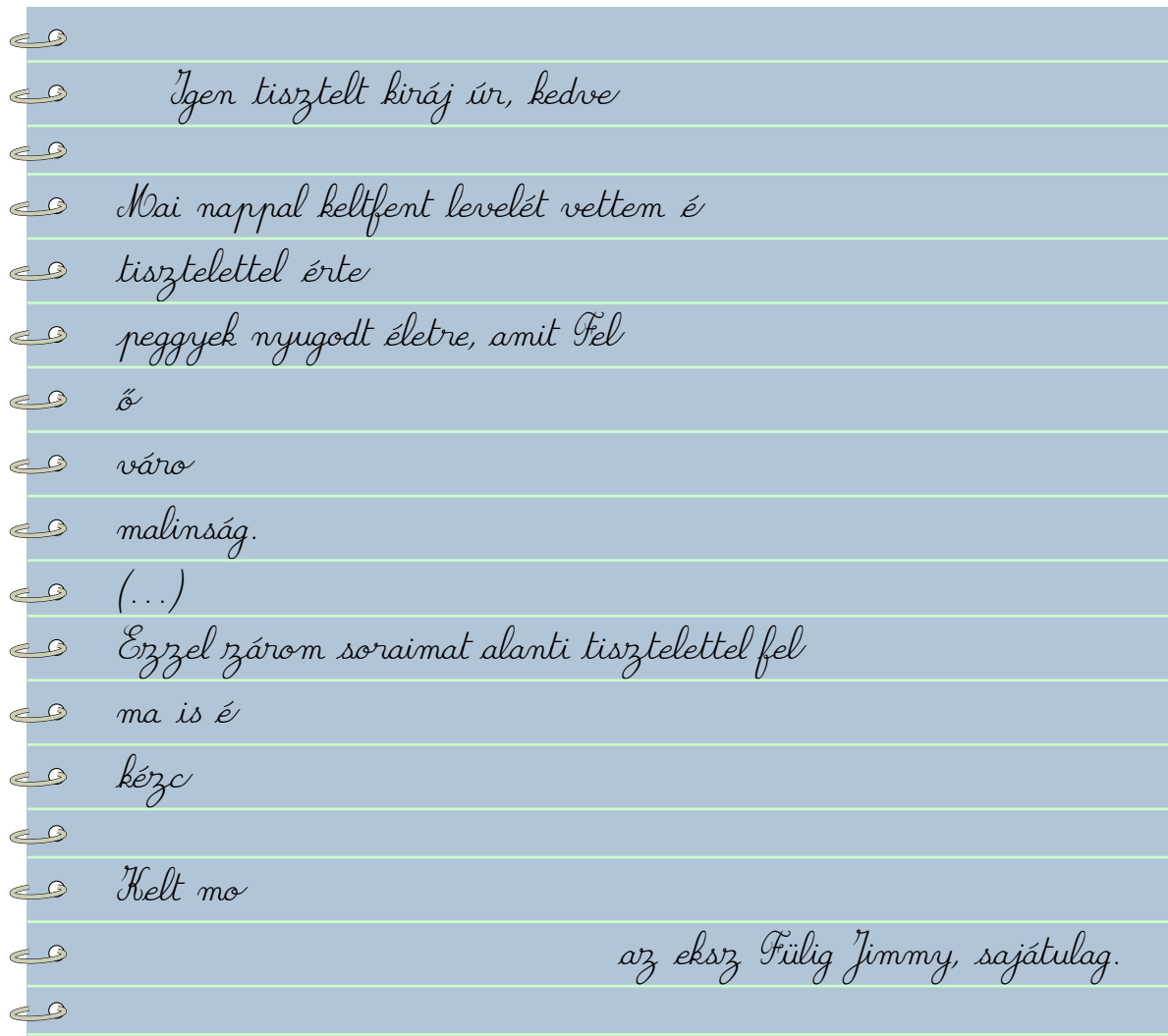
Különleges bekezdések készíthetők a **fancy par** csomaggal is. Példaként megmutatunk egy részletet Fülíg Jimmy leveléből az uralkodóhoz (Rejtő Jenő: Piszkos Fred, a kapitány):

```
\NotebookPar{%
\usefont{T1}{frc}{m}{sl}
\mbox{}\
\mbox{}\hfill
Igen tisztelt kiráj úr, kedves mamája és T. neje őfelsége!
\hfill\mbox{}\
Mai nappal keltfent levelét vettem és kibontám. Ezennel felelek
tisztelettel értesíteni! Szíves mekhívására, hogy udvarára telepeggyek
```

```

nyugodt életre, amit Felség gondtalanít, van szerencsém őszinte
sajnálattal. Mer ott nekem nagy strapa a tétlenség. Én városi lakós
vagyok, ha nem is bejelentett, ami csak egy üres formalinság.\\
(\dots)\\
Ezzel zárom soraimat alanti tisztelettel felségednek régi barátja,
ma is és a kedves mama őfelségének és az uralkodó őnacsságának
kézcsókkal.\\ \\ \\
Kelt most lent: Néhai kollégája:\\
\mbox{}\hfill az eksz Fülíg Jimmy, sajátulag.\\}

```



15.21. Vágójelek nyomdai előkészítéshez

Amikor elkészítette a pdf fájlt, ami majd a nyomdába kerül, célszerű az oldalakra vágójeleket tenni a nyomdászok részére, ami megkönnyíti a munkájukat. Ehhez készítse el a következő tartalmú tex fájlt, tegye mellé a vágójelekkel ellátandó pdf fájlt, majd fordítsa le a tex fájlt.

```

\documentclass{minimal}
\usepackage{pdfpages,tikz}
\setlength{\paperwidth}{\langle pdf szélessége \rangle+30mm}
\setlength{\paperheight}{\langle pdf magassága \rangle+30mm}

```



```
\begin{document}
\AddToShipoutPictureFG{\begin{tikzpicture}[overlay]
\draw ([yshift= 15mm] current page.south west) -- +(10mm,0);
\draw ([xshift= 15mm] current page.south west) -- +(0,10mm);
\draw ([yshift= 15mm] current page.south east) -- +(-10mm,0);
\draw ([xshift=-15mm] current page.south east) -- +(0,10mm);
\draw ([yshift=-15mm] current page.north west) -- +(10mm,0);
\draw ([xshift= 15mm] current page.north west) -- +(0,-10mm);
\draw ([yshift=-15mm] current page.north east) -- +(-10mm,0);
\draw ([xshift=-15mm] current page.north east) -- +(0,-10mm);
\node at ([yshift=-5mm] current page.north) {\textcolor{green}{információ}};
\end{tikzpicture}}
\includepdf[noautoscale,pages=1-]{\textcolor{green}{pdf fájl}}
\end{document}
```

16. fejezet

Strukturált művek

Hosszabb, strukturált dokumentumokat a következő módon szoktuk tagolni:

- cím
- kivonat (`book` osztályban nincs)
- tartalomjegyzék
- úszó objektumok jegyzéke
- főszöveg szintjei
 - részek
 - fejezetek (`article` osztályban nincs)
 - szakaszok
 - alszakaszok
 - al-alszakaszok
 - paragrafusok
 - alparagrafusok
 - bármelyiken belül tételszerű bekezdések
- függelék
- bibliográfia
- tárgymutató

16.1. Főcím, címlap, kivonat

A dokumentumtestbe írja a következőket:

```
\title{<cím>}  
\author{<szerző>}  
\date{<dátum>}  
\maketitle
```

A dátumot automatikusra is veheti a `\date{\today}` paranccsal. Ekkor a fordítás kezdetén aktuális dátum jelenik meg. Ezen parancsok argumentumaiba lábjegyzetek is írhatók a `\thanks{<szöveg>}` paranccsal. Ez a `\footnote`-tól függetlenül számoz. A `\maketitle` a rendelkezésre álló adatokból elkészíti a címet. Ha a `\documentclass` parancs `titlepage` opcióval lett betöltve, akkor ezt külön oldalra teszi.

Ezután nyithat egy `abstract` környezetet (kivéve a `book` osztályt), melybe a mű rövid kivonatát írhatja.

16.2. A főszöveg szintjei

A főszöveg szintjeinek hierarchiáját a következő táblázat foglalja össze:

név	parancs	szintszám	
		article	book/report
rész	<code>\part[⟨rövid cím⟩]{⟨cím⟩}</code>	0	−1
fejezet	<code>\chapter[⟨rövid cím⟩]{⟨cím⟩}</code>	—	0
szakasz	<code>\section[⟨rövid cím⟩]{⟨cím⟩}</code>	1	1
alszakasz	<code>\subsection[⟨rövid cím⟩]{⟨cím⟩}</code>	2	2
al-alszakasz	<code>\subsubsection[⟨rövid cím⟩]{⟨cím⟩}</code>	3	3
paragrafus	<code>\paragraph[⟨rövid cím⟩]{⟨cím⟩}</code>	4	4
alparagrafus	<code>\subparagraph[⟨rövid cím⟩]{⟨cím⟩}</code>	5	5

A *⟨cím⟩* az adott szint címe, míg a *⟨rövid cím⟩* az a cím, ami a tartalomjegyzékben és a fejlécben jelenik meg. Ennek alapértéke a *⟨cím⟩*, azaz ha nem adja meg, akkor a tartalomjegyzékbe és fejlécbe az a cím kerül, ami a szövegbe is. Ha a szövegben, tartalomjegyzékben és a fejlécbe is más-más címet akar kiírni, akkor használja a következő kódot:

```
\chapter[⟨tartalomjegyzékbeli cím⟩]{⟨cím⟩}\chaptermark{⟨fejlécbeli cím⟩}
\section[⟨tartalomjegyzékbeli cím⟩]{⟨cím⟩}\sectionmark{⟨fejlécbeli cím⟩}
\subsection[⟨tartalomjegyzékbeli cím⟩]{⟨cím⟩}\subsectionmark{⟨fejlécbeli cím⟩}
```

A szintek automatikusan sorszámot kapnak a `secnumdepth` számláló által megadott szintszámig (article-ben ez 3, a többiben 2.) Ha ezen változtatni akar, például még a paragrafusokat is szeretné számozni (melynek 4 a szintszáma), akkor írja be a következőt a preambulumba:

```
\setcounter{secnumdepth}{4}
```

Ha egy számozott szint esetén csak egyetlen szintnek nem akar sorszámot, akkor használja az előző parancsok ún. csillagos változatát (például `\section*{⟨cím⟩}`). Ilyenkor a cím nem kerül a tartalomjegyzékbe és a fejlécbe sem.

A részek számozása alapesetben római számozással, a fejezeteké pedig arab számozással történik. Lehetőség van magyar nyelv esetén arra, hogy a számozás betűzve jelenjen meg. Ehhez a `magyar.ldf` fájlt

```
\partnumber=Huordinal,chapternumber=Huordinal
```

opciókkal kell betölteni. Ekkor például

```
\part{A rész címe}
```

Első rész
A rész címe

```
\chapter{A fejezet címe}
```

Első fejezet
A fejezet címe

A szintekre pontosan úgy lehet hivatkozni, mint azt az általános esetre leírtuk. Például

```
\subsection{Ez az alszakasz címe}\label{subsec-pelda}
...
Lásd \aref{subsec-pelda}.~alszakaszban.
```

1.1. Ez az alszakasz címe

...
Lásd az 1.1. alszakaszban.

A szintek címének megjelenési formáját szabályozhatja is a `titlesec` csomaggal. Ezt itt nem részletezzük, csak egy példát említünk, hogyan lehet fejezet címet középre igazítani. A dokumentumtörzsbe írja be a következőt:

```
\titleformat{\chapter}[display]{\normalfont\bfseries\filcenter}
{\huge\thechapter.~\chaptertitlename}{20pt}{\Huge} ∈ titlesec
```

A book osztályban további három parancs van a könyv szerkezetének kialakítására:

`\frontmatter` római számozású oldalak, fejezetek sorszám nélkül,
`\mainmatter` arab számozású oldalak 1-től,
`\backmatter` fejezetek sorszám nélkül.

Ezek elhelyezkedése:

```
\begin{document}
<cím>
\frontmatter
<jegyzékek, előszó, bevezetés>
\mainmatter
<szöveg fő része, függelék>
\backmatter
<bibliográfia, tárgymutató>
\end{document}
```

A magyar tipográfiában `\frontmatter` esetén az oldalszámozás nagy római számokkal történik, míg az angolban kis római számokkal. Ezt a `magyar.ldf` nem kezeli. A hibát úgy tudja javítani, hogy a `\frontmatter` után kiadja a `\pagenumbering{Roman}` parancsot is.

16.3. Fattyúsorok

A tipográfiában a hosszú dokumentumok tördelésének súlyos hibája az úgynevezett fattyúsor. Két előfordulása van,

özvegy sor: egy oldal vagy hasáb egy bekezdés utolsó sorával kezdődik;
árvasor: egy oldal vagy hasáb utolsó sorában kezdődik egy bekezdés.

Ezek letiltására használja a `nowidow` csomagot `all` opcióval. Ezzel azt is be lehet állítani, hogy az oldal vagy hasáb tetején illetve alján egy bekezdésnek minimum hány sora legyen, ha az egyáltalán lehetséges. Például ha azt akarja, hogy ez a szám 4 legyen, akkor használja még a `defaultlines=4` opciót is.

16.4. Fej- és láblécek

16.4.1. Alapbeállítások

Egy hosszabb dokumentumban célszerű, ha minden oldalon találunk utalást arra, hogy az a dokumentum mely részén van: hányadik oldalon, melyik szinten és melyik alszinten. Ezek `book` osztályban automatikusan megjelennek. A másik két osztályban (`article`, `report`) ehhez adja ki a

■ `\pagestyle{headings}`

parancsot. Ennek hatása:

- lábléc üres
- oldalszám a fejléc külső margójánál
- szint információi a fejlécben
 - egyoldalas szedésnél minden oldalon a belső margónál
 - kétoldalas szedésnél páros oldalon a belső margónál
- alszint információi a fejlécben
 - egyoldalas szedésnél nincs
 - kétoldalas szedésnél páratlan oldalon a belső margónál.

További oldalstílusok:

`\pagestyle{empty}` Üres fej- és lábléc.

`\pagestyle{plain}` Üres fejléc, a lábléc közepén oldalszám.

`\pagestyle{myheadings}\markboth{<infó1>}{<infó2>}`

- lábléc üres
- oldalszám a fejléc külső margójánál
- `<infó1>` a fejlécben
 - egyoldalas szedésnél nincs
 - kétoldalas szedésnél páros oldalon a belső margónál
- `<infó2>` a fejlécben
 - egyoldalas szedésnél minden oldalon a belső margónál
 - kétoldalas szedésnél páratlan oldalon a belső margónál
- `<infó1>` és `<infó2>` bármikor megváltoztatható a `\markboth` paranccsal. Külön csak az `<infó2>` is megadható a `\markright{<infó2>}` paranccsal.

Ha egy konkrét oldalra vonatkozóan meg akarja az oldalstílust változtatni, akkor az adott szövegrészhez gépelje be:

■ `\thispagestyle{<stílus>}`

ahol a `<stílus>`: `headings`, `myheadings`, `empty` vagy `plain`. A `report` és `book` osztályokban az új részt és az új fejezetet nyitó oldalak `plain` stílusra váltanak, majd a következő oldaltól visszatér az eredeti stílusra.

Ha a `book` vagy `report` osztályt `openright` opcióval töltötte be, akkor előfordulhat, hogy a dokumentumban lesz üres oldal. A magyar tipográfiai szabály előírja, hogy ezeken az oldalakon a fej- és láblécnek is üresnek kell lennie. A `magyar.ldf` ezt nem oldja meg. Ennek eléréséhez tölts be az `emptypage` csomagot. Ha mi kényszerítettünk ki üres oldalt, akkor oda írja be a

■ `\thispagestyle{empty}`

parancsot.

16.4.2. Fej- és láblécek testreszabása

A fej- és láblécek beállításaira a következő parancsok használhatók:

`\thepage` kiírja az aktuális oldalszámot.

`\thechapter` kiírja az aktuális fejezet számát.

`\thesection` kiírja az aktuális szakasz számát.

`\thesubsection` kiírja az aktuális alszakasz számát.

`@chapapp` kiírja az aktuális fejezet címkéjét: „fejezet” vagy „függelék”.

`\markboth{<infó1>}{<infó2>}` hatására a `\leftmark` kifejtése `<infó1>`, míg a `\rightmark` kifejtése `<infó2>` lesz.

`\markright{<infó>}` hatására a `\rightmark` kifejtése `<infó>` lesz. Az aktuális szint és al-szint információit a `\rightmark` és `\leftmark` parancsok tárolják. A `headings` stílus ezt a következő táblázat szerint teszi:

	article		report/book	
	egyoldalas	kétoldalas	egyoldalas	kétoldalas
<code>\leftmark</code>		szakasz		fejezet
<code>\rightmark</code>	szakasz	alszakasz	fejezet	szakasz

`\chaptermark{<cím1>}` a `\chapter[<cím1>]{<cím2>}` parancs kiadásakor végrehajtódik.

`\sectionmark{<cím1>}` a `\section[<cím1>]{<cím2>}` parancs kiadásakor végrehajtódik.

`\subsectionmark{<cím1>}` a `\subsection[<cím1>]{<cím2>}` parancs kiadásakor végrehajtódik.

`@oddfoot` tartalma kerül a láblécbe egyoldalas nyomtatás esetén minden oldalon, kétoldalas nyomtatás esetén a páratlan oldalon.

`@evenfoot` tartalma kerül a láblécbe kétoldalas nyomtatás esetén a páros oldalon.

`@oddhead` tartalma kerül a fejlécbe egyoldalas nyomtatás esetén minden oldalon, kétoldalas nyomtatás esetén a páratlan oldalon.

`@evenhead` tartalma kerül a fejlécbe kétoldalas nyomtatás esetén a páros oldalon

`\ps@<stílusnév>` a `\pagestyle{<stílusnév>}` illetve `\thispagestyle{<stílusnév>}` parancs kiadásakor végrehajtódik.

A következő példában definiálunk egy saját nevű stílust, melynek `\pagestyle{sajat}` parancssal történő bekapcsolása után a következő beállítások érvényesülnek `report` vagy `book` osztály esetén:

- lábléc üres
- oldalszám a fejléc külső margójánál
- fejezet információi a fejlécben
 - egyoldalas szedésnél minden oldalon a belső margónál
 - kétoldalas szedésnél páros oldalon a belső margónál
- szakasz információi a fejlécben
 - egyoldalas szedésnél nincs
 - kétoldalas szedésnél páratlan oldalon a belső margónál.

```
\makeatletter
\def\ps@sajat{%
\def\chaptermark##1{\markboth{%
\thechapter.\@chapapp.\enspace##1}{}}
\def\sectionmark##1{\markright{\thesection.\enspace##1}}
```

```
\def\@oddfoot{}
\def\@evenfoot{}
\def\@oddhead{\rightmark\hfill\thepage}
\def\@evenhead{\thepage\hfill\leftmark}}
\makeatother
```

(A `\makeatletter` és `\makeatother` parancsok csak a `@` jelet tartalmazó parancsok miatt kellenek, lásd később.)

A következő kódot, ha a dokumentumtestbe írja, akkor onnan a következő beállítások érvényesülnek:

- lábléc üres
- oldalszám a fejléc külső margójánál
- szakasz információi a fejlécben
 - egyoldalas szedésnél minden oldalon a belső margónál
 - kétoldalas szedésnél páros oldalon a belső margónál
- alszakasz információi a fejlécben
 - egyoldalas szedésnél nincs
 - kétoldalas szedésnél páratlan oldalon a belső margónál.

```
\makeatletter
\def\sectionmark#1{\markboth{\thesection.\enspace#1}{}}
\def\subsectionmark#1{\markright{\thesubsection.\enspace#1}}
\def\@oddfoot{}
\def\@evenfoot{}
\def\@oddhead{\rightmark\hfill\thepage}
\def\@evenhead{\thepage\hfill\leftmark}
\makeatother
```

A következő példában definiált `\nouppercase` hatástalanítja a `\MakeUppercase` parancsot. A `\HeadRule` aláhúzza a fejléc tartalmát 0.4 pt vastag vonallal, aminek lénia a neve. A lénia és a fejléc szövegének alapvonala 1 ex távolságra lesznek. Ezután betölti a `headings` stílust. Ebben az `\@oddhead`, `\@evenhead` parancsokat átdefiniálja úgy, hogy a szintinformációk ne csupa nagy betűvel jelenjenek meg (mint ahogy ezt ezen kód nélkül tenné), továbbá megjelenik a lénia is.

```
\newcommand{\nouppercase}[1]
  {\let\uppercase\relax\let\MakeUppercase\relax
  \expandafter\let\cename MakeUppercase \endcsname\relax#1}}
\newcommand{\HeadRule}[1]
  {\lower-1ex\hbox{\makebox[\textwidth]{#1}}%
  \llap{\rule{\textwidth}{0.4pt}}}
\pagestyle{headings}
\makeatletter
\def\@oddhead{%
\HeadRule{\nouppercase{\rightmark}\hfill\thepage}}
\def\@evenhead{%
\HeadRule{\thepage\hfill\nouppercase{\leftmark}}}
\makeatother
```

A következő példában szintinformációk nincsenek a fej- és láblécben. Az oldalszám a külső margónál lesz a fejlécben. Végül a `plain` stílust hatástalanítja, hogy a rész és fejezet nyitó oldalakon ne változzon meg az oldalstílus.

```

\makeatletter
\def\@oddfoot{}
\def\@evenfoot{}
\def\@oddhead{\hfill\thepage}
\def\@evenhead{\thepage\hfill}
\def\ps@plain{}
\makeatother

```

Az eddigi példákban még érdemes megadni az oldalszám és a szintinformációk betűtípusát. Például `\thepage` helyett írhatja, hogy `{\normalsize\normalfont\thepage}` vagy `\leftmark` helyett `{\footnotesize\sffamily\leftmark}`.

Az oldalszámozás alapesetben arab számokkal történik. Ennek átállítása a következő parancsokkal történhet:

`\pagenumbering{arabic}` hatására az oldalak arab számokkal jelennek meg.
`\pagenumbering{roman}` hatására az oldalak kis római számokkal jelennek meg.
`\pagenumbering{Roman}` hatására az oldalak nagy római számokkal jelennek meg.
`\pagenumbering{alpha}` hatására az oldalak az angol ábécé kis betűivel számozódnak.
`\pagenumbering{Alpha}` hatására az oldalak az angol ábécé nagy betűivel számozódnak.

Ezek nemcsak az oldalszámozás stílusát változtatják meg, hanem egyúttal annak értékét visszaállítják 1-re.

Amikor `report` vagy `book` osztály esetén a `magyar.ldf` fájl `chapternumber=Huordinal` opcióval van betöltve, akkor a fejlécben a fejezet számozása `Huordinal` típusú (Első, Második, stb.). De ha sok fejezet van, akkor például a „Tizenötödik fejezet” kiírása a címmel együtt már nem biztos, hogy elfér a fejlécben. A következő kód visszaállítja a fejlécben a fejezet számozását arabra:

```

\renewcommand{\chaptermark}[1]{%
\markboth{\MakeUppercase{\arabic{chapter}. fejezet.\ #1}}{}}

```

Amikor `\chapter*`, `\section*` vagy `\subsection*` parancsokat, azaz számozatlan szinteket használunk, akkor azok a `\leftmark` és `\rightmark` parancsokat nem definiálják át az aktuális címre. Így, ha korábban ezek a parancsok már tartalmaztak valamilyen információt, akkor a fejlécben rossz címek jelennek meg. Ennek bemutatására próbálja ki a következő kódot:

```

\documentclass{book}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\begin{document}
\chapter{Számozott fejezet címe}
szöveg\newpage szöveg
\chapter*{Számozatlan fejezet címe}
szöveg\newpage szöveg
\end{document}

```

Ilyen esetekben a `\leftmark` és `\rightmark` tartalmát át kell definiálni a `\markboth` illetve `\markright` parancsokkal. Például az előbbi esetben, ha a számozatlan fejezetben nem akar szintinformációt, akkor a `\chapter*{Számozatlan fejezet címe}` után gépelje be a `\markboth{}{}` parancsot.

Testreszabás fancyhdr csomaggal

A testreszabáshoz az eddigiek helyett használható a **fancyhdr** csomag is. Ezt a csomagot már a **babel** előtt be kell tölteni. Ennek a csomagnak a használatakor a szintinformációk az alábbi táblázat szerint töltődnek be:

	article	report/book
<code>\leftmark</code>	szakasz	fejezet
<code>\rightmark</code>	alszakasz	szakasz

Ennek a csomagnak van egy saját stílusa **fancy** néven. Ennek hatása:

- lábléc közepén az oldalszám
- szint információi a fejlécben
 - egyoldalas szedésnél a külső margónál
 - kétoldalas szedésnél a belső margónál
- alszint információi a fejlécben
 - egyoldalas szedésnél a belső margónál
 - kétoldalas szedésnél a külső margónál.

Ezt a stílust testre szabhatja a

```
\fancyhead[<hely>]{<szöveg>}
\fancyfoot[<hely>]{<szöveg>}
```

parancsokkal. A *<hely>* lehetséges értékei: **L**, **C**, **R**, **LE**, **CE**, **RE**, **LO**, **CO**, **RO**. (Alapopció: LCR.) A betűk jelentései: **L** = bal mező, **C** = közép mező, **R** = jobb mező, **E** = páros oldal, **O** = páratlan oldal. Tehát például **LE** a bal mezőt jelenti a páros oldalakon.

Minden testreszabás előtt adja ki a

```
\fancyhf{}
```

parancsot, mely a korábban definiált fej- és lábléc beállításokat törli. Lehetőség van a főszöveget elválasztani egy vonallal, az ún. léníával, a fejléctől és lábléctől. Ezeknek a vonalaknak a vastagságát a következő parancsokkal állíthatja be:

```
\renewcommand{\headrulewidth}{<vastagság>} fejléc alatti lénia vastagsága,
\renewcommand{\footrulewidth}{<vastagság>} lábléc feletti lénia vastagsága.
```

A `\headrulewidth` alapértéke 0.4pt, a `\footrulewidth` alapértéke pedig 0pt.

Egy létező stílust átdefiniálhat, vagy egy újat létrehozhat a következő paranccsal:

```
\fancypagestyle{<stílusnév>}{<stílus>}
```

A következő kódot, ha a dokumentumtestbe írja, akkor onnan kétoldalas szedésnél a következő beállítások érvényesülnek:

- lénia nincs
- lábléc üres
- oldalszám a fejléc külső margójánál
- szint információi a fejlécben páros oldalon a belső margónál
- alszint információi a fejlécben páratlan oldalon a belső margónál.

```
\pagestyle{fancy}
\fancyhf{}
\fancyhead[LE,RO]{\normalfont\normalsize\thepage}
```

```
\fancyhead[LO]{\sffamily\small\rightmark}
\fancyhead[RE]{\sffamily\small\leftmark}
\renewcommand{\headrulewidth}{0pt}
```

A következő kódot, ha a dokumentumtestbe írja, akkor onnan kétoldalas szedésnél a fejléc üres, a láblécben a külső margónál lesz az oldalszámozás. Amikor fejezetkezdő oldalra érünk, akkor a `report` és `book` osztály `plain` stílusra vált, így az oldalszám bekerül középre, ami zavaró lehet ennél a beállításnál. A `plain` stílus hatástalanítására alkalmas a `\fancypagestyle{plain}{}` parancs.

```
\pagestyle{fancy}
\fancyhf{}
\fancyfoot[LE,RO]{\normalfont\normalsize\thepage}
\renewcommand{\headrulewidth}{0pt}
\fancypagestyle{plain}{}%
```

Ha `m/n` alakú oldalszámozást szeretne, ahol `m` az aktuális oldalszám, `n` pedig az utolsó oldalé, akkor töltsse be a `lastpage` csomagot, majd írja be a következőket:

```
\pagestyle{fancy}
\fancyhf{}
\fancyfoot[C]{\normalfont\normalsize\thepage/\pageref{LastPage}}
\renewcommand{\headrulewidth}{0pt}
\fancypagestyle{plain}{}%
```

vagy

```
\fancypagestyle{sajat}{
  \fancyhf{}
  \fancyfoot[C]{\normalfont\normalsize\thepage/\pageref{LastPage}}
  \renewcommand{\headrulewidth}{0pt}
  \fancypagestyle{plain}{}%
}
```

Ezután bárhol használható a `\pagestyle{sajat}` vagy `\thispagestyle{sajat}` parancs.

Az előző példákban nem csak oldalszámok és szintinformációk jeleníthetők meg, hanem saját információk is. Ezek betűtípusait tetszőlegesen beállíthatja. Használhatja a korábban már megismert `\nouppercase{<szöveg>}` parancsot is, amit a `fancyhdr` csomag alaphoz definiál, így ezt nem nekünk kell megtenni, mint tettük ezt korábban.

16.5. Jegyzékek

16.5.1. Tartalomjegyzék

A dokumentumnak arra a pontjára, ahol a tartalomjegyzéket meg akarja jeleníteni, adja ki a

```
\tableofcontents
```

parancsot. Ha meg akarja változtatni a címet például „Tartalom”-ra, akkor még írja elé a következőt:

```
\def\contentsname{Tartalom}
```

Fordításnál a szintcímek és a hozzátartozó oldalszámok egy `toc` (table of contents) kiterjesztésű fájlba íródnak (a neve és a könyvtára a forrásállományéval egyezik meg). A fordítás végén ennek a fájlnek a segítségével ténylegesen megjelenik a tartalomjegyzék.

A tartalomjegyzék mélységét, azaz hogy mely szintek címei jelenjenek meg a tartalomjegyzékben, a `tocdepth` számláló tartalmazza. Átállítása pl. 4-re:

```
\setcounter{tocdepth}{4}
```

Ekkor a 4-es és annál kisebb szintszámú címek jelennek meg a tartalomjegyzékben.

Amikor egy szintnyitó parancsnak a csillagos verzióját alkalmazza, akkor ez a cím nem lesz sorszámozva, nem kerül az élőfejbe és a tartalomjegyzékbe. Hogy mégis bekerüljön a tartalomjegyzékbe az oldalszámmal együtt, a szintnyitó parancs után gépelje be a következőt:

```
\addcontentsline{toc}{<szint>}{<cím>}
```

Például

```
\section*{Előszó}
\addcontentsline{toc}{section}{Előszó}
```

A `hyperref` csomag használata esetén, ha `\addcontentsline` paranccsal írunk a tartalomjegyzékbe, akkor az oldalszám linkje nem fog működni. Ennek javítása az, hogy az `\addcontentsline` elé be kell még írni a `\phantomsection` \in `hyperref` parancsot is.

Oldalszám nélküli feliratok és parancsok is kiíratathatók a tartalomjegyzékbe az

```
\addtocontents{toc}{<szöveg>\par}
```

paranccsal. Hogy az `\addtocontents` parancsnak hol lesz hatása a tartalomjegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

Alaphelyzetben a tartalomjegyzékben nem jelennek meg a jegyzékek (tartalom, táblázatok és ábrák jegyzéke, bibliográfia) továbbá a név- és tárgymutatók. Ha mégis szükség van rá, a `tocbibind` csomag mindezeket megjeleníti. Ez a következő dokumentumosztályokkal tud együttműködni: `book`, `report`, `article`, `proc`, `ltxdoc`. A lehetséges opciók:

`notbib` Az irodalomjegyzék nem jelenik meg a tartalomjegyzékben.

`notindex` A tárgymutató nem jelenik meg a tartalomjegyzékben.

`nottoc` A tartalomjegyzék nem jelenik meg a tartalomjegyzékben.

`notlot` A táblázatok jegyzéke nem jelenik meg a tartalomjegyzékben.

