

Tómács Tibor

TATEX

Legutóbbi módosítás 2025. 10. 04. 7:43

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[hungarian]{babel}
\begin{document}
Hello, világ!
\end{document}
```

```

\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\begin{document}
Hello World!
\end{document}

\documentclass[12pt,twoside]{report}
\usepackage[T1]{fontenc}
\usepackage[fancyhdr]{fancyhdr}
\usepackage[colorlinks,linktocpage,allcolors=blue,pdfstartview=Fifth,
bookmarksnumbered]{hyperref}
\usepackage[a4paper,top=40mm,bottom=40mm,inner=40mm,outer=30mm,headsep=8mm,
headheight=15pt]{geometry}
\PassOptionsToPackage{defaults=hu-min}{magyar.lrf}
\usepackage[magyar]{babel}
\usepackage{fullipsum}
\pagestyle{fancy}
\fancyhf{}
\fancyhead[LE,RO]{\normalfont\normalsize\thepage}
\fancyhead[RE]{\small\sffamily\small\leftmark}
\fancyhead[LO]{\small\sffamily\small\rightmark}
\title{Cím}
\author{Szerző neve}
\date{Készült \today}
\begin{document}
\maketitle
\tableofcontents
\chapter*(Bevezetés)
\markboth{Bevezetés}{Bevezetés}
\fullipsum
\chapter{Fejezet címe}\label{fejezet-pelda}
\section{Szakasz címe}
\fullipsum
\end{document}
\azt(\ref{fejezet-pelda}). -fejezetben \dots
Lásd még \cite[oldal]{DAROCZY}.
Ajánlott feladatgyűjtemények: \cite{DENKINGER,SOLT}.

\begin{thebibliography}{3}
\bibitem[DAROCZY]{DAROCZY} \textsc{Daróczy Zoltán}: \emph{Mérték és integrál}, Budapest, 1984, Tankönyvkiadó.
\bibitem[DENKINGER]{DENKINGER} \textsc{Denninger Géza}: \emph{Valószinúségszámítási gyakorlatok}, Budapest, 1986, Tankönyvkiadó.
\bibitem[SOLT]{SOLT} \textsc{Solt György}: \emph{Valószinúségszámítás}, Budapest, 1993, Műszaki Könyvkiadó.
\end{thebibliography}
\end{document}