`notlof` Az ábrák jegyzéke nem jelenik meg a tartalomjegyzékben.

`numbib` A bibliográfia számozott címet kap.

`numindex` A tárgymutató számozott címet kap.

16.5.2. Táblázatok jegyzéke

A táblázatok címeiből is készíthet jegyzéket, melyben a táblázat száma, címe és oldalszáma jelenik meg. Ehhez a dokumentum megfelelő pontjára be kell írni a

```
\listoftables
```

parancsot. Ha meg akarja változtatni a címet például „Táblázatlista”-ra, akkor még írja elé a következőt:

```
\def\listtablename{Táblázatlista}
```

Fordításnál a `table` környezetbe írt `\caption` parancs, illetve tetszőleges helyre írt `\captionof{table}` opciója illetve annak hiányában az argumentuma az aktuális oldalszámmal kiíródik egy `lot` (list of tables) kiterjesztésű fájlba (a neve és a könyvtára a forrásállományával egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik a táblázatok jegyzéke.

Ha a táblázatok jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lot}{table}{<szöveg>}
```

Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{lot}{<szöveg>\par}
```

parancssal. Hogy az `\addtocontents` parancsnak hol lesz hatása a jegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

16.5.3. Ábrák jegyzéke

Ábrák jegyzékének készítéséhez a dokumentum megfelelő pontjára be kell írni a

```
\listoffigures
```

parancsot. Ha meg akarja változtatni a címet például „Ábralista”-ra, akkor még írja elé a következőt:

```
\def\listfigurename{Ábralista}
```

Fordításnál a `figure` környezetbe írt `\caption` parancs, illetve tetszőleges helyre írt `\captionof{figure}` opciója illetve annak hiányában az argumentuma az aktuális oldalszámmal kiíródik egy `lof` (list of figures) kiterjesztésű fájlba (a neve és a könyvtára a forrásállományával egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik az ábrák jegyzéke.

Ha az ábrák jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lof}{figure}{<szöveg>}
```

Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{lof}{<szöveg>\par}
```

parancssal. Hogy az `\addtocontents` parancsnak hol lesz hatása a jegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

16.5.4. Kódok jegyzéke

A `listings` illetve `listingsutf8` csomagokkal készített programkódok jegyzékének készítéséhez a dokumentum megfelelő pontjára be kell írni a

```
\def\lstlistlistingname{Kódok jegyzéke}
\lstlistoflistings
```

parancsokat. Fordításnál a számozott kódok címe az aktuális oldalszámmal kiíródik egy `lol` (list of listings) kiterjesztésű fájlba (a neve és a könyvtára a forrásállományával egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik a kódok jegyzéke.

Ha a kódok jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lol}{lstlisting}{<szöveg>}
```

Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{lol}{<szöveg>\par}
```

paranccsal. Hogy az `\addtocontents` parancsnak hol lesz hatása a jegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

16.5.5. Saját úsztatott objektumok jegyzéke

Láttuk korábban, hogy a `caption` csomaggal saját úsztató környezetet is létrehozhatunk. Az ilyen környezettel úsztatott objektumokból is készíthet jegyzéket a

```
\listof<környezet>s
```

paranccsal. Fordításnál a `<környezet>` környezetbe írt `\caption` illetve tetszőleges helyre írt `\captionof{<környezet>}` opciója illetve annak hiányában az argumentuma az aktuális oldalszámmal kiíródik egy `lo<környezet>` (list of `<környezet>`) kiterjesztésű fájlba (a neve és a könyvtára a forrásállományával egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik a jegyzék.

Ha az úsztatott objektum jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lo<környezet>}{<környezet>}{<szöveg>}
```

Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{lo<környezet>}{<szöveg>\par}
```

paranccsal. Hogy az `\addtocontents` parancsnak hol lesz hatása a jegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

Legyen például az általunk definiált úsztató környezet neve `prog`. Ekkor a jegyzék elkészítéséhez a dokumentum megfelelő pontjára be kell írni a

```
\listofprogs
```

parancsot. A jegyzék adatai a `loprog` (list of progs) kiterjesztésű fájlban vannak. Ha a jegyzékébe akar szöveget írni, akkor használja a következő parancsokat:

```
\addcontentsline{loprog}{prog}{<szöveg>}  
\addtocontents{loprog}{<szöveg>\par}
```

16.5.6. Jegyzékek stílusának szerkesztése

Lehetőség van saját stílus kialakítására is a következő kóddal.

```
\makeatletter  
\def\l@<stílusnév>{<fontstílus>\@dottedtocline{<színtszám>}{<behúz1>}{<behúz2>}}  
\makeatother
```

Például a

```
\makeatletter  
\def\l@sajat{\large\@dottedtocline{1}{2em}{3em}}  
\makeatother
```

kóddal definiálunk egy 1 szintszámú saját nevű stílust. Ezután a

```
\addcontentsline{toc}{sajat}{\cím}
```

hatására megjelenik a $\langle\text{cím}\rangle$ a tartalomjegyzékben (large méretben), ha a `tocdepth` értéke nem kisebb 1-nél. A $\langle\text{cím}\rangle$ felirat 2em behúzással kezdődik. Ha a cím olyan hosszú, hogy sort kell törni, akkor a második és az azt követő sorokat az első sorhoz képest 3em behúzással kezdi. A cím végét és az oldalszámot pontsor köti össze.

Ha azt szeretné, hogy ne legyen a cím vége és az oldalszám között pontsor, akkor bonyolultabb a kód. Ehhez először a `\@dottedtocline` parancs mintájára definiálni kell egy `\@nodottedtocline` parancsot:

```
\makeatletter
\def\@nodottedtocline#1#2#3#4#5{%
\ifnum #1>\c@tocdepth \else
\vskip \z@ \@plus.2\p@
{\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
\parindent #2\relax\@afterindenttrue
\interlinepenalty\@M
\leavevmode
\@tempdima #3\relax
\advance\leftskip \@tempdima \null\nobreak\hskip -\leftskip
\#4}\nobreak
\leaders\hbox{}\hfill
\nobreak
\hb@xt@\@pnumwidth{\hfil\normalfont \normalcolor #5}%
\par}%
\fi}
\makeatother
```

Ezután a

```
\makeatletter
\def\l@<stílusnév>{\fontstílus}\@nodottedtocline{\szintszám}{\behúz1}{\behúz2}}
\makeatother
```

pontosan úgy használható, mint az előbb.

16.6. Tételszerű bekezdések

Sokszor lehet szükség olyan bekezdések írására, melyeknek típuscímet vagy sorszámot kell adni. Ilyen például a matematikában a tétel, bizonyítás, definíció, vagy a törvénykönyvben a paragrafusok stb. Ezek az ún. tételszerű bekezdések, melyeket a `\newtheorem` paranccsal definiált környezetekkel hozhat létre.

```
\newtheorem{<tételnév>}{<tétalcím>}
\newtheorem{<tételnév>}{<tétalcím>}[<számlálóos>]
\newtheorem{<tételnév>}[<együttnév>]{<tétalcím>}
```

$\langle\text{tételnév}\rangle$: Létrejön egy $\langle\text{tételnév}\rangle$ környezet és egy $\langle\text{tételnév}\rangle$ számláló, mely minden újabb ilyen környezet megnyitásakor növekszik eggyel.

$\langle\text{tétalcím}\rangle$: Ez lesz a tételszerű bekezdés típuscíme (definíció, megjegyzés stb.). Ezen cím mellett megjelenik a $\langle\text{tételnév}\rangle$ számláló aktuális értéke is.

⟨számlálóós⟩: Egy már korábban definiált számláló, általában valamelyik szint számlálója (chapter, section stb.). Ennek változásakor a *⟨tételnév⟩* nevű számláló lenullázódik. A *⟨számlálóós⟩* és a *⟨tételnév⟩* számláló együtt jelenik meg (például 2.1. tétel).

⟨együttnév⟩: Egy másik tételszerű környezet neve. A *⟨tételnév⟩* és *⟨együttnév⟩* környezetek számlálói együtt fognak növekedni.

A létrehozott tételszerű környezetet az alábbi módon használhatja:

```
\begin{⟨tételnév⟩}[⟨egyedi cím⟩]
⟨A bekezdés szövege⟩
\end{⟨tételnév⟩}
```

Az *⟨egyedi cím⟩* megadása esetén, az a *⟨tételcím⟩* után jelenik meg zárójelben. Hivatkozni a tételszerű bekezdésekre az általános leírásnak megfelelően lehet. Például

```
\newtheorem{tetel}{tétel}
...
\begin{tetel}
A tétel szövege.
\end{tetel}
\begin{tetel}[Cauchy]\label{cauchy}
A következő tétel szövege.
\end{tetel}
\Aref{cauchy}.~tételből következően\dots
```

1. tétel. *A tétel szövege.*
 2. tétel (Cauchy). *A következő tétel szövege.*
- A 2. tételből következően...

```
\newtheorem{tetel}{tétel}[section]
\newtheorem{defin}[tetel]{definíció}
...
\section{Szakasz címe}
\begin{tetel}
A tétel szövege.
\end{tetel}
\begin{defin}
A definíció szövege.
\end{defin}
```

1. Szakasz címe

- 1.1. tétel. *A tétel szövege.*
- 1.2. definíció. *A definíció szövege.*

Az eddigi példákból látható, hogy a tételszerű bekezdésekben a cím félkövéren, a szöveg pedig dőlten jelenik meg. A tételszerű bekezdések stílusait magunk is beállíthatjuk az `ntheorem` vagy az `amsthm` csomaggal. Mi most csak az `amsthm` csomaggal foglalkozunk. Ha ezt az `amsmath` csomaggal együtt használja, akkor az `amsmath` előbb legyen betöltve, mint az `amsthm`. A stílus beállítása a következő paranccsal lehetséges:

`\theoremstyle{<stílusnév>} ∈ amsthm`

A `<stílusnév>` értékei a következők lehetnek:

`plain` A cím félkövér, a szöveg dőlt. Ez az alapérték.

`definition` A cím félkövér, a szöveg álló antikva.

`remark` A cím dőlt, a szöveg álló antikva.

Ezen stílusokon kívül sajátokat is definiálhat a következő paranccsal:

`\newtheoremstyle{<sn>}{<fe>}{<le>}{<sz>}{<be>}{<cf>}{<cp>}{<ct>}{<cd>} ∈ amsthm`

`<sn>`: Az új stílus neve.

`<fe>`: A bekezdés feletti térköz mérete. Üresen hagyva az alapértéket veszi fel.

`<le>`: A bekezdés alatti térköz mérete. Üresen hagyva az alapértéket veszi fel.

`<sz>`: A szöveg fonttípusa.

`<be>`: A behúzás mérete. Ha normál bekezdésnyi méretű behúzást akarunk, akkor ide írjuk be a `\parindent` parancsot. Üresen hagyva nincs behúzás.

`<cf>`: A cím fontja.

`<cp>`: A címet a szövegtől elválasztó írásjel.

`<ct>`: A címet a szövegtől elválasztó térköz mérete. Ha a cím után sortörést akarunk, akkor ide írjuk be a `\newline` parancsot.

`<cd>`: A tétel címének felépítése. Üresen hagyva az alapbeállítás érvényesül. A beállításához három parancs használható: `\thmname`, `\thmnumber`, `\thmnote`.

Ezek használatára nézzük a következő példát:

```
\newtheoremstyle{sajat}{3ex}{3ex}{\slshape}{0pt}{\slshape\bfseries}{.}{.}
{2ex}{\thmnumber{#2}\thmname{. #1}\thmnote{ (#3)}}
\theoremstyle{sajat}
\newtheorem{tetel}{tétel}
...
\begin{tetel}[Cauchy]
A tétel szövege.
\end{tetel}
```

1. tétel (Cauchy). A tétel szövege.

Ha egy tételszerű környezetnek nem akar számozást, akkor használja a következőt:

`\newtheorem*{<tétnév>}{<tételcím>} ∈ amsthm`

Például

```
\newtheorem{tetel}{tétel}[section]
\theoremstyle{definition}
\newtheorem{defin}[tetel]{definíció}
\theoremstyle{remark}
\newtheorem*{megj}{Megjegyzés}
...
\section{Szakasz címe}
\begin{tetel}
A tétel szövege.
\end{tetel}
\begin{defin}
A definíció szövege.
```



```
\end{defin}
\begin{megj}
A megjegyzés szövege.
\end{megj}
```

1. Szakasz címe

1.1. tétel. *A tétel szövege.*

1.2. definíció. A definíció szövege.

Megjegyzés. A megjegyzés szövege.

Matematikai tételek, lemmák, következmények bizonyítására van egy előre definiált **proof** környezet az **amsthm** csomagban. Például

```
\begin{proof}
A bizonyítás szövege.
\end{proof}
```

Bizonyítás. A bizonyítás szövege. □

A □ az ún. Q.E.D. jel, ami a latin *quod erat demonstrandum* (ami bizonyítandó volt) kifejezés rövidítése. A Q.E.D. rövidítést matematikai levezetések végére szokták odaírni. Ma már ritkábban használják, helyette inkább □ vagy ■ módon jelölik, melyet angol nyelvterületen *sírkőnek* (tombstone), illetve *halmosnak* is neveznek Halmos Pál után, aki az 1950-es években vezette be a használatát az *iff*-fel (akkor és csak akkor) együtt.

A Q.E.D. jelet átdefiniálhatja például ■ jelre az alábbi módon:

```
\renewcommand{\qedsymbol}{\blacksquare}
```

Ha nem akar Q.E.D. jelet, akkor írja be a következőt:

```
\renewcommand{\qedsymbol}{} 
```

Ha a bizonyítás kiemelt matematikai képlettel zárul, akkor a képlet utáni sorba kerül a Q.E.D. jel, ami csúnya:

$$e^{i\pi} + 1 = 0.$$

□

Ilyenkor használja a `\qedhere` parancsot:

```
\begin{proof}
...
\[\mathrm{e}^{\mathrm{i}\pi}+1=0.\qedhere\]
\end{proof}
```

$$e^{i\pi} + 1 = 0.$$

□

Ez a megoldás természetesen nem ad jó eredményt, ha a képlet jobbról számozott. Ilyenkor a bizonyítást mindenképpen szöveggel zárja le. Ha a bizonyítás utolsó eleme egy többsoros képlet, például `align*` környezettel megadva, akkor a `\qedsymbol` parancsot az utolsó képletsor után írja. Például

```
\begin{proof}
...
```

```
\begin{align*}
a&=2,\\
b&=3.\qedhere
\end{align*}
\end{proof}
```

$$\begin{aligned} a &= 2, \\ b &= 3. \end{aligned}$$

□

Ha a bizonyítás utolsó eleme `array` környezet, akkor használja a `b` opcióját, továbbá a `\qedsymbol` parancsot az `\end{array}` után tegye. Például

```
\begin{proof}
...
\[
\begin{array}[b]{|c|c|c|}
\hline
a&b&c\\
\hline
d&e&f\\
\hline
\end{array}\qedhere
\]
```

a	b	c
d	e	f

□

Fontos, hogy a `\qedhere` parancsot akkor is tegye ki az előbbi esetekben és módon, ha előtte a `\renewcommand{\qedsymbol}{}` paranccsal a Q.E.D. jelet üresre állította. Ugyanis ellenkező esetben a bizonyítás utolsó képlete után generálódik egy üres sor, ahová a Q.E.D. jel kerülne, melynek eredményeként a bizonyítás után túl nagy függőleges térköz keletkezne.

A `proof` környezet opcióval is használható. Például

```
\begin{tétel}\label{xy}
A tétel szövege.
\end{tétel}
\begin{proof}[\Aref{xy}.~tétel bizonyítása]
A bizonyítás szövege.
\end{proof}
```

1. tétel. *A tétel szövege.*

Az 1. tétel bizonyítása. A bizonyítás szövege.

□

Ha „Bizonyítás” cím helyett például „Megoldás” feliratot akar, akkor használja a

```
\renewcommand{\proofname}{Megoldás}
```

parancsot a dokumentumtestben (különben a `magyar.ldf` felülbírálja).

Ha a „*Bizonyítás.*” feliratot például „**Bizonyítás.**” típusúra szeretné átállítani, akkor használja a következő kódot:

```

\makeatletter
\renewenvironment{proof}[1][\proofname]{\par
\pushQED{\qed}%
\normalfont \topsep6\p@\@plus6\p@\relax
\trivlist
\item[\hskip\labelsep
\bfseries%           itt van beállítva a félkövér típus
#1\@addpunct{.}]\ignorespaces}
{\popQED\endtrivlist\@endpfalse}
\makeatother

```

A már definiált tételszerű környezeteket nem tudja átdefiniálni a `\newtheorem` paranccsal, csak akkor, ha előtte kiadja a következő parancsokat:

```

\let\<tétnév>\relax
\let\end<tétnév>\relax

```

Tételszerű környezet lezárása új bekezdést nyit, függetlenül attól, hogy tett-e üres sort utána vagy sem. Ezt a hatást felülbíráhatja a következő kóddal:

```

\makeatletter
\AfterEndEnvironment{\<tétnév>}{\@doendpe} ∈ etoolbox
\makeatother

```

Ekkor az `\end{\<tétnév>}` után üres sort hagyva új bekezdés indul, ellenkező esetben nem. Ha nem akar új bekezdést az `\end{\<tétnév>}` után, ha rak utána üres sort, ha nem, akkor a következő kódot használja:

```

\makeatletter
\AfterEndEnvironment{\<tétnév>}
{\everypar{\setbox\z@\lastbox\everypar{}}} ∈ etoolbox
\makeatother

```

Ha egy tételszerű környezetet egy adott jellel szeretne lezárni, úgy mint a bizonyítás esetén a Q.E.D. jel, akkor ezt a következő példa alapján megteheti. Töltse be az `amsthm` csomagot, majd írja a preambulumba a következőket:

```

\newtheorem{tetel}{Tétel}
\let\oldtetel\tetel
\let\endoldtetel\endtetel
\renewenvironment{tetel}[1][\%
\begin{oldtetel}[#1]\pushQED{\qed}}
{\def\qedsymbol{$\bigcirc$}\popQED\end{oldtetel}}

```

Ezzel egy olyan `tetel` környezetet definiál, amely mindig \bigcirc (`\bigcirc`) jellel zárul. Ez nem definiálja át a `proof` környezet végén található Q.E.D. jelet. Tehát ekkor

```

\begin{tetel}
Szöveg.
\end{tetel}
\begin{proof}
Szöveg.
\end{proof}

```

1. Tétel. *Szöveg.*



Bizonyítás. Szöveg.



16.7. Bibliográfia

A bibliográfia címe `article` osztályban „Hivatkozások”, melyet a `\refname` parancs tárol, `report` és `book` osztályban „Irodalomjegyzék”, melyet a `\bibname` parancs tárol. Átdefiniálásuk például „Irodalom”-ra:

```
\renewcommand{\refname}{Irodalom}
\renewcommand{\bibname}{Irodalom}
```

Az átdefiniálást a dokumentumtestben kell megtenni, különben a `magyar.ldf` felülbírálja.

16.7.1. Bibliográfia készítése környezettel

Bibliográfiát `thebibliography` környezettel lehet készíteni, a bibliográfiai elemeket pedig a `\bibitem` paranccsal adhatja meg.

```
\begin{thebibliography}{\langle példacímke \rangle}
\bibitem[\langle címke \rangle]{\langle kulcs \rangle} \langle elemleírás \rangle
...
\end{thebibliography}
```

\langle példacímke \rangle A bibliográfiai elemek címkéi közül a legszélesebb.

\langle címke \rangle Ezzel adhatja meg, hogy a bibliográfiai elem milyen szöveggel legyen azonosítva. Elhagyása esetén automatikus sorszám lesz a címke.

\langle kulcs \rangle A bibliográfiai elemre `\cite[\langle szöveg \rangle]{\langle kulcs \rangle}` paranccsal lehet hivatkozni a dokumentumban. Ilyenkor az adott ponton az adott elem címkéje `[]` jelek között jelenik meg. Egyszerre több kulcsot is megadhat, ezeket vesszővel kell elválasztani. A *\langle szöveg \rangle*-ben például megadhatja, hogy melyik oldalra hivatkozik.

Magyar nyelvű dokumentumban a hivatkozások elé automatikus névelőt is rakhat az

```
\acite[\langle szöveg \rangle]{\langle kulcs \rangle}
\Acite[\langle szöveg \rangle]{\langle kulcs \rangle}
```

vagy az ezzel egyenértékű

```
\az{\cite[\langle szöveg \rangle]{\langle kulcs \rangle}}
\Az{\cite[\langle szöveg \rangle]{\langle kulcs \rangle}}
```

parancsokkal.

Ezek illusztrálására itt egy példa:

```
Lásd \cite{PlainTeX} és \cite[134.~oldal]{LaTeX}\dots
Lásd \cite{PlainTeX,LaTeX}\dots
Lásd \acite{PlainTeX,LaTeX} könyvekben\dots
...
\begin{thebibliography}{2}
\bibitem{PlainTeX} Bujdosó Gyöngyi, Fazekas Attila:
    \TeX\ kezdőlépések, Budapest, 1997, Tertia Kiadó.
\bibitem{LaTeX} Wettl Ferenc, Mayer Gyula, Szabó Péter:
    \LaTeX\ kézikönyv, Budapest, 2004, Panem Könyvkiadó.
\end{thebibliography}
```

Lásd [1] és [2, 134. oldal]...Lásd [1, 2]...Lásd az [1, 2] könyvekben...

Hivatkozások

- [1] Bujdosó Gyöngyi, Fazekas Attila: \TeX kezdőlépések, Budapest, 1997, Tertia Kiadó.
- [2] Wettl Ferenc, Mayer Gyula, Szabó Péter: \LaTeX kézikönyv, Budapest, 2004, Panem Könyvkiadó.

Lásd `\cite{PlainTeX}` és `\cite[134.~oldal]{LaTeX}\dots`

Lásd `\cite{PlainTeX,LaTeX}\dots`

Lásd `\acite{PlainTeX,LaTeX}` könyvekben\dots

...

`\begin{thebibliography}{Bujdosó 1997}`

`\bibitem[Bujdosó 1997]{PlainTeX}` Bujdosó Gyöngyi, Fazekas Attila:
 \TeX kezdőlépések, Budapest, 1997, Tertia Kiadó.

`\bibitem[Wettl 2004]{LaTeX}` Wettl Ferenc, Mayer Gyula, Szabó Péter:
 \LaTeX kézikönyv, Budapest, 2004, Panem Könyvkiadó.

`\end{thebibliography}`

Lásd [Bujdosó 1997] és [Wettl 2004, 134. oldal]...Lásd [Bujdosó 1997, Wettl 2004]...Lásd a [Bujdosó 1997, Wettl 2004] könyvekben...

Hivatkozások

- [Bujdosó 1997] Bujdosó Gyöngyi, Fazekas Attila: \TeX kezdőlépések, Budapest, 1997, Tertia Kiadó.
- [Wettl 2004] Wettl Ferenc, Mayer Gyula, Szabó Péter: \LaTeX kézikönyv, Budapest, 2004, Panem Könyvkiadó.

16.7.2. Bib \TeX

Lehetőség van rá, hogy bibliográfiát adatbázisból készítsen. Ennek előnyei:

- Több dokumentumhoz is használható ugyanaz az adatbázis, mert csak azok a művek jelennek meg a bibliográfiában, amelyekre valóban történt hivatkozás a `\cite` paranccsal. De arra is van lehetőség, hogy az adatbázis minden eleme megjelenjen, függetlenül attól hogy hivatkoztunk-e rá vagy sem.
- Az ábécérendbe rendezés és a leghosszabb címke beállítása automatikusan történik.
- A stílus átállítható az adatbázis változtatása nélkül.

Erre a munkára több eszköz is van. Most az ún. Bib \TeX működését tekintjük át. Az adatbázis elemeit egy `bib` kiterjesztésű fájlba kell írni ASCII kódolással. Ez azt jelenti, hogy az ékezetes betűket repülő ékezzettel kell begépelni, de nem a megszokott formában, azaz `\'{o}`, `\''{o}`, `\H{o}` stb., hanem `{\`o}`, `{\''o}`, `{\H o}` stb. módon.

Pontosabban, használhat ugyanolyan kódolású fontokat is a `bib` fájlban, mint a `tex` fájlban – azaz akár UTF-8 kódolásút is –, de ebben az esetben az ékezetes betűket úgy kezeli, mintha azok az ábécérend végén helyezkednének el.

A TeXstudióban érdemes a végeredményt a `latexmk` programmal előállítani (lásd a Bevezetésben), mert ez automatikusan futtatja a `bibtex.exe` programot.

A `bib` kiterjesztésű fájl tartalma elemtípusokból és mezőnevekből áll. Az elemtípus határozza meg, hogy az adott elem cikk vagy könyv vagy valami egyéb. A mezőnév adja meg, hogy az adott elemnek milyen adatát adjuk meg (szerző, cím, stb.). Ennek szerkezete a következő:

```
@<elemtípus>{<kulcs>,
  <mezőnév> = {<szöveg>},
  <mezőnév> = {<szöveg>},
  <mezőnév> = {<szöveg>},
  ...
}
```

Az adott elemre a dokumentumban a `\cite[<szöveg>]{<kulcs>}` paranccsal vagy annak valamely automatikus névelős verziójával lehet hivatkozni. Sokféle elemtípus létezik, mi csak néhányat emelünk ki. Részletesebben lásd a T_EX dokumentációját, illetve ezen a címen: <http://www.math.bme.hu/latex/magyarldf-doc.pdf>.

A következő kód egy lehetséges séma egy könyv adatainak bevitelére:

```
@book{<kulcs>,
  author    = {<szerző(k)>},
  editor    = {<szerkesztő (vagy ez vagy a szerző kötelező)>},
  title     = {<cím>},
  publisher = {<kiadó>},
  year      = {<kiadás éve>},
  volume    = {<kötetszám>},
  address   = {<városnév>},
  ISBN      = {<ISBN szám>},
  url       = {<URL cím>},
}
```

A következő kód egy lehetséges séma egy folyóirat cikk adatainak bevitelére:

```
@article{<kulcs>,
  author = {<szerző(k)>},
  title  = {<cikk címe>},
  journal = {<folyóirat neve>},
  year   = {<kiadás éve>},
  volume = {<kötetszám>},
  number = {<folyóirat száma>},
  pages  = {<a cikk ezeken az oldalakon van>},
  ISSN   = {<ISSN szám>},
  url    = {<URL cím>},
}
```

A következő kód egy lehetséges séma egy konferencia kötetben megjelent előadás adatainak bevitelére:

```
@inproceedings{<kulcs>,
```

```

author      = {\szerző(k)},
booktitle   = {\konferenciakötet címe},
title       = {\előadás címe},
year        = {\kiadás éve},
editor      = {\szerkesztő},
volume      = {\kötetszám},
number      = {\konferencia kötet száma},
address     = {\városnév},
pages       = {\az előadás ezeken az oldalakon van},
ISSN        = {\ISSN szám},
ISBN        = {\ISBN szám},
url         = {\URL cím},
}

```

A szerzők illetve szerkesztők neveit az angol szabálynak megfelelően kell megadni, azaz „Keresztnév Családnév” sorrendben:

```

{\Keresztnév1} {\Családnév1} and
{\Keresztnév2} {\Családnév2} and
...
{\KeresztnévX} {\CsaládnévX} and others}

```

Az **and** csak akkor kell, ha több nevet sorol fel, illetve az **and others** akkor, ha nem sorolja fel az összes nevet.

A megjelenés stílusa egy **bst** kiterjesztésű fájlban van megadva. Angol nyelvű dokumentumokhoz használhatja például a **plain.bst** fájlt, mely része a \TeX -rendszernek. A magyar stílushoz a **huplain.bst** fájlra van szükség, melyet le kell tölteni a dokumentumunk könyvtárába: [klikk ide](#).

Van egy másik magyar stílus is, a **huszak.bst**, ami a szerzők vezetéknévét kiskapitálissal szedi. Letöltés: [klikk ide](#). Ennek a fájlnak van egy hibája, a vezetéknévben szereplő kötőjelet kiszedi (például Szőkefalvi-Nagy). Ilyenkor az a megoldás, hogy a kötőjelet **{-}** módon kell begépelni.

Ha magyar nyelvű a dokumentum, akkor a magyar neveknek „Családnév Keresztnév” sorrendben kell megjeleníteni. Ehhez nem jó megoldás az adatbázisban a két név sorrendjét megcserélni, mert ekkor a keresztnév alapján fog névsorba rendezni. Ehelyett írja be a **huname = 1** mezőt a magyar stílusfájl használatával mellett.

A magyar stílusfájl a címek szavait (kivéve az elsőt) kisbetűsíti. Ha ezt nem akarja egy adott szó esetén, mert az például egy város neve, akkor az adott szót tegye kapcsos zárójelek közé.

A **huplain.bst** használatával a szerzők nagyköötőjellel vannak elválasztva. Ha ehelyett vesszőt szeretne használni, akkor a **\bibliography** parancs elé gépelje a következőt:

```

\def\bibOverride{\def\bibAnd##1{, }}

```

A dokumentum forrásába a bibliográfia megjelenéséhez a következőket kell beírni:

```

\bibliography{\bib fájl neve kiterjesztés nélkül}
\bibliographystyle{\bst fájl neve kiterjesztés nélkül}

```

Ekkor a **bib** fájlban megadott művekből csak azok fognak megjelenni a bibliográfiában, melyekre **\cite** paranccsal hivatkozott. Ha olyan elemeket is meg akar jeleníteni az adatbázisból, melyekre nem hivatkozik a dokumentumban, akkor a megfelelő elemek kulcsát vesszővel elválasztva be kell írni a **\nocite** parancsba:

```
\nocite{<kulcs1>,<kulcs1>,...}
```

Ha minden elemet be akar illeszteni az adatbázisból, akkor használja a `\nocite{*}` parancsot.

Például legyen az `irodalom.bib` fájl tartalma:

```
@book{KOLMOGOROV,
  author    = {Andrej Nyikolajevics Kolmogorov and
               Szergej Vasziljevics Fomin},
  title     = {A f{"u}ggv{"e}nyelm{"e}let {"e}s a
               funkcion{"a}lanal{"i}zis elemei},
  publisher = {M{"H} u}szaki K{"o}nyvkiad{"o}},
  address   = {Budapest},
  year      = {1981}
}
@book{KATONA,
  huname    = 1,
  author    = {Gyula Katona and Andr{"a}s Recski and Csaba Szab{"o}},
  title     = {A sz{"a}m{"i}t{"a}studom{"a}ny alapjai},
  publisher = {Typotex},
  address   = {Budapest},
  year      = {2006}
}
```

A dokumentum forrásfájljának tartalma azon a helyen, ahová a bibliográfiát szánja:

```
\bibliography{irodalom}
\bibliographystyle{huplain}
\nocite{*}
```

Az `irodalom.bib` és `huplain.bst` fájlok legyenek a dokumentum könyvtárában. Az eredmény a következő:

Hivatkozások

- [1] Katona Gyula–Recski András–Szabó Csaba: *A számítástudomány alapjai*. Budapest, 2006, Typotex.
- [2] Andrej Nyikolajevics Kolmogorov–Szergej Vasziljevics Fomin: *A függvényelmélet és a funkcionálanalízis elemei*. Budapest, 1981, Műszaki Könyvkiadó.

16.8. Függelék

Ha függelékét készít, akkor az adott ponton írja be az `\appendix` parancsot. Ennek hatására a szakasz- illetve fejezetszámlálók lenullázódnak és a számozásuk alfabetikusra vált (A, B, C, ...).

A magyar.ldf fájl `defaults=hu-min` opciója ezen alfabetikus sorszámok után nem tesz pontot `report` és `book` osztályokban (A függelék, B függelék, ...). Ezt a tipográfiát felülbíráhatja az `appendixdot=yes` opcióval (A. függelék, B. függelék, ...).

Az `\appendix` nem írja ki tartalomjegyzékbe, hogy „Függelék”, és `article` osztályban folyószövegbe sem kerül címként ez a felirat. Ha ezt mégis meg akarja tenni, akkor másolja be a következő kódot:


```

\makeatletter
\let\old@appendix\appendix
\def\appendix{\old@appendix
\@ifundefined{chapter}
{\section*{Függelék}\addcontentsline{toc}{section}{Függelék}}
{\addtocontents{toc}{\bigskip\noindent\textbf{Függelék}\par}}}
\makeatother

```

16.9. Tárgymutató

Hosszabb műveknél fontos feladat lehet az ún. tárgymutató készítése, amely a műben előforduló fontosabb fogalmaknak az ábécérendbe szedett jegyzéke. Ez a mű végén szokott elhelyezkedni. Ehhez írja be a preambulumba a következőket:

```

\usepackage{makeidx}
\makeindex

```

Ahová el akarja helyezni a tárgymutatót, írja be a

```
\printindex
```

parancsot. A „Tárgymutató” címet például „Szójegyzék”-re a következő parancs dokumentumtestbe írásával definiálhatja át:

```
\renewcommand{\indexname}{Szójegyzék}
```

A tárgymutatót az

```
\index{<tárgyszó>}
```

paranccsal bővítheti. Például

```

...
\usepackage{makeidx}
\makeindex
\begin{document}
...
A szórásnégyzetet\index{szórásnégyzet} a következőképpen értelmezzük.
...
\printindex

```

A tárgymutatóba kerülő tárgyszó formázható is a következő módon:

`\index{<csoport>!\<tárgyszó>}` Ekkor a `<tárgyszó>` a `<csoport>`-hoz lesz besorolva. Például `\index{eloszlás!normális}`.

`\index{<csoport>!\<alcsoport>!\<tárgyszó>}` Ekkor a `<tárgyszó>` a `<csoport>`-hoz, azon belül az `<alcsoport>`-hoz lesz besorolva. Például `\index{eloszlás!normális!standard}`.

`\index{<besorolás>@\<tárgyszó>}` Ekkor a `<tárgyszó>` úgy sorolódik be a betűrendbe, mint a `<besorolás>` szó. Például `\index{gamma-eloszlás@Γ-eloszlás}` esetén a „ Γ -eloszlás” tárgyszó kerül a tárgymutatóba, de „gamma-eloszlás”-ként sorolódik betűrendbe. Az előző megoldásokban a `<csoport>`, `<alcsoport>` és `<tárgyszó>` is lehet `<besorolás>@\<tárgyszó>` alakú. Például

```

\index{eloszlás!gamma@$\Gamma$-eloszlás}
\index{gamma-eloszlás@$\Gamma$-eloszlás!sűrűségfüggvény}.

```

`\index{<tárgyszó>|\<oldalszámformázás>}` Például `\index{eloszlás|textbf}` esetén az oldalszám félkövér betűtípussal jelenik meg az „eloszlás” tárgyszó után. A `|` jel

után álló kifejezés parancsként lesz értelmezve, ezért nincs a példában a `textbf` előtt `\` jel. A `\langle tárgyszó \rangle` helyére bármilyen korábban ismertetett verzió beírható:

```
\index{eloszlás!normális|textbf}
```

```
\index{gamma-eloszlás@$\Gamma$-eloszlás|textbf}
```

```
\index{gamma-eloszlás@$\Gamma$-eloszlás!sűrűségfüggvény|textbf}.
```

`\index{\langle tárgyszó \rangle|{ }}, \index{\langle tárgyszó \rangle|{ }}` párral több, egymást követő oldalszámot vonhatunk össze.

`\index{$\langle tárgyszó \rangle$}` Ekkor a `\langle tárgyszó \rangle` egy képlet, amely a jelekhez lesz besorolva a tárgymutató elején. Például `\index{\mathbb{R}}`.

Az `\index` parancsban négy speciális jel van, melyek nem jelennek meg a kiírásnál: `@ ! | "`. Ha ezeket meg akarja jeleníteni, akkor eléjük kell írni egy `"` jelet.



A `showidx` csomagot használva a L^AT_EX minden tárgyszót feltüntet a szöveg bal margóján. Ez nagyon hasznos ellenőrzési lehetőséget nyújt. Természetesen a szerkesztés befejezése után, a végleges verzióban ez feleslegessé válik.

A tárgymutató kéthasábos szedéssel jelenik meg. Ennek több gondja is van. A cím új oldalon kezdve nem pontosan ott jelenik meg, ahol kellene, továbbá a két hasáb nem lesz kiegyenlítve az utolsó oldalon. További problémát okoz, hogy a tartalomjegyzékben nem jelenik meg a tárgymutató. Ezeket orvosolja a következő kód `\makeindex` után való beírása a preambulumba. A kód értelemszerű módosításával az is elérhető, hogy három hasábban szedje ki a tárgymutatót.

```
\usepackage{multicol}
\makeatletter
\renewenvironment{theindex}
{
  \@ifundefined{chapter}{\section*{\indexname}}
  {\addcontentsline{toc}{section}{\indexname}}
  {\chapter*{\indexname}}
  {\addcontentsline{toc}{chapter}{\indexname}}
  \@mkboth{\MakeUppercase\indexname}{\MakeUppercase\indexname}
  \begin{multicols}{2}
  \let\item\@idxitem
  \setlength{\parindent}{0pt}
  \setlength{\parskip}{0pt plus .3pt}
  {\end{multicols}}
  \makeatother
```

Az `\index` parancs először beírja a tárgyszót és a hozzá tartozó oldalszámot egy `idx` kiterjesztésű fájlba. Ezután a `makeindex.exe` programmal az adatok ábécérendbe bekerülnek egy `ind` kiterjesztésű fájlba. Végül ennek az `ind` kiterjesztésű fájlbanak a tartalma betöltődik a forrásállományba, majd elkészül a végeredmény.

A TeXstudióban érdemes a végeredményt a `latexmk` programmal előállítani (lásd a Bevezetésben), mert ez automatikusan futtatja a `makeindex.exe` programot.

A `makeindex.exe` program csak angol és német nyelvű szavakat képes helyesen ábécérendbe szedni. A magyar nyelv jóval bonyolultabb algoritmust igényel a speciális betűk miatt (ékezetek, többjegyű mássalhangzók, stb.). Ezért magyar nyelvű dokumentum készítésekor nem a `makeindex.exe` programmal kell a rendezést elvégezni, hanem egy `husort.pl` nevű programmal (Szabó Péter a szerzője). Ennek használatához először töltsse le a forrásfájl könyvtárába a `husort.pl` fájlt innen: [klikk ide](#). Ha a forrásállomány például `C:\próba dokumentum\dokumentum.tex`, akkor nyissa meg a „Futtatás” ablakot a  +  gombokkal, írja be, hogy

cmd

majd **Enter**. Az így megjelenő parancssorba írja be, hogy

```
latexmk -pdf -e "$makeindex='perl -x husort.pl -s gind -C circum2 -C
latin2 -C separate_tags -C single_symbols -C shadow_untagged -C
no_vowel_equiv %0 %S'" "C:\próba dokumentum\dokumentum"
```

majd **Enter**. Ettől jóval kényelmesebb megoldás, ha a TeXstudioból futtatja az előző parancssort. Ehhez használja az 1.10. szakasz 2. beállítását.

Ekkor minden csoport elején, megjelenik középen az adott csoporthoz tartozó kezdőbetű. Ha ezt a sor elejére akarja rakni, akkor a `\printindex` parancs elé gépelje be a következőt:

```
\def\IdxGroupHead#1{\indexspace\par\noindent{\bfseries#1}
\par\nopagebreak\smallskip}
```

Ha nem akar ilyen csoportcímet, akkor a `\printindex` parancs elé gépelje be a következőt:

```
\def\IdxGroupHead#1{\indexspace}
```

vagy az előző parancssorban az `-s gind` kapcsolót ne használja.

Az `\index` bejegyzésbe kerülhet néhány, csak a `husort.pl`-re jellemző szekvencia, melyek hasznosak lehetnek:

`\~{}` Például `\index{idő!tér és \~{}}` az „idő”-n belül „tér és idő”-ként sorolódik be, de a tárgymutatóban „tér és ~” jelenik meg.

`\empty_` Ez eltűnik, de például a „`t\empty_type`” a `t` betűhöz és nem a `ty`-hez sorolódik.

`{_}` Fontos szóköz, mely a besoroláskor nem nyelődik el.

`{.}` Fontos pontkarakter, mely a besoroláskor nem nyelődik el.



Videó: Tárgymutató készítése

16.10. Hosszabb művek szervezése

Ha hosszú művet ír, nem kell az egészét egyetlen fájlban megírni. Használhat egy főfájlt, ami betölti az egyes fejezeteket vagy szakaszokat tartalmazó alfájlokat. Például egy `bevezetes.tex` alfájl következésképpen olvashatja be a főfájlba:

```
\input{bevezetes}
```

vagy

```
\include{bevezetes}
```

Mindkét esetben elhagyható a `.tex` kiterjesztés. Ha más az alfájl kiterjesztése, akkor azt ki kell írni. Ha az aktuális mappán belül a `bevezetes.tex` fájlt például a `fejezetek` nevű almappába teszi, akkor a beolvasása a következőképpen történik:

```
\input{fejezetek/bevezetes}
```

vagy

```
\include{fejezetek/bevezetes}
```

Az `\include` nemcsak beolvassa az adott fájlt, mint az `\input`, hanem annak tartalmát új oldalon is kezdi, továbbá az utolsó oldalt `\clearpage` paranccsal zárja, így az utána következő szöveg is új oldalon kezdődik, továbbá a még függőben lévő úsztatásokat lezárja. Példa főfájltra

```
\documentclass{book}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\usepackage{makeidx}
\makeindex
\begin{document}
\input{cim}
\tableofcontents
\input{bevezetes}
\input{...}
...
\appendix
\input{...}
...
\input{irodalom}
\printindex
\end{document}
```

17. fejezet

Elektronikus publikáció

Az elkészült dokumentumot átalakíthatja elektronikus publikációvá is. Ehhez töltsse be a `hyperref` csomagot. Ekkor az elkészült PDF fájlban automatikusan készül vázlatfa (bookmarks) és kis vázlatképek (thumbnails), továbbá linkké válnak a hivatkozások, URL címek.

Ha a `babel` és `geometry` csomagokat is használja, akkor azokat a `hyperref` után hívja meg. A `hyperref` és `setspace` csomagok együttes használatánál ügyelni kell arra, hogy a `setspace` előbb legyen betöltve, különben a lábjegyzetek linkjei hibás helyre fognak ugrani.

A `hyperref` csomag néhány hasznos parancsa:

`\url{<URL cím>}` Internetcímet adhat meg. Ez nem rakható parancsok argumentumaiba.

`\href{<URL cím>}{<szöveg>}` Internetcímet adhat meg. A pdf-ben a `<szöveg>` jelenik meg, melyre kattintva betölti az `<URL cím>`-et. Ez nem rakható parancsok argumentumaiba.

`\href{mailto:<e-mail cím>}{\nolinkurl{<e-mail cím>}}` E-mail cím megadása.

`\href{run:<fájl>}{<szöveg>}` Ennek helyén a `<szöveg>` felirat jelenik meg linkként. Erre kattintva betölti a `<fájl>`-t. A `<fájl>` nem lehet `exe`, `bat`, `zip` és más hasonló önállóan futtatható illetve tömörített fájl.

`\hyperref[<címke>]{<szöveg>}` Ennek helyén a `<szöveg>` felirat jelenik meg linkként. Erre kattintva a `\label{<címke>}`-vel létrehozott címkére ugrik.

`\hyperlink[<címke>]{<szöveg1>}` Ennek helyén a `<szöveg1>` felirat jelenik meg linkként. Erre kattintva a `\hypertarget[<címke>]{<szöveg2>}` által megcímkézett `<szöveg2>`-re ugrik.

`\phantomsection` Ha `\addcontentsline` paranccsal ír a tartalomjegyzékbe, akkor az oldalszám linkje nem működik. Ennek javításaként a `\addcontentsline` elé be kell írni a `\phantomsection` parancsot.

Hasznos lehet még a `NoHyper` környezet használata, melyben hatástalanná válik a `hyperref` csomag.

A `hyperref` csomag néhány opciója:

`unicode` A vázlatfában helyesen jelenjenek meg az ő Ő ú Ű betűk.

`bookmarks=false` Ne készüljön vázlatfa. Alaphelyzetben készül.

`bookmarksopen` Alaphelyzetben a vázlatfában csak a legfelső szint látszik. Ezzel az opcióval minden szint nyitott lesz.

`bookmarksopenlevel=<szintszám>` A vázlatfa az adott `<szintszám>`-ig nyitott.

bookmarksnumbered A vázlatfában a címek legyenek számozottak.

linktocpage A jegyzékekben az oldalszámok legyenek a linkek. Alaphelyzetben a címek a linkek.

breaklinks Linkek sorvégi törésének engedélyezése. (pdf_latex.exe fordító esetén alapopció.)

colorlinks A linkek színes karakterrel legyenek kiemelve. Alaphelyzetben színes kerettel jelennek meg.

hidelinks A linkek ne legyenek színnel vagy kerettel kiemelve.

hyperfootnotes=false A lábjegyzet jelölője ne legyen link.

pdfpagemode=FullScreen A pdf megnyitásakor csak a lap jelenik meg a teljes képernyőn, a lehető legnagyobb nagyításban.

pdfstartview=<érték> Ha az <érték> **Fit**, akkor a pdf megnyitásakor az ablakban a lehető legnagyobb nagyítást alkalmazza. Ha **FitH**, akkor a pdf megnyitásakor az ablak teljes szélességére nagyít. Ha **FitV**, akkor a pdf megnyitásakor az ablak teljes magasságára nagyít.

linkcolor=<szín> A \ref által létrehozott link színe.

pagecolor=<szín> A \pageref által létrehozott link színe.

citecolor=<szín> A \cite által létrehozott link színe.

urlcolor=<szín> Az \url és \href által létrehozott link színe.

runcolor=<szín> A run: protokoll linkjének a színe.

allcolors=<szín> Minden link színe.

linkbordercolor=<szín> A \ref által létrehozott link keretének színe.

citebordercolor=<szín> A \cite által létrehozott link keretének színe.

urlbordercolor=<szín> Az \url és \href által létrehozott link keretének színe.

runbordercolor=<szín> A run: protokoll link keretének a színe.

allbordercolors=<szín> Minden link keretének színe.

pdfborder={0 0 <szám>} A link keretének vastagsága <szám> pont (ha ez 0, akkor nincs keret).

A hyperref csomag opciói a

```
\hypersetup{<opció1>,<opció1>,...}.
```

paranccsal is megadhatók. Például

```
\hypersetup{bookmarks=false,colorlinks}
```

Előfordulhat, hogy például egy szakasz címében olyan karakter szerepel, ami nem jelenik meg a pdf könyvjelzőjében. Például

```
\section{$\sigma$-gyűrű}
```

esetén a könyvjelzőben csak „-gyűrű” fog megjelenni, a σ jel nem. Ezt oldja meg a következő kódban a \textorpdfstring parancs:

```
\section{\textorpdfstring{$\sigma$-gyűrű}{~cf~83-gyűrű}}
```

ahol cf 83 a σ UTF-8 hexadecimális kódja (lásd <http://www.utf8-chartable.de/>). De még ez sem tökéletes, mert így a σ nem félkövér módban jelenik meg a címben. A legjobb megoldás a következő kód:

```
\section[\textorpdfstring{$\sigma$-gyűrű}{~cf~83-gyűrű}]{~cf~83-gyűrű}
```

Ekkor a σ a szövegben félkövéren, a tartalomjegyzékben pedig normál módban fog megjelenni, továbbá a pdf könyvjelzőjében is látható.

18. fejezet

Szakdolgozat készítése

A `thesis-ekf` osztály olyan szakdolgozatok megírására alkalmas, amely megfelel az Eszterházy Károly Egyetem szabályzatának. Az oldal- és fontparaméterek beállításán túl a megfelelő címoldal elkészítését is elvégzi. A formai követelmények a következők:

- A4-es lap- és 12 pt betűméret;
- a margó a kötés oldalon 30 mm, a többi 25 mm;
- oldalszámozás a láblécben arab számozással;
- a fejezetcímek középre, a további szintek címei balra igazítva;
- a főszöveg antikva betűcsaláddal kisédvé;
- sorkizárt igazítás, másfeles sortávolság.

Ebben a dokumentumosztályban a `geometry` és `hyperref` csomagok automatikusan betöltődnek, így ezeket nem szabad ismét betölteni! A lehetséges opciók:

`twoside` Ha a szakdolgozatot kétoldalasán szeretné kinyomtatni, akkor ezt az opciót alkalmazza! Ne használja egyoldalas nyomtatáshoz illetve elektronikus verzióhoz!

`colorlinks` A linkek színes karakterekkel jelennek meg. Ezt csak a szakdolgozat elektronikus verziójához használja, a nyomtatott verzióhoz nem kell!

A címoldal a `\maketitle` paranccsal hozható létre. Ehhez előtte az adatokat a következő parancsokkal lehet megadni:

`\logo{<képbetöltés>}` Logó betöltéséhez kell használni. Például

```
■ \logo{\includegraphics{eke-logo}}
```

Előtte töltsse be a `graphicx` csomagot!

`\institute{<intézet neve>}` Ezzel adja meg az intézmény nevét. Ha az Eszterházy Károly Egyetem logóját használja, akkor az egyetem nevét nem kell feltüntetni, mert azt a logó már tartalmazza. Ekkor elég csak az intézet neve. Például

```
■ \institute{Matematikai és Informatikai Intézet}
```

`\title{<dolgozat címe>}` Ezzel adja meg a dolgozat címét.

`\author{<név>\<szak>}` Ezzel adja meg a szerző nevét és szakját. Például

```
■ \author{Tóth István\matematika BSc}
```

`\supervisor{<név>\<beosztás>}` Ezzel adja meg a témavezető nevét és beosztását. Például

```
■ \supervisor{Dr. Nagy János\főiskolai docens}
```

`\city{<város>}` Ezzel adja meg a város nevét, ahol az intézmény található. Például

■ `\city{Eger}`

`\date{<évszám>}` Ezzel adja meg a dolgozat leadásának évét. Az évszám után ne tegyen pontot! Az `<évszám>` alapértéke az aktuális évszám.

Egy példa a használatra:

```
\documentclass[colorlinks]{thesis-ekf}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\usepackage{graphicx}

\begin{document}
\logo{\includegraphics{eszterhazy-logo-hu}}
\institute{Matematikai és Informatikai Intézet}
\title{A szakdolgozat címe}
\author{Szerző neve\\szak}
\supervisor{Tanár neve\\beosztás}
\city{Eger}
\date{2018}
\maketitle
\tableofcontents

\chapter{Fejezet címe}
\section{Szakasz címe}

\begin{thebibliography}{1}
\bibitem{cimke} \textsc{Szerző}: Cím, Kiadó, Hely, évszám.
\end{thebibliography}
\end{document}
```

Még egyszerűbb, ha a következő sablont használja: [klikk ide](#).

19. fejezet

Prezentációk

L^AT_EX-ben elektronikus prezentáció készítésére a **beamer** dokumentumosztály a legalkalmasabb. A pdf alapú prezentációk előnye, hogy a végeredmény minden platformon levetíthető és ugyanúgy fog működni. Így nem kell attól tartani, hogy egy idegen gépen nem indul el vagy más jelenik meg, mint a saját gépünkön. A **beamer** osztály jellemzői:

- Oldalméret: 128 mm × 96 mm (4 : 3 arány). Az `aspectratio=169` opció esetén 160 mm × 90 mm (16 : 9 arány).
- Alap betűméret: 11 pt. Opcióban a következő további méretek adhatók meg: `8pt` `9pt` `10pt` `12pt` `14pt` `17pt` `20pt`.
- Alap betűtípus: álló, normál, groteszk.
- Főszöveg sortörése: balra zárt, így nincsenek szóelválasztások.
- Új bekezdés elején nincs behúzás.
- Keret (lásd később) tartalmának függőleges pozíciója: közép. Opcióban másik két lehetőség: `t` (fent), `b` (lent).
- Ezzel az osztállyal automatikusan betöltődnek a következő csomagok: `graphicx`, `amsthm`, `xcolor`, `enumerate`, `hyperref`.

19.1. Témák

A nyomtatott illetve elektronikus publikációk szerkesztésénél a tipográfiai munka jelentős részét a L^AT_EX-re bíztuk. Ez itt is megoldható, ugyanis a **beamer** rengeteg ún. témát tartalmaz, melyek mindegyike egy-egy tipográfiai beállítást, stílust jelent. A témák betöltése a preambulumban történik a következő parancsokkal:

`\useinnertheme[⟨opciók⟩]{⟨név⟩}` Belső szerkezeti elemekből (címloldal, listák, tömbök, tételszerű környezetek, képek, táblázatok, lábjegyzetek, irodalomjegyzék) mi jelenjen meg és milyen geometriával.

`\useoutertheme[⟨opciók⟩]{⟨név⟩}` Külső szerkezeti elemekből (fej- és lábléc, oldalsávok, logó, keret címe) mi jelenjen meg és milyen geometriával.

`\usecolortheme[⟨opciók⟩]{⟨név⟩}` Belső és külső szerkezeti elemek színvilága.

`\usefonttheme[⟨opciók⟩]{⟨név⟩}` Belső és külső szerkezeti elemek betűtípusai.

`\usetheme[<opciók>]{<név>}` Teljes témák. Szerkezeti, szín- és betűtípus témák összehangolása.

Célszerű először egy teljes témát választani. Ha ebben valamilyen részlet nem tetszik, akkor alkalmazhat még valamilyen belső vagy külső szerkezeti, szín- vagy betűtípus témát is.

19.1.1. Teljes témák

Oldalsáv nélkül

<i><név></i>	<i><opciók></i>
Bergen	—
Boadilla	<code>secheader</code> (fejléc bekapcsolása)
Madrid	<code>secheader</code> (fejléc bekapcsolása)
AnnArbor	—
CambridgeUS	—
Pittsburgh	—
Rochester	<code>height=<i><magasság></i></code> (keretcím magassága)

Fa navigáció

<i><név></i>	<i><opciók></i>
Antibes	—
JuanLesPins	—
Montpellier	—

Oldalsávval

<i><név></i>	<i><opciók></i>
Berkeley	<code>hideallsubsections</code> (oldalsávon nincs alszakasz cím) <code>hideothersubsections</code> (oldalsávon csak az aktuális alszakasz címe van) <code>left</code> (oldalsáv bal oldalon) <code>right</code> (oldalsáv jobb oldalon) <code>width=<i><szélesség></i></code> (oldalsáv szélessége)
PaloAlto	lásd Berkeley
Goettingen	lásd Berkeley
Marburg	lásd Berkeley
Hannover	lásd Berkeley, de nincs <code>left</code> és <code>right</code>

Mini keret a fejlécben

<i><név></i>	<i><opciók></i>
Berlin	<code>compress</code> (egysoros a mini keret)
Ilmenau	lásd Berlin
Dresden	lásd Berlin
Darmstadt	—
Frankfurt	—
Singapore	—
Szeged	—

Fejlécben az aktuális szakasz és alszakasz címe

<i><név></i>	<i><opciók></i>
Copenhagen	—
Luebeck	—
Malmoe	—
Warsaw	—

19.1.2. Belső témák

<i><név></i>	<i><opciók></i>
circles	—
rectangles	—
rounded	shadow (árnyékolt tömbök)
inmargin	—

19.1.3. Külső témák

<i><név></i>	<i><opciók></i>
infolines	—
miniframes	footline=authorinstitute (láblécben: szerző, intézet) footline=authortitle (láblécben: szerző, cím) footline=institutetitle (láblécben: intézet, cím) footline=authorinstitutetitle (láblécben: szerző, intézet, cím) subsection=true (alszakasz címet mutassa) subsection=false (alszakasz címet ne mutassa)
smoothbars	subsection=true (alszakasz címet mutassa) subsection=false (alszakasz címet ne mutassa)
sidebar	hideallsubsections (tartalomban nincs alszakasz cím) hideothersubsections (tartalomban csak az aktuális alszakasz cím) left (oldalsáv bal oldalon) right (oldalsáv jobb oldalon) width= <i><szélesség></i> (oldalsáv szélessége) height= <i><magasság></i> (keretcím magassága)
split	—
shadow	—
tree	hooks („faágak” behúzása)
smoothtree	—

19.1.4. Színtémák

<i><név></i>	<i><opciók></i>
structure	named= <i><színnév></i> (strukturális elemek előterének színe)
sidebartab	—

Teljes színtémák

<i>⟨név⟩</i>	<i>⟨opciók⟩</i>
albatross	overlystylish
beetle	—
crane	—
dove	—
fly	—
seagull	—
wolverine	—
beaver	—

Belső elemek színtémái

<i>⟨név⟩</i>	<i>⟨opciók⟩</i>
lily	—
orchid	—
rose	—

Külső elemek színtémái

<i>⟨név⟩</i>	<i>⟨opciók⟩</i>
whale	—
seahorse	—
dolphin	—

19.1.5. Betűtípus témák

<i>⟨név⟩</i>	<i>⟨opciók⟩</i>
serif	stillsansserifmath stillsansserifsmall stillsansseriflarge stillsansseriftext
structurebold	onlysmall onlylarge
structureitalicserif	lásd structurebold
structuresmallcapserif	lásd structurebold

19.2. Keretek

A beamer-ben a prezentáció keretek sorozatából, a keretek pedig diák sorozatából áll. Egy keretnek címet és alcímet is adhat. Ha egy keret több diából álló diasorozatot tartalmaz, akkor az adott keretben egymásután fognak megjelenni a diasorozat tagjai. Ha egy keret tartalma nem fér el egy dián, akkor az széttörhető több keretre is. Az eredeti keret címe és alcíme megjelenik minden „megtört” kereten. Az ilyen megtört keretekben csak egy-egy dia szerepelhet.

Minden keretet `frame` környezetbe kell rakni.

```
\begin{frame}[⟨opció⟩]{⟨keret címe⟩}{⟨keret alcíme⟩}{⟨keret tartalma⟩}
```

```
\end{frame}
```

vagy

```
\begin{frame}[\textit{opció}]
\frametitle{\textit{keret címe}}
\framesubtitle{\textit{keret alcíme}}
\textit{keret tartalma}
\end{frame}
```

A `frame` környezet opciói

`t`, `b`, `c` A keret tartalma függőlegesen felülre, alulra, középre igazított. (Alapopció `c`.)

`plain` A keretben a fejléc, lábléc és az oldalsávok nem jelennek meg.

`shrink=<kicsinyítés>` Aktiválja a `t` opciót és a keret tartalmát `<kicsinyítés>`% mértékben kicsinyíti. A `<kicsinyítés>` alapértéke 0.

`fragile` Alapesetben verbatim szöveg vagy kód nem írható a keretbe. Ezt a korlátozást oldja fel ez az opció.

`squeeze` Listák függőleges extra térközök nélkül jelennek meg.

`allowframebreaks=<kitöltés>` A kitöltés egy 0 és 1 közötti szám, alapértéke 1. A keretet kitöltés arányú telítettség után több keretre töri. A keret ezen opció esetén a `\framebreak` paranccsal közvetlenül is megtörhető. Ez az opció nem támogatja a keretben több dia használatát.

Ha aktiválja az `allowframebreaks` opciót, akkor alapesetben a keret címe után megjelenik a megtört keret sorszáma nagy római számokkal. Például ha a keret címe „Példa”, akkor a megjelenő címek az egymást követő kereteken:

Példa I → Példa II → Példa III → ...

Ennek átállítására nézzünk néhány példát.

```
\setbeamertemplate{frametitle continuation}[from second]
[\insertcontinuationcountroman.]
```

Példa → Példa II. → Példa III. → ...

```
\setbeamertemplate{frametitle continuation}[from second]
[\insertcontinuationcount.]
```

Példa → Példa 2. → Példa 3. → ...

```
\setbeamertemplate{frametitle continuation}[from second][(folyt.)]
```

Példa → Példa (folyt.) → Példa (folyt.) → ...

19.3. Egy keretben több dia

Emlékeztetünk arra, hogy a `frame` környezet `allowframebreaks` opcióval nem támogatja a kereten belüli több dia használatát. A keret tartalmának több dián való megjelenítésére a legegyszerűbb megoldás a `\pause` parancs használata. Vigyázat, ez a parancs nem használható az `amsmath` csomag által definiált környezetekben, mint például az `align`. Például

```
\begin{frame}{Példa}{}
Ez látható a keret 1. diáján.\par\pause
Ez látható a keret 2. diáján.\par\pause
Ez látható a keret 3. diáján.
\end{frame}
```

19.3.1. Overlay specifikációk

Ennél bonyolultabb diasorozatok is létrehozhatók az úgynevezett overlay specifikációk használatával. A `beamer` sok standard parancsot kiegészít overlay specifikációval. Például listák esetén az `\item` parancsot. A használata és működése megérthető a következő példán:

```
\begin{frame}{Példa}{}
\begin{itemize}
\item<1-2> 1. listaelem
\item<2> 2. listaelem
\item<3> 3. listaelem
\item<3-4> 4. listaelem
\end{itemize}
\end{frame}
```

Tehát az overlay specifikációt a `<` és `>` jelek közé rakjuk. Egyszerre több overlay specifikációt is beírhat, amiket vesszővel kell elválasztani. Példák:

```
<0>          Egyetlen dián sem látható.
<1>          Az 1. dián látható.
<1-3>        Az 1–3. diákon látható.
<1-3,5-6>    Az 1–3. és 5–6. diákon látható.
<1,5>        Az 1. és 5. diákon látható.
<3->         A 3. diától az utolsóig látható.
<-3>         Az 1–3. diákon látható.
<-2,4-6,8->  A 3. és 7. dia kivételével minden dián látható.
```

Ún. léptető overlay specifikációk is írhatók a számok helyére. Ezek egy `beamerpauses` nevű számlálót használnak, melynek a kezdeti értéke a keret elején 1.

Az egyik léptető overlay specifikáció a `+(<szám>)`, ahol a *<szám>* bármilyen egész érték lehet, akár negatív is. Ennek hatása:

- A `+(<szám>)` helyére a `beamerpauses + <szám>` értékét írja. A `+(0)` helyett írható egyszerűen csak `+` jel is.
- Az overlay specifikációt lezáró `>` jel után a `beamerpauses` értékét 1-gyel megnöveli. (Akkor is csak 1-gyel nő az érték, ha több `+` is szerepel az overlay specifikációk között.)

A következő példák mindegyikében tételezzük fel, hogy az overlay specifikáció kifejtése előtt a `beamerpauses` értéke 2. Ekkor

```
<+(1)> = <3>
<+(-1)> = <1>
<+(-2)> = <0>
<+(-4)> = <-2> = <-+>
<+(0)> = <+> = <2>
<+-+(2)> = <2-4>
```

Ezen példák mindegyike után a `beamerpauses` értéke 3-ra nő.

A másik ilyen léptető specifikáció a `pont`. Ennek használatánál ügyeljen arra, hogy a `beamerpauses` értéke már legalább 2 legyen. Ennek hatása:

- A pont helyére a `beamerpauses` értékénél 1-gyel kisebbet ír.
- Az overlay specifikációt lezáró `>` jel után a `beamerpauses` értéke változatlan marad.