\documentclass{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.lrf}
\usepackage[magyar]{babel}
\begin{document}
Hello, világ!
\end{document}

```

Tómács Tibor

LATEX

2025

© Tómács Tibor, 2025

A könyv minden részlete *L^AT_EX*-ben készült, beleértve a borítót és az ábrákat is. Ez alól csak Donald Ervin Knuth és Leslie Lamport fotói, Duane Bibby rajza, továbbá a Könyvborító című fejezetben található borító fotója és ugyanezen fejezet példájához szükséges ábrák kivételek.

A pdf generálásához használt *T_EX*-rendszer:
pdfTeX, Version 3.141592653-2.6-1.40.28 (TeX Live 2025) kpathsea version 6.4.1

A könyv szabadon letölthető az alábbi linkről:
<https://tibortomacs.github.io/latex-tutorial-hu/latex.pdf>

latex-tutorial-hu

Legutóbbi módosítás: 2025. október 4. 7:43

Tartalomjegyzék

Előszó	13
A szerzőről	14
Jelölések	16
1. Bevezetés	17
1.1. A L ^A T _E X koncepciója és jellemzői	18
1.2. T _E X-disztribúciók	19
1.3. L ^A T _E X-editorok	19
1.4. L ^A T _E X használata online (Overleaf)	19
1.5. Terminál, parancssor	19
1.6. Telepítés	20
1.6.1. TeX Live és TeXstudio telepítése Windowsra	20
1.6.2. TeX Live és TeXstudio telepítése Ubuntu Linuxra	21
1.6.3. MiKTeX és TeXstudio telepítése Windowsra	22
1.6.4. MiKTeX és TeXstudio telepítése Ubuntu Linuxra	23
1.6.5. Texfireplace telepítése Windowsra	24
1.7. L ^A T _E X-csomagok frissítése	24
1.7.1. TeX Live frissítése	25
1.7.2. MiKTeX frissítése	25
1.7.3. Texfireplace frissítése	25
1.8. Fontosabb fájlkiterjesztések	25
1.9. A T _E X-rendszer fontosabb programjai	26
1.10. TeXstudio beállítások	28
2. Az első lépések	30
2.1. A L ^A T _E X alapfogalmai	30
2.1.1. Parancs	30
2.1.2. Kötelező argumentum	30
2.1.3. Opcionális argumentum	30
2.1.4. Környezet	31
2.1.5. Blokk	31
2.1.6. Deklarációs parancs	32
2.1.7. Komment	32
2.1.8. Dokumentumosztály, preambulum, dokumentumtest	32
2.1.9. Csomag	33
2.2. Fontosabb standard dokumentumosztályok	33
2.3. Az első dokumentum elkészítése	34

3. A dokumentum nyelve	38
3.1. A babel csomag	38
3.2. A szavak elválasztása	39
3.3. Sorvégi túlcsordulás	41
3.4. A magyar.ldf aktív karakterei	42
4. Alapvető formai elemek	44
4.1. Karakterek	44
4.1.1. Foglalt karakterek	44
4.1.2. Ékezetes betűk	44
4.1.3. Speciális betűk	45
4.1.4. Kerning	45
4.1.5. Ligatúrák	45
4.1.6. Különleges karakterek	46
4.2. Szóközök	47
4.3. Központozás	48
4.3.1. Pont, vessző, kettőspont, pontosvessző, kérdőjel, felkiáltójel .	48
4.3.2. Kötőjel	48
4.3.3. Nagykötőjel	48
4.3.4. Gondolatjel, kvírtmínusz	49
4.3.5. Zárójelek	49
4.3.6. Hármaspont	49
4.3.7. Idézőjel	49
4.4. Betűváltozatok	51
4.4.1. Osztályozás	51
4.4.2. Kurzív kiegyenlítés	53
4.4.3. Kiemelés	53
4.5. Betűméretek	55
4.5.1. Alapbetűméret	55
4.5.2. Betűméretet beállító deklarációs parancsok	55
4.5.3. Relatív betűméretek	56
4.5.4. Abszolút betűméretek	56
4.6. Térközök	57
4.6.1. Fix méretű vízszintes térközök	57
4.6.2. Rugalmas méretű vízszintes térközök	58
4.6.3. Fix méretű függőleges térközök	58
4.6.4. Rugalmas méretű függőleges térközök	59
4.6.5. Sortávolság	60
4.7. Törések	60
4.7.1. Sortörések	60
4.7.2. Oldaltörések	61
4.8. Bekezdések	61
4.8.1. Bekezdések balra zárása	62
4.8.2. Bekezdések jobbra zárása	63
4.8.3. Bekezdések középre zárása	63
4.8.4. Többsoros idézetek	63
4.8.5. Versek	64
4.8.6. Párbeszédek	64

4.9.	Tabulálás	65
4.10.	Lábjegyzetek	66
4.11.	Széljegyzetek	67
4.12.	Színek kezelése	68
4.12.1.	Színmodellek és paraméterek	68
4.12.2.	Színnevek	69
4.12.3.	Színes szöveg	69
4.12.4.	Átlátszóság	70
4.12.5.	Szöveg kiemelése színes háttérrel	70
4.12.6.	Szöveg kiemelése színes aláhúzással	71
4.12.7.	Szöveg kiemelése színes áthúzással	71
4.12.8.	Színes lapok	71
4.13.	Dátumtípusok	72
4.14.	Számírás	72
4.15.	Mértékegységek	74
5.	Oldalak kinézete	76
5.1.	Oldalak szerkezete és méretei	76
5.2.	Lapmátrix megváltoztatása dokumentumon belül	77
5.3.	Oldalak nagyítása/kicsinyítése	78
5.4.	Többhasábos szedés	79
5.5.	Oldal elforgatása	79
5.6.	Vágójelek nyomdai előkészítéshez	80
5.7.	Méretek ellenőrzése	81
6.	Kereszthivatkozások	83
6.1.	Címkék	83
6.2.	Hivatkozás címkézett elemekre	84
7.	Listák	86
7.1.	Számozatlan listák	86
7.1.1.	Felsorolásjelek megváltoztatása	87
7.1.2.	Számozatlan listák extra függőleges térközök nélkül	88
7.2.	Leíró listák	88
7.3.	Számozott listák	89
7.3.1.	Számozott listák számozási stílusának megváltoztatása	90
7.3.2.	Hivatkozás számozott listaelemre	95
7.3.3.	Számozott listák extra függőleges térközök nélkül	96
7.3.4.	Sorfollytonos számozott listák	96
7.4.	Lista paramétereinek beállítása az enumitem csomaggal	97
7.4.1.	Korlátozások	97
7.4.2.	Listák globális formázása	97
7.4.3.	Listák lokális formázása	100
7.4.4.	Sorfollytonos listák	101
7.4.5.	Új listakörnyezet definiálása	101

8. Képek	103
8.1. Kép beillesztése	103
8.2. Hát- illetve előtérbe illesztés	105
8.3. Külső pdf oldalak beszúrása	106
9. Ábrák készítése	108
9.1. Koordináta-rendszer, referenciaPont és vonalvastagság	108
9.2. Szakaszok, törött vonalak és vektorok	109
9.3. Körvonalak	111
9.4. Lekerekített sarkú téglalapok	112
9.5. Bézier-görbék	113
9.6. Útvonalak	113
9.7. Vonalak végeinek és útvonalak csatlakozási pontjainak stílusa	116
9.8. Betűk elhelyezése ábrában	118
9.9. Koordináta megadása hosszmérettel	120
10. Táblázatok	121
10.1. Standard táblázatkezelési eszközök	121
10.1.1. Példatáblázatok	121
10.1.2. Hosszú táblázatok	127
10.1.3. Kiadói minőségű táblázatok	128
10.1.4. Táblázatok alapvonalhoz igazítása	129
10.2. A tabulararray csomag	130
10.2.1. Normál tabulararray táblázatok	130
10.2.2. Hosszú tabulararray táblázatok	143
11. Objektumok úsztatása	147
11.1. Képek és táblázatok úsztatása	147
11.2. Úsztatás kéthasábos szedés esetén	148
11.3. Úsztatott objektumok feliratozása	148
11.4. Úsztatott objektumok feliratainak testreszabása	151
11.5. Saját úsztatott objektumok létrehozása	156
11.6. Úsztatás mellőzése	156
11.7. Több objektum egy úsztató környezetben	157
12. Objektumok körbefuttatása	162
12.1. A wrapstuff csomag	162
12.2. A wrapfig2 csomag	165
12.3. A floatflt csomag	166
13. Dobozok	168
13.1. Egysoros dobozok	168
13.2. Bekezdésdobozok	170
13.3. Vonaldobozok	172
13.4. Dobozok nyújtása, tükrözése	172
13.5. Dobozok átméretezése	173
13.6. Dobozok forgatása	173
13.7. Dobozok transzformálása a pdf-trans csomaggal	174
13.7.1. Tükrözés	174

13.7.2. Eltolás	175
13.7.3. Döntés	176
13.7.4. Foramatás	177
13.7.5. Nyújtás	178
13.7.6. Doboz átméretezése	180
13.7.7. Egyebek	181
13.8. Doboz méreteinek nullázása	182
13.9. Láthatatlan dobozok	183
13.10. Dobozok halmozása	184
14. Verbatim, programkód, URL	187
14.1. Verbatim	187
14.2. Verbatim szöveg kiírása fájlba	189
14.2.1. A newfile csomag	189
14.2.2. Az answers csomag	191
14.3. Programkódok	193
14.3.1. A listings csomag	193
14.3.2. A minted csomag	199
14.4. URL címek megadása	204
15. Képletek	206
15.1. Matematikai mód	206
15.2. Matematikai betűváltozatok	209
15.3. Kalligrafikus, dupla szárú betűk és fraktúrák	210
15.4. Görög betűk	210
15.5. Matematikai ékezetek	211
15.6. Közönséges matematikai jelek	211
15.7. Műveleti jelek	212
15.8. Három pont	212
15.9. Relációjelek	213
15.10. Matematikai zárójelek	214
15.11. Matematikai jelek több szerepben	218
15.12. Esetek szétválasztása	219
15.13. Változó hosszúságú vízszintes jelek	220
15.14. Gyökvonás	221
15.15. Mátrixok	222
15.16. Matematikai jelek egymásra helyezése	224
15.17. Matematikai indexek	225
15.18. Törtek, binomiális együtthatók	226
15.19. Operátorok, függvények	227
15.19.1. Nagy operátorok	227
15.19.2. „Nolimits” függvények	228
15.19.3. „Limits” függvények	229
15.19.4. Új függvények definiálása	230
15.19.5. Differenciál operátor, differenciálás	234
15.20. Kommutatív diagramok	234
15.21. Színek használata képletekben	235
15.22. Képletek bekeretezése	235
15.23. Kiemelt képletek sorszámozása	236

15.24. Képletek eltörése	238
15.25. Több képlet egymás alatt	240
15.26. Több képlet egymás alatt illesztéssel	243
15.27. Részformulák számozása	250
15.28. Oldaltörés többsoros képletekben	250
15.29. Táblázat matematikai módban	250
15.30. Számolás L ^A T _E X segítségével	251
15.31. HTML-ben képletek írása L ^A T _E X szintaxissal	255
16. További formai elemek	256
16.1. Görög betűk	256
16.2. Cirill betűk	257
16.3. Gótikus írás	258
16.4. Iniciálék	258
16.4.1. Latin iniciálé	258
16.4.2. Díszes latin iniciálé	258
16.4.3. Gótikus iniciálé	259
16.4.4. Díszes gótikus iniciálé	260
16.5. Betűk kontúrozása és árnyékolása	261
16.6. Alá- és föléhúzás egyszerre	262
16.7. Díszítő elemek	262
16.8. Vonalazott lapok	263
16.9. Négyzetrácsos lapok	263
16.10. Az oldal két pontjának összekötése vonallal	263
16.11. A pdf készítésének ideje óra percben	265
16.12. Nem vízszintes alapvonalú szöveg szedése	265
16.13. Lorem ipsum	267
16.14. Vízjel	268
16.15. Különleges bekezdések	269
16.16. Vonalkód	270
16.17. QR-kód	271
16.18. Dátumtípusok automatikus toldalékolása	271
16.19. Számok automatikus toldalékolása	272
16.20. Automatikus magyar határozott névelő	272
16.21. T _E X-hel kapcsolatos logók	274
17. Strukturált művek	275
17.1. Főcím, címlap, kivonat	275
17.2. A főszöveg szintjei	276
17.3. Fattyúsorok	277
17.4. Fej- és láblécek	278
17.4.1. Alapbeállítások	278
17.4.2. Fej- és láblécek testreszabása	279
17.5. Tételszerű bekezdések	284
17.6. Jegyzékek	289
17.6.1. Tartalomjegyzék	289
17.6.2. Táblázatok jegyzéke	290
17.6.3. Ábrák jegyzéke	292
17.6.4. Kódok jegyzéke	293

17.6.5. Saját úsztatott objektumok jegyzéke	293
17.6.6. Új jegyzék készítése	294
17.6.7. Jegyzékek stílusának testreszabása a tocloft csomaggal	295
17.6.8. Jegyzékek készítésének elvi alapjai	299
17.7. Végjegyzetek	305
17.8. Bibliográfia	306
17.8.1. Bibliográfia készítése környezettel	306
17.8.2. A biblatex csomag	308
17.9. Tárgymutató	321
17.9.1. Az indextools és texindy együttes használata	321
17.9.2. Az indextools és upmendex együttes használata	326
17.10. Függelék	331
17.11. Hosszabb művek szervezése	331
18. Könyvborító	332