Például a következő két kód ekvivalens:

```
\begin{frame}{Példa}{}
\begin{itemize}
\item<+> 1. listaelem
\item<.-> 2. listaelem
\item<+> 3. listaelem
\item<.-> 4. listaelem
\end{itemize}
\end{frame}
```

és

```
\begin{frame}{Példa}{}
\begin{itemize}
\item<1-> 1. listaelem
\item<1-> 2. listaelem
\item<2-> 3. listaelem
\item<2-> 4. listaelem
\end{itemize}
\end{frame}
```

Az overlay specifikációval ellátott parancsoknak lehet alapspecifikációjuk is. Például az `\item` az alapspecifikációja `<1->`, azaz `\item` ekvivalens az `\item<1->` paranccsal. A többi parancs alapspecifikációját az adott parancs tárgyalásánál közöljük.

19.3.2. Diasorozat átlátszósága

Arra is lehetőség van, hogy a keret diáin halványan megjelenjen a kerethez tartozó minden más dia erre engedélyezett tartalma. Ezt a következő módon állíthatja be:

```
\setbeamercovered{transparent=<szám>}
```

Ezután a keretben a diákon `<szám>%` intenzitással látható a többi dia tartalma.

19.3.3. Overlay specifikációval rendelkező parancsok

```
\uncover<spec>{<szöveg>}
```

vagy

```
\begin{uncoverenv}<spec><szöveg>\end{uncoverenv}
```

Csak a megadott diákon fog megjelenni a szöveg, a többin csak foglalja a helyet, illetve a `transparent` értékének megfelelően látjuk. (`<spec>` alapértéke `<1->`.)

```
\visible<spec>{<szöveg>}
```

vagy

```
\begin{visibleenv}<spec><szöveg>\end{visibleenv}
```

Ugyanaz mint az `uncover`, csak a `transparent` pozitívrá állítása erre a parancsra nem hat. (`<spec>` alapértéke `<1->`.) Az `\invisible` parancs illetve `invisibleenv` környezet az előbbihez hasonlóan használható, de a hatása azzal ellentétes. Erre sem hat a pozitív `transparent` érték.

```
\only<spec>>{<szöveg>}
```

vagy

```
\begin{onlyenv}<spec>>{<szöveg>}\end{onlyenv}
```

Ugyanaz mint a `visible`, de a helyet nem foglalja el a `<spec>`-en kívül eső diákon.

```
\alt<spec>>{<szöveg1>}{<szöveg2>}
```

A megadott diákon fog megjelenni a `<szöveg1>`, a többin a `<szöveg2>`. A `transparent` pozitívrá állítása erre a parancsra nem hat. (`<spec>` alapértéke `<1->`.)

```
\begin{altenv}<spec>>{<start1>}{<vége1>}{<start2>}{<vége2>}
<szöveg>
\end{altenv}
```

A megadott diákon ez fog megjelenni: `<start1>` `<szöveg>` `<vége1>`. A többin ez fog megjelenni: `<start2>` `<szöveg>` `<vége2>`. A `transparent` pozitívrá állítása erre a parancsra nem hat. (`<spec>` alapértéke `<1->`.)

```
\temporal<spec>>{<szöveg-előtte>}{<szöveg>}{<szöveg-utána>}
```

A megadott diák előtt fog megjelenni a `<szöveg-előtte>`, a megadott diákon fog megjelenni a `<szöveg>` és a megadott diák után fog megjelenni a `<szöveg-utána>`. (`<spec>`-nek itt nincs alapértéke, kötelező megadni.) Például

```
\begin{frame}{Példa}{
\temporal<3-4>{1., 2. dia}{3., 4. dia}{5., 6., \dots dia}\\
\temporal<3,5>{1., 2., 4. dia}{3., 5. dia}{6., 7., \dots dia}\\
\temporal<2>{1. dia}{2. dia}{3., 4., \dots dia}
\end{frame}
```

```
\begin{overlayarea}{<szélesség>}{<magasság>}
\only<spec1>>{<szöveg1>}
\only<spec2>>{<szöveg2>}
...
\end{overlayarea}
```

A keret minden diáján lefoglal egy `<szélesség>` és `<magasság>` méretű dobozt, melyben a `<spec1>`, `<spec2>`, stb. overlay specifikációknak megfelelően kerül be a `<szöveg1>`, `<szöveg2>`, stb.

```
\begin{overprint}[<szélesség>]
\onslide<spec1>>{<szöveg1>}
\onslide<spec2>>{<szöveg2>}
...
\end{overprint}
```


A keret minden diáján lefoglal egy *<szélesség>* széles dobozt, melynek alapértéke a szövegtükör szélessége. A doboz magassága a *<szöveg1>*, *<szöveg2>*, stb. által meghatározott dobozok természetes magasságai közül a legnagyobb. A *<spec1>*, *<spec2>*, stb. overlay specifikációk között nem lehet átfedés. A dobozban a *<spec1>*, *<spec2>*, stb. overlay specifikációknak megfelelően kerül be a *<szöveg1>*, *<szöveg2>*, stb.

A következő betűtípusra és színre vonatkozó parancsok is rendelkeznek overlay specifikációval (alapspecifikáció *<1->*): `\textbf`, `\textit`, `\textsl`, `\textrm`, `\textsf`, `\textcolor`, `\color`. Például

```
\begin{frame}{1. példa}{
\textbf<1>{Ez félkövér az 1. dián, a többin normál.}\\
\textcolor<2>{red}{Ez a 2. dián piros, a többin fekete.}\\
\textcolor<3>[RGB]{43,52,223}{Ez a 3. dián kék, a többin fekete.}
\end{frame}

\begin{frame}{2. példa}{
\begin{itemize}
\item\textcolor<+>{red}{1. listaelem}
\item\textcolor<+>{red}{2. listaelem}
\item\textcolor<+>{red}{3. listaelem}
\end{itemize}
\end{frame}
```

19.4. Diaváltás látványeffektekkel

Amikor egy keret következő diájára, vagy a következő keret első diájára váltunk, akkor az eddigiekben csak annyi történt, hogy az előző dia képe egyszerűen átváltott az újra. Ezeket a váltásokat látványosabbá is teheti különböző effektekkel. Sajnos nem minden pdf néző támogatja ezeket az effektekkel, ezért idegen gépen nem biztos, hogy fog működni. Például az Adobe Reader esetén működnek, de csak akkor, ha teljes képernyős üzemmódba váltunk, ahogy ez egy prezentáció bemutatásánál szokásos.

Ezeket az effektekkel a következő parancsok `frame` környezetbe írásával érheti el:

```
\transblindshorizontal<spec>[<opció>]
\transblindsvertical<spec>[<opció>]
\transboxin<spec>[<opció>]
\transboxout<spec>[<opció>]
\transcover<spec>[<opció>]
\transdissolve<spec>[<opció>]
\transfade<spec>[<opció>]
\transglitter<spec>[<opció>]
\transreplace<spec>[<opció>]
\transsplitverticalin<spec>[<opció>]
\transsplitverticalout<spec>[<opció>]
\transsplithorizontalin<spec>[<opció>]
\transsplithorizontalout<spec>[<opció>]
\transwipe<spec>[<opció>]
```

Ezekben a parancsokban az overlay specifikáció alapértéke *<1->*. A lehetséges opciók:

`duration=<idő>` Ennyi másodpercig tart az effekt.

`direction=<szög>` Ennyi fokos szögben megy végbe az effekt. A `<szög>` lehetséges értékei 0, 90, 180, 270, illetve `\transglitter` esetén még lehet 315 is.

Az eddigiekben diaváltás mindig gombnyomásra történt. Ez bizonyos idő megadásával automatizálható is, de ez is csak teljes képernyős üzemmódban lehetséges, a következő paranccsal:

```
\transduration<spec2>>{<idő>}
```

Az overlay specifikáció alapértéke `<1->`. Az `<idő>` helyére annyi másodpercet kell írni, ameddig a specifikációkkal megadott diákat látni akarjuk gomb megnyomása nélkül.

19.5. A prezentáció tagolása

19.5.1. Címoldal

A prezentáció első oldala a címoldal, melynek elkészítéséhez szükséges adatokat a következő parancsokkal adhatja meg.

```
\title[<rövid cím>]{<cím>}
\subtitle[<rövid alcím>]{<alcím>}
\author[<rövid név>]{<név>}
\institute[<intézet rövid neve>]{<intézet>}
\date[<rövid dátum>]{<dátum>}
\logo{\includegraphics[<opció>]{<képfájl>}}
\titlegraphic{\includegraphics[<opció>]{<képfájl>}}
```

Ezután a címoldal a következőképpen készül el:

```
\begin{frame}[plain]
\titlepage
\end{frame}
```

vagy

```
\maketitle
```

ami a következővel ekvivalens:

```
\begin{frame}
\titlepage
\end{frame}
```

19.5.2. A főszöveg tagolása

A `beamer`-ben a szöveg tagolása az `article` osztályhoz hasonló, de nincs paragrafus és alparagrafus.

Ha nagyon hosszú prezentációt készít, akkor szükség lehet a több részre való bontásra. Új részt a

```
\part[<rész rövid címe>]{<rész címe>}
```

parancs kereten kívüli kiadásával indíthat. Az opcióban megadott cím alapesetben a rész címével egyezik meg. Ez így még nem jelenik meg sehol, csak a pdf néző könyvjelzői között (a rövid cím), ha az aktiválva van, illetve a navigációs sávban (legtöbbször a

rövid cím), ha az úgy van beállítva. Ha azt akarja, hogy az előző parancs kiadásakor egy külön keret jöjjön létre a címmel, akkor használhatja az `\AtBeginPart`, `\insertpart`, `\insertpartnumber` és `\insertromanpartnumber` parancsokat. Például írja be a következőket a preambulumba:

```
\AtBeginPart{
  \begin{frame}[plain]
  \begin{center}
  {\Large\insertromanpartnumber. rész\}[10mm]}
  {\large\insertpart\}
  \end{center}
  \end{frame}}
```

Ezután a

```
\part{A rész címe}
```

parancs kiadása egy keretet generál a rész sorszámaival és címével. Új szakaszt a

```
\section[<szakasz rövid címe>]{<szakasz címe>}
```

parancs kereten kívüli kiadásával indíthat. Az opcióban megadott cím alapesetben a szakasz címével egyezik meg. Ez így még nem jelenik meg sehol, csak a pdf néző könyvjelzői között (a rövid cím), ha az aktiválva van, illetve a navigációs sávban (legtöbbször a rövid cím), ha az úgy van beállítva. Ha azt akarja, hogy az előző parancs kiadásakor egy külön keret jöjjön létre a címmel, akkor használhatja az `\AtBeginSection`, `\insertsection` és `\insertsectionnumber` parancsokat. Például írja be a következőket a preambulumba:

```
\AtBeginSection{
  \begin{frame}[plain]
  \begin{center}
  {\Large\insertsectionnumber. \insertsection\}
  \end{center}
  \end{frame}}
```

Ezután a

```
\section{Szakasz címe}
```

parancs kiadása egy keretet generál a szakasz sorszámaival és címével. Értelmszerű változtatásokkal hasonlóan járhat el az alszakasz és al-alszakasz esetében is.

19.5.3. Tartalomjegyzék

A rész, szakasz, alszakasz, al-alszakasz tartalmi felosztást linkek formájában megjelenítheti egy külön keretben is. Ha nem használt `\part` parancsot, akkor például a címoldal után beírhatja a következő kódot:

```
\begin{frame}[plain]{Tartalomjegyzék}
\tableofcontents
\end{frame}
```

Ha használt `\part` parancsot, akkor az előző kód csak akkor hatásos, ha a `\part` parancs kiadása után van. Ekkor a hatása nem az egész tartalomjegyzék, hanem csak az adott részé. Ha azt akarja, hogy minden rész tartalma még a `\part` parancs előtt megjelenjen például a címoldal után közvetlenül, akkor a következőt teheti:

```
\begin{frame}[plain]{I. rész tartalomjegyzéke}
\tableofcontents[part=1]
\end{frame}
\begin{frame}[plain]{II. rész tartalomjegyzéke}
\tableofcontents[part=2]
\end{frame}
```

Ha a tartalomjegyzéket tartalmazó keretben az első szakaszt kivéve minden szakasz címe elé egy `\pause` parancs hatását akarja elérni, akkor használja a `\tableofcontents` parancs `pausesections` opcióját. Ha a `pausesubsections` opciót használja, akkor azt a hatást érjük el, mintha a tartalomjegyzéket tartalmazó keretben az első alszakaszt kivéve minden alszakasz és al-alszakasz címe elé egy `\pause` parancsot írnánk.

Ha a tartalomjegyzékben nem akar például al-alszakasz címeket, vagy az éppen nem aktuális címeket csak halványan akarja megjeleníteni, akkor lehet használni a `\tableofcontents` alábbi opcióit:

```
sectionstyle=<stílus>
subsectionstyle=<stílus>
subsubsectionstyle=<stílus>
```

ahol a `<stílus>` lehet: `show` (mutat), `hide` (rejt), `shaded` (halványan). Például

```
\tableofcontents[subsubsectionstyle=hide]
```

esetén a tartalomjegyzékben nem szerepelnek az al-alszakasz címek. Ha egy adott szakszhoz készít al-tartalomjegyzéket, akkor a stílusokat kombinálhatja is.

```
sectionstyle=<stílus1>/<stílus2>
```

`<stílus1>`: aktuális szakasz címének stílusa

`<stílus2>`: többi szakasz címének stílusa

```
subsectionstyle=<stílus1>/<stílus2>
```

`<stílus1>`: aktuális alszakasz címének stílusa

`<stílus2>`: többi alszakasz címének stílusa

```
subsectionstyle=<stílus1>/<stílus2>/<stílus3>
```

`<stílus1>`: aktuális alszakasz címének stílusa

`<stílus2>`: aktuális szakasz többi alszakasz címének stílusa

`<stílus3>`: többi alszakasz címének stílusa (`hide` esetén nem csak ezen alszakasz címek, hanem azok al-alszakasz címei sem jelennek meg a tartalomjegyzékben)

```
subsubsectionstyle=<stílus1>/<stílus2>
```

`<stílus1>`: aktuális al-alszakasz címének stílusa

`<stílus2>`: többi al-alszakasz címének stílusa

```
subsubsectionstyle=<stílus1>/<stílus2>/<stílus3>
```

`<stílus1>`: aktuális al-alszakasz címének stílusa

`<stílus2>`: aktuális alszakasz többi al-alszakasz címének stílusa

`<stílus3>`: többi al-alszakasz címének stílusa

```
subsubsectionstyle=<stílus1>/<stílus2>/<stílus3>/<stílus4>
```

`<stílus1>`: aktuális al-alszakasz címének stílusa

`<stílus2>`: aktuális alszakasz többi al-alszakasz címének stílusa

`<stílus3>`: aktuális szakasz többi al-alszakasz címének stílusa

`<stílus4>`: többi al-alszakasz címének stílusa

Például a következő kódot használva az adott szakasz al-tartalomjegyzékét kapjuk:

```
\section{...}
...
\section{...}
\begin{frame}[plain]{}{}
\tableofcontents[sectionstyle=show/hide,subsectionstyle=show/show/hide]
\end{frame}
```

19.5.4. Irodalomjegyzék

Irodalomjegyzéket pontosan úgy készíthet egy kereten belül, mint a nyomtatott dokumentumok esetében. Annyi csak a különbség, hogy a `\bibitem` parancsnak itt lehet adni overlay specifikációt (alap `<1->`). Például

```
\begin{frame}[plain]{Irodalomjegyzék}
\begin{thebibliography}{12}
\bibitem<+>{Salomaa1973} A.~Salomaa, ...
\bibitem<+>{Dijkstra1982} E.~Dijkstra, ...
...
\end{thebibliography}
\end{frame}
```

19.6. Tartalmi elemek

19.6.1. Listák

A `beamer` betölti az `enumerate` csomagot. Ez nem kompatibilis a `paralist` csomaggal, így azt ne töltsse be. Ezért nem használhatja a `compactenum` és `compactitem` listakörnyezeteket sem. Ha a listákat *függőleges* extra térközök nélkül akarja, akkor a `frame` környezetet `squeeze` opcióval töltsse be. A standard környezetek használhatók: `itemize`, `enumerate`, `description`. Ezen környezeteknek nincs, de az `\item` parancsnak van overlay specifikációja, melynek alapértéke `<1->`. Például

```
\begin{frame}{}{}
\begin{itemize}
\item<+> 1. listaelem
\item<+> 2. listaelem
\item<+> 3. listaelem
\end{itemize}
\end{frame}
```

Ha az alap overlay specifikációt egy adott listában át akarja állítani például `<+>` értékre, akkor azt az alábbi módon teheti meg. (Ez ekvivalens az előző kóddal.)

```
\begin{frame}{}{}
\begin{itemize}[<+>]
\item 1. listaelem
\item 2. listaelem
\item 3. listaelem
```

```
\end{itemize}
\end{frame}
```

Ha az alap overlay specifikációt egy adott keret minden listájára át akarja állítani például $\langle + - \rangle$ értékre, akkor azt az alábbi módon teheti meg.

```
\begin{frame}[\langle + - \rangle]{\}{}
\begin{itemize}
\item 1. listaelem
\item 2. listaelem
\end{itemize}
\begin{enumerate}
\item 1. listaelem
\item 2. listaelem
\end{enumerate}
\end{frame}
```

Ha a keretnek kell például `t` opció, akkor az előző kódban az 1. sort így módosítsa:

```
\begin{frame}[\langle + - \rangle][t]{\}{}{\}
```

Ha az `\item` parancsban egyszerre használ overlay specifikációt és opciót, akkor azt ebben a sorrendben tegye. Például

```
\item\langle + - \rangle[--]
```

Ha egy adott számozott lista adott szintjének számozását akarja megváltoztatni, akkor használhatja a

```
\begin{enumerate}[\langle stílus \rangle]
```

környezetnyitást, pontosan úgy, mint például az `article` osztályban. Ha az `enumerate` környezetnél az opción túl még az alap overlay specifikációt is be akarja állítani például $\langle + - \rangle$ értékre, akkor azt így lehet megtenni:

```
\begin{enumerate}[\langle + - \rangle][\langle stílus \rangle]
```

19.6.2. Tömbök, tételszerű környezetek

A tömbök a keret olyan részei, amelyek saját fejrészsel és címmel rendelkeznek. Létrehozásuk:

```
\begin{block}<spec>{\langle tömb címe \rangle}
\langle szöveg \rangle
\end{block}
```

Az alap overlay specifikáció $\langle 1 - \rangle$. Ha ezt át akarja állítani egy adott keret tömbjeire vonatkozólag, akkor pontosan úgy kell eljárni, mint a listák esetében. Két speciális tömb is van, melyek alapvetően a színezésben térnek el: `alertblock` és `exampleblock` környezetek, melyek használata az előzőhöz hasonló.

A `beamer`-ben a tételszerű környezetek tömbként viselkednek, ahol a cím a tételszerű környezet címe. Mivel az `amsthm` csomag alapról betöltődik, ezért a `proof` környezet is használható. A tételszerű környezeteket definiálni és használni pontosan úgy kell, mint azt taglaltuk a normál esetben illetve az `amsthm` csomag tárgyalásánál, két különbséggel.

Az egyik különbség, hogy a definiált tételszerű környezetek overlay specifikációval is használhatók (alapérték $\langle 1 - \rangle$). Ha ezt át akarja állítani egy adott keret tételszerű

környezeteire vonatkozólag, akkor pontosan úgy kell eljárni, mint a listák esetében. A másik különbség, hogy a tétel számozása alapesetben nem jelenik meg. Ez átállítható a

```
\setbeamertemplate{theorems}[numbered]
```

preambulumba írásával, de magyar nyelv esetén ekkor nem kapunk jó eredményt, mert az erre vonatkozó angol tipográfiát a `magyar.ldf` nem állítja át. Ha magyar nyelv esetén mégis szeretné tételszámozást, akkor használhatja a következő megoldást:

```
\setbeamertemplate{theorems}[default]
\newtheorem{tétel}{\inserttheoremnumber. tétel}
```

19.6.3. Dobozok

Dobozok pontosan úgy használhatók a `beamer`-ben, mint normál esetben, de itt még kiegészül két bekezdésdobozzal. Ezek ismertetése előtt pár szót a `beamer` színkezeléséről. A `beamer` előre definiál saját elnevezésű színösszeállításokat, és mi is készíthetünk ilyet. Például

```
\setbeamercolor{sajat szín}{fg=blue,bg=yellow}
```

`sajat szín` néven definiál egy olyan színösszeállítást, amelyben a háttér sárga, az előtér, azaz a tartalom pedig kék. Az egyik `beamer` bekezdésdoboz a következő:

```
\begin{beamercolorbox}[\langle opció \rangle]{\langle színösszeállítás \rangle}
\langle doboz tartalma \rangle
\end{beamercolorbox}
```

Az opciók:

`wd=\langle szélesség \rangle` doboz szélessége (alapérték `\textwidth`)
`dp=\langle mélység \rangle` doboz mélysége
`ht=\langle magasság \rangle` doboz magassága
`left` doboz tartalma balra zárt
`right` doboz tartalma jobbra zárt
`center` doboz tartalma középre zárt
`sep=\langle távolság \rangle` doboz tartalma körüli extra tér nagysága
`shadow=true` doboz árnyékolt
`shadow=false` doboz nem árnyékolt
`rounded=true` doboz sarkai kerekítettek
`rounded=false` doboz sarkai nem kerekítettek

Például

```
\setbeamercolor{sajat szín}{fg=blue,bg=yellow}
\begin{frame}{}{}
\begin{beamercolorbox}[wd=6cm,shadow=true,rounded=true,center]{sajat szín}
Doboz tartalma
\end{beamercolorbox}
\end{frame}
```

A másik `beamer` által definiált bekezdésdoboz kerekített sarkú és adhatunk neki címet egy fejrészben:

```
\begin{beamerboxesrounded}[\langle opció \rangle]{\langle cím \rangle}
\langle doboz tartalma \rangle
```

```
\end{beamerboxesrounded}
```

Az opciók:

`width=<szélesség>` doboz szélessége (alapérték `\textwidth`)
`shadow=true` doboz árnyékolt
`shadow=false` doboz nem árnyékolt
`lower=<színösszeállítás>` doboz tartalmának színösszeállítása
`upper=<színösszeállítás>` doboz fejrészének színösszeállítása

Például

```
\setbeamercolor{sajat szín1}{fg=white,bg=blue}
\setbeamercolor{sajat szín2}{fg=black,bg=yellow}
\begin{frame}{}{}
\begin{beamerboxesrounded}[upper=sajat szín1,lower=sajat szín2]{Cím}
Doboz tartalma
\end{beamerboxesrounded}
\end{frame}
```

19.6.4. Többhasábos terület

```
\begin{columns}[<opció>]
\begin{column}{<1. oszlop szélessége>}
<1. oszlop tartalma>
\end{column}
\begin{column}{<2. oszlop szélessége>}
<2. oszlop tartalma>
\end{column}
...
\end{columns}
```

Az opciók:

`totalwidth=<szélesség>` a többhasábos terület teljes szélessége
b az oszlopok alsó sorainak alapvonalát igazítja össze
c az oszlopok vertikális közepét igazítja össze
t az oszlopok felső sorainak alapvonalát igazítja össze
T az oszlopok felső sorainak tetejét igazítja össze

19.6.5. Háttér

A háttér színe a következő kóddal állítható be:

```
\setbeamercolor{background canvas}{bg=<színnév>}
```

Lehetőség van többszínű háttér készítésére is:

```
\setbeamertemplate{background canvas}
[vertical shading][top=<színnév>,middle=<színnév>,bottom=<színnév>]
```

A `top`, `middle` és `bottom` opciók mellett használható még a `midpoint` opció is, amivel azt lehet megadni, hogy hol legyen a függőleges pozíciója a `middle`-ben megadott szín középszintjének. Ez egy 0 és 1 közötti arányszám, ahol 0 jelenti a legalsó szintet, 1 pedig a legfelsőt. Például


```
\setbeamertemplate{background canvas}
[vertical shading][midpoint=0.3,middle=yellow]
```

Ha háttérképet akar a diáknak, akkor a preambulumba írja a következőt:

```
\setbeamertemplate{background canvas}
{\includegraphics[width=\paperwidth]{\langle kép \rangle}}
```

Ekkor a *\langle kép \rangle* minden dia hátterén megjelenik. Ha ugyanezt egyetlen keret diáira akarja elérni, akkor ezt kell tenni:

```
{\setbeamertemplate{background canvas}
{\includegraphics[width=\paperwidth]{\langle kép \rangle}}
\begin{frame}
...
\end{frame}}
```

19.6.6. Képek

A képek beillesztése, hasonlóan a normál esethez, történhet az `\includegraphics` paranccsal, de itt van overlay specifikációja, melynek alapértéke `<1->`.

```
\includegraphics<\langle spec \rangle>[\langle opció \rangle]{\langle kép \rangle}
```

A nem jelölt diákon nem foglalja a helyet a kép. Ez azért van így, hogy könnyebben lehessen egy képsorozatból animációt csinálni. Például

```
\includegraphics<+>[width=5cm]{figure0}
\includegraphics<+>[width=5cm]{figure1}
\includegraphics<+>[width=5cm]{figure2}
\includegraphics<+>[width=5cm]{figure3}
```

19.6.7. Animáció

Az előző kód már tekinthető animációnak, de ha sok képből áll, akkor a kód is sok sorból áll, ami kényelmetlen. Ez a probléma megoldható az `xmpmulti` csomaggal. Tegyük fel, hogy az animáció a `fig-0.png`, `fig-1.png`, `fig-2.png`, ..., `fig-20.png` képekből áll. Ekkor a

```
\multiinclude<+>[format=jpg,graphics={width=5cm}]{fig} ∈ xmpmulti
```

kód ekvivalens ezzel:

```
\includegraphics<+>[width=5cm]{fig-0}
\includegraphics<+>[width=5cm]{fig-1}
\includegraphics<+>[width=5cm]{fig-2}
...
\includegraphics<+>[width=5cm]{fig-20}
```

Ez a megoldás kódírás szempontjából már kényelmes, de a prezentáció használata még nem az, hiszen minden képváltáshoz léptetni kell a számítógépen. Ez a gond megoldható például a korábban már ismertetett `\transduration` paranccsal is, de sokkal szebb megoldást ad az

```
\animategraphics[\langle opció \rangle]{\langle sebesség \rangle}{\langle alapnév \rangle}{\langle első \rangle}{\langle utolsó \rangle} ∈ animate
```

parancs. Ez a kód képsorozatot videóként fogja lejátszani, feltéve, hogy ez a funkció a pdf nézőben támogatott. Az Adobe Reader ilyen, de a SumatraPDF vagy a TeXstudio beépített pdf-nézője nem. Ezt a parancsot csak olyan keretben használja, ahol egyetlen dia van.

`<sebesség>` Pozitív egész, ennyi kép/másodperc sebességgel játssza le.

`<alapnév>` Például ha a képfájlok sorra `fig0.png`, `fig1.png`, ..., `fig20.png`, akkor ide `fig` kerül.

`<első>` Az előző példában ide 0 kerül.

`<utolsó>` Az előző példában ide 20 kerül.

A lehetséges opciók:

`autoplay` Az oldal megnyitásakor automatikusan indul a lejátszás.

`loop` A lejátszás végén automatikusan újraindul.

`width=<szélesség>` A képek szélessége.

`height=<magasság>` A képek magassága.

`controls` Lejátszó gombok jelenjenek meg.

`buttonsize=<gombméret>` Lejátszó gombok mérete.

`buttonbg=<szín>` Lejátszó gombok hátterének a színe.

`buttonfg=<szín>` Lejátszó gombok vonalának a színe. A `<szín>` megadása szürke skálával vagy rgb palettával történhet. Például `buttonbg=0.8` vagy `buttonbg=0.36:0.08:0.88` (Ha a `magyar.ldf`-et használjuk, akkor a kettőspont aktív váételét ki kell kapcsolni, különben a `buttonbg` és `buttonfg` opciók nem használhatók, csak szürke skálával.)

Az `\animategraphics` parancs természetesen nem csak a `beamer` dokumentumosztályban használható, de akkor az `animate` mellett töltsse be a `graphicx` csomagot is.

Animált gif közvetlenül nem építhető pdf-be. Ilyenkor a gif fájlt konvertálni kell png képekből álló sorozatba, amely már az előző módon megjeleníthető pdf-ben is. A konvertáláshoz használhatja például az `ImageMagick` programot. Telepítés után a következő parancssorral végezheti a konvertálást:

```
convert -coalesce <fájlnév>.gif <fájlnév>.png
```

19.6.8. Videó

Arra is lehetőség van, hogy videót játsszon le a prezentáció egy keretén belül a következő parancssal:

`\movie[<opció>]{<poszter>}{<videófájl>} \in multimedia`

Figyeljünk arra, hogy videó lejátszása idegen gépen nem feltétlenül fog működni. Például ha a gépre a lejátszáshoz szüksége codec nincs telepítve, hibát fog jelezni.

Csak a lejátszás történik a pdf fájlban belül, a videó fájl nem épül be a pdf-be. Így vetítéskor a videó fájlt be kell másolni a pdf fájl mellé.

A másik ami gondot jelenthet, hogy a pdf néző program biztonsági kockázatnak tarthatja a videók lejátszását. Ezt külön be kell állítani vetítés előtt.

Amíg nem indul el a videó, a `<poster>` látható a videónak kijelölt területen, hacsak nem adta meg a `poster` opciót (lásd később). Erre kattintva indul a lejátszás. A `<poster>` lehet szöveg és `\includegraphics` paranccsal betöltött kép is.

A lehetséges opciók:

`width=<szélesség>` A videó szélessége.

`height=<magasság>` A videó magassága.

`poster` Amíg a videó nem indul el, nem a `<poster>` látható, hanem a videó első képkockája. Erre kattintva indul a lejátszás.

`showcontrols` Mutatja a videó alatt a navigációs sávot.

`start=<idő>` A videó lejátszási kezdőpontjának megadása. Például `start=5s` azt jelenti, hogy a lejátszási kezdőpont az 5. másodperc.

`duration=<idő>` A videóból milyen hosszú részt játsszon le. Például `duration=25s` azt jelenti, hogy 25 másodpercnyi részt játszik le.

Például

```
\movie[width=8cm,height=6cm,showcontrols,poster]{}{video.avi}
```

Arra is lehetőség van a `label` opció és `\hyperlinkmovie` \in `multimedia` parancs együttes használatával, hogy a videónak különböző időintervallumait játssza le egy-egy linkre kattintással. Ezeknek a linkeknek ugyanazon a dián kell lenniük, mint ahol a videó van.

Például

```
\begin{frame}{}{}
\movie[label=cimke,width=8cm,height=6cm,showcontrols,poster]{}{video.avi}
\par\medskip
\hyperlinkmovie[start=5s,duration=10s]{cimke}{5--15\,sec}
\par
\hyperlinkmovie[start=20s,duration=25s]{cimke}{20--45\,sec}
\end{frame}
```

Ha a videót nem a pdf fájlban, hanem csak egy linkre kattintva, külső alkalmazással akarja lejátszani, akkor nincs szükség a `multimedia` csomagra:

```
\href{run:<videófájl>}{<link szövege>}
```

Ez a parancs természetesen csak akkor működik, ha a gépen az `avi`-hoz külső alkalmazás van rendelve.

19.6.9. Nagyítás

Lehetőség van arra, hogy a dia egy adott területét kinagyítsa a `\framezoom` paranccsal. Például a

```
\framezoom<1><2>[border=3](1cm,2cm)(4cm,3cm)
```

parancsot a keret elejére írva a következő történik. Az 1. dián meg fog jelenni egy 3 pixel vastag keret egy 4 cm \times 3 cm méretű téglalap körül, melynek a bal felső sarka 1 cm távolságra van a szövegtükör bal oldalától és 2 cm-re a szövegtükör tetejétől. A kijelölt terület linkként működik, rákattintva a 2. diához jutunk, melyen az előbbi kijelölt részt láthatjuk a teljes dia méretére kinagyítva. A 2. dia teljes területe is linkként működik, rákattintva visszajutunk az 1. diára. (A linkek akkor fognak helyesen működni, ha teljes képernyős üzemmódban van a pdf néző.) Például

```
\begin{frame}{}{}{}
```

```
\framezoom<2><3>[border=3](1cm,0.5cm)(5cm,3.75cm)
\framezoom<2><4>[border=3](6.2cm,0.2cm)(4.5cm,3.375cm)
\framezoom<2><5>[border=3](2cm,5cm)(4cm,3cm)
\includegraphics[width=\textwidth]{pic}
\end{frame}
```

létrehoz egy 5 diából álló keretet. Az 1. dián betölt egy `pic.jpg` képet, majd a másodikon kijelöli a nagyítandó részeket. Ezekre kattintva megnézhetjük a nagyítást.

19.6.10. Kereszthivatkozás

Próbálja ki a következő kódot:

```
\begin{frame}{Példa}{}
\begin{itemize}
\item<+> 1. listaelem
\item<+> 2. listaelem
\begin{equation}\label{egyenlet}
a^2+b^2=c^2
\end{equation}
\end{itemize}
\end{frame}

\begin{frame}
\eqref{egyenlet}
\end{frame}
```

Azt fogja tapasztalni, hogy az `\eqref{egyenlet}` által létrehozott linkre kattintva nem az egyenlethez ugrik a prezentáció, azaz nem a „Példa” című keret 2. diájához, hanem az 1. diájához. Ennek a problémának a megoldására kapott a `\label` parancs is overlay specifikációt, melynek alapértéke `<1>` (ezért ugrik a link az előző esetben az 1. diára). Így az előző kód helyesen:

```
\begin{frame}{Példa}{}
\begin{itemize}
\item<+> 1. listaelem
\item<+> 2. listaelem
\begin{equation}\label<2>{egyenlet}
a^2+b^2=c^2
\end{equation}
\end{itemize}
\end{frame}

\begin{frame}
\eqref{egyenlet}
\end{frame}
```

Ha egy keret adott diájára akar hivatkozni, akkor használja a `frame` környezet `label` opcióját:

```
\begin{frame}[label=cimke]{Példa}{}
\begin{itemize}
\item<+> 1. listaelem
\item<+> 2. listaelem
```

```

\end{itemize}
\end{frame}

\begin{frame}
\ref{cimke<2>}
\end{frame}

```

Ekkor a `\ref{cimke<2>}` létrehoz egy keretszámot tartalmazó linket, melyre kattintva a keret 2. diájára ugrik.

Ha `\ref` helyett a `\hyperlink` parancsot használja, akkor a link szövegét mi adhatjuk meg. Például

```

\begin{frame}[label=cimke]{Példa}{ }
\begin{itemize}
\item<+>-> 1. listaelem
\item<+>-> 2. listaelem
\end{itemize}
\end{frame}

\begin{frame}
\hyperlink{cimke<2>}{Az előző keret 2. diájára ugrás.}
\end{frame}

```

19.6.11. Nyomógombok

Linknek nem csak szöveg, hanem gomb is megadható:

```

\beamerbutton{<gomb szövege>}
\beamergetobutton{<gomb szövege>}
\beamerskipbutton{<gomb szövege>}
\beamerreturnbutton{<gomb szövege>}

```

Például az előző kód második keretét javítsa ki erre:

```

\begin{frame}
\hyperlink{cimke<2>}{\beamerreturnbutton{Előző keret 2. diája}}
\end{frame}

```

A nyomógombok szimbólumait az

```

\insertgotosymbol
\insertskipsymbol
\insertreturnsymbol

```

parancsok átdefiniálásával változtathatja meg. A gomb színeit és a szöveg betűtípusát is állíthatja. Például

```

\renewcommand{\insertgotosymbol}{$\ggg$}
\setbeamercolor{button}{fg=black,bg=yellow}
\setbeamercolor{button border}{fg=red}
\setbeamerfont{button}{family=\rmfamily,shape=\itshape,series=\bfseries}

```

19.6.12. Keret ismétlése

Ha egy keretet `label` opcióval töltötte be, akkor lehetőség van az `\againframe` paranccsal a keret tartalmát egy másik ponton is megjeleníteni, esetleg más overlay specifikációval, más opcióval. Például

```
\begin{frame}[<+>][label=cimke]{Példa}{}  
  \begin{itemize}  
    \item 1. listaelem  
    \item 2. listaelem  
    \item 3. listaelem  
  \end{itemize}  
\end{frame}  
  
\againframe<2->[<+>][t]{cimke}
```

20. fejezet

A L^AT_EX programozása

20.1. ASCII kódolás és kategória kódok

A forrás fordítása során a L^AT_EX először minden karaktert megvizsgál. Ha az benne van a következő két táblázatban – azaz ún. ASCII karakter –, akkor megtartja, de ha nincs, akkor egy megfelelő parancsot ír be a helyére, ami már csak ASCII karaktereket tartalmaz. Például az ũ karakter helyére berakja a \H{0} parancsot. Ha a forrásfájl UTF-8 kódolású, akkor ezt a munkát 2018-tól már alapesetben elvégzi a L^AT_EX, de korábban az `inputenc` csomag `utf8` opciója konvertált. Ha más kódolású a forrás, akkor kötelező használni az `inputenc` csomagot a kódolásnak megfelelő opcióval.

Az ASCII kódolás 8 bites, de ebből az első 0, és csak a többi hetet variálja. Így ASCII kódolású karakterekből $2^7 = 128$ darab van.

Nyomtatható karakterek decimális ASCII-kódjai

karakter	kód	karakter	kód	karakter	kód	karakter	kód
szóköz	32	8	56	P	80	h	104
!	33	9	57	Q	81	i	105
"	34	:	58	R	82	j	106
#	35	;	59	S	83	k	107
\$	36	<	60	T	84	l	108
%	37	=	61	U	85	m	109
&	38	>	62	V	86	n	110
'	39	?	63	W	87	o	111
(40	@	64	X	88	p	112
)	41	A	65	Y	89	q	113
*	42	B	66	Z	90	r	114
+	43	C	67	[91	s	115
,	44	D	68	\	92	t	116
-	45	E	69]	93	u	117
.	46	F	70	^	94	v	118
/	47	G	71	_	95	w	119
0	48	H	72	`	96	x	120
1	49	I	73	a	97	y	121
2	50	J	74	b	98	z	122
3	51	K	75	c	99	{	123
4	52	L	76	d	100		124
5	53	M	77	e	101	}	125
6	54	N	78	f	102	~	126
7	55	O	79	g	103		

Vezérlő karakterek decimális ASCII-kódjai

leírás	utalás	kód
lezáró nulla	<code>^^@</code>	0
fejléc kezdete	<code>^^A</code>	1
szöveg kezdete	<code>^^B</code>	2
szöveg vége	<code>^^C</code>	3
adatátvitel vége	<code>^^D</code>	4
vizsgálat	<code>^^E</code>	5
viSSzaigazolás	<code>^^F</code>	6
csengetés	<code>^^G</code>	7
viSSzalépés	<code>^^H</code>	8
vízszintes tabulátor	<code>^^I</code>	9
új sor	<code>^^J</code>	10
függőleges tabulátor	<code>^^K</code>	11
lapdobás	<code>^^L</code>	12
kocsi vissza	<code>^^M</code>	13
karakterkészlet váltása	<code>^^N</code>	14
karakterkészlet visszaállítása	<code>^^O</code>	15
nyers adat következik	<code>^^P</code>	16
eszközvezérlés 1	<code>^^Q</code>	17
eszközvezérlés 2	<code>^^R</code>	18
eszközvezérlés 3	<code>^^S</code>	19
eszközvezérlés 4	<code>^^T</code>	20
negatív viSSzaigazolás	<code>^^U</code>	21
szinkron üresjárat	<code>^^V</code>	22
adatátviteli blokk vége	<code>^^W</code>	23
mégsem	<code>^^X</code>	24
adathordozó vége	<code>^^Y</code>	25
helyettesítő karakter	<code>^^Z</code>	26
feloldójel	<code>^^[</code>	27
állományelválasztó	<code>^^\</code>	28
csoportelválasztó	<code>^^]</code>	29
rekordelválasztó	<code>^^^</code>	30
egységelválasztó	<code>^^_</code>	31
törlés	<code>^^?</code>	127

Nem csak vezérlő karakterekre lehet utalni `^^` után szereplő valamilyen karakterrel. Általánosságban, ha a karakter decimális ASCII-kódja x és a karakterre utalásban a `^^` után álló karakter decimális ASCII-kódja y , akkor

$$x = \begin{cases} y + 64, & \text{ha } y = 0, 1, \dots, 63, \\ y - 64, & \text{ha } y = 64, 65, \dots, 127, \end{cases}$$

illetve

$$y = \begin{cases} x + 64, & \text{ha } x = 0, 1, \dots, 63, \\ x - 64, & \text{ha } x = 64, 65, \dots, 127. \end{cases}$$

Például a szóköz decimális ASCII-kódja $x = 32$, így $y = 32 + 64 = 96$, amely a ``` karakternek az ASCII-kódja. Tehát a szóközre a `^^`` jelsorozattal utalhatunk. A `^^I`-ben az `I` decimális ASCII-kódja $y = 73$, így $x = 73 - 64 = 9$, ami a vízszintes tabulátor decimális ASCII-kódja. Tehát a `^^I` a vízszintes tabulátorra utal.

Egy ASCII-kódolású karakter decimális ASCII-kódját a következő kód tárolja:

■ `\<karakter>`

ahol a `<karakter>` lehet utalás is. Ha ezt ki is akarja írni a dokumentumában, akkor elé kell tenni a `\number` parancsot. Amennyiben a ``` karakter aktív – ahogyan az a `magyar.ldf` használatakor is van –, ezen jel elé tegye ki a `\string` parancsot. Például az A betű ASCII-kódja

■ `\number\string`\A`

65

A „vízszintes tabulátor” ASCII-kódja

■ `\number\string`\^^I`

9

Minden ASCII-kódolású karakternek van a \LaTeX -ben egy úgynevezett kategória kódja is, amellyel ezeket a karakterek 16 különböző osztályba soroljuk:

- 0 → Parancsot bevezető karakter (alapesetben a `\` jel).
- 1 → Blokk nyitó karakter (alapesetben a `{` jel).
- 2 → Blokk csukó karakter (alapesetben a `}` jel).
- 3 → Sorközi matematikai módváltó karakter (alapesetben a `$` jel).
- 4 → Tabulátort jelölő karakter (alapesetben a `&` jel).
- 5 → Sorvégét jelölő karakter (alapesetben a 13 ASCII-kódú „kocsi vissza” vezérlőkarakter).
- 6 → Makróparamétert jelölő karakter (alapesetben a `#` jel).
- 7 → Felső indexet jelölő karakter (alapesetben a `^` jel).
- 8 → Alsó indexet jelölő karakter (alapesetben a `_` jel).
- 9 → Figyelmen kívül hagyható karakter (plain \TeX -ben ilyen a „lezáró nulla” vezérlőkarakter, \LaTeX -ben nincs ilyen).
- 10 → Szóközt jelölő karakter (alapesetben a 9 és 32 ASCII-kódú karakterek, azaz a „vízszintes tabulátor” vezérlőkarakter és a szóköz).
- 11 → Betűt jelölő karakter (alapesetben az a-tól z-ig, illetve A-tól Z-ig terjedő karakterek).
- 12 → Egyéb karakterek (alapesetben a 10, 33, 34, 39, 40–64, 91, 93, 96, 124 ASCII-kódú karakterek).
- 13 → Aktív karakter, amely parancsot bevezető karakter nélkül, önmagában is parancsnak minősül (alapesetben a 1–8, 11, 12, 14–31, 126 ASCII-kódú karakterek, azaz három kivétellel az összes vezérlőkarakter és a `~` jel).
- 14 → Kommentet jelölő karakter (alapesetben a `%` jel).
- 15 → Érvénytelen karakter, amely fordítási hibát eredményez (alapesetben a 0 és 127 ASCII-kódú karakterek).

Az ASCII-kód ugyan 8 bites, de alapesetben az első bit mindig 0, és csak a következő 7 bitet variálja, így jön ki az összesen $2^7 = 128$ darab karakter, melyek decimális kódjai 0-tól 127-ig terjednek. A kiterjesztett ASCII-kódolás használja az első bitet is, így ebben már $2^8 = 256$ karakter szerepel 0-tól 255-ig terjedő decimális kódokkal. A \LaTeX a 128 és 255 közötti ASCII-kódú karakterekhez a 13 kategóriakódot rendeli.

Egy karakter kategóriakódját a következő kód tárolja:

■ `\catcode<karakter ASCII-kódja>`

ahol a $\langle \text{karakter ASCII-kódja} \rangle$ megadható decimális, oktális és hexadecimális formában is. Ha oktális formát használ, akkor a `'` jelet, míg ha hexadecimális formát használ, akkor a `"` jelet kell elé gépelni. Ha a kategóriakódot ki akarja íratni a dokumentumában, akkor a `\catcode` elé kell írni a `\number` parancsot. Például a „törlés” vezérlőkarakter kategóriakódját (15) a következő sorok bármelyike kiírja:

```
\number\catcode127
\number\catcode\string`^^?
\number\catcode'177
\number\catcode"7F
```

Ha egy karakter kategóriakódját át szeretné állítani, akkor használja a

```
\catcode\langle \text{karakter ASCII-kódja} \rangle = \langle \text{kategóriakód} \rangle
```

parancsot, ahol a $\langle \text{karakter ASCII-kódja} \rangle$, hasonlóan az előzőekben leírtakkal, megadható decimális, oktális és hexadecimális formában is. Például a „vízszintes tabulátor” vezérlőkarakter a következő sorok bármelyikével a 10 kategóriakóddal lesz ellátva:

```
\catcode9=10
\catcode\string`^^I=10
\catcode'11=10
\catcode"9=10
```

Az eddigi példákban azért szerepelt a `\string` parancs, hogy a magyarban aktív ``` karakter szerepét ideiglenesen kikapcsolja. Valójában az történik, hogy a `\string` után álló karakter 12 kategóriakódot kapja erre az egy esetre. Ha a `\string` után parancs áll, akkor a parancsot bevezető karakter és a parancsszó minden karaktere is 12 kategóriakódot kapja erre az egy esetre. Tehát például

```
\string\TeX\TeX
```

esetén az első `\` jel és az azt követő `T e X` betűk mindegyike 12 kategóriakóddal fognak szerepelni a fordítás során, de a második `\` jel már ismét 0 kategóriakódú, így az utána következő karaktereket már parancsnak tekinti. Tehát az előző kód eredménye

```
\TeXTeX
```

20.2. Hosszúságparancsok

A \LaTeX -ben vannak olyan parancsok, melyek hosszúságokat hordoznak. Ezek az ún. hosszúságparancsok. Ilyen például a `\textwidth`.

`\newlength{ $\langle \text{új hosszúságparancs} \rangle$ }` Új hosszúságparancs definiálása. Ennek kezdőértéke 0 pt. Például `\newlength{\sajathossz}`.

`\setlength{ $\langle \text{hosszúságparancs} \rangle$ }{ $\langle \text{hossz} \rangle$ }` A $\langle \text{hosszúságparancs} \rangle$ értéke $\langle \text{hossz} \rangle$ lesz. Például `\setlength{\sajathossz}{2cm}` esetén az előbb definiált `\sajathossz` értéke 2 cm lesz.

`\setlength{ $\langle \text{hosszúságparancs} \rangle$ }{ $\langle \text{hossz} \rangle$ plus $\langle \text{hossz1} \rangle$ minus $\langle \text{hossz2} \rangle$ }` A hosszúságparancs beállítása rugalmas méretre.

Például `\setlength{\sajathossz}{15pt plus 7pt minus 3pt}`

`\setlength{ $\langle \text{hosszúságparancs} \rangle$ }{ $\langle \text{más hosszúságparancs} \rangle$ }` A $\langle \text{hosszúságparancs} \rangle$ átveszi a $\langle \text{más hosszúságparancs} \rangle$ értékét.

Például `\setlength{\sajathossz}{\textwidth}`

`\setlength{<hosszúságparancs>}{<szorzó><más hosszúságparancs>}` A `<hosszúságparancs>` a `<más hosszúságparancs>` értékének `<szorzó>` szorosát veszi át. Ha rugalmas hosszt szoroztunk, akkor az rugalmatlanná válik.

Például `\setlength{\sajathossz}{0.3\textwidth}`.

`\settowidth{<hosszúságparancs>}{<szöveg>}` felveszi a `<szöveg>` szélességét.

`\settoheight{<hosszúságparancs>}{<szöveg>}` felveszi a `<szöveg>` magasságát (az alapvonal fölötti rész magassága).

`\settodepth{<hosszúságparancs>}{<szöveg>}` felveszi a `<szöveg>` mélységét (az alapvonal alatti rész magassága).

`\addtolength{<hosszúságparancs>}{<hossz>}` a `<hosszúságparancs>` értékét `<hossz>`-al megnöveli. A `<hossz>` lehet negatív is, ami csökkenést eredményez. (Rugalmas hosszúságok is összeadhatók.)

`\the<hosszúságparancs>` megadja a `<hosszúságparancs>` aktuális értékét pt-ban mérve. Például `\the\textwidth`.

`\uselengthunit{<mértékegység>}\printlength{<hosszúságparancs>}` `\in printlen` megadja a `<hosszúságparancs>` aktuális értékét `<mértékegység>`-ben mérve. Például `\uselengthunit{cm}\printlength{\paperwidth}`.

Például

```
\newlength{\hossz}
\newlength{\melyseg}
\settoheight{\hossz}{vizsga}
\settodepth{\melyseg}{vizsga}
A vizsga szó magassága \the\hossz, mélysége \the\melyseg,
ezek összege pedig \addtolength{\hossz}{\melyseg}\the\hossz.
```

A vizsga szó magassága 7.96234pt, mélysége 2.33276pt, ezek összege pedig 10.2951pt.

20.3. Számlálók

A \LaTeX számlálói egész számokat tárolnak. A beépített számlálók a következők:

<code>part</code>	rész sorszáma
<code>chapter</code>	fejezet sorszáma
<code>section</code>	szakasz sorszáma
<code>subsection</code>	alszakasz sorszáma
<code>subsubsection</code>	al-alszakasz sorszáma
<code>paragraph</code>	paragrafus sorszáma
<code>subparagraph</code>	alparagrafus sorszáma
<code>tocdepth</code>	mi kerül a tartalomjegyzékbe
<code>secnumdepth</code>	szintek számozásának mélysége
<code>page</code>	oldalszám
<code>equation</code>	egyenlet sorszáma
<code>figure</code>	ábra sorszáma
<code>table</code>	táblázat sorszáma
<code>enumi</code>	lista 1. szintjének sorszáma
<code>enumii</code>	lista 2. szintjének sorszáma
<code>enumiii</code>	lista 3. szintjének sorszáma

<code>enumiv</code>	lista 4. szintjének sorszáma
<code>footnote</code>	lábjegyzet sorszáma
<code>mpfootnote</code>	lábjegyzet sorszáma <code>minipage</code> környezetben

A `\newtheorem` paranccsal létrehozott tételszerű környezet is generál egy számlálót. Például, ha a `lemma` tételszerű környezetet definiáltuk, akkor annak számozására létrejön egy `lemma` számláló is.

A számlálók kezelésére használt parancsok:

`\newcounter{<új számláló>}` Új számlálót definiál. Ennek kezdőértéke 0.
`\newcounter{<új számláló>}[<számláló>]` Új számlálót definiál. Ennek kezdőértéke 0. Ha a `<számláló>` értéke megváltozik, akkor az `<új számláló>` értéke lenullázódik.
`\setcounter{<számláló>}{<szám>}` A `<számláló>` értéke `<szám>` lesz.
`\addtocounter{<számláló>}{<szám>}` A `<számláló>` értékét megnöveli a `<szám>` értékével, ami lehet negatív is.
`\stepcounter{<számláló>}` A `<számláló>` értékét megnöveli 1-gyel.
`\refstepcounter{<számláló>}` A `<számláló>` értékét megnöveli 1-gyel, továbbá, ha ezután helyezünk el egy címkét a `\label` paranccsal, akkor egy erre való hivatkozás a `\ref` paranccsal, a `<számláló>` itteni értékét írja ki.
`\value{<számláló>}` A `<számláló>` értékének átadására használható. Például `\setcounter{secnumdepth}{\value{tocdepth}}` hatására a `secnumdepth` értéke felveszi a `tocdepth` aktuális értékét.
`\multiply\value{<számláló>} by <szám>` A `<számláló>` értékét megszorozza a `<szám>`-mal.
`\divide\value{<számláló>} by <szám>` A `<számláló>` értékét elosztja a `<szám>`-mal és veszi az egész részét.

A számlálók kiírására használt parancsok:

`\arabic{<számláló>}` A `<számláló>` értékének kiírása arab számozással.
`\Roman{<számláló>}` A `<számláló>` értékének kiírása nagy római számozással.
`\roman{<számláló>}` A `<számláló>` értékének kiírása kis római számozással.
`\Alph{<számláló>}` A `<számláló>` értékének kiírása nagy alfanumerikus számozással.
`\alph{<számláló>}` A `<számláló>` értékének kiírása kis alfanumerikus számozással.
`\fnsymbol{<számláló>}` A `<számláló>` értékének kiírása szimbólumokkal: *, †, ‡, §, ¶, ||, **, ††, ‡‡
`\the<számláló>` Ez a parancs a `<számláló>` létrehozásakor definiálódik, ami a `<számláló>` értékét arab számozással írja ki. Átdefiniálható például a következőképpen:
`\renewcommand{\thepage}{\roman{page}}` vagy
`\renewcommand{\thesubsection}{\thesection.\arabic{subsection}}`.
`\numberwithin{<számláló1>}{<számláló2>}` \in **amsmath** Ha a `<számláló2>` értéke megváltozik, akkor a `<számláló1>` értéke lenullázódik, továbbá a `\the<számláló1>` kifejtése `\the<számláló2>.\arabic{<számláló1>}`.

Néhány példa:

```
\newcounter{szamA}
\newcounter{szamB}
\numberwithin{szamB}{szamA}
\stepcounter{szamA}
\setcounter{szamB}{2}
\theszamB;
```

```
\stepcounter{szamA}
\theszamB
```

1.2; 2.0

```
\newcounter{egyik}
\newcounter{masik}
\newcounter{szorzat}
\setcounter{egyik}{5}
\setcounter{masik}{2}
\setcounter{szorzat}{\value{egyik}}
\Roman{egyik} és \Roman{masik} szorzata
\multiply\value{szorzat}by\value{masik}\Roman{szorzat}.
```

V és II szorzata X.

```
\newcounter{szamA}
\newcounter{szamB}
\newcounter{szamC}
\setcounter{szamA}{2015}
\setcounter{szamB}{44}
\setcounter{szamC}{\value{szamA}}
\multiply\value{szamC} by \value{szamB}
\divide\value{szamC} by 100
$\theszamC=\left[\theszamA\cdot\frac{\theszamB}{100}\right]$\
```

$886 = \left[2015 \cdot \frac{44}{100}\right]$

20.4. Vezérlő utasítások

A továbbiakban többször fogunk találkozni olyan parancsokkal, melyek @ karaktert tartalmaznak. Ezeket belső parancsoknak nevezzük, melyek egy egyszerű L^AT_EX forrásállományban nem használhatók, csak az osztály- (.cls) és csomagfájlokban (.sty). (Az osztályfájl `\documentclass` míg a csomagfájl `\usepackage` segítségével töltjük be.) Ha mégis szeretne egy belső parancsot használni egyszerű L^AT_EX forrásállományban, akkor azt zárja a `\makeatletter` és `\makeatother` parancsok közé. Ezen parancsok nem kerülhetnek új parancsot leíró makróba. Például helytelen:

```
\def\hacsillag#1#2{\makeatletter\@ifstar{#1}{#2}\makeatother} % ROSSZ KÓD!
```

A következő kód a helyes:

```
\makeatletter
\def\hacsillag#1#2{\@ifstar{#1}{#2}}
\makeatother
```

20.4.1. Feltételes utasítások

`\if@twoside<igaz>\fi` Az eredmény *<igaz>*, ha a dokumentumosztályt `twoside` opcióval töltöttük be.

`\if@twoside<igaz>\else<hamis>\fi` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a dokumentumosztályt `twoside` opcióval töltöttük-e be vagy sem.

`\@ifnextchar<karakter>{<igaz>}{<hamis>}` Az eredmény aszerint lesz *<igaz>* vagy *<hamis>*, hogy a következő első karakter *<karakter>* vagy sem.

Például `(\@ifnextchar c{a}{b}c)` eredménye (ac).

`\@ifstar{<igaz>}{<hamis>}` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a következő első karakter *** vagy sem. Ha ***, akkor azt elnyeli. Például `(\@ifstar{a}{b}*)` eredménye (a), illetve `(\@ifstar{a}{b}x)` eredménye (bx).

`\@ifundefined{<parancs>}{<igaz>}{<hamis>}` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<parancs>* nem definiált vagy definiált.

Például `(\@ifundefined{section}{a}{b})` eredménye (b), hiszen a `\section` definiált.

`\@ifundefined{DeclareUnicodeCharacter}{<igaz>}{<hamis>}` aszerint lesz az eredmény *<igaz>* vagy *<hamis>*, hogy az `inputenc` csomag `utf8` opcióval volt-e betöltve vagy sem.

`\@ifclassloaded{<osztály>}{<igaz>}{<hamis>}` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy az *<osztály>* dokumentumosztályt töltöttük-e be vagy sem. Csak preambulumban használható!

`\@ifpackageloaded{<csomag>}{<igaz>}{<hamis>}` Aszerint *<igaz>* vagy *<hamis>* az eredmény, hogy a *<csomag>* csomag be volt-e korábban töltve vagy sem. Csak preambulumban használható!

`\if<kód><igaz>\fi` Az eredmény *<igaz>*, ha a *<kód>* eredményében az első két karakter megegyezik.

`\if<kód><igaz>\else<hamis>\fi` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<kód>* eredményében az első két karakter megegyezik-e vagy sem.

`\ifx<kód1><kód2><igaz>\fi` Az eredmény *<igaz>*, ha a *<kód1>* és *<kód2>* eredménye ugyanaz.

`\ifx<kód1><kód2><igaz>\else<hamis>\fi` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<kód1>* és *<kód2>* eredménye ugyanaz-e vagy sem.

`\ifx<kód1><kód2><igaz>\else<hamis>\fi` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<kód1>* és *<kód2>* eredménye ugyanaz-e vagy sem.

`\ifx\@onlypreamble\@notprerr<igaz>\else<hamis>\fi` Aszerint *<igaz>* vagy *<hamis>* az eredmény, hogy ez a kód a dokumentumtestben vagy a preambulumban van.

`\ifnum<numerikus feltétel><igaz>\fi` Az eredmény *<igaz>*, ha a *<numerikus feltétel>* teljesül. Például `\ifnum10<20(a)\fi` eredménye (a).

`\ifnum<numerikus feltétel><igaz>\else<hamis>\fi` Az eredmény aszerint lesz *<igaz>* vagy *<hamis>*, hogy a *<numerikus feltétel>* teljesül-e vagy sem.

Például `\ifnum\value{page}>10(a)\else(b)\fi` eredménye (a).

`\ifodd<egész szám><igaz>\fi` Az eredmény *<igaz>*, ha az *<egész szám>* értéke páratlan.

`\ifodd<egész szám><igaz>\else<hamis>\fi` Az eredmény aszerint lesz *<igaz>* vagy *<hamis>*, hogy az *<egész szám>* értéke páratlan vagy páros.

Például `\ifodd\value{page}(a)\else(b)\fi` eredménye (b).

`\ifdim<hosszúság feltétel><igaz>\fi` Az eredmény *<igaz>*, ha a *<hosszúság feltétel>* teljesül. Például `\ifdim\textwidth<\textheight(a)\fi` eredménye (a).

`\ifdim<hosszúság feltétel><igaz>\else<hamis>\fi` Az eredmény aszerint lesz *<igaz>* vagy *<hamis>*, hogy a *<hosszúság feltétel>* teljesül-e vagy sem.

Például `\ifdim 1in<2cm(a)\else(b)\fi` eredménye (b).

`\ifmmode<igaz>\fi` Az eredmény *<igaz>*, ha matematikai módban vagyunk.

`\ifmmode<igaz>\else<hamis>\fi` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy matematikai módban vagyunk-e vagy sem. Például

`$2\ifmmode^4\else\textsuperscript5\fi$` eredménye 2^4 , illetve

`2\ifmmode^4\else\textsuperscript5\fi` eredménye 2^5 .

`\ifpdf<igaz>\fi` `\ifpdf` Az eredmény *<igaz>*, ha `pdflatex.exe`-vel fordítjuk a forrásfájlt.

`\ifpdf<igaz>\else<hamis>\fi` `\ifpdf` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy `pdflatex.exe`-vel fordítjuk a forrásfájlt vagy sem.

`\iflanguage{<nyelv>}{<igaz>}{<hamis>}` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<nyelv>* aktív-e vagy sem. Csak akkor működik, ha a `babel` vagy `polyglossia` csomagokkal betöltöttük a *<nyelv>*-et.

`\IfLanguagePatterns{<nyelv>}{<igaz>}{<hamis>}` `\IfLang` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<nyelv>* aktív-e vagy sem. Sajnos néhány esetben más csomagokkal összeakadhat (pl. a `siunitx` csomag esetén, ha rossz sorrendben töltjük be).

`\ifnum\pdf@strcmp{\language}{<nyelv>}=z@<igaz>\else<hamis>\fi` `\pdfifnum` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<nyelv>* aktív-e vagy sem.

`\bbl@iflanguage\language{<nyelv>}`

`\expandafter\ifx\csname date<nyelv>\endcsname\relax<hamis>\else<igaz>\fi`

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a `babel` vagy `polyglossia` csomagokkal betöltöttük-e a *<nyelv>*-et.

`\IfFileExists{<fájl>}{<igaz>}{<hamis>}` Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<fájl>* létezik-e vagy sem.

`\InputIfFileExists{<fájl>}{<igaz>}{<hamis>}` Ha a *<fájl>* létezik, akkor az eredmény *<igaz>*, majd beolvassa a *<fájl>*-t, különben az eredmény *<hamis>*.

Magunk is létrehozhatunk új feltételes utasítást:

`\newif\if<név>`

ennek alapértéke hamis, vagyis ezután

`\if<név><igaz>\else<hamis>\fi`

kifejtése *<hamis>*. Igaz értékre a következő módon állíthatja:

`\<név>true`

(Ez lokális hatású. Globális használatához tegye elé a `\global` parancsot.) Ezután

`\if<név><igaz>\else<hamis>\fi`

kifejtése *<igaz>*. Visszaállítása hamisra:

`\<név>>false`

(Ez lokális hatású. Globális használatához tegye elé a `\global` parancsot.) Például

```
\newif\ifproba
\ifproba(a)\else(b)\fi
\probatrue
\ifproba(c)\else(d)\fi
\probafalse
```

```
\ifproba(e)\else(f)\fi
```

```
(b)(c)(f)
```

A feltételek kezelését megkönnyíti az `ifthen` csomag használata.

`\ifthenelse{<feltétel>}{<igaz>}{<hamis>} ∈ ifthen` Az eredmény aszerint `<igaz>` vagy `<hamis>`, hogy a `<feltétel>` igaz vagy hamis.