18.1. A könyvborító részei	332
18.2. A bookcover dokumentumosztály	333
19. Elektronikus publikáció	337
19.1. A hyperref csomag	337
19.2. Fájlok csatolása pdf-be	340
20. Szakdolgozat készítése	342
21. Prezentációk	344
21.1. Témák	344
21.1.1. Teljes témák	344
21.1.2. Belső témák	345
21.1.3. Külső témák	345
21.1.4. Színtémák	346
21.1.5. Betűtípus témák	347
21.2. Keretek	347
21.3. Egy keretben több dia	348
21.3.1. Overlay specifikációk	348
21.3.2. Diasorozat átlátszósága	350
21.3.3. Overlay specifikációval rendelkező parancsok	350
21.4. Diaváltás látványeffektekkel	351
21.5. A prezentáció tagolása	352
21.5.1. Címoddal	352
21.5.2. A főszöveg tagolása	353
21.5.3. Tartalomjegyzék	354
21.5.4. Irodalomjegyzék	355
21.6. Tartalmi elemek	355
21.6.1. Listák	355
21.6.2. Többhasábos terület	356
21.6.3. Tömbök, tételeszerű környezetek	357
21.6.4. Dobozok	357
21.6.5. Háttér	359
21.6.6. Képek	359

21.6.7. Animáció	359
21.6.8. Videó	360
21.6.9. Nagyítás	361
21.6.10. Kereszthivatkozások, nyomógombok	362
21.6.11. Keret ismétlése	363
22. A L^AT_EX programozása	365
22.1. ASCII kódolás és kategória kódok	365
22.2. Kontrollszo, token, makró	368
22.3. Belső parancsok	369
22.4. Hosszúságparancsok	369
22.5. Szóközök méretének beállítása	372
22.6. Számlálók	373
22.7. Vezérlő utasítások	376
22.7.1. Feltételes utasítások	376
22.7.2. Ciklusok	381
22.8. Makrók definiálása	382
22.8.1. A <code>\newcommand</code> parancs	382
22.8.2. A <code>\def</code> parancs	384
22.8.3. Az <code>\edef</code> és <code>\noexpand</code> parancsok	388
22.8.4. Az <code>\expandafter</code> parancs	388
22.8.5. Parancsok másolása	390
22.8.6. Lokális és globális hatás	391
22.8.7. Makrónév definiálása	392
22.8.8. Mozgó argumentum, gyenge és erős parancsok	393
22.8.9. A <code>\NewDocumentCommand</code> parancs	393
22.8.10. Az etoolbox csomag	397
22.8.11. Létező parancs bővítése	398
22.8.12. Létező parancs egy részletének cseréje	398
22.8.13. Új verbatim parancs	399
22.8.14. Új overlay specifikációval rendelkező parancs	399
22.8.15. Új <code>kulcs=érték</code> típusú opcióval rendelkező parancs	400
22.9. Környezetek definiálása	402
22.9.1. A <code>\newenvironment</code> parancs	402
22.9.2. A <code>\NewDocumentEnvironment</code> parancs	404
22.9.3. Új verbatim típusú környezet	405
22.9.4. Új comment típusú környezet	405
22.9.5. Új overlay specifikációval rendelkező környezet	406
22.9.6. Új listakörnyezet	406
22.10. Kapserek	407
22.10.1. Környezetkapserek	407
22.10.2. Dokumentumtest-kapserek	408
22.10.3. Parancskapserek	409
22.10.4. Oldalkapserek	409
22.10.5. Bekezdéskapserek	410
22.10.6. Fájlkapserek	410
22.11. Csomag betöltésének megakadályozása	412

23. Stílusfájlok írása	413
23.1. Csomag készítése	413
23.2. Dokumentumosztály készítése	417
24. Fontok kiválasztása	419
24.1. L ^A T _E X fontkatalógus	419
24.2. A forrásfájl fontkódolása és a L ^A T _E X belső kódkészlete	419
24.3. Globális beállítás	421
24.3.1. Család	421
24.3.2. Testesség	422
24.3.3. Alak	423
24.4. Lokális beállítás	425
24.5. Testességek kombinálása	425
24.6. Családkód deklarálása	426
24.6.1. Több családkód összevonása új kóddal	426
24.6.2. Új családkód deklarálása	426
24.7. Fontváltó csomagok	427
24.8. Fontok információi és tesztelése	428
25. LuaL^AT_EX és X_EL^AT_EX	431
25.1. Jellemzőik	431
25.2. Fontok betöltése	432
25.3. Lua szkript használata lualatex fordítóval	433
25.4. A fordító detektálása	434
26. További információk	435
26.1. A TeX Live és a MiKTeX pdf tömörítési szintje	435
26.2. A generált pdf verziószáma	435
26.3. Ha a magyar nem alapnyelvként van beállítva	435
26.4. Rendszerparancsok végrehajtása fordítás közben	436
26.5. Szöveg másolása pdf-ből	437
26.6. Hasznos csomagok	438
27. L^AT_EX-kurzus videókkal	440
28. Linkek	442
28.1. A könyvben található videók	442
28.2. Sablonok	442
28.3. T _E X-rendszerök	442
28.4. T _E X-hez fejlesztett editorok	443
28.5. L ^A T _E X oldalak	443
28.6. Leírások	443
28.7. L ^A T _E X fórumok	443
28.8. L ^A T _E X fontok	443
28.9. Segédprogramok	444
Irodalomjegyzék	445

Előszó

Ma már a világ minden felsőoktatási intézményében ismert és standardként használt kiadványszerkesztő eszköz a \LaTeX -rendszer. Ennek része a \LaTeX magas szintű dokumentumleíró nyelv is. Ez a könyv ennek a világába vezeti be az Olvasót. Az itt ismertetett tananyag alapjait az egri Eszterházy Károly Katolikus Egyetem Matematikai és Informatikai Intézetének „*Számítógépes szöveg- és kiadványszerkesztés*” című előadásai és gyakorlatai képezik, de azt jelentősen kibővítve. Ennek a könyvnek elérhető egy jóval kisebb terjedelmű kivonatos verziója is $\text{\LaTeX} \text{ LPOK}$ címmel.

A könyvet próbáltam gyakorlatias oldalról megközelíteni, de nem mindig lehet megkerülni az elméletet. Így van ez a „*Bevezetés*” című fejezetted is, ahol az Olvasó sok olyan dologgal találkozhat, amit még a gyakorlatban nem próbált ki. Erre azért van szükség, mert a későbbiekbén az itt bevezetett fogalmakat gyakran fogjuk használni.

Szeretném felhívni a figyelmet a könyvben található [videókra](#) és az Eszterházy Károly Katolikus Egyetem [\LaTeX-kurzusára](#), melyek jelentősen megkönnyíthetik az önálló tanulást.

Ha egy könyvben található példát nem tud reprodukálni a saját telepített rendszerén, akkor valószínűleg abban egy olyan kód szerepel, amihez szükség van egy csomagra vagy egy csomagfrissítésre, amely az Ön gépére még nincs telepítve.

Reményeim szerint a \LaTeX megismerése után az Olvasó természetesnek veszi majd, hogy szakdolgozatának vagy bármely más jellegű publikációjának, dokumentumának elkészítéséhez ezt a rendszert használja. Ha észrevétele, megjegyzése van, kérem írjon a következő címre:

tomacs.tibor@gmail.com

A \LaTeX -rendszer használók többmilliós taborának jelmondatával kívánok az Olvasónak sok türelmet és kitartó munkát a tanuláshoz!



A szerzőről

A \TeX -rendszerrel az 1990-es évek elején az egri Eszterházy Károly Katolikus Egyetem (akkori nevén Ho Si Minh Tanárképző Főiskola) oktatójaként találkoztam először. Egy kollégám, RIMÁN JÁNOS mutatta meg az EBERHARD MATTES által kifejlesztett [em\TeX](#) nevű \TeX -disztribúciót, amit MS-DOS, OS/2 és Windows 3.1 operációs rendszereken lehetett telepíteni. Ezen belül a Plain \TeX makrócsomagot használtam, pontosabban annak JÁRAI ANTAL és JÁRAI ZOLTÁN által újraimplementált verzióját (\LaTeX), ami magyar specifikus eszközökkel bővítette ki az eredeti rendszert. Szerkesztésre a [KEDIT](#) editort használtam.

Több cikket, egy könyvet és a doktori disszertációt is ezzel a rendszerrel írtam. Ezekben keresztül megtapasztalhattam a Plain \TeX korlátait, ami egy összetettebb dokumentum megírását rendkívül kényelmetlenebb teszi. A Plain \TeX hátránya az is, hogy tipográfiai ismeretek nélkül használva, nagy eséllyel kifogásolható kimenetet eredményez. Emiatt mai szemmel már rengeteg formai hibát látok az akkor írt műveimben.

Ezek érlelték bennem a gondolatot, hogy a strukturált művekre jóval alkalmasabb \LaTeX makrócsomagra térjek át, amely a helyes tipográfiára is sokkal nagyobb hangsúlyt fektet. A váltásra a végső lökést 2004-ben egy konferenciakiadvány elkészítésére való felkérés adta. Első lépésként búcsút vettettem az akkorra már elavultnak számító [em\TeX](#)-től, és a [MiK\TeX](#)-re tértem át, szerkesztőként pedig a [WinEdt](#) programot választottam. (Ma már leginkább a [TeX Live](#) és [TeXstudio](#) párosítást használom.) A konferenciakiadvány elkészítésére elég szűk határidő volt, így a \LaTeX megismerése is csak a legfontosabb dolgokra korlátozódott. De ezek sem szakkönyveken keresztül, hanem mások tanácsaira hagyatkozva történtek, így a különböző problémák megoldásainak helyessége is sokszor kétséges volt.

Ezután 2005-ben felkérést kaptam az [Annales Mathematicae et Informaticae](#) folyóirat műszaki szerkesztésére. Ehhez már nem volt elég az addigi \LaTeX tapasztalatom, sokkal mélyebb és hitelesebb tudásra volt szükségem. Először WETTL FERENC, MAYER GYULA és SZABÓ PÉTER \LaTeX kézikönyv [18] című leírását olvastam el. Ez nagyon jó kiindulási alapot adott, sok dolgot helyretett és kijavított a korábbi ismereteimben. Azóta rengeteg külföldi lektorált szakkönyvön rágtam át magam, továbbá a tipográfia mesterségébe is igyekeztem minél jobban elmélyülni. Az utóbbi GYURGYÁK JÁNOS *Szerzők és szerkesztők kézikönyve* [5] című leírása alapozta meg.

A felhasználói szinten túl, a fejlesztői munkában is aktívan részt veszek. A következő csomagoknak és osztályoknak vagyok a szerzője és fejlesztője: [bookcover](#), [fgruler](#), [huaz](#), [hulipsum](#), [numspell](#), [thesis-ekf](#). Ezek a csomagok a [Comprehensive TeX Archive Network \(CTAN\)](#) oldalról érhetők el, de részét képezik a \TeX Live és MiK\TeX disztribúcióknak is.

2025-ben átvettettem a [babel](#) csomag magyar nyelvi moduljának ([babel-hungarian](#)) karbantartását, mivel a szerzője, Szabó Péter 2019-től már nem foglalkozik a fejlesztésével illetve hibajavítással.

A L^AT_EX Windowson történő használatához kifejlesztettem egy könnyen és gyorsan telepíthető kompakt keretrendszeret [TeXfireplace](#) néven, amely egyszerre telepíti a MiKTeX és TeXstudio programokat, továbbá néhány fontos kiegészítő eszközt.

2009-től átvettettem az Eszterházy Károly Katolikus Egyetem Matematikai és Informatikai Intézetében a L^AT_EX oktatását. Ennek a kurzusnak az elsődleges célja, hogy a hallgatók L^AT_EX-ben tudják megírni a TDK illetve szakdolgozataikat.

Elkötelezett híve vagyok a L^AT_EX népszerűsítésének és oktatásának. Ebben a téma-
ban írt könyveim a [latex-tutorial-hu](#) oldalról töltethetők le. Ezekhez számos kiegészítő
feladat és videó is készült, melyek az előbbi könyvekből, illetve a [honlapomról](#) is elér-
hetőek.

Tómács Tibor
egyetemi docens

Jelölések

A L^AT_EX-ben a szerkesztés során ún. parancsokat kell használni. Ezek általános leírására a következő példában látható jelölést vezetjük be:

```
\textbf{<szöveg>}
```

A <> jelek közé írt rész helyére olyan kódot kell beírni, melyet az adott parancs magyarázatánál adunk meg. Például ebben az esetben a <szöveg> helyére beírt betűket ez a parancs félkövéren fogja kiszedni. Példakódot a következő módon jelöljük:

```
\textbf{ABC}
```

Ennek eredménye így lesz jelölve:

```
ABC
```

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

```
\euro \in eurosym
```

A csomagokat többféle opcióval is be lehet tölteni. Például az \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
```

Egy program menüjére [Menü] > [Almenü] > [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 linkekkel ilyen színű szöveggel jelöljük, míg a videókat a



ikonra kattintva nézheti meg. Ha egy kód mellett </> jelet lát a margón, akkor arra kattintva – internetes böngészőben a The TeXLive.net Server illetve az Overleaf segítségével – megnézheti a teljes fordítható kódot (*Minimal Working Example*) és annak eredményét.

1. fejezet

Bevezetés

DONALD ERVIN KNUTH stanfordi matematikus 1977-ben egy olyan számítógépes programot fejlesztett ki, amely a nyomdászat minden tudását képes modellezni. Tette mindenért 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 TeX-nek keresztelte el, ami egyúttal a *text* (szöveg) szóra is utal. Így a kiejtése nem *tekisz*, hanem *tekh* vagy *tek*, mint a *technika* szóban. A TeX márka jelet egy egyszerű szövegfájlba TeX módon kell beírni.

Számítástechnikai hasonlattal, a TeX nevezhető a nyomdászat assemblerének is, mellyel minden tipográfiai feladat megoldható. Ezzel azonban csak fáradságos úton, sok száz elemi parancs használatával tudunk dolgozni. Ezért szükség volt olyan makrócsomag létrehozására, mely magasabb szintű programozási nyelven, könnyebben kezelhető.

Az első ilyen makrócsomagot Knuth írta, és Plain TeX-nek nevezte el. Ennek a dokumentációját is elkészítette *The TeXbook* címmel [6]. Egy másik makrócsomagot MICHAEL SPIVAK fejlesztett ki, melyet az American Mathematical Society (AMS) támogatott, és *AMS-Tex*-nek neveztek el. Ez a fő hangsúlyt a matematikai képletek tipográfiájára helyezte. Magyar nyelven a Plain TeX és az *AMS-Tex* használatáról [3] ad rövid áttekintést. További két makrócsomag még az OpTeX és ConTeXt.



L. B. LAMPORT²



D. E. KNUTH¹

LESLIE B. LAMPORT amerikai informatikus a *Great American Concurrency Book* című könyvéhez az 1980-as évek elején tervezett makrócsomagot LATEX néven (ejtsd: *latekh* vagy *latek*). 1989-ben a stanfordi TeX-találkozó után létrejött egy munkacsoport, mely a LATEX újraírását és kiterjesztését tűzte ki célul. 1994-ben jelent meg a LATEX 2_ε (ejtsd: *latekh kettő e* vagy *latek kettő e*), ami az *AMS-Tex* tudását is magába olvasztotta. A LATEX 2_ε folyamatosan bővül ún. csomagokkal, melyek bizonyos speciális feladatok elvégzését könnyítik meg. Mára a LATEX 2_ε (a továbbiakban röviden csak LATEX) vált a legnépszerűbb makrócsomaggá. Már elérhető a LATEX3 is, amely a fejlesztők számára egy programozási nyelv, a felhasználók közvetlenül nem találkoznak vele. A LATEX márka jelet egyszerű szövegfájlba LaTeX módon kell beírni.

Amikor TeX-rendszerről beszélünk, akkor ezalatt az egész rendszert értjük, ami magába foglalja a Plain TeX, *AMS-Tex*, OpTeX, ConTeXt, LATEX makrócsomagokat,

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

² Forrás: <https://alchetron.com/Leslie-Lamport>

fontkészleteket és számos olyan dolgot, amit itt nem részletezünk, például a METAFONT nevű programot is, mely betűkészletek létrehozására alkalmas.

Évente sok ezer könyv, cikk, oktatási segédanyag, szakdolgozat, doktori disszertáció stb. jelenik meg L^AT_EX-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 L^AT_EX-rendszerben készül.

1.1. A L^AT_EX koncepciója és jellemzői

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

A L^AT_EX koncepciója szerint, a szerző a tipográfusi munka jelentős részét a L^AT_EX-re bízza, a szedési munkát pedig a T_EX végzi el. Ettől függetlenül természetesen a végső forma minden apró részlete befolyásolható, sőt saját stílusállomány is írható, de ez csak tipográfiai tudással és a L^AT_EX mélyebb ismeretével ajánlott.

A teljes T_EX-rendszer – így a L^AT_EX is – ingyenes és nyílt forráskódú program. A L^AT_EX segítségével professzionális tipográfia érhető el, beleértve a matematikai képleteket is.

Az irodalomjegyzékek, tartalomjegyzékek, szójegyzékek, lábjegyzetek és kereszthivatkozások automatikusan számozódnak, sőt a számok előtti határozott névelők (az 1., a 2.) is automatikusan kezelhetők.

A mai számítógépes programok közül a L^AT_EX 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 eredmény jelentő pdf fájl is csekély méretű, így internetes publikálásra ideális.

Ha a T_EX-et a nyomdászat assemblerének neveztük, akkor a L^AT_EX egy magas szintű dokumentumleíró nyelvnek is tekinthető. A dokumentum L^AT_EX-forrása egy szöveges állomány, mely együtt tartalmazza a kiadvány szövegét és a L^AT_EX parancsait, hasonlóan egy html dokumentumhoz, csak ott nem parancsok, hanem tagek vannak. Így a szerkesztés során nem azt látja amit a végén kap. Ez a kezdő felhasználónak hátrány, de a gyakorlat megszerzése után már előnyként fogja élvezni, mert ezáltal vizuális szerkesztésre nincs szükség, csak a tartalomra kell figyelni.

A dokumentum L^AT_EX-forrását a szerkesztés után konvertálni kell pdf fájlba. Ehhez a konvertáláshoz lesz majd szükség a T_EX-rendszerre.

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.



A T_EX szimbóluma³

³ Tervezte Duane Bibby (CTAN lion drawing by Duane Bibby; <https://ctan.org/lion>).

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 telepítő csomagja, amely Mac OS-re telepíti a TeX Live rendszert. Másik \TeX -disztribúció a szintén népszerű MiKTeX. Ez először csak Windowson működött, de ma már telepíthető Linuxra és Mac OS-re is.

1.3. \LaTeX -editorok

A szerkesztett dokumentum forrása egy szöveges állomány, ami bármelyik editoron létrehozható. Azonban célszerű olyat használni, amely a \LaTeX -re lett optimalizálva (automatikus parancs kiegészítő \LaTeX -parancsokhoz, pdf és forrás közötti szinkronizálás, automatikus hibakereső, parancssori programokhoz rendelt ikonok, stb.). Számos ilyen szerkesztő létezik: TeXworks, Kile, TeXnicCenter, WinEdt, Texmaker, TeXstudio, stb. A legjobbnak a TeXstudiót tartom, ezért ebben a könyvben ennek a telepítése és használata lesz bemutatva.

1.4. \LaTeX használata online (Overleaf)

Az **Overleaf** (<https://www.overleaf.com>) weboldal regisztrálás után internetes böngészőben ad szerkesztési lehetőséget, továbbá a végeredményt jelentő pdf fájlt egy szerencsével található TeX Live rendszer generálja. A rendszer előnyei:

- Saját gépre nem kell telepíteni \TeX -disztribúciót illetve \LaTeX -editort,
- a dokumentumokat felhőben tárolja,
- a kezelése könnyű és gyorsan átlátható,
- jól dokumentált,
- az alapszolgáltatások ingyenesek,
- az ingyenes verzióban egy másik felhasználóval megosztható a dokumentumunk (prémium tagság esetén többel is),
- rengeteg sablont tartalmaz.

Vannak hátrányai is az Overleaf használatának:

- Csak online használható,
- nem a legfrissebb TeX Live rendszerrel dolgozik,
- a TeX Live rendszerbe integrált dokumentációja a csomagoknak (texdoc) nem elérhető,
- a szerkesztő funkciói szegényesek a TeXstudióhoz képest,
- nagyobb terjedelmű dokumentumokat (mint például ez a könyv is) még a prémium verzióban sem lehet pdf-be konvertálni a fordítási idő korlátozása miatt (ingyenesen 10 másodperc, prémiumban 4 perc a limit),
- a menü és a dokumentáció nem érhető el magyar nyelven.

1.5. Terminál, parancssor

Ez a szakasz nem tartozik a \TeX -rendszer leírásához, csak arra térünk ki, hogy a továbbiakban többször emlegetett *parancssor* beviteléhez hogyan indítsa el a *terminált*.

Ehhez Windows esetén nyissa meg a „Futtatás” ablakot a + gombokkal, írja be, hogy

cmd

majd **Enter**. Linux esetén **Ctrl**+**Alt**+**T**.

Sok esetben a terminálban először be kell lépni abba a mappába, ahol éppen dolgozunk. Például, ha ez Windows esetén a

D:\dokumentumok\LaTeX próba

mappa, akkor parancssorba írja be, hogy

cd /d "D:\dokumentumok\LaTeX próba"

majd **Enter**. Linux esetén, ha a mappa például

~/Dokumentumok/LaTeX próba

akkor parancssorba írja be, hogy

cd ~/'Dokumentumok/LaTeX próba'

majd **Enter**.

1.6. Telepítés

A TeX Live rendszernek érdemes a teljes verzióját telepíteni, ugyanis a munkánk során szükségesse váló hiányzó csomagok telepítése körülményes.

MiKTeX esetén elég a basic verziót telepíteni, mert a munkánk során szükségesse váló hiányzó csomagok telepítése automatikusan a háttérben történik.

\LaTeX -editorként a TeXstudio telepítése lesz bemutatva, de más editor is választható igény szerint.

1.6.1. TeX Live és TeXstudio telepítése Windowsra

1. lépés: TeX Live telepítése. Tölts le a TeX Live telepítésvezérlőjét [innen](#) (vagy [innen](#)). Futtassa a fájlt, majd kövesse az utasításokat: **Install** → **Next** → **Install** → **Install**. A telepítés akkor fejeződik be sikeresen, ha megjelenik a „Welcome to TeX Live!” felirat. Ezután **Close** → **Close**.

Telepítési hibák elhárítása

(A) Ha az előző módszerrel nem sikerül a telepítés, akkor elköpzelhető, hogy a telepítő szerveren szinkronizálási problémák miatt nem a legfrissebb telepítő fájlok találhatóak. Ebben az esetben az előbb letöltött `install-tl-windows.exe` fájlt ne közvetlenül, hanem az

install-tl-windows.exe -select-repository

parancssorral futtassa. (Természetesen előtte a terminálban lépjen be abba a mappába, ahol az `install-tl-windows.exe` található.) Ekkor a telepítés elején ki kell választani a **Specific mirror...** gomb segítségével egy másik telepítő szervert.

(B) A leggyakrabban előforduló hiba okozója a Windows-fiók nevében található ékezetes betű vagy szóköz. Ezt a következő módon lehet megoldani. Nyissa meg a „Futtatás” ablakot a + gombokkal, írja be, hogy

notepad

majd **Enter**. A megjelenő ablakba másolja be a következő sorokat:

```
rmdir /S /Q C:\texlive
md C:\tlinst
cd /d C:\tlinst
curl -L -o inst.zip ^
https://mirror.ctan.org/systems/texlive/tlnet/install-tl.zip
tar -xf inst.zip -C C:\tlinst
for /f %%i in ('dir /B /A:D C:\tlinst') do set inst=C:\tlinst\%%i
start /min %inst%\install-tl-windows.bat
```

Mentse el egy bat kiterjesztésű fájlba, majd futtassa.

Ugyanezt elérheti úgy is, ha letölти az `install-tl.zip` fájlt [innen](#) (vagy [innen](#)), kicsomagolja egy olyan helyre, aminek az elérési útvonalában nincs ékezetes betű és szóköz, majd futtatja az `install-tl-windows.bat` fájlt.

(C) Ha így sem sikerül a telepítés, akkor töltse le a `texlive.iso` fájlt, amely tartalmazza az összes csomagot, majd kattintson rá kétszer. A megjelenő fájlkezelőben futtassa az `install-tl-windows.bat` fájlt. Ezzel a módszerrel nem a legfrissebb csomagok települnek, így ajánlott az utólagos frissítés (lásd az [1.7.1.](#) alszakaszban).