`\isodd{<szám>} ∈ ifthen` értéke aszerint igaz vagy hamis, hogy a `<szám>` értéke páratlan vagy páros.

`\lengthtest{<hosszúság feltétel>} ∈ ifthen` értéke aszerint igaz vagy hamis, hogy a `<hosszúság feltétel>` igaz vagy hamis.

`\equal{<szöveg1>}{<szöveg2>} ∈ ifthen` értéke aszerint lesz igaz vagy hamis, hogy a `<szöveg1>` és `<szöveg2>` megegyezik-e vagy sem.

`\not <feltétel> ∈ ifthen` értéke aszerint igaz vagy hamis, hogy a `<feltétel>` hamis vagy igaz.

`<feltétel1>\and<feltétel2> ∈ ifthen` értéke csak akkor igaz, ha a `<feltétel1>` és `<feltétel2>` is igaz.

`<feltétel1>\or<feltétel2> ∈ ifthen` értéke csak akkor igaz, ha a `<feltétel1>` vagy `<feltétel2>` valamelyike igaz.

`\(<feltétel>\) ∈ ifthen` feltételek zárójelezése.

`\newboolean{<logikai kapcsoló>} ∈ ifthen` Új logikai kapcsoló definiálása. Alapértéke `false` (hamis).

`\setboolean{<logikai kapcsoló>}{<logikai érték>} ∈ ifthen` Beállítja a `<logikai kapcsoló>` értékét. A `<logikai érték>` lehet `false` (hamis) vagy `true` (igaz).

`\boolean{<logikai kapcsoló>} ∈ ifthen` A `<logikai kapcsoló>` értékét fejt ki.

Például

```
\newcounter{szam}
\setcounter{szam}{2}
\ifthenelse{\value{szam}<2}{(a)}{(b)}
\ifthenelse{\isodd{szam}}{(c)}{(d)}
```

```
(b)(d)
```

```
\ifthenelse{\lengthtest{\textwidth<\textheight}}{(a)}{(b)}
```

```
(a)
```

```
\newcounter{szam}
\setcounter{szam}{2}
\ifthenelse{
  \(&\not\value{szam}<2&\and\value{szam}<11&\)\or\isodd{\value{szam}}}{
  {$2\geq\mathtt{szam}<11$ vagy páratlan}
  {Nem igaz, hogy $2\geq\mathtt{szam}<11$ vagy páratlan}}
```

```
2 ≤ szam < 11 vagy páratlan
```

```
\newboolean{kapcsoló}
\ifthenelse{\boolean{kapcsoló}}{(a)}{(b)}
\setboolean{kapcsoló}{true}
```



```
\ifthenelse{\boolean{kapcsoló}}{(c)}{(d)}
\setboolean{kapcsoló}{false}
\ifthenelse{\boolean{kapcsoló}}{(e)}{(f)}
```

(b) (c) (f)

20.4.2. Esetszétválasztás

A következő parancs is feltételes utasítás, de esetek szétválasztására alkalmas:

```
\ifcase<egész szám><0>\or<1>\or<2>\or...\or<n>\fi
```

Ha az *<egész szám>* értéke például 2, akkor az eredmény *<2>*.

```
\ifcase<egész szám><0>\or<1>\or<2>\or...\or<n>\else<n+>\fi
```

Ha az *<egész szám>* értéke például 2, akkor az eredmény *<2>*, míg ha *n*-nél nagyobb, akkor *<n+>*. Például

```
\ifcase\value{page} nulla\or egy\or kettő\else sok\fi
```

sok

```
\newcounter{szam}\setcounter{szam}{7}
\ifcase\value{szam}\or
hétfő\or kedd\or szerda\or csütörtök\or péntek\or szombat\or vasárnap\fi
```

vasárnap

20.4.3. Ciklusok

A következő parancsok belső ciklus utasítások:

```
\@whilenum<numerikus feltétel>\do{<parancsok>}
```

Mindaddig végrehajtja a *<parancsok>*-at, amíg a *<numerikus feltétel>* fennáll. Például

```
\newcounter{szam}
\@whilenum\theszam<10\do{\stepcounter{szam}\theszam\ }
```

1 2 3 4 5 6 7 8 9 10

```
\@for\parancs:={<lista1>,<lista2>,...}\do{<parancsok>}
```

A listaelemeken végrehajtja a *<parancsok>*-at. Például

```
\@for\mitcsinal:={felkel,lenyugszik}\do{A Nap \mitcsinal. }
```

A Nap felkel. A Nap lenyugszik.

A `\@for` parancs dokumentumtestben való használata összeakad a `magyar.ldf` fájl `defaults=hu-min` opciójával. (Preambulumban nincs gond.) Egy lehetséges megoldás, ha a dokumentumtestben újra definiáljuk a `\@for` parancsot:

```
\makeatletter
\long\def\@for#1:=#2\do#3{%
\expandafter\def\expandafter\@fortmp\expandafter{#2}%
```

```
\ifx\@fortmp\@empty \else
\expandafter\@forloop#2,\@nil,\@nil\@@#1{#3}\fi}
\makeatother
```

A következő ciklus utasítások minden korlátozás nélkül használhatók:

```
\loop<parancsok1>\ifnum<numerikus feltétel><parancsok2>\repeat
```

Kifejti a $\langle parancsok1 \rangle$ kódot, majd megvizsgálja, hogy teljesül-e a $\langle numerikus feltétel \rangle$. Ha igen, akkor kifejti a $\langle parancsok2 \rangle$ kódot, majd kezdi az egészet előlről. Ha a $\langle numerikus feltétel \rangle$ nem teljesül, akkor a ciklus lezárul. Például

```
\newcounter{szam}
\loop\theszam\ifnum\value{szam}<9\stepcounter{szam}\repeat
```

```
0 1 2 3 4 5 6 7 8 9
```

```
\whiledo{<numerikus feltétel>}{<parancsok>} \in ifthen
```

Mindaddig végrehajtja a $\langle parancsok \rangle$ -at, amíg a $\langle numerikus feltétel \rangle$ fennáll. Például

```
\newcounter{szam}\setcounter{szam}{2}
\whiledo{\value{szam}<5}{\theszam\stepcounter{szam}}
```

```
234
```

20.5. Parancsok definiálása

A \LaTeX már meglévő parancsai mellé sajátokat is definiálhat a $\backslash newcommand$ paranccsal.

```
\newcommand{\<parancs>}{<kód>}
```

Ekkor $\backslash \langle parancs \rangle$ kifejtése $\langle kód \rangle$ lesz. Például

```
\newcommand{\EKE}{Eszterházy Károly Egyetem}
```

után

```
\EKE
```

```
Eszterházy Károly Egyetem
```

Paraméteres parancsok is definiálhatók:

```
\newcommand{\<parancs>}[<paraméterek száma>]{<kód>}
```

A $\langle paraméterek száma \rangle$ maximum 9 lehet. A $\langle kód \rangle$ -ban például a 2. paraméterre $\#2$ módon utalhat. Például

```
\newcommand{\ora}[2]{\#1\textsuperscript{\underline{\#2}}}
```

után

```
\ora{12}{45}
```

```
1245
```

Opcionális parancs is definiálható:

```
\newcommand{\<parancs>}[<paraméterek száma>][<alapérték>]{<kód>}
```

Ekkor az első paraméter opcióvá minősül, melynek alapértéke *alapérték*. Például

```
\newcommand{\ora}[2][12]{#1\textsuperscript{\underline{#2}}}
```

után

```
\ora{45}, \ora[13]{45}
```

12⁴⁵, 13⁴⁵

Korábban említettük, hogy maximum csak 9 paraméter lehet. Ha valamiért mégis többre van szükség, akkor a következő példában leírtak szerint járhat el:

```
\newcommand{\vektor}[9]{%
\newcommand{\koordA}{#1}\newcommand{\koordB}{#2}%
\newcommand{\koordC}{#3}\newcommand{\koordD}{#4}%
\newcommand{\koordE}{#5}\newcommand{\koordF}{#6}%
\newcommand{\koordG}{#7}\newcommand{\koordH}{#8}%
\newcommand{\koordI}{#9}\vektorfolyt}
\newcommand{\vektorfolyt}[2]{%
\newcommand{\koordJ}{#1}\newcommand{\koordK}{#2}%
$(\koordA,\koordB,\koordC,\koordD,\koordE,\koordF,
\koordG,\koordH,\koordI,\koordJ,\koordK)$}
```

Ezután a `\vektor` parancsnak $9 + 2 = 11$ paramétere lesz. Például

```
A \vektor{1}{2}{3}{4}{5}{6}{7}{8}{9}{10}{11} második koordinátája \koordB.
```

A (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11) második koordinátája 2.

Ha létező parancsot akarunk `\newcommand`-dal átdefiniálni, akkor a fordítás ezt jelzi és leáll. Ez azért hasznos, mert így elkerülhető a véletlen átdefiniálás.

Ha szándékosan akar átdefiniálni, akkor a `\renewcommand`-dal tegye, melyet pontosan úgy kell használni, mint a `\newcommand`-ot.

Ha csak abban az esetben akar egy parancsot definiálni, ha az még nem létezik, ugyanakkor, ha létezik, akkor nincs szándékában átdefiniálni, akkor használja a `\providecommand` parancsot, amit pontosan úgy kell használni, mint a `\newcommand`-ot.

Ezeknek a parancsoknak van egy csillagos verziója is: `\newcommand*`, `\renewcommand*`, `\providecommand*`. Ezeket akkor használja, ha a kifejtésük csak egy bekezdésből áll. A használatuk módja megegyezik a nem csillagos parancsokéval.

Egy parancs definíciójában lehet másik parancsot definiálni. Hogy a külső és belső definíciók argumentumaira való hivatkozások ne keveredjenek össze, az utóbbiakat `##1`, `##2`, stb. módon jelöljük. Például

```
\newcommand{\irogep}[1]{\renewcommand{\emph}[1]{\underline{##1}}}%
\texttt{#1}}
\irogep{Ki van \emph{emelve}.}
```

Ki van emelve.

A `\newcommand` és a `\renewcommand` hatása lokális. Ez azt jelenti, hogy a belső parancsnak blokkon kívül nincs hatása. Például az előbb definiált `\irogep` parancs esetén

```
{\irogep{Ki van \emph{emelve},} \emph{emelve},} \emph{emelve}.
```

Ki van emelve, emelve, *emelve*.

Új parancsokat a `\def` paranccsal is létrehozhat. Ennek hatása szintén lokális. Vigyázat, ez nem jelzi, ha létező parancsot definiál át.

```
\def\ora#1#2{#1\textsuperscript{\underline{#2}}}  
\ora{12}{45}
```

12⁴⁵

```
\def\ora(#1.#2){#1\textsuperscript{\underline{#2}}}  
\ora(12.45)
```

12⁴⁵

```
\def\FirstUppercase#1{\MakeUppercase#1}  
\FirstUppercase{tamási} \FirstUppercase{{á}ron}
```

Tamási Áron

A belső definíciókban itt is `##1`, `##2`, stb. hivatkozásokat használunk.

Ha `\def` helyett `\global\def`-et, vagy annak rövid verzióját, `\gdef`-et ír, akkor globális parancsot kap, azaz annak hatásköre blokkon kívül is megmarad.

A `\def` által definiált parancs argumentumában csak egy bekezdés állhat, úgy, mint a `\newcommand*` esetében. Ha ezt a korlátozást fel akarja oldani, akkor a `\def` elé írja be a `\long` parancsot is.

Ha azt akarja, hogy a parancs definíciójában szereplő makró a definiálás pillanatában érvényes értékek szerint fejtődjön ki, akkor `\edef`-et használjon. Például

```
\newcounter{szam}  
\def\szamkiir{\theszam}  
\edef\kiirszam{\theszam}  
\setcounter{szam}{1}  
\szamkiir  
\kiirszam
```

10

Az `\edef` lokális parancsokat definiál. Ennek globális megfelelője az `\xdef`.

Az `\edef` és `\xdef` parancsok definíciójában minden makró a definiáláskori értékek szerint fejtődik ki, kivéve azok, melyek előtt `\noexpand` áll. Például

```
\newcounter{szam}  
\edef\kiirszam{\noexpand\theszam\theszam}  
\setcounter{szam}{1}  
\kiirszam
```

10

Egy létező parancsot átdefinálás helyett, tovább is tudja bővíteni a `\g@addto@macro` belső paranccsal. Például

```
\def\EKE{Eszterházy Károly}  
\EKE\\  
\makeatletter
```

```
\g@addto@macro\EKE{ Egyetem}
\makeatother
\EKE
```

Eszterházy Károly
Eszterházy Károly Egyetem

Ha egy meglévő parancsnak szeretne másik nevet adni, akkor használja a `\let` parancsot. Például, ha `\section` helyett a `\szakasz` parancsot akarja használni, akkor ezt kell beírni:

```
\let\szakasz\section
```

A következő példában definiált `\ora` parancs második paramétere opcionális, melynek 00 az alapértéke:

```
\makeatletter
\def\@ora#1[#2]{#1\textsuperscript{#2}\ }
\def\ora#1{\@ifnextchar[{\@ora#1}{\@ora#1[00]}}
\makeatother
\ora{11}
\ora{11}[15]
```

11⁰⁰ 11¹⁵

A következő példában definiált `\eredmeny` parancs első paramétere opcionális, melynek alapértéke a második paraméter:

```
\makeatletter
\def\@eredmeny[#1]#2{\textbf{#1}\,:\,\textbf{#2}}
\def\@@eredmeny#1{\@eredmeny[#1]{#1}}
\def\eredmeny{\@ifnextchar[{\@eredmeny}{\@@eredmeny}}
\makeatother
\eredmeny{1}\
\eredmeny[5]{0}
```

1 : 1
5 : 0

A következő példában átdefiniáljuk a `\section` parancsot úgy, hogy a címet csupa nagybetűvel írja ki, de a tartalomjegyzékben változatlanul hagyja:

```
\makeatletter
\let\old@section\section
\def\@szakasz[#1]#2{\old@section[#1]{\MakeUppercase{#2}}}
\def\@@szakasz#1{\@szakasz[#1]{#1}}
\def\@@@szakasz#1{\@szakasz[#1]{#1}}
\def\section{\@ifnextchar[{\@szakasz}
{\@ifstar{\@@szakasz}{\@@@szakasz}}}
\makeatother
```

A következő példában `\alph{<számláló>}` mintájára definiáljuk a `\greekalph{<számláló>}` parancsot:

```
\newcommand{\greekalph}[1]{\ensuremath{%
\ifcase\value{#1}\or\alpha\or\beta\or\gamma\or\delta%
\or\varepsilon\or\zeta\or\eta\or\vartheta\or\iota\or\kappa%
\or\lambda\or\mu\or\nu\or\xi\or\varphi\or\varpi\or\varrho%
\or\varsigma\or\tau\or\upsilon\or\chi\or\psi\or\omega\fi}}
```

Ekkor például

```
\newcounter{szam}
\setcounter{szam}{4}
\greekalph{szam}
```

δ

A következő makrókkal nem nekünk kell az összeget kiszámolni:

```
\newcounter{szumma}
\def\szummatag#1{\addtocounter{szumma}{#1}#1}
\def\szumma{\theszumma\setcounter{szumma}{0}}
```

Ezután például

```
\szummatag{1234} és \szummatag{367} összege \theszumma.
```

1234 és 367 összege 1601.

vagy

```
\begin{tabular}{@{}r@{}r@{}}
&\szummatag{12345}\\
&\szummatag{1234}\\
+&\szummatag{123}\\
\hline
&\szumma
\end{tabular}
```

```
12345
 1234
+  123
-----
13702
```

A következőkben definiálunk egy `\ifdivisible{<szám1>}{<szám2>}{<igaz>}{<hamis>}` parancsot, amelynek az eredménye aszerint *igaz* vagy *hamis*, hogy a *szám1* osztható-e *szám2*-vel.

```
\newcounter{checknum}
\def\ifdivisible#1#2#3#4{%
\setcounter{checknum}{#1}%
\divide\value{checknum}by#2\relax%
\multiply\value{checknum}by#2\relax%
\ifnum\value{checknum}=#1\relax#3\else#4\fi}
```

Ekkor például

```
\ifdivisible{20449}{143}{Osztható!}{Nem osztható!}
```

Osztható!

A következőkben definiálunk egy `\lentounit{<hossz>}` parancsot, amelynek az eredménye a `<hossz>` mértéke centiméterben.

```
\newlength{\unit}
\setlength{\unit}{1cm}
\def\lentounit#1{\strip@pt\dimexpr#1*\p@/\unit}
```

Ekkor például

```
1\,inch = \lentounit{1in}\,cm
```

1 inch = 2.54 cm

Verbatim típusú parancs definiálása nem történhet az előzőek alapján, hiszen a `\verb` nem kerülhet más parancs argumentumába. Elemezze ki, hogy a következő kód miért hibás:

```
\newcommand{\kod}{kód: \color{red}\rmfamily\bfseries\verb*} % HIBÁS KÓD!
```

Ehelyett a `fancyvrb` csomag betöltése után használja a következő kódot:

```
\newcommand{\kod}{\SaveVerb[after save={%
kód: \UseVerb[format com={\color{red}\rmfamily\bfseries}, show spaces]{kod}%
}]{kod}}
```

Ezután a `\kod` parancs pontosan úgy használható, mint a `\verb`, csak csillagos verziója nincs. Például

```
\kod+$ $ \+
```

kód: \$ \$ \

Ha `beamer` osztályt használ prezentáció készítéséhez, akkor saját overlay specifikációval rendelkező parancsokat is definiálhat. Erre használhatja a `\newcommand<>` illetve `\renewcommand<>` parancsokat. Ezek pontosan úgy működnek, mint a `<>` jel nélküli verziók, csak ha a definícióban n darab paraméter van ($n = 0, 1, \dots, 9$), akkor az kibővül egy $n + 1$ -edikkel, melyben az overlay specifikáció adható meg. Például

```
\newcommand<>{\textblue}[1]{\textcolor#2{blue}{#1}}
```

vagy

```
\newcommand<>{\blue}{\color#1{blue}}
```

Az így definiált `\textblue` illetve `\blue` parancsok overlay specifikációjának alapértéke `<1->`. Használatuk:

```
\textblue<spec>{<szöveg>}
\blue<spec>{<szöveg>}
```

20.6. Környezetek definiálása

A `LATEX` már meglévő környezetei mellé sajátokat is definiálhat a `\newenvironment` paranccsal:

```
\newenvironment{<név>}[<argumentumszám>][<alapérték>]{<nyitódef>}{<végdef>}
```

$\langle név \rangle$ a környezet neve.

$\langle argumentumszám \rangle$ az argumentumok száma. Az n -edik argumentumra $\#n$ hivatkozik, ahol $n = 1, 2, \dots, 9$.

$\langle alapérték \rangle$ az első argumentum alapértéke. Ha ez adott, akkor az első argumentum opció lesz.

$\langle nyitódef \rangle$ a környezet megnyitásakor hajtódik végre.

$\langle végdef \rangle$ a környezet bezárásakor hajtódik végre. Ebben nem lehetnek $\#n$ hivatkozások, azaz a környezet argumentumai.

A $\langle név \rangle$ környezetet nem definiálhatja, ha már létezik ilyen környezet, vagy létezik $\backslash \langle név \rangle$ parancs. Viszont definiálható, ha már létezik $\langle név \rangle$ számláló. Ha egy már létező környezetet akar átdefiniálni, akkor azt a `\renewenvironment` paranccsal teheti meg. Ennek használata megegyezik a `\newenvironment` használatával.

Amikor a `\newenvironment` paranccsal definiál például egy $\langle név \rangle$ környezetet, akkor tulajdonképpen két parancsot definiál:

```
\langle név \rangle = \begin{\langle név \rangle}
\end{\langle név \rangle} = \end{\langle név \rangle}
```

Ez a magyarázata, hogy a $\langle végdef \rangle$ -ben miért nem lehet $\#n$ hivatkozás. Tekintsünk néhány példát.

```
\newenvironment{rend}[1][Napirend]
{\noindent\textbf{\#1}\[2mm]\begin{tabular}{|r|p{5cm}|}\hline}
{\hline\end{tabular}\par}
```

Ezután

```
\begin{rend}
6:30 & Kelés után reggeli torna, mosakodás és étkezés.\\
7:30 & Munkába indulás.\\
\dots & \dots
\end{rend}
```

Napirend

6:30	Kelés után reggeli torna, mosakodás és étkezés.
7:30	Munkába indulás.
...	...

illetve

```
\begin{rend}[Órater]
8:00 & Hetes jelentése, napló beírása.\\
8:02 & Házi feladat ellenőrzése, értékelése.\\
\dots & \dots
\end{rend}
```

Órater

8:00	Hetes jelentése, napló beírása.
8:02	Házi feladat ellenőrzése, értékelése.
...	...


```
\newenvironment{rend}[1]
{\noindent\textbf{#1}\[2mm]\begin{tabular}{|r|p{5cm}|}\hline}
{\hline\end{tabular}\par}
```

Ezután

```
\begin{rend}{Óraterv}
8:00 & Hetes jelentése, napló beírása.\\
8:02 & Házi feladat ellenőrzése, értékelése.\\
\dots & \dots
\end{rend}
```

Óraterv

8:00	Hetes jelentése, napló beírása.
8:02	Házi feladat ellenőrzése, értékelése.
...	...

Az előző módszerekkel verbatim típusú környezetet nem tud definiálni, mert az ehhez szükséges parancsok nem tehetők más parancs argumentumába. Példaként elemezze, hogy a következő kód, a `fancyvrb` csomag betöltése után, miért nem működhet:

```
\newenvironment{frameverb}
{\begin{Verbatim}[frame=single]}\end{Verbatim}} % ROSSZ KÓD!
```

Az ilyen jellegű problémákra ad megoldást a `\DefineVerbatimEnvironment` \in `fancyvrb` parancs. Használatára példaként itt van az előző kód helyes változata:

```
\DefineVerbatimEnvironment{frameverb}{Verbatim}{frame=single}
```

Verbatim környezetbe Verbatim környezet nem ágyazható be, de az előbb definiált `frameverb` környezet már beágyazható Verbatim környezetbe, vagy fordítva.

Programkód környezet definiálásához használja a `\lstnewenvironment` \in `listings` parancsot, melynek használata megegyezik a `\newenvironment` használatával. Például:

```
\lstnewenvironment{delphi}[1] []
{\lstset{language=Delphi,numbers=left,numberstyle=\tiny,#1}}
{}
```

Tegyük fel, hogy definiált egy `megjegyzes` tételszerű környezetet. Ha a dokumentumnak egy olyan verzióját akarja előállítani, amelyből a megjegyzések hiányoznak, akkor azt kell megoldani, hogy a `megjegyzes` környezet úgy viselkedjen, mint a `comment` csomag `comment` környezete, azaz, hogy ennek a környezetnek a tartalmát a \LaTeX -fordító figyelmen kívül hagyja. Ezt tudja elérni az `\excludecomment` \in `comment` paranccsal, a következő módon:

```
\renewenvironment{megjegyzes}{}{}
\excludecomment{megjegyzes}
```

Tegyük fel, hogy az előző példában a `megjegyzes` környezet együtt számozódik egy `tétel` tételszerű környezettel. Ekkor a dokumentum eredeti verziójában és az előző kóddal ellátott verzióban a tételek számozása nem fog megegyezni, hiszen a kiiktatott megjegyzéseknél a sorszám nem emelkedik. Ha ezt nem szeretné, akkor a megoldás a `\processcomment` \in `comment` parancs:

```
\renewenvironment{megjegyzes}{}{}
\processcomment{megjegyzes}{\stepcounter{tetel}\def\ThisComment##1{}{}{}}
```

Ha ezen felül még a megjegyzések helyét például egy — jellel akarja megjelölni, akkor ez a megoldás:

```
\renewenvironment{megjegyzes}{}{}
\processcomment{megjegyzes}
{\stepcounter{tetel}\def\ThisComment##1{}{}{---}}{}
```

A beamer dokumentumosztályban saját overlay specifikációval rendelkező környezeteket is definiálhat. Erre használhatja a `\newenvironment<>` illetve `\renewenvironment<>` parancsokat. Ezek pontosan úgy működnek, mint a `<>` jel nélküli verziók, csak ha a definícióban n darab paraméter van ($n = 0, 1, \dots, 9$), akkor az kibővül egy $n+1$ -edikkel, melyben az overlay specifikáció adható meg. Például

```
\newenvironment<>{boldornormal}
{\begin{altnv}#1{\begin{bfseries}}{\end{bfseries}}{}{}}
{\end{altnv}}
```

Az így definiált `boldornormal` környezet overlay specifikáció alapértéke `<1->`. Használata:

```
\begin{boldornormal}<spec>
<szöveg>
\end{boldornormal}
```

20.7. Környezet horgonyok

```
\BeforeBeginEnvironment{<környezet>}{<kód>} ∈ etoolbox
\AtBeginEnvironment{<környezet>}{<kód>} ∈ etoolbox
\AtEndEnvironment{<környezet>}{<kód>} ∈ etoolbox
\AfterEndEnvironment{<környezet>}{<kód>} ∈ etoolbox
```

Ezeknek a működése érthetővé válik, ha megvizsgálja a következő példát:

```
...
\BeforeBeginEnvironment{quote}{<kód1>}
\AtBeginEnvironment{quote}{<kód2>}
\AtEndEnvironment{quote}{<kód3>}
\AfterEndEnvironment{quote}{<kód4>}
...
\begin{document}
...
\begin{quote}
<kód5>
\end{quote}
...
\begin{quote}
<kód6>
\end{quote}
...
\end{document}
```

Ennek eredménye ugyanaz lesz, mintha a következőt használta volna:

```
...
...
\begin{document}
...
\langle kódd1 \rangle { \langle kódd2 \rangle \begin{quote}
\langle kódd5 \rangle \langle kódd3 \rangle
\end{quote} } \langle kódd4 \rangle
...
\langle kódd1 \rangle { \langle kódd2 \rangle \begin{quote}
\langle kódd6 \rangle \langle kódd3 \rangle
\end{quote} } \langle kódd4 \rangle
...
\end{document}
```

A document környezetnek is vannak horgonyai:

```
\AtEndPreamble{ \langle kód \rangle } \in etoolbox
\AfterEndPreamble{ \langle kód \rangle } \in etoolbox
\AtBeginDocument{ \langle kód \rangle }
\AtEndDocument{ \langle kód \rangle }
```

Ezeknek a működése érthetővé válik, ha megvizsgálja a következő példát:

```
...
\AtEndPreamble{ \langle kódd1 \rangle }
\AfterEndPreamble{ \langle kódd2 \rangle }
\AtBeginDocument{ \langle kódd3 \rangle }
\AtEndDocument{ \langle kódd4 \rangle }
\langle kódd5 \rangle
\begin{document}
\langle kódd6 \rangle
\end{document}
```

Ennek eredménye ugyanaz lesz, mintha a következőt használta volna:

```
...
\langle kódd5 \rangle
\langle kódd1 \rangle
\begin{document}
\langle kódd3 \rangle \langle kódd2 \rangle \langle kódd6 \rangle \langle kódd4 \rangle
\end{document}
```

21. fejezet

Stílusfájlok írása

Saját dokumentumosztályokat és csomagokat is összeállíthat. Akkor készítsen csomagot, ha az több dokumentumosztállyal is működik. Ellenkező esetben dokumentumosztályt írjon. Arra is lehetőség van, hogy ezeket a fájlokat a hivatalos T_EX-disztribúciók részévé tegye. Erre vonatkozólag itt talál információt: [klikk ide](#). A beadás itt lehetséges: [klikk ide](#).

21.1. Csomag készítése

Csomag írásánál a következőket vegye figyelembe:

- A csomag forrásfájlja legyen `sty` kiterjesztésű, és rakja a `tex` kiterjesztésű főfájl könyvtárába.
- A csomag forrásfájlja csak ascii karaktereket tartalmazzon, így az ékezetes betűket repülő ékezetekkel gépelje be. Ez azért kell, hogy bármilyen kódolású főfájlba be lehessen tölteni a `\usepackage` paranccsal.
- A csomag forrásfájljába minden olyan parancs írható, amely a főfájl preambulumban szerepelhet, egyedül a `\usepackage` helyett használjon `\RequirePackage` parancsot.
- A belső parancsok csomagban a `\makeatletter` és `\makeatother` parancsok nélkül is működnek.

Egy `sty` kiterjesztésű fájl szerkezete a következő:

```
\NeedsTeXFormat{LaTeX2e}[\langle dátum1 \rangle]
\ProvidesPackage{\langle csomagnév \rangle}[\langle dátum2 \rangle \langle verzió \rangle \langle leírás \rangle]
% tartalom
\endinput
```

A `\langle dátum1 \rangle` a csomag használatához szükséges L^AT_EX verzió dátuma `éééé/hh/nn` formátumban, pl. 1999/12/01. A `\langle csomagnév \rangle` az `sty` kiterjesztésű fájl nevével egyezik meg. Ebben lehetőleg csak az angol ábécé betűit és számokat használjon. Tehát, ha pl. a fájl neve `sajat.sty`, akkor a `\langle csomagnév \rangle` helyére `sajat` kerül. A `\langle dátum2 \rangle` a csomag publikálásának dátuma ugyanolyan formátumban, mint az előbb. A `\langle verzió \rangle` a csomag verziószáma, pl. `v1.0`. A `\langle leírás \rangle` a csomag céljának pár szavas leírása parancsok használata nélkül, ascii karakterekkel.

Csomagnak opciókat is adhat. Legyen például a `sajat.sty` tartalma:

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{sajat}[2016/09/28 v1.0 Ez a csomag csak egy pelda]
\DeclareOption{<opció1>}{<kód1>}
\DeclareOption{<opció2>}{<kód2>}
\ExecuteOptions{<opció1>}
\ProcessOptions\relax
\endinput
```

Ekkor a `sajat` csomagnak két opciója lesz: `<opció1>` (alapopció) és `<opció2>`. Ha az `<opció2>` opciót használja, akkor a `<kód2>` kód lesz érvényben. Viszont akár kiadja akár nem az `<opció1>` opciót, a `<kód1>` mindenképpen érvényben lesz. Ha az `\ExecuteOptions` parancsban több opciót is megad alapopcióként, akkor azokat vesszővel kell elválasztani:

```
\ExecuteOptions{<opció1>,<opció2>,...}
```

Egy opcióinak értéket (számláló, hossz, sztring, logikai érték) is adhat. Például a `geometry` és a `hyperref` csomagokban is vannak ilyen opciók:

```
\usepackage[width=150mm]{geometry}
\usepackage[linktocpage=false,linkcolor=blue]{hyperref}
```

Ha a saját csomagjában is szeretne ilyen opciókat, akkor használja a `kvoptions` csomagot. Legyen például a `sajat.sty` tartalma:

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{sajat}[2016/09/28 v1.0 Ez a csomag csak egy pelda]
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=sajat,prefix=sajat@}
\DeclareVoidOption{<opció1>}{<kód1>}
\DeclareBoolOption[true]{<opció2>}
\DeclareComplementaryOption{<opció3>}{<opció2>}
\DeclareStringOption[<kód4>]{<opció4>}
\ProcessKeyvalOptions{sajat}
\ifsajat@<opció2> <kód2-igaz>\else<kód2-hamis>\fi
\endinput
```

Ez a kód az alábbiak szerint működik:

opció	hatása
<code><opció1></code>	<code><kód1></code>
<code><opció2></code> , <code><opció2>=true</code> , <code><opció3>=false</code> (alapopció)	<code><kód2-igaz></code>
<code><opció3></code> , <code><opció3>=true</code> , <code><opció2>=false</code>	<code><kód2-hamis></code>
<code><opció4>=<kód5></code> (alapopció <code><opció4>=<kód4></code>)	<code>\def\sajat@<opció4>{<kód5>}</code>

Ezután, ha a dokumentumban alapopciókkal tölti be a `sajat.sty` csomagot

```
\usepackage{sajat}
```

akkor az alapopciók által kifejtett kódok érvényesülnek:

- `<kód2-igaz>`
- `\sajat@<opció4>` kifejtése `<kód4>`.

Ha például így tölti be

```
\usepackage[<opció1>,<opció2>=false,<opció4>=<kód5>]{sajat}
```

akkor a következő kódok érvényesülnek:

- `<kód1>`

- `<kód2-hamis>`
- `\sajat@<opció4>` kifejtése `<kód5>`.

A `sajat` csomag bármelyik opciója parancsban is aktiválható. Például

```
\setkeys{sajat}{<opció2>=false,<opció1>}
```

Egy csomag opciója öröközhető a saját csomagunkra is a

```
\PassOptionsToPackage{<opció>}{<csomag>}
```

paranccsal. Legyen például a `sajat.sty` tartalma:

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{sajat}[2016/09/28 v1.0 Ez a csomag csak egy pelda]
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=sajat,prefix=sajat@}
\DeclareVoidOption{unicode}{\PassOptionsToPackage{unicode}{hyperref}}
\DeclareBoolOption{colorlinks}
\DeclareStringOption{urlcolor}
\ProcessKeyvalOptions{sajat}
\ifsajat@colorlinks\PassOptionsToPackage{colorlinks}{hyperref}\fi
\PassOptionsToPackage{urlcolor=\sajat@urlcolor}{hyperref}
\RequirePackage[bookmarksopen]{hyperref}
\endinput
```

Ekkor a `sajat` csomag opciójaként használható a `hyperref` csomagnak a `unicode`, `colorlinks` és `urlcolor` opciói.

Lehetőség van arra, hogy adott esetben a fordításnál valamilyen figyelmeztetést küldjön a felhasználónak:

```
\PackageWarning{<csomagnév>}{<figyelmeztetés>}
```

Azt is megteheti, hogy adott esetben a fordítás leálljon egy hibaüzenettel:

```
\@latexerr{<hibaüzenet>}{<segítség>}
```

Az üzenetek szövegében a `\MessageBreak` paranccsal tud sort törni.

21.2. Dokumentumosztály készítése

Dokumentumosztály készítésénél ugyanaz az eljárás, mint a csomagnál, néhány kivétellel:

- A dokumentumosztály forrásfájljának kiterjesztése `cls`.
- A `\ProvidesPackage` helyett a `\ProvidesClass` parancsot kell használni.
- Nem a `\PackageWarning`, hanem a `\ClassWarning{<osztálynév>}{<üzenet>}` paranccsal kell figyelmeztetést generálni.

Célszerű egy létező dokumentumosztályt betölteni alapnak a `\LoadClass` paranccsal. Nézzük a következő példát. Legyen a `sajat.cls` tartalma:

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesClass{sajat}[2016/09/28 v1.0 Ez az osztály csak egy pelda]
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=sajat,prefix=sajat@}
```

```

\DeclareBoolOption[true]{<opció>}
\ProcessKeyvalOptions{sajat}
\ifsajat@<opció> <kód-igaz>\else<kód-hamis>\fi

\LoadClass[12pt,a4paper]{article}
\RequirePackage[utf8]{inputenc}
\RequirePackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\RequirePackage[magyar]{babel}
\endinput

```

Ezután, ha a dokumentumban például a következőképpen tölti be a `sajat.cls` osztályfájlt

```

\documentclass[<opció>=false]{sajat}

```

akkor egy 12 pt-os alap betűméretű, A4-es oldalméretű, magyar tipográfiájú dokumentumot kap az `article` osztálynak megfelelően, amelyben a `<kód-hamis>` fejtődik ki.

Egy dokumentumosztály opciója örökíthető a saját dokumentumosztályunkra is. Ezt hasonlóan lehet, mint a csomagok esetében, csak ekkor `\PassOptionsToPackage` helyett `\PassOptionsToClass` parancsot kell használni. Legyen például a `sajat.cls` tartalma:

```

\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesClass{sajat}[2016/09/28 v1.0 Ez az osztaly csak egy pelda]
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=sajat,prefix=sajat@}
\DeclareVoidOption{11pt}{\PassOptionsToClass{11pt}{article}}
\DeclareVoidOption{12pt}{\PassOptionsToClass{12pt}{article}}
\ProcessKeyvalOptions{sajat}
\LoadClass[a4paper]{article}
\endinput

```

Ekkor a `sajat` dokumentumosztály opciójaként használható az `article` dokumentumosztály 11pt és 12pt opciói.

22. fejezet

Fontok kiválasztása

A betűváltozatok osztályozásáról a 4.4.1. alszakaszban volt szó. Most azt vizsgáljuk, hogy az alapbeállításoktól eltérő fontokat hogyan választhatjuk ki. Ebben segítségére lehet még a [The L^AT_EX Font Catalogue](#) internetes oldal és az [fntguide.pdf](#) dokumentáció is.

22.1. L^AT_EX fontkatalógus

Ebben a szakaszban összefoglaljuk a T_EX-rendszerekben installált latin fontcsaládokat és a használatukhoz szükséges kódokat. Ezen kódok megértéséhez szüksége lesz ezen fejezet további szakaszainak tanulmányozására is.

1. [Antikva](#)
2. [Groteszk](#)
3. [Írógép](#)
4. [Egyéb](#)

22.2. A forrásfájl fontkódolása és a L^AT_EX belső kódkészlete

A forrásfájlban található ASCII karaktereknek (lásd a 20.1. szakaszban) minden fontkódolás esetén ugyanaz a kódszámuk. Például az O karakter ASCII és UTF-8 kódja is 79. A nem ASCII karakterek egy jó részét az `inputenc` csomag (a forrásfájl kódolásának megfelelő opcióval) – illetve UTF-8 kódolás esetén 2018-tól a L^AT_EX már e nélkül is – parancs alakra konvertálja. Például az Ő karakter helyére berakja a `\H{O}` parancsot. Ha egy nem ASCII karakternek nincs parancs megfelelője, akkor a fordítás hibával leáll.

Alapesetben a L^AT_EX a pdf-ben antikva, normál vastagságú, álló, 10 pt nagyságú fontokat jelenít meg, melyhez a `cmr10` nevű fontkészletet használja. A fontkészletekben minden karakternek van egy kódszáma. Ha egy ASCII karaktert kell beilleszteni a pdf-be, akkor az ASCII kódnak megfelelő kódú karaktert választja a fontkészletből. Tehát például az O betű helyére – aminek az ASCII kódja 79 – a `cmr10` fontkészletbeli 79 kódú O betűt illeszti. Azonban nem minden esetben felel meg a `cmr10` fontkészlet kódolása az ASCII-nek. Például a < karakter ASCII kódja 60, ugyanakkor a `cmr10` fontkészlet 60 kódú karaktere a j, ami meglepő eredményhez vezet:

```
■ \documentclass{article}
```



```
\usepackage[utf8]{inputenc}
\begin{document}
<O
\end{document}
```

ıO

A \LaTeX úgynevezett belső kódkészlete fogja azt meghatározni, hogy egy nem ASCII karakternek megfelelő parancsnak a pdf-ben a fontkészlet melyik kódszámú karaktere feleljen meg. Alapesetben a \LaTeX az OT1 jelű belső kódkészletet használja, amely olyan fontkészletekhez lett kitalálva, amelyek nem tartalmaznak ékezetes karaktereket. A **cmr10** is ilyen. Így ékezetes betűk esetén nem egy fontkészletbeli elem lesz hozzárendelve, hanem kettő: az alapbetű és az ékezet külön. Például az Ő betű helyére beillesztett `\H{0}` parancs azt fogja jelenteni az OT1 belső kódkészlet szerint, hogy a **cmr10** fontkészletben 125 decimális számmal kódolt " karaktert tegye a 79 decimális számmal kódolt O karakterre. Az eredmény: Ő.

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\begin{document}
Ő
\end{document}
```

Ő

Az ékezetes karakterek két karakterből történő összerakása a pdf-ben néhány gondot okoz:

- Ékezetes betűket tartalmazó szótagok után nem tud elválasztani a sor végén.
- Az elkészült pdf-ben nem lehet rákeresni ékezetes betűket tartalmazó szavakra.
- Ha a pdf fájból ékezetes betűket tartalmazó szöveget másol ki, akkor az ékezetes betűk rosszul fognak megjelenni.

A megoldás az, hogy a \LaTeX -ben telepített fontkészletek közül olyat kell használni, amelyben vannak ékezetes karakterek. Ilyen például az **ecrm1000**. Ebben 142 kóddal az Ő karakter található. Ráadásul ebben a fontkészletben a kódszámok összhangban vannak az ASCII kódolással, így például a < karakter sem fog rosszul megjelenni. Még azt kell megoldani, hogy az Ő karakterből keletkező `\H{0}` parancs ne azt a metódust kövesse, mint az OT1 belső kódkészlettel, hanem a 142 kódú karaktert rendelje hozzá. Ezt csinálja a T1 jelű belső kódkészlet. Erre áttérni a **fontenc** csomag T1 opciójával lehet. A \LaTeX úgy van beállítva, hogy T1 belső kódkészletre áttérve, alapból az **ecrm1000** fontkészletet használja. Így a megoldáshoz elég a következő:

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\begin{document}
Ő<
\end{document}
```

Ő<

Általánosan a következő kóddal állíthatjuk be a belső kódkészletet a preambulumban:

`\usepackage[<kódolás>]{fontenc}`

A *<kódolás>* alapértéke **OT1**. Egyszerre több kódolás is beírható. Ilyenkor ezeket vesszővel kell elválasztani, és az utolsó lesz az alapértelmezett.

22.3. Globális beállítás

A pdf-ben használt fontkészlet kiválasztásához öt információra van szükség:

- Belső kódolás kódja (alapérték: **OT1**).
- Család kódja (alapérték: **cmr**).
- Testesség kódja (alapérték: **m**).
- Alak kódja (alapérték: **n**).
- Betűméret (alapérték: **10pt**).

Ezután – hacsak a forráskódban nincs erre más irányú utasítás – a \LaTeX -fordító a belső kódolás és a családkód alapján betölt egy **fd** (font definition) kiterjesztésű fájlt. Alapesetben **OT1** a belső kódolás és **cmr** a családkód, így az **ot1cmr.fd** fájlt tölti be. Ezután lesz szükség a testesség- és alakkódra, illetve a betűméretre, melyek alapesetben **m**, **n** és **10pt**. Az **ot1cmr.fd** fájlban ezekre vonatkozóan azt az utasítást fogja találni a \LaTeX -fordító, hogy a **cmr10** nevű fontkészletet használja.

Ha áttérünk **T1** belső kódolásra, akkor az **ot1cmr.fd** helyett a **t1cmr.fd** fájlt tölti be, melyben az **m**, **n** és **10pt** értékekhez az **ecrm1000** fontkészlet van társítva.

22.3.1. Család

`\renewcommand{\rmdefault}{<család>}`

Alapértelmezett antikva család. Az `\rmfamily` és `\textrm` ezt a családot tölti be.

A *<család>* alapértéke **cmr** (Computer Modern Roman).

`\renewcommand{\sfdefault}{<család>}`

Alapértelmezett groteszk család. Az `\sffamily` és `\textsf` ezt a családot tölti be. A *<család>* alapértéke **cmss** (Computer Modern Sans Serif).

`\renewcommand{\ttdefault}{<család>}`

Alapértelmezett írógép család. Az `\ttfamily` és `\texttt` ezt a családot tölti be.

A *<család>* alapértéke **cmtt** (Computer Modern Typewriter).

`\renewcommand{\familydefault}{<családparancs>}`

Alapértelmezett család. A `\normalfont` és `\textnormal` ezt a családot tölti be. A *<családparancs>* alapértéke **rmdefault** (antikva). Lehet még **sfdefault** (groteszk) és **ttdefault** (írógép).

A *<család>* további lehetséges értékeit megtalálja a [22.1.](#) szakaszban.

22.3.2. Testesség

`\renewcommand{\mddefault}{<testesség>}`

Alapértelmezett normál testesség. Az `\mdseries` és `\textmd` ezt a testességet tölti be. A *<testesség>* alapértéke **m** (normál).

`\renewcommand{\bfdefault}{<testesség>}`

Alapértelmezett félkövér testesség. A `\bfseries` és `\textbf` ezt a testességet tölti be. A *<testesség>* alapértéke **bx** (félkövér).

```
\renewcommand{\seriesdefault}{\testességparancs}
```

Alapértelmezett testesség. A `\normalfont` és `\textnormal` ezt a testességet tölti be. A `\testességparancs` alapértéke `\mddefault` (normál). Lehet még `\bfdefault` (félkövér).

A `\testesség` további lehetséges értékeit megtalálja a 22.1. szakaszban.

22.3.3. Alak

```
\renewcommand{\updefault}{\alak}
```

Alapértelmezett álló alak. Az `\upshape` és `\textup` ezt az alakot tölti be. Az `\alak` alapértéke `n` (álló).

```
\renewcommand{\sldefault}{\alak}
```

Alapértelmezett döntött alak. Az `\slshape` és `\textsl` ezt az alakot tölti be. Az `\alak` alapértéke `sl` (döntött).

```
\renewcommand{\itdefault}{\alak}
```

Alapértelmezett dőlt alak. Az `\itshape` és `\textit` ezt az alakot tölti be. Az `\alak` alapértéke `it` (dőlt).

```
\renewcommand{\scdefault}{\alak}
```

Alapértelmezett kiskapitális alak. Az `\scshape` és `\textsc` ezt az alakot tölti be. Az `\alak` alapértéke `sc` (kiskapitális).

```
\renewcommand{\shapedefault}{\alakparancs}
```

Ez az alapértelmezett alak. A `\normalfont` és `\textnormal` ezt az alakot tölti be. Az `\alakparancs` alapértéke `\updefault` (álló). Lehet még `\sldefault` (döntött), `\itdefault` (dőlt) és `\scdefault` (kiskapitális).

Az `\alak` további lehetséges értékeit megtalálja a 22.1. szakaszban.

22.4. Lokális beállítás

A következő parancsokkal egy adott helyen ideiglenesen áttérhetünk az alapbeállítástól különböző fontokra is.

```
\fontencoding{\kódolás}
\fontfamily{\család}
\fontseries{\testesség}
\fontshape{\alak}
\selectfont
```

Ezután az adott paraméterekkel töltődik be a kódolás, család, testesség és alak. A kódolást a `fontenc` csomag opciójában is be kell tölteni, kivéve, ha a kódjel OT1, T1 vagy U. Az előző öt parancs egyben is megadható:

```
\usefont{\kódolás}{\család}{\testesség}{\alak}
```

Például

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
```

```
\begin{document}
{\usefont{T1}{qzc}{m}{it},,Lenni vagy nem lenni: az itt a kérdés\dots'}
(William Shakespeare)
\end{document}
```

„Lenni vagy nem lenni: az itt a kérdés...” (William Shakespeare)

22.5. Fontcsaládnév deklarációja

22.5.1. Több fontcsalád összevonása új néven

Akár több fontcsalád is összevonható egy új családnév alatt, amivel nagyon rugalmasá tehetjük a fontok kezelését. Egy új fontcsaládnév a következő paranccsal deklarállható:

```
\DeclareFontFamily{<kódolás>}{<új családnév>}{}
```

Ezután az *<új családnév>* alatt egy adott testességhez és alakhoz a következő módon rendelhetünk egy korábban már definiált fontcsaládnevet, testességet és alakot:

```
\DeclareFontShape{<kódolás>}{<új családnév>}{<új testességnév>}{<új alaknév>}
{<->ssub*<család>/<testesség>/<alak>}}{}
```

Fontos, hogy a *<kódolás>* ugyanaz legyen, mint a *<család>*-hoz tartozó belső kódkészlet. Például

```
\DeclareFontFamily{T1}{myroman}{}
\DeclareFontShape{T1}{myroman}{m}{n}{<->ssub*clm/m/n}{}
\DeclareFontShape{T1}{myroman}{m}{sl}{<->ssub*clm/m/sl}{}

```

Ezután

```
\usefont{T1}{myroman}{m}{n} szöveg
\usefont{T1}{myroman}{m}{sl} szöveg
```

és

```
\usefont{T1}{clm}{m}{n} szöveg
\usefont{T1}{clm}{m}{sl} szöveg
```

kódok ugyanazt eredményezik:

szöveg szöveg

Bemutatunk egy összetettebb példát is:

```
% A következőkben deklarált fontcsaládnevekhez
% clm, fcm stb. családokat rendelünk, melyek T1 belső kódolásúak:
\usepackage[T1]{fontenc}

% Deklarálunk egy T1 belső kódolású 'myrm' fontcsaládnevet:
\DeclareFontFamily{T1}{myrm}{}

% A 'myrm' fontcsaládhoz m és b kódú testességet,
% illetve n, sl, it, sc kódú alakokat rendelünk:
\DeclareFontShape{T1}{myrm}{m}{n}{<->ssub*clm/m/n}{}
\DeclareFontShape{T1}{myrm}{m}{sl}{<->ssub*clm/m/sl}{}
\DeclareFontShape{T1}{myrm}{m}{it}{<->ssub*clm/m/it}{}
\DeclareFontShape{T1}{myrm}{m}{sc}{<->ssub*clm/m/sc}{}

```

```

\DeclareFontShape{T1}{myrm}{b}{n} {<->ssub*clm/bx/n}{}
\DeclareFontShape{T1}{myrm}{b}{sl}{<->ssub*clm/bx/sl}{}
\DeclareFontShape{T1}{myrm}{b}{it}{<->ssub*clm/bx/it}{}
\DeclareFontShape{T1}{myrm}{b}{sc}{<->ssub*fcm/b/sc}{}

% Deklarálunk egy T1 belső kódolású 'mysf' fontcsaládnevet:
\DeclareFontFamily{T1}{mysf}{}
% A 'mysf' fontcsaládhoz m és b kódú testességet,
% illetve n, sl, it, sc kódú alakokat rendelünk:
\DeclareFontShape{T1}{mysf}{m}{n} {<->ssub*FiraSans-TLF/1/n}{}
\DeclareFontShape{T1}{mysf}{m}{sl}{<->ssub*jkpsos/m/sl}{}
\DeclareFontShape{T1}{mysf}{m}{it}{<->ssub*FiraSans-TLF/1/it}{}
\DeclareFontShape{T1}{mysf}{m}{sc}{<->ssub*FiraSans-TLF/1/sc}{}
\DeclareFontShape{T1}{mysf}{b}{n} {<->ssub*FiraSans-TLF/mb/n}{}
\DeclareFontShape{T1}{mysf}{b}{sl}{<->ssub*jkpsos/bx/sl}{}
\DeclareFontShape{T1}{mysf}{b}{it}{<->ssub*FiraSans-TLF/mb/it}{}
\DeclareFontShape{T1}{mysf}{b}{sc}{<->ssub*FiraSans-TLF/mb/sc}{}

% Deklarálunk egy T1 belső kódolású 'mytt' fontcsaládnevet:
\DeclareFontFamily{T1}{mytt}{}
% A 'mytt' fontcsaládhoz m és b kódú testességet,
% illetve n, sl, it, sc kódú alakokat rendelünk:
\DeclareFontShape{T1}{mytt}{m}{n} {<->ssub*hfott/m/n}{}
\DeclareFontShape{T1}{mytt}{m}{sl}{<->ssub*hfott/m/sl}{}
\DeclareFontShape{T1}{mytt}{m}{it}{<->ssub*hfott/m/it}{}
\DeclareFontShape{T1}{mytt}{m}{sc}{<->ssub*hfott/m/sc}{}
\DeclareFontShape{T1}{mytt}{b}{n} {<->ssub*clmt/bx/n}{}
\DeclareFontShape{T1}{mytt}{b}{sl}{<->ssub*clmt/bx/sl}{}
\DeclareFontShape{T1}{mytt}{b}{it}{<->ssub*clmt/bx/sl}{}
\DeclareFontShape{T1}{mytt}{b}{sc}{<->ssub*clmt/bx/n}{}

% Alapértelmezett antikva betűcsalád: myrm
\renewcommand{\rmdefault}{myrm}
% Alapértelmezett groteszk betűcsalád: mysf
\renewcommand{\sfdefault}{mysf}
% Alapértelmezett írógép betűcsalád: mytt
\renewcommand{\ttdefault}{mytt}
% Alapértelmezett betűcsalád: \rmdefault (antikva)
\renewcommand{\familydefault}{\rmdefault}
% Normál testesség alapértelmezett kódja: m
\renewcommand{\mddefault}{m}
% Félkövér testesség alapértelmezett kódja: b
\renewcommand{\bfdefault}{b}
% Alapértelmezett testesség: \mddefault (normál)
\renewcommand{\seriesdefault}{\mddefault}
% Álló alak alapértelmezett kódja: n
\renewcommand{\updefault}{n}
% Döntött alak alapértelmezett kódja: sl
\renewcommand{\sldefault}{sl}
% Dőlt alak alapértelmezett kódja: it
\renewcommand{\itdefault}{it}

```

```
% Kiskapitális alak alapértelmezett kódja: sc
\renewcommand{\scdefault}{sc}
% Alapértelmezett alak: \updefault (álló)
\renewcommand{\shapedefault}{\updefault}
```

Ezután a begépelt szöveg alapértelmezetten antikva, normál vastagságú és álló alakú. Így a következő sorok ugyanazt eredményezik:

```
szöveg
{\normalfont szöveg}
{\rmfamily\mdseries\upshape szöveg}
{\usefont{T1}{myroman}{m}{n}szöveg}
{\usefont{T1}{clm}{m}{n}szöveg}
```

szöveg szöveg szöveg szöveg szöveg

Hasonlóan ugyanazt eredményezik a következő sorok is:

```
{\ttfamily\bfseries\slshape szöveg}
{\usefont{T1}{mytype}{b}{sl}szöveg}
{\usefont{T1}{clmt}{bx}{sl}szöveg}
```

szöveg szöveg szöveg

22.5.2. Új fontcsaládnév deklarációja

Az előző alszakasznak a példája azt mutatta meg, hogyan lehet új családnevet létrehozni korábban definiált családnevek segítségével. De ezeket a családneveket hogyan definiálták? Ennek illusztrálására nézzük meg, hogy T1 belső kódkészlettel az lmr családnév hogyan van definiálva m testességekód és n alakkód esetén (lásd a t1lmr.fd fájlban):

```
\DeclareFontFamily{T1}{lmr}{}
\DeclareFontShape{T1}{lmr}{m}{n}{%
  <-5.5> ec-lmr5 <5.5-6.5> ec-lmr6
  <6.5-7.5> ec-lmr7 <7.5-8.5> ec-lmr8
  <8.5-9.5> ec-lmr9 <9.5-11> ec-lmr10
  <11-15> ec-lmr12 <15-> ec-lmr17}{}
```

Eszerint, ha az lmr család aktív m testességekóddal és n alakkóddal, akkor 5.5pt betűméret alatt az ec-lmr5, 5.5pt és 6.5pt közötti betűméret esetén az ec-lmr6, és így tovább, 15pt betűméret fölött az ec-lmr17 néven installált fontkészletet fogja betölteni. Azért töltenek be a különböző mérettartományokban más-más fontkészleteket, mert egy font 5pt méretben lehet, hogy jól néz ki, de kinagyítva 20pt méretre már nem biztos, hogy a legideálisabb. Ennek illusztrálására nézzük meg a „szöveg” kiszedését különböző fontkészletekkel és méretekből:

fontkészlet	méret (pt)	
ec-lmr5	20	szöveg
ec-lmr17	20	SZÖVEG
ec-lmr5	5	szöveg
ec-lmr17	5	szöveg

Vannak olyan installált fontok is, amelyeknek nincsenek variációik a különböző méretekre. Ilyenkor a `<->` kód azt jelenti, hogy minden méret esetén ugyanazt használja. Például:

```
\DeclareFontFamily{T1}{aur}{}
\DeclareFontShape{T1}{aur}{m}{n}{<-> AuriocusKalligraphicus}{}
\DeclareFontShape{T1}{aur}{m}{sl}{<-> AuriocusKalligraphicusSlant}{}
\DeclareFontShape{T1}{aur}{bx}{n}{<-> AuriocusKalligraphicusBold}{}
\DeclareFontShape{T1}{aur}{bx}{sl}{<-> AuriocusKalligraphicusBoldSlant}{}

```

Ha a fontnevek elé például azt írja, hogy `[1.2]`, akkor az adott fontot kinagyítja 1,2-szeresére. Azaz, ha például 10pt-os betűmérettel tölti be, akkor valójában 12pt méretben fog megjelenni:

```
\DeclareFontFamily{T1}{aur}{}
\DeclareFontShape{T1}{aur}{m}{n}{<-> [1.2] AuriocusKalligraphicus}{}

```

22.6. Új családosztály definiálása

Alaphelyzetben három családosztály definiált: antikva, groteszk és írógép. Lehetőség van további családosztályok definiálására is a következő kóddal:

```
\newcommand{\<jel>default}{\<új család kód>}
\DeclareRobustCommand{\<jel>family}{\fontfamily{\<jel>default}\selectfont}
\DeclareTextFontCommand{\text<jel>}{\<jel>family}
\DeclareFontFamily{T1}{\<jel>default}{}
\DeclareFontShape{T1}{\<jel>default}{\<1>default}{\<2>default}{
  <->ssub*<család>/<testesség>/<alak>}{}

```

ahol

`<1>` lehetséges értékei: `md`, `bf`.

`<2>` lehetséges értékei: `up`, `sl`, `it`, `sc`.

Például

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}

% Új családosztály (gótikus) bevezetése 'mygoth' kóddal.
% A definiált \textgt és \gtfamily ezt a családot tölti be. Használata:
%   \textgt{egy bekezdés...}
%   {\gtfamily több bekezdés...}
\newcommand{\gtdefault}{mygoth}
\DeclareRobustCommand{\gtfamily}{\fontfamily{\gtdefault}\selectfont}
\DeclareTextFontCommand{\textgt}{\gtfamily}
\DeclareFontFamily{T1}{\gtdefault}{}
\DeclareFontShape{T1}{\gtdefault}{\mddefault}{\updefault}{
  <->ssub*yfrak/m/n}{}
\DeclareFontShape{T1}{\gtdefault}{\mddefault}{\sldefault}{
  <->ssub*yfrak/m/n}{}
\DeclareFontShape{T1}{\gtdefault}{\mddefault}{\itdefault}{
  <->ssub*yfrak/m/n}{}

```

```

\DeclareFontShape{T1}{\gtdefault}{\mddefault}{\scdefault}{
  <->ssub*yfrak/m/n}{ }
\DeclareFontShape{T1}{\gtdefault}{\bfdefault}{\updefault}{
  <->ssub*yfrak/b/n}{ }
\DeclareFontShape{T1}{\gtdefault}{\bfdefault}{\sldefault}{
  <->ssub*yfrak/b/n}{ }
\DeclareFontShape{T1}{\gtdefault}{\bfdefault}{\itdefault}{
  <->ssub*yfrak/b/n}{ }
\DeclareFontShape{T1}{\gtdefault}{\bfdefault}{\scdefault}{
  <->ssub*yfrak/b/n}{ }

\begin{document}
\noindent
% Elhagyható, mert alapbeállítás: \mdseries \upshape
{\gtfamily\mdseries\upshape Szöveg}\\
{\gtfamily\mdseries\slshape Szöveg}\\
{\gtfamily\mdseries\itshape Szöveg}\\
{\gtfamily\mdseries\scshape Szöveg}\\
{\gtfamily\bfseries\upshape Szöveg}\\
{\gtfamily\bfseries\slshape Szöveg}\\
{\gtfamily\bfseries\itshape Szöveg}\\
{\gtfamily\bfseries\scshape Szöveg}\\
\end{document}

```

22.7. Új testességosztály definiálása

Alaphelyzetben két testességosztály definiált: normál és félkövér. Lehetőség van további testességosztályok definiálására is a következő kóddal:

```

\newcommand{\<jel>default}{\<új testességkód>}
\DeclareRobustCommand{\<jel>series}{\fontseries{\<jel>default}\selectfont}
\DeclareTextFontCommand{\text<jel>}{\<jel>series}
\fontfamily{\<1>default}\selectfont
\DeclareFontShape{T1}{\<1>default}{\<jel>default}{\<2>default}{
  <->ssub*<család>/<testesség>/<alak>}{ }

```

ahol

<1> lehetséges értékei: **rm**, **sf**, **tt**.

<2> lehetséges értékei: **up**, **sl**, **it**, **sc**.

Például

```

\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}

% Új testességosztály (vékony) bevezetése 'l' kóddal.
% A definiált \textlt és \ltseries ezt a testességet tölti be. Használata:
%   \textlt{egy bekezdés...}
%   {\ltseries több bekezdés...}
\newcommand{\ltdefault}{l}
\DeclareRobustCommand{\ltseries}{\fontseries{\ltdefault}\selectfont}

```



```

\DeclareTextFontCommand{\textlt}{\ltseries}

% Alapértelmezett antikva fontcsalád bővítése vékony testességgel:
\fontfamily{\rmdefault}\selectfont
\DeclareFontShape{T1}{\rmdefault}{\ltdefault}{\updefault}{
  <->ssub*jkpx/1/n}{ }
\DeclareFontShape{T1}{\rmdefault}{\ltdefault}{\sldefault}{
  <->ssub*jkpx/1/sl}{ }
\DeclareFontShape{T1}{\rmdefault}{\ltdefault}{\itdefault}{
  <->ssub*jkpx/1/it}{ }
\DeclareFontShape{T1}{\rmdefault}{\ltdefault}{\scdefault}{
  <->ssub*jkpx/1/sc}{ }

% Alapértelmezett groteszk fontcsalád bővítése vékony testességgel:
\fontfamily{\sfdefault}\selectfont
\DeclareFontShape{T1}{\sfdefault}{\ltdefault}{\updefault}{
  <->ssub*AlegreyaSans-LF/t/n}{ }
\DeclareFontShape{T1}{\sfdefault}{\ltdefault}{\sldefault}{
  <->ssub*ComicNeueAngular-TLF/1/sl}{ }
\DeclareFontShape{T1}{\sfdefault}{\ltdefault}{\itdefault}{
  <->ssub*AlegreyaSans-LF/t/it}{ }
\DeclareFontShape{T1}{\sfdefault}{\ltdefault}{\scdefault}{
  <->ssub*AlegreyaSans-LF/t/sc}{ }

% Alapértelmezett írógép fontcsalád bővítése vékony testességgel:
\fontfamily{\ttdefault}\selectfont
\DeclareFontShape{T1}{\ttdefault}{\ltdefault}{\updefault}{
  <->ssub*lmttos/1/n}{ }
\DeclareFontShape{T1}{\ttdefault}{\ltdefault}{\sldefault}{
  <->ssub*lmttos/1/sl}{ }
\DeclareFontShape{T1}{\ttdefault}{\ltdefault}{\itdefault}{
  <->ssub*IBMPlexMono-TLF/1/it}{ }
\DeclareFontShape{T1}{\ttdefault}{\ltdefault}{\scdefault}{
  <->ssub*yesj/1/sc}{ }

\begin{document}
\noindent
% Elhagyható, mert alapbeállítás: \rmfamily \upshape
{\rmfamily\ltseries\upshape Szöveg}\\
{\rmfamily\ltseries\slshape Szöveg}\\
{\rmfamily\ltseries\itshape Szöveg}\\
{\rmfamily\ltseries\scshape Szöveg}\\
{\sfamily\ltseries\upshape Szöveg}\\
{\sfamily\ltseries\slshape Szöveg}\\
{\sfamily\ltseries\itshape Szöveg}\\
{\sfamily\ltseries\scshape Szöveg}\\
{\ttfamily\ltseries\upshape Szöveg}\\
{\ttfamily\ltseries\slshape Szöveg}\\
{\ttfamily\ltseries\itshape Szöveg}\\
{\ttfamily\ltseries\scshape Szöveg}
\end{document}

```

22.8. Új alakosztály definiálása

Alaphelyzetben négy alakosztály definiált: álló, dőlt, döntött és kiskapitális. Lehetőség van további alakosztályok definiálására is a következő kóddal:

```
\newcommand{\<jel>default}{\<új alak kód>}
\DeclareRobustCommand{\<jel>shape}{\fontshape{\<jel>default}\selectfont}
\DeclareTextFontCommand{\text<jel>}{\<jel>shape}
\fontfamily{\<1>default}\selectfont
\DeclareFontShape{T1}{\<1>default}{\<2>default}{\<jel>default}{
  <->ssub*<család>/<testesség>/<alak>}{}
```

ahol

<1> lehetséges értékei: **rm**, **sf**, **tt**.