(D) 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 is oka lehet, hogy a vírusirtó nem engedi valamelyik fájl bemásolását. Ebben az esetben kapcsolja ki a vírusirtót a telepítés idejére.

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

A TeXstudio néhány praktikus beállításához mentse le a `user.txsprofile` fájlt, majd azt a TeXstudio **Beállítások** > **Profil betöltése** menüpontjával töltse be. Néhány további beállítási lehetőségről az [1.10.](#) szakaszban lesz szó.

3. lépés: A gép újraindítása. A TeX Live elérési útja a `%PATH%` környezeti változóban bizonyos esetekben akkor fog frissülni, ha újraindítja a számítógépet.

4. lépés: Telepítés ellenőrzése. Indítsa el a TeXstudiót, majd válassza ki a következő menüpontot: **Súgó** > **LaTeX-telepítés ellenőrzése**. Ha a `pdflatex.exe` elérési útja látható a 2. sorban, akkor a telepítés rendben van.



Videó: `TeX`-rendszer telepítése Windowsra

1.6.2. TeX Live és TeXstudio telepítése Ubuntu Linuxra

1. lépés: TeX Live telepítése. A legújabb TeX Live telepítéséhez töltse le az `install-tl-unx.tar.gz` fájlt és csomagolja ki. Ha szerverhiba miatt nem sikerül a letöltés, akkor próbálja ezt a [linket](#). A kicsomagolt mappában nyissa meg a terminált. Parancsorba írja be a következőt:

```
sudo ./install-tl
```

majd **Enter**. Amikor választani kell a TeX Live telepítési lehetőségei közül, akkor írja be, hogy

I

majd **Enter**. Várjon a telepítés végéig.

A TeX Live kényelmes használatához meg kell adni a `pdflatex` program elérési útvonalát. Tegyük fel például, hogy ez az

```
/usr/local/texlive/2025/bin/x86_64-linux
```

mappa. Írja parancssorba a következőt:

```
sudo gedit /etc/environment
```

majd **Enter**. Az `environment` fájlban ki kell egészíteni a

```
PATH="útvonalak"
```

sort erre:

```
PATH="/usr/local/texlive/2025/bin/x86_64-linux:útvonalak"
```

Mentse az `environment` fájlt, majd lépjen ki a `gedit` programból. Ubuntu esetén a gyökér PATH más, mint a normál PATH, ezért még ezt is be kell állítani. Parancssorba írja be:

```
sudo gedit ~/.bashrc
```

Enter, majd a `.bashrc` fájlban írja be a következő sort az első sorba:

```
alias sudo='sudo env PATH=$PATH'
```

Mentse a `.bashrc` fájlt, majd lépjen ki a `gedit` programból. Végül indítsa újra a számítógépet!

2. lépés: **TeXstudio telepítése.** A legújabb TeXstudio telepítéséhez töltse le a [TeXstudio](#) honlapjáról a megfelelő telepítőfájlt, majd indítsa el. A másik lehetőség a következő parancsok használata:

```
sudo add-apt-repository ppa:sunderme/texstudio
sudo apt-get remove texstudio-d*
sudo apt-get update
sudo apt-get install texstudio
```

Természetesen minden parancs után **Enter**. A TeXstudio néhány praktikus beállításához mentse le a `user.txsprofile` fájlt, majd azt a TeXstudio [Beállítások](#) > [Profil betöltése](#) menüpontjával töltse be. Néhány további beállítási lehetőségről az [1.10.](#) szakaszban lesz szó.

3. lépés: **Telepítés ellenőrzése.** Indítsa el a TeXstudiót, majd válassza ki a következő menüpontot: [Súgó](#) > [LaTeX-telepítés ellenőrzése](#). Ha a `pdflatex.exe` elérési útja látható a 2. sorban, akkor a telepítés rendben van.

1.6.3. MiKTeX és TeXstudio telepítése Windowsra

1. lépés: **MiKTeX telepítése.** Menjen a [MiKTeX](#) honlapjára, válassza ki a *Windows / Installer* ablakot, majd kattintson a *Download* gombra. Töltsé le a telepítőfájlt, majd indítsa el. Indítás után kövesse az utasításokat.

2. lépés: **TeXstudio telepítése.** Töltsé le a [TeXstudio telepítőjét](#), indítsa el a telepítő fájlt, majd kövesse az utasításokat.

A TeXstudio néhány praktikus beállításához mentse le a `user.txsprofile` fájlt, majd azt a TeXstudio [Beállítások](#) > [Profil betöltése](#) menüpontjával töltse be. Néhány további beállítási lehetőségről az [1.10.](#) szakaszban lesz szó.

3. lépés: **Strawberry Perl telepítése.** Amennyiben a forrásfájlok fordítását a kézőbb ismertetett `latexmk` fordításvezérlővel szeretné elvégezni, akkor szükség lesz

még egy Perl futtatóra is. A TeX Live tartalmaz ilyen futtatót, ezért ott nem kellett külön telepíteni. A MiKTeX esetében viszont szükség van rá. Ehhez menjen a [Strawberry Perl](#) honlapjára, válassza ki a *Recommended version* ablakban a megfelelő verziót. Tölts le a telepítőfájlt, majd indítsa el. Indítás után kövesse az utasításokat.

4. lépés: A gép újraindítása. A MiKTeX és a Strawberry Perl elérési útja a %PATH% környezeti változóban bizonyos esetekben akkor fog frissülni, ha újraindítja a számítógépet.

5. lépés: MiKTeX frissítése. Ajánlott a MiKTeX csomagjainak a frissítése. Ehhez a Windowson menjen a **Start** ➤ **MiKTeX** ➤ **MiKTeX Console** menüpontra. Először egy figyelmeztető ablak jelenik meg, hogy frissítenie kell a rendszert. Az **OK** megnyomása után: **Updates** → **Check for updates** → **Update now** → **OK**.

Ugyanezt elvégezheti parancssorban is:

```
miktex packages update
echo []>"%APPDATA%\MiKTeX\miktex\config\issues.json"
```

Természetesen minden parancs után **Enter**.

6. lépés: Alapértelmezett fontkészlet telepítése. A MiKTeX előbb ismertetett telepítésekor az ún. basic verzió települ, amelyben T1 belső kódolás esetén az alapértelmezett fontkészlet bittérképes, azaz a PDF kinagyításával a betűk szélei „recések” lesznek. Ezt célszerű átállítani, azaz az alapértelmezett fontkészletnek vektorgrafikus típust választani. Ehhez a Windowson menjen a **Start** ➤ **MiKTeX** ➤ **MiKTeX Console** menüpontra, majd **Packages**. Ezután válassza ki a **cm-super** sort, majd kattintson a **+** jelre (Install). Ezután **OK** → **Close**. Ugyanezt elvégezheti parancssorban is:

```
miktex packages install cm-super
```

7. lépés: Telepítés ellenőrzése. Indítsa el a TeXstudiót, majd válassza ki a következő menüpontot: **Súgó** ➤ **LaTeX-telepítés ellenőrzése**. Ha a pdflatex.exe elérési útja látható a 2. sorban, akkor a telepítés rendben van.

1.6.4. MiKTeX és TeXstudio telepítése Ubuntu Linuxra

1. lépés: MiKTeX telepítése. Futtassa a következő parancsot a terminálban:

```
curl -fsSL https://miktex.org/download/key |
sudo tee /usr/share/keyrings/miktex-keyring.asc > /dev/null
```

Az előző két sort egy sorba gépelje be. (Itt csak helyhiány miatt látható két sorban.) A következő parancsot szintén egy sorba kell begépelní:

```
echo "deb [signed-by=/usr/share/keyrings/miktex-keyring.asc]
https://miktex.org/download/ubuntu <típus> universe" |
sudo tee /etc/apt/sources.list.d/miktex.list
```

A *<típus>* helyére *focal*, *jammy* vagy *noble* kerül aszerint, hogy az Ubuntu 20.04 LTS, 22.04 LTS vagy 24.04 LTS verziójára telepít. Végül futtassa a következő két parancsot:

```
sudo apt-get update
sudo apt-get install miktex
```

Ezután nyissa meg a MiKTeX Console programot: **Finish private setup** → **OK** → **OK**.