<2> lehetséges értékei: **md**, **bf**.

Például

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}

% Új alakosztály bevezetése (dőlt kiskapitális) 'scit' kóddal.
% A definiált \textscit és \scitshape ezt az alakot tölti be. Használata:
%   \textscit{egy bekezdés...}
%   {\scitshape több bekezdés...}
\newcommand{\scitdefault}{scit}
\DeclareRobustCommand{\scitshape}{\fontshape{\scitdefault}\selectfont}
\DeclareTextFontCommand{\textscit}{\scitshape}

% Alapértelmezett antikva fontcsalád bővítése dőlt kiskapitális alakokkal:
\fontfamily{\rmdefault}\selectfont
\DeclareFontShape{T1}{\rmdefault}{\mddefault}{\scitdefault}{
  <->ssub*LinuxLibertineT-LF/m/scit}{}
\DeclareFontShape{T1}{\rmdefault}{\bfdefault}{\scitdefault}{
  <->ssub*LinuxLibertineT-LF/b/scit}{}

% Alapértelmezett groteszk fontcsalád bővítése dőlt kiskapitális alakokkal:
\fontfamily{\sfdefault}\selectfont
\DeclareFontShape{T1}{\sfdefault}{\mddefault}{\scitdefault}{
  <->ssub*AlegreyaSans-LF/l/scit}{}
\DeclareFontShape{T1}{\sfdefault}{\bfdefault}{\scitdefault}{
  <->ssub*AlegreyaSans-LF/b/scit}{}

% Alapértelmezett írógép fontcsalád bővítése dőlt kiskapitális alakokkal:
\fontfamily{\ttdefault}\selectfont
\DeclareFontShape{T1}{\ttdefault}{\mddefault}{\scitdefault}{
  <->ssub*qcr/m/scit}{}
\DeclareFontShape{T1}{\ttdefault}{\bfdefault}{\scitdefault}{
  <->ssub*qcr/b/scit}{}

\begin{document}
\noindent
```

```
% Elhagyható, mert alapbeállítás: \rmfamily \mdseries
\noindent
{\rmfamily\mdseries\scitshape Szöveg}\\
{\rmfamily\bfseries\scitshape Szöveg}\\
{\sffamily\mdseries\scitshape Szöveg}\\
{\sffamily\bfseries\scitshape Szöveg}\\
{\ttfamily\mdseries\scitshape Szöveg}\\
{\ttfamily\bfseries\scitshape Szöveg}
\end{document}
```

22.9. Alapértelmezett osztálykombinációk bővítése

A család, testesség és alak kombinálhatóak. Például

```
{\sffamily\bfseries\slshape Szöveg}
```

Szöveg

Azonban nem feltétlenül tartozik minden kombinációhoz megfelelő fontkészlet. Például alaphelyzetben az írógép családhoz a félkövér testesség úgy van hozzárendelve, mint normál testesség. Így a következő két sor ugyanazt eredményezi:

```
{\ttfamily Szöveg}
{\ttfamily\bfseries Szöveg}
```

Szöveg Szöveg

Ilyen esetekben módunk van ezen hiányosságok pótlására a következő kóddal:

```
\fontfamily{\<1>default}\selectfont
\DeclareFontShape{T1}{\<1>default}{\<2>default}{\<3>default}{
  <->ssub*{\<család>/\<testesség>/\<alak>}}{}
```

ahol

<1> lehetséges értékei: **rm**, **sf**, **tt**.

<2> lehetséges értékei: **md**, **bf**.

<3> lehetséges értékei: **up**, **sl**, **it**, **sc**.

Például

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\fontfamily{\ttdefault}\selectfont
\DeclareFontShape{T1}{\ttdefault}{\bfdefault}{\updefault}{
  {\<->ssub*lmtt/b/n}}{ }
\begin{document}
{\ttfamily Szöveg}
{\ttfamily\bfseries Szöveg}
\end{document}
```

Szöveg Szöveg

22.10. Fontok információi és tesztelése

Ha arra kíváncsi, hogy egy bizonyos család adott testesség, alak és méret esetén melyik fontkészletet tölti be, akkor használja a következő kódot:

```
\fontsize{<fontméret>}{\the\baselineskip}
\usefont{<belső kód>}{<család>}{<testesség>}{<alak>}
\xdef\thisfont{\fontname\font}
\thisfont
```

Például

```
\fontsize{16}{\the\baselineskip}
\usefont{T1}{lmr}{m}{n}
\xdef\thisfont{\fontname\font}
\thisfont
```

ec-lmr17 at 16.0pt

Az éppen aktuális font adatainak kiírására használja a következő kódot:

```
\makeatletter
\xdef\thisfont{%
  \f@encoding/\f@family/\f@series/\f@shape/\f@size/\fontname\font}
\makeatother
{\usefont{OT1}{cmr}{m}{n}\thisfont}
```

Arra is lehetőség van, hogy egy család adott testesség, alak és méret esetén betöltött fontjaiban megtervezett összes karaktert megnézzük egy táblázatban. Ehhez használja a következő kódot tartalmazó fájlt:

```
\documentclass{article}
\usepackage[<belső kód>]{fontenc}
\begin{document}
\input{fntproof}
\fontsize{<fontméret>}{\the\baselineskip}
\usefont{<belső kód>}{<család>}{<testesség>}{<alak>}
\initcurrentfont
\fonttable
\end{document}
```

Például

```
\documentclass{article}
\usepackage[T1]{fontenc}
\begin{document}
\input{fntproof}
\fontsize{12}{\the\baselineskip}
\usefont{T1}{lmr}{m}{n}
\initcurrentfont
\fonttable
\end{document}
```

lefordítása után, a kapott táblázatban megnézheti az `ec-lmr12` nevű fontban megtervezett összes karaktert. (Ugyanis T1 belső kódkészlet, lmr család, m testesség, n alak és 12pt méret esetén ezt a fontot tölti be.)

Minden karakterhez tartozik egy kódszám is, melyeket a táblázat első és utolsó soraiból és oszlopaiból tudhatunk meg. A kódszám megadható decimális, oktális és hexadecimális értékkel is. A táblázat csak az oktális és hexadecimális kódokat tartalmazza. Például a K karakter oktális kódja 113, míg a hexadecimális kódja 4B. Ebből a decimális kódja $1 \cdot 8^2 + 1 \cdot 8^1 + 3 \cdot 8^0 = 75$.

Egy karakter a kódszámával is meghívható

```
\symbol{<decimális kód>}
\symbol{'<oktális kód>}
\symbol{"<hexadecimális kód>}
```

vagy

```
\char<decimális kód>
\char'<oktális kód>
\char"<hexadecimális kód>
```

módon. Tehát például

```
\usefont{T1}{lmr}{m}{n}
\symbol{75}
\symbol{'113}
\symbol{"4B}
```

K K K

A következő kóddal a karakterek decimális kódjait írathatjuk ki:

```
\documentclass{article}
\usepackage[<belső kód>]{fontenc}
\def\FONT{\fontsize{<fontméret>}{\the\baselineskip}%
\usefont{<belső kód>}{<család>}{<testesség>}{<alak>}}
\usepackage[a4paper,margin={1cm,1cm},landscape]{geometry}
\usepackage[T1]{fontenc}
\usepackage{multicol,amsmath,xcolor}
\setlength{\columnseprule}{.4pt}
\pagestyle{empty}
\newcounter{currchar}
\renewcommand{\ttdefault}{lmtt}
\renewcommand{\familydefault}{\ttdefault}
\newlength{\fonht}
\newlength{\fonhtd}
\newlength{\fonhtnext}
\newlength{\fonhtnextd}
\settoheight{\fonht}{\FONT\char0}
\settodepth{\fonhtd}{\FONT\char0}
\addtolength{\fonht}{\fonhtd}
\loop
\ifnum\value{currchar}<255
\stepcounter{currchar}
\settoheight{\fonhtnext}{\FONT\char\arabic{currchar}}
\settodepth{\fonhtnextd}{\FONT\char\arabic{currchar}}
\addtolength{\fonhtnext}{\fonhtnextd}
\ifdim\fonhtnext>\fonht\setlength{\fonht}{\fonhtnext}\fi
\repeat
```

```

\addtolength{\fontht}{5pt}
\setcounter{currchar}{0}
\begin{document}
\noindent{\FONT\xdef\thisfont{\fontname\font}}%
\kern1em\framebox{\thisfont}
\begin{multicols}{10}
\noindent
\loop
\ifnum\value{currchar}<256
\phantom{\rule{0pt}{\fontht}}%
\kern1em\smash{\FONT\char\arabic{currchar}}\hfill
{\footnotesize\color{blue}\arabic{currchar}}\kern1em\\
\stepcounter{currchar}
\repeat
\end{multicols}
\end{document}

```

22.11. Fontváltó csomagok

A következő táblázat első oszlopában fontváltó csomagokat tüntettünk fel. A további oszlopokból azt lehet megtudni, hogy az adott csomag milyen kódú fontcsaládokat tölt be az antikva, groteszk és írógép betűcsaládok helyére, továbbá, hogy a matematikai fontokat is átállítja-e. Érdeemes elolvasni a csomagok leírásait is, mert egyesekhez opciók is tartoznak.

csomag	antikva	groteszk	írógép	mat.
lmodern	lmr	lmss	lmtt	☑
times	ptm	phv	pcr	-
txfonts	txr	txss	txtt	☑
pxfonts	pxr	pxss	pxtt	☑
bera	fve	fvs	fvm	-
lxfonts	-	llcmss	llcmtt	☑
newtxtext	ntxtlf	qhv	ntxtt	-
cyklop	cyklop	-	-	-
tgbonum	qbk	-	-	-
tgadventor	-	qag	-	-
tgchorus	qzc	-	-	-
tgcursor	-	-	qcr	-
tgheros	-	qhv	-	-
tgpagella	qpl	-	-	-
tgschola	qcs	-	-	-
tgtermes	qtm	-	-	-
anttor	antt	-	-	☑
arev	fav	fav	fvm	☑
cmbright	-	cmbr	cmtl	☑

23. fejezet

X_ƎL^AT_EX

A X_ƎL^AT_EX név az eXtended (kiterjesztett) L^AT_EX kifejezésre utal. Kiejtése: zílatekh. Ezt a programot Jonathan Kew (a TeXworks és a TeXShop szerzője) 2004-ben készítette. A TeX Live-nak 2007-től része. A program honlapja: [klikk ide!](#)

A X_ƎL^AT_EX a L^AT_EX egy olyan változata, melyben külső fontok is betölthetők. Sajnos nem teljesen kompatibilis a L^AT_EX-fordítókkal (`latex.exe`, `pdflatex.exe`), ugyanis a fejlesztést nem a L^AT_EX3 munkacsoport végzi. A külső fontok használata nagyon hasznos lehet abban az esetben, ha nem a T_EX-rendszer által kezelt saját fontokat akarja használni. Ugyanakkor ebben az esetben számolni kell azzal, hogy a dokumentum nem lesz hordozható, hiszen más gépen nem biztos, hogy a forrás által használt fontok telepítve vannak.

23.1. Fordítás

Használata TeXstudióban   , parancssorból

```
xelatex dokumentum.tex
```

Együttműködik a `latexmk`-val is a következő parancssorral:

```
latexmk -xelatex dokumentum
```

23.2. Jellemzők

- A `xelatex.exe` fordító csak UTF-8 kódolású forráskóddal működik.
- A forrást `xdv`-be (extended dvi), majd `xdvipdfmx.exe`-vel az `xdv`-t `pdf`-be konvertálja. Ezután az `xdv` fájlt törli.
- A forrásfájlban az `inputenc` csomag nem használható, helyette a `fontspec` csomagot kell betölteni.
- Ha `fontenc` csomagot tölt be, akkor azt előbb kell, mint a `fontspec`-et. Ekkor EU1 lesz az alapértelmezés. Az EU1 is jó a magyar ékezetes betűkhöz.
- Alapból a Latin Modern belső fontkészletet tölti be EU1 belső kódkészlettel. Ha European Computer Modern fontkészletet akar használni, akkor a `fontspec` csomagot `cm-default` opcióval töltsse be.

- Matematika fontok kezelése a `mathspec` vagy `unicode-math` csomagokkal lehetséges. Az `ams` csomagokat ezek előtt kell betölteni.
- Képek `eps`, `pdf`, `jpg`, `png` formátumban is betölthetők.
- A `husort`-tal nem kompatibilis, de az `idx` kiterjesztésű fájlban az ékezetes betűk utólagos konvertálása repülő ékezetekre, megoldja a problémát.
- Az `f{}f` nem akadályozza meg a ligatúrát. Helyette: `f\mbox{}f`.

23.3. Fontok betöltése

Korábban láttuk, hogy a fontokat három családba oszthatjuk: antikva, groteszk, írógép. Ezeket rendre a `fontspec` csomag `\setmainfont`, `\setsansfont` és `\setmonofont` parancsaival töltheti be. Például:

```
\documentclass{article}
\usepackage{fontspec}
\setmainfont{Times New Roman}
\setsansfont{Arial}
\setmonofont{Courier New}
\begin{document}
Times New Roman \textsf{Arial} \texttt{Courier New}
\end{document}
```

Ha ideiglenesen át akar térni egy ezektől különböző betűcsaládra, akkor használja a `\fontspec` \in `fontspec` parancsot:

```
\documentclass{article}
\usepackage{fontspec}
\setmainfont{Times New Roman}
\setsansfont{Arial}
\setmonofont{Courier New}
\begin{document}
Times New Roman \textsf{Arial} \texttt{Courier New}
{\fontspec{Book Antiqua} Book Antiqua}
Times New Roman
\end{document}
```

A Windows-ban telepített fontok neveit megnézheti, ha parancssorba a következőt írja:

```
%windir%\fonts
```

A T_EX-rendszerben nem csak a Latin Modern és a Computer Modern fontkészlet található. Betölthetők még az előző parancsokkal a következő fonttípusok:

TeX Gyre Termes
TeX Gyre Adventor
TeX Gyre Bonum
TeX Gyre Chorus
TeX Gyre Cursor
TeX Gyre Heros
TeX Gyre Pagella
TeX Gyre Schola

Például

```
\documentclass{article}
\usepackage{fontspec}
\setmainfont{TeX Gyre Termes}
\setsansfont{TeX Gyre Adventor}
\setmonofont{TeX Gyre Cursor}
\begin{document}
...
\end{document}
```

23.4. Az ifxetex csomag

Ha figyelmeztetni akarja a felhasználót, hogy `xelatex.exe` fordítót kell alkalmaznia, akkor használja a `\RequireXeTeX ∈ ifxetex` parancsot.

Ha olyan forrást akar, amely többféle fordítóval, közöttük `xelatex.exe`-vel is használható, akkor alkalmazza az `\ifxetex ∈ ifxetex` feltételes utasítást. Például:

```
\documentclass{article}
\usepackage{ifxetex}
\usepackage[T1]{fontenc}
\ifxetex
  \usepackage{fontspec}
\else
  \usepackage[utf8]{inputenc}
\fi
\begin{document}
...
\end{document}
```

24. fejezet

További információk

24.1. Hasznos csomagok

afterpage Megadhatja, hogy egy oldal befejezése után mi történjen.

calc Számoláshoz alkalmas.

comma Számlálók ezres csoportosítása.

dirtree Könyvtárszerkezet megjelenítéséhez. Például

```
\renewcommand*{\DTstylecomment}{\color{blue}}
\dirtree{%
.1/images\DTcomment{képek helye}.
.2/sources.
}
```

bookcover Könyvborító készítéséhez.

empheq Többsoros képlet keretezésére.

fancypar Bekezdések háttérének beállítása.

fancytooltips Hivatkozások külön ablakban bukkanjanak fel.

fp Fixpontos aritmetikánál használható, maximum 18 számjegyig.

hfoldsty Régi típusú számok.

keystroke Billentyűzet rajzolására.

lwrap L^AT_EX konvertálása HTML formátumba

menukeys Programleírások esetén a menü leírására.

minitoc Al tartalomjegyzékek létrehozására.

moresize A relatív betűméretek listája bővül.

numspell Maximum 66 jegyű nemnegatív egész szám betűzése.

pdfcomment PDF-ben felbukkanó megjegyzések írása.

pdfmarginpar PDF-ben felbukkanó megjegyzések írása.

picinpar Képek körbefuttatására.

prettyref A `\ref` parancs tudását bővíti.

pst-3d 3D árnyékoláshoz (csak latex.exe-vel megy).

pst-3dplot 3D rajzokhoz (csak latex.exe-vel megy).


pst-fr3d 3D dobozhoz (csak latex.exe-vel megy).

pst-text Görbén vezetett szöveghez (csak latex.exe-vel megy).

refcheck A pdf-be írja a kereszthivatkozások label-jeit széljegyzetként. Azt is mutatja, hogy melyekre hivatkoztunk, melyekre nem.

relsize Aktuális betűmérethez viszonyított relatív betűméret használata.

rotating Objektumok elforgatása.
selectp A dokumentumnak csak bizonyos oldalai jelennek meg.
sepnum Számok automatikus ezres csoportosítása.
seqsplit Hosszú karakterlánc választható el bárhol, elválasztó jel nélkül. Például a π értékét írjuk ki nagyon sok tizedesjeggyel.
shadethm Tételszerű környezetek árnyékolására.
sidecap A kép címét a kép oldalára lehet rakni.
siunitx Számok és SI mértékegységek írása. Például automatikus ezres csoportosítás a következő módon lehetséges:

 `\num{<szám>} \in siunitx`

Az ezres csoportosító jel a `\`, (átállítás például pontra a következő opcióval lehetséges: `group-separator={.}`).

sketch A tikz csomagot kiegészíti 3D lehetőségekkel.
splitindex Több tárgymutató is készíthető egy dokumentumban.
spot Különlegesen lehet kiemelni.
subfigure Számozott képeknél alszámozás esetén.
stringstrings Sztringek kezelése
sverb Például `\begin{demo}{Cím}\frac{1}{2}\end{demo}`
tablists Sorfolytonos számozott listákhoz.
tcolorbox Színes dobozok készítése.
tdclock A pdf-ben az aktuális időpont írható ki, azaz nem a fordítás időpontja. Ez csak Adobe esetén jelenik meg jól.
tex4ht \LaTeX konvertálása HTML illetve XML formátumba
textpos Szöveget az adott oldal tetszőleges pozíciójába rakhatunk.
tocloft Tartalomjegyzék stílus készítés.
todonotes Dokumentumban megjegyzéseket lehet ezzel készíteni.
tram Szöveg háttérét kipontozza.
umoline Többsoros szöveg aláhúzásához.
varioref A `\ref` parancs tudását bővíti.
venndiagram Egyszerűen rajzolhatunk Venn-diagramokat.
wcol Többhasábos szedést lehet csinálni úgy, hogy a hasábok különböző szélesek legyenek.
xargs Többopciós parancsok definiálásának megkönnyítése
xstring Sztringek kezelése

24.2. Ha PDF-ben a betűk nem vektorgrafikusan jelennek meg

Ez az eset akkor fordulhat elő, ha nem teljes \TeX -rendszer telepített. Például, ha a MiKTeX portable verziójával dolgozik, akkor az helytakarékoság miatt, alaphelyzetben a Computer Modern fontkészletnek csak a bitmap-es verzióját tartalmazza, vagyis a címbe szereplő gond lép fel. Ilyenkor csak annyit kell tenni, hogy utólag kiegészítjük a rendszerünket a szükséges fájlokkal. Indítsa el a MiKTeX package manager-t, válassza ki a `cm-super` sort, majd a `+` jelű gombbal telepítse.

24.3. PDF-ből kimásolt szöveg

Ha PDF-ből kimásolva egy szövegrészt, majd azt egy editorba beszúrva rossz karaktereket kapunk, és valamiért fontos, hogy ez ne így legyen, akkor a forrásfájlban tölts be a `cmap` csomagot a preambulum elején és az `upquote` csomagot a preambulum végén.

24.4. HTML oldalakon képletek megjelenítése közvetlenül \LaTeX forrásból

Erre alkalmas a MathJax nevű JavaScript. A honlapja itt található: [klikk ide!](#) Egy példát is megnézhet itt: [klikk ide!](#)

24.5. A hyperref csomag egy hibája

Próbálja ki a következő kódot:

```
\documentclass{report}
\usepackage{hyperref}
\author{A}
\title{A}
\begin{document}
\maketitle
\tableofcontents
\chapter{A}
\end{document}
```

Lefordítva `pdflatex.exe`-vel, a következő figyelmeztetést fogja kapni:

```
destination with the same identifier (name{page.1}) has been already used,
duplicate ignored<to be read again>
```

Megoldás, hogy a `\maketitle` parancs helyett használja a következő kódot:

```
\hypersetup{pageanchor=false}\maketitle\hypersetup{pageanchor}
```

Ha valakit nem zavar ez a figyelmeztetés, akkor nem kell tennie semmit.

24.6. dottedtocline=fix

A `magyar.ldf defaults=hu-min` opciója bekapcsolja a `dottedtocline=fix` opciót is, ami néha gondot okoz. Például próbálja ki a következő kódot:

```
\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\begin{document}
\tableofcontents
\section{A}
\subsection{A}
\end{document}
```

Lefordítva `pdflatex.exe`-vel, a következő figyelmeztetést fogja kapni:

```
pdflatex.exe: pop empty color page stack 0]Package atveryend Info: Empty hook `AfterLastShipout'
```

Megoldásként a `defaults=hu-min` után töltsse be a `dottedtocline=unchanged` opciót, vagy használja az `xcolor` csomagot. Ha valakit nem zavar ez a figyelmeztetés, akkor nem kell tennie semmit.

24.7. Ha a magyar nem alapnyelvként van beállítva

Próbálja ki a következő kódot, melyben nem a magyar az alapnyelv.

```
\documentclass{book}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar,english]{babel}
\begin{document}
\chapter{Title}
Text\newpage Text
\end{document}
```

Az eredmény 2. oldalán „1. CHAPTER” jelenik meg „CHAPTER 1.” helyett. Megoldásként a `defaults=hu-min` után töltsse be a `classmod=unchanged` opciót is.

24.8. A magyar.ldf téves kódolási figyelmeztetése

Az `inputenc` csomag legújabb verziója nem kompatibilis a `magyar.ldf` kódolási figyelmeztető rendszerével. Így, ha az `inputenc` csomagot `latin2` vagy `utf8` opcióval töltötte is be a `babel` csomag `magyar` opciója mellett, akkor is a következő figyelmeztetést kapja:

```
Please use
\usepackage[latin2]{inputenc} or \usepackage[utf8]{inputenc}
with \usepackage[magyar]{babel}.
```

Ha ez zavarja, akkor a `magyar.ldf` `suggestions=no` opcióját írja be a `defaults=hu-min` opció után:

```
\PassOptionsToPackage{defaults=hu-min,suggestions=no}{magyar.ldf}
```

Ez a hiba már korrigálva van a `magyar.ldf`-ben, de a TeX Live-nak egyelőre még nem része a javított verzió.

25. fejezet

Linkek

25.1. Videóleckék

- [Telepítés menete](#)
- [Az első L^AT_EX-dokumentum készítése](#)
- [Betűtípusok és -méretek, térközök, törések](#)
- [Bekezdések, lábjegyzetek, színek, kereszthivatkozások](#)
- [Listák](#)
- [Képek és táblázatok](#)
- [Tárgymutató készítése](#)

25.2. Gyakorlatok

- [1. gyakorlat](#) – bekezdések, központosítás, betűméretek, betűtípusok, igazítások, listák, térközök
- [2. gyakorlat](#) – listák, táblázatok, úsztatás, kereszthivatkozások, lábjegyzetek
- [3. gyakorlat](#) – URL, képek, úsztatás, kereszthivatkozások
- [4. gyakorlat](#) – saját úsztatott környezet, dobozok, többhasábos szedés, színek
- [5. gyakorlat](#) – matematikai képletek
- [6. gyakorlat](#) – verbatim, programkódok
- [7. gyakorlat](#) – strukturált mű `article` dokumentumosztályban, tételszerű környezetek, matematikai képletek
- [8. gyakorlat](#) – strukturált mű `report` dokumentumosztályban, margók, tartalomjegyzék, fej- és lábléc, irodalomjegyzék
- [9. gyakorlat](#) – szakdolgozat készítése `thesis-ekf` dokumentumosztályban
- [10. gyakorlat](#) – prezentáció készítése `beamer` dokumentumosztályban
- [Beadandó feladatok](#)

25.3. Sablonok

- [LaTeX Templates](#)
- [TeXample.net](#)
- [Magyar nyelvű dokumentumalap](#) **Overleaf**
- [Szakdolgozat – thesis-ekf](#) (Forrás a CTAN-en: [klikk ide.](#)) **Overleaf**
- [Dolgozat](#) **Overleaf**
- [Prezentáció](#) **Overleaf**
- [Határidőnapló](#) **Overleaf**
- [Curriculum Vitae](#) **Overleaf**
- [Levél](#) **Overleaf**
- [Angol nyelvű cikk article osztállyal](#) **Overleaf**
- [Angol nyelvű cikk amsart osztállyal](#) **Overleaf**
- [Annales Mathematicae et Informaticae folyóirat cikksablonja](#) **Overleaf**

25.4. T_EX-rendszerek

- [TeX Live](#) ([svn](#) – [source](#) – [bug](#) – bug report: tex-live@tug.org)
- [MacTeX](#)
- [MiKTeX](#)
- [proTeXt](#) MiKTeX-alapú rendszer Windowsra
- [BaKoMa TeX](#) „Amit látsz, azt kapod” típusú szerkesztő felülettel rendelkező L^AT_EX-rendszer, fizetős.

25.5. Installálás nélkül, online működő T_EX-rendszerek

- [Overleaf](#)
- [ShareLaTeX](#)
- [LaTeX Base](#)
- [Papeeria](#)

25.6. Mobil eszközökön működő T_EX-rendszerek

- [LaTeX Editor](#) Androidon futtatható ingyenes alkalmazás. Ennek alapját egy 2014-es T_EX-rendszer képezi, amely offline is használható. Internetes kapcsolat akkor kell hozzá, ha egy hiányzó csomagot tölt le.
- [VerbTeX](#) Androidon futtatható ingyenes alkalmazás. Ez offline nem használható. Fordításkor egy online elérhető szerverre telepített TeX Live rendszert használ.

25.7. T_EX-hez fejlesztett editorok

- [TeXstudio](#) ([Sourceforge](#), [GitHub](#))
- [Texmaker](#)
- [WinEdt](#) (shareware)
- [Kile](#)
- [TeXworks](#)
- [TeXnicCenter](#) ([Sourceforge](#))
- [LyX](#) „Rich text” szerkesztő felületet biztosít, amely félig „Amit láatsz, azt kapod” típusú rendszer. Hátránya, hogy más szerkesztő által létrehozott L^AT_EX-forrást nem tud kezelni.
- [JabRef](#) A BibT_EX használatát segítő editor.
- [Editorok összehasonlítása](#)

25.8. Leírások

- [Donald Ervin Knuth: The TeXbook](#)
- [LaTeX2e unofficial reference manual](#)
- [LaTeX Wikibook](#)
- [Dickimaw LaTeX Books](#)
- [TeXdoc Online](#)
- [TeX tips](#)
- [Egy nem túl rövid bevezető a LaTeX2e használatába avagy LaTeX2e 78 percben](#)
- [Bujdosó Gyöngyi: L^AT_EX kezdőlépések](#)
- [Szabó Péter: Magyar nyelvű műszaki-tudományos tipográfia](#)
- [Szabó Péter: Magyar nyelvű szöveg szedése MagyarL^AT_EX-hel](#) (A magyar tipográfiát követő segédfájlok dokumentációja: `magyar.ldf`, `huplain.bst`, `husort.pl`.)
- [Mayer Gyula, Pröhle Péter: LaTeX – platformfüggetlen általános célú dokumentum készítő rendszer](#) ([videó1](#), [videó2](#), [videó3](#), [videó4](#))

25.9. Magyar tipográfiát követő segédfájlok

- [magyar.ldf](#) (magyar Babel stílus)
- [huplain.bst](#) (magyar nyelvű irodalomjegyzék-stílus BibTeX-hez)
- [huszak.bst](#) (ugyanaz, mint a [huplain.bst](#), csak a szerzők vezetéknévét kiskapitálissal szedi)
- [husort.pl](#) (makeindex-et felváltó index processzor, a magyar nyelvet követi, Perl szkript)

25.10. L^AT_EX oldalak

- [TeX Users Group](#)
- [The Comprehensive TeX Archive Network](#)
- [The LaTeX Project](#)
- [BME Math LaTeX](#)

25.11. L^AT_EX fórumok

- [LaTeX Community](#)
- [TeX - LaTeX Stack Exchange](#)
- [\howto TeX](#)

25.12. L^AT_EX fontok

- [The L^AT_EX Font Catalogue](#)
- [Detexify - LaTeX symbol classifier](#)
- [The Comprehensive LaTeX Symbol List](#)

25.13. Segédprogramok

- [TikzEdt](#) (Tikz csomag használatát segítő WYSIWYG/text editor)
- [Asymptote: The Vector Graphics Language](#) (TeX Live tartalmazza)
- [Ghostscript, GSview](#)
- [Sumatra PDF](#)
- [MathJax](#) (HTML oldalakon képletek jeleníthetők meg L^AT_EX-parancsokkal. [Itt](#) egy példa.)

- [LaTeX Tables Generator](#) (L^AT_EX táblázat online)
- [Equation Editor](#) (L^AT_EX egyenletszerkesztő online)

Irodalomjegyzék

- [1] Bujdosó Gyöngyi, Fazeka Attila: T_EX kezdőlépések. Budapest, 1997, Tertia Kiadó.
- [2] Donald Ervin Knuth: The T_EXbook. Reading/Ma. etc., 1984, Addison-Wesley.
- [3] L^AT_EX News, Issue 28, April 2018.
- [4] LaTeX2e unofficial reference manual.
- [5] Szabó Péter: Magyar nyelvű szöveg szedése MagyarL^AT_EX-hel
- [6] Tobias Oetiker, Hubert Partl, Irene Hyna, Elisabeth Schlegl: Egy nem túl rövid bevezető a L^AT_EX 2_ε használatába.
- [7] Wettl Ferenc, Mayer Gyula, Szabó Péter: L^AT_EX kézikönyv, Budapest, 2004, Panem Könyvkiadó. (1. és 2. fejezet)
- [8] Wikibooks.org: LaTeX.
- [9] Tibor Tómacs: Thesis class for the Eszterházy Károly University