2. lépés: TeXstudio telepítése. A legújabb TeXstudio telepítéséhez töltse le a [TeXstudio](#) honlapjáról a megfelelő telepítőfájlt, majd indítsa el. A másik lehetőség a következő parancsok használata:

```
sudo add-apt-repository ppa:sunderme/texstudio
sudo apt-get remove texstudio-d*
sudo apt-get update
sudo apt-get install texstudio
```

Természetesen minden parancs után **Enter**. A TeXstudio néhány praktikus beállításához mentse le a [user.txsprofile](#) fájlt, majd azt a TeXstudio **Beállítások** → **Profil betöltése** menüpontjával töltse be. Néhány további beállítási lehetőségről az [1.10.](#) szakaszban lesz szó.

3. lépés: A gép újraindítása. A környezeti változók bizonyos esetekben akkor fognak frissülni, ha újraindítja a számítógépet.

4. lépés: Telepítés ellenőrzése. Indítsa el a TeXstudiót, majd válassza ki a következő menüpontot: **Súgó** → **LaTeX-telepítés ellenőrzése**. Ha a **pdflatex.exe** elérési útja látható a 2. sorban, akkor a telepítés rendben van.

5. lépés: MiKTeX frissítése. A rendszer már az első két lépés után is működőképes, de ajánlott a MiKTeX csomagjainak a frissítése. Ehhez a MiKTeX Console programban **Updates** → **Check for updates** → **Update now** → **OK**.

6. lépés: Alapértelmezett fontkészlet telepítése. A MiKTeX előbb ismertetett telepítésekor az ún. basic verzió települ, amelyben T1 belső kódolás esetén az alapértelmezett fontkészlet bittérképes, azaz a PDF kinagyításával a betűk szélei „recések” lesznek. Ezt célszerű átállítani, azaz az alapértelmezett fontkészletnek vektorgrafikus típust választani. Ehhez a MiKTeX Console programban **Packages**, válassza ki a **cm-super** sort, majd kattintson a **+** jelre (Install). Ezután **OK** → **Close**. Ugyanezt elvégezheti parancssorban is:

```
miktex packages install cm-super
```

1.6.5. TeXfireplace telepítése Windowsra

A [TeXfireplace](#) a L^AT_EX Windowson történő használatához biztosít egy könnyen és gyorsan telepíthető kompakt keretrendszert, amit ennek a könyvnek a szerzője készített. A telepítő a következő programok legfrissebb verziót tölti le és telepíti:

MiKTeX (hordozható, basic verzió)

Perl A MiKTeX Perl szkriptjeinek a futtatásához (pl. **latexmk**).

Python/Pygments/latextmited A **minted** csomag használatához.

TeXstudio Hordozható verzió, módosított alapbeállításokkal. Alapértelmezetten a **latexmk** automatizálja a L^AT_EX dokumentumok fordításának folyamatát.

1.7. L^AT_EX-csomagok frissítése

A L^AT_EX-rendszert folyamatosan bővíti újabb csomagokkal, illetve a meglévőket frissítik. Ezért célszerű ezeket néha letölteni illetve frissíteni a meglévő rendszerünkön.

1.7.1. TeX Live frissítése

Ez a TeX Live Shell segítségével oldható meg, mely Windowson a **Start** ➤ **TeX Live** ➤ **TLShell TeX Live Manager** menüvel érhető el, míg Linuxon a következő parancssorral:

```
sudo tlshell
```

A megnyitott ablakban: **• Updatable** → **Update all**.

A TeX Live Manager és a csomagok frissítését parancssorból is elvégezheti. Windows esetén

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

illetve Linux esetén

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

Amennyiben a frissítés alapértelmezett szervere nem elérhető, az előbbi parancs hibát fog jelezni. Ekkor az előzőeket egészítse még ki a következő kapcsolóval is, amely keres egy működő tükörszervet:

```
-repository ctan
```

A frissítés során a frissített csomagokról biztonsági másolat készül. Ha ezt például helytakarékossági célokból nem szeretné, akkor adja ki Windows esetén a

```
tlmgr option autobackup 0
```

illetve Linux esetén a

```
sudo tlmgr option autobackup 0
```

parancsot. A TeX Live minden évben új verzióval jön ki, melynek pontos dátumáról a TeX Live honlapjáról értesülhet. Ennek telepítését az 1.6. szakaszban leírtak szerint végezheti el. Az előző verziót nem kell eltávolítani, csak ha helytakarékossági okokból muszáj.

1.7.2. MiKTeX frissítése

A MiKTeX Console programban **Updates** → **Check for updates** → **Update now** → **OK**. Ugyanezt elvégezheti parancssorban is:

```
miktex packages update
```

1.7.3. TeXfireplace frissítése

A TeXfireplace részét képező MiKTeX L^AT_EX-csomagjainak, illetve a TeXstudio frissítéséhez használja a **Start** ➤ **TeXfireplace** ➤ **TeXfireplace update** menüt.

1.8. Fontosabb fájlkiterjesztések

tex A Plain T_EX és L^AT_EX forrásfájlt jelentő szöveges állomány kiterjesztése.

bib Bibliográfiai adatbázist tartalmazó szöveges állomány kiterjesztése.

dvi (Device Independent) Dokumentumformátum, melyben 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 dvout programok képesek.

ps (PostScript) Dokumentumformátum, mely a nyomdászathoz lett optimalizálva.

eps (Encapsulated PostScript) Postscript alapú vektorgrafikus képformátum.

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

1.9. A TeX-rendszer fontosabb programjai

Az itt ismertetett programok TeXstudióból vezérelhetők, de parancssorból is futtathatók. Utóbbi esetben először nyisson egy terminál ablakot, majd lépjön abba a mappába, ahol a lefordítandó tex fájl van. A parancssori használatot csak a teljesség kedvéért írjuk le, a TeXstudióban ezek sokkal egyszerűbben elérhetők. A programok áttekintéséhez tegyük fel, hogy a L^AT_EX forrásfájlnak a **dokumentum.tex** nevet adta.

pdftex Plain TeX forrásból készít pdf fájlt. Használata parancssorból

```
pdftex dokumentum.tex
```

A **tex**, **latex** és **pdflatex** fordítók (lásd később) is ennek a fordítónak a speciális kapcsolóival érhetők el. Ezen fordítók motorja ASCII kódolású, így az ettől különböző kódolású forrásfájlok feldolgozásához szükség van olyan csomagokra, melyek a nem ASCII karaktereket parancsokra konvertálja (vagy eleve az ilyen karaktereket parancsként gépeljük be, ami a forrás olvashatóságát rontja). Ezen fordítók másik közös tulajdonsága, hogy csak a TeX-rendszerben telepített TFM (TeX Font Metric) és Type 1 típusú fontkészletek érhetők el.

tex Plain TeX forrásból készít dvi fájlt. Használata parancssorból

```
tex dokumentum.tex
```

mely a következő parancsor rövidítése:

```
pdftex -output-format=dvi dokumentum.tex
```

latex Ez egy ún. L^AT_EX-fordító, mely L^AT_EX forrásból készít dvi kiterjesztésű fájlt. Kereszthivatkozások, jegyzékek esetén többször kell futtatni. Használata TeXstudióban **Eszközök** **Parancsok** **LaTeX**, parancssorból

```
latex dokumentum.tex
```

mely a következő parancsor rövidítése:

```
pdftex -output-format=dvi -progname=latex dokumentum.tex
```

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

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

A TeXstudio alapbeállítások esetén használja ezt a kapcsolót.

pdflatex L^AT_EX-fordító, mely L^AT_EX forrásból pdf kiterjesztésű fájlt készít. A TeXstudióban ez az alapértelmezett fordító.

Ebben a könyvben alapesetben a pdflatex fordító használatát feltételezzük. Az ettől való eltérést mindig feltüntetjük.

Kereszthivatkozások, jegyzékek esetén többször kell futtatni. Használata TeXstudióban **Eszközök** **Parancsok** **PDFLaTeX**, parancssorból



pdflatex dokumentum.tex

mely a következő parancssor rövidítése:

pdftex -output-format=pdf -progname=latex dokumentum.tex

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

lualatex és **xelatex** L^AT_EX-fordítók, melyek L^AT_EX forrásból pdf kiterjesztésű fájlt készítenek. Bővebben erről a két fordítóról és a használatukról a 25. fejezetben olvashat.

biber A bib kiterjesztésű fájlbeli bibliográfiai adatbázist kezeli a **biblatex** csomag használata mellett. Használata során, először a **dokumentum.tex** fájlt fordítsa a kívánt formátumba (pdf, dvi), utána TeXstudióban **Eszközök** \gg **Parancsok** \gg **Biber** vagy parancssorban

biber dokumentum

A **dokumentum.tex** fájlt ezután ismét fordítsa le kétszer.

texindy A **xindy** egy rugalmas tárgymutató készítő rendszer (2014 óta nem fejlesztik).

A **texindy** ennek egy L^AT_EX-specifikus parancsa. A névsorba rendezés során a megadott nyelv szabályait követi. Használata során, először a **dokumentum.tex** fájlt fordítsa a kívánt formátumba (pdf, dvi), utána TeXstudióban **Eszközök** \gg **Parancsok** \gg **TexIndy** vagy parancssorban

texindy dokumentum.idx

A **dokumentum.tex** fájlt ezután ismét fordítsa le pdf-be vagy dvi-be. A **texindy**nek van néhány fontos kapcsolója is, továbbá az **indextools** csomaggal együtt könnyebben is használható, melyekről a tárgymutató készítésekor még lesz szó.

upmendex A **texindy**-nél modernebb tárgymutató készítő rendszer, mely a névsorba rendezés során a megadott nyelv szabályait követi. Használata során először a **dokumentum.tex** fájlt fordítsa a kívánt formátumba (pdf, dvi), utána parancssorban

upmendex dokumentum.idx

A **dokumentum.tex** fájlt ezután ismét fordítsa le pdf-be vagy dvi-be. Az **upmendex** az **indextools** csomaggal együtt könnyebben is használható, melyekről a tárgymutató készítésekor lesz szó.

dvips Ezzel dvi kiterjesztésű fájlokat tud konvertálni ps-be. Használata TeXstudióban **Eszközök** \gg **Parancsok** \gg **DVI->PS**, parancssorból

dvips -o dokumentum.ps dokumentum.dvi

ps2pdf Ezzel ps kiterjesztésű fájlokat lehet konvertálni pdf-be. Használata TeXstudióban **Eszközök** \gg **Parancsok** \gg **PS->PDF**, parancssorból

ps2pdf dokumentum.ps

latexmk Ez egy ún. fordításvezérlő. Meghívja a **latexmk.pl** Perl szkriptet, mely a L^AT_EX forrásfájlt a megfelelő külső programok használatával a megfelelő számban lefordítja. Ezzel egy menetben kaphat végeredményt. A megfelelő kapcsoló használatával ki tudjuk választani, hogy a **latexmk** melyik fordítót használja.

Ha **pdflatex** fordítóra van szüksége, akkor TeXstudióban használja az **Eszközök** \gg **Parancsok** \gg **Latexmk** menüpontot. Parancssorból

latexmk -pdf dokumentum

Ha `latex` fordítóra van szüksége, akkor az előbbi `-pdf` kapcsolót hagyja el vagy cserélje a `-dvi` kapcsolóra.

Ha nem akarja, hogy a fordítás minden hiba után leálljon, csak naplózza azokat, akkor használja a `latexmk` program `-interaction=nonstopmode` kapcsolóját. Ehelyett a `-silent` kapcsoló is használható, de ez még csökkenti a generált diagnosztikai adatok mennyiségét is. A `-silent` kapcsolót a TeXstudio is használja alapbeállítások esetén.

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

```
latexmk -pdf -c
```

vagy

```
latexmk -c
```

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

latexdiff 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
```

A generált `dokumentum-diff.tex` lefordításával megnézhető a különbség.

SyncTeX (Synchronize TeXnology) Nagyobb terjedelmű dokumentum esetén rengeteg munkát meg lehet spórolni, ha a tex fájl adott pozíójából a pdf fájl megfelelő pozíciójába tud ugrani és viszont. Ezt a célt szolgálja a SyncTeX program. A működéséhez használja a `pdflatex`, `latex` illetve `latexmk` programok `-synctex=1` kapcsolóját. Természetesen dvi fájl generálása esetén ennek csak akkor van értelme, ha azt konvertálja a `dvips` és `ps2pdf` 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 kattintson a tex fájlból a megfelelő szövegrészre. Ekkor a TeXstudio átugrik a pdf fájl megfelelő részére. Ez visszafelé is működik.

texdoc A TeX-rendszer dokumentációját kezelő program. Ha egy csomag útmutatóját szeretné elolvasni, akkor TeXstudióban használja a **Súgó** ▶ **Csomagleírások** menüt, vagy parancssorban

```
texdoc <csomagnév>
```

Előfordulhat, hogy parancssorral nem a megfelelő leírást kapja. Például

```
texdoc wrapstuff
```

a kínai dokumentációt találja meg. Ilyenkor először listázza ki a találatokat:

```
texdoc --list <csomagnév>
```

Ezután ki kell választani a megfelelő sorszámot.

1.10. TeXstudio beállítások

- Ha tárgymutatót készít vagy `biblatex`-het használ az irodalomjegyzékhez, akkor célszerű az alapértelmezett fordítót `pdflatex`-ről átállítani `latexmk`-ra:

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

2. A `latexmk` használata még rugalmasabbá válik például az `auto-pst-pdf` vagy az `indextools` csomagok esetén, ha használja a `-shell-escape` kapcsolót. Ehhez a következő beállítást kell elvégezni: **Beállítások** **A TeXstudio beállításai** **Parancsok**
`Latexmk [latexmk -gg -pdf -silent -synctex=1 -shell-escape %]` majd **OK**.
 Speciális beállítások megjelenítése
Fordítás A bibliográfia ellenőrzése és frissítése fordítás előtt
3. Ha a pdf-et szeretné külön ablakban megjeleníteni, és nem 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** Speciális beállítások megjelenítése
Haladó szerkesztő Zárójelpárok automatikus bezárása
5. A TeXstudio a listájában nem szereplő parancsokat kiemeli színes háttérrel. Ha ezt nem akarja, akkor: **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. 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 tudását szkriptekkel bővítheti:
Makrók **Makrók szerkesztése** **Hozzáadás**
Név (nevezze el a szkriptet)
Típus Szkript (gépelje be a szkriptet)
 - Általam használt szkriptek
 - A TeXstudio készítőinek szkriptgyűjteménye

2. fejezet

Az első lépések

2.1. A L^AT_EX alapfogalmai

2.1.1. Parancs

A L^AT_EX-ben a dokumentum formázását *parancsokkal* végezzük. A parancs \ (backslash) jellel kezdődik, majd ezt követi a parancs neve, melyben ékezes betű, szám és szóköz nem szerepelhet, továbbá kis- és nagybetű között különbséget tesz. Például

```
\LaTeX
```

```
LaTeX
```

2.1.2. Kötelező argumentum

Vannak olyan parancsok, amelyek csak bizonyos *paraméterek* megadásával működnek. Ezeket a paramétereket a parancs *argumentumába* kell beírni {} jelek közé. Például a

```
\textit{szöveg}
```

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

```
\textit{ szöveg}
```

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

```
\textit{ és még valami}
```

esetén hibát kapunk. Egy parancsnak több paramétere is lehet. Például

```
\setcounter{page}{1}
```

az oldalszámot 1-re állítja.

2.1.3. Opcionális argumentum

Egy parancsnak lehet *opciónja* is, amit nem kötelező megadni. Ha nem adja meg, akkor az *alapopció* lép érvénybe. Az opciókat a parancs *opcionális argumentumában* kell megadni [] jelek között. Például egy listaelem bevezethető az

```
\item
```

parancsal, ami az alapértelmezett jelet teszi ki a listaelem előtt, de írhat

```
\item[-]
```

parancsot is, amely egy kötőjelet tesz a listaelem előtt. Előfordulhat, hogy egy parancsnak opciója és paramétere 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 az

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

parancs betölti az `abra.jpg` képet 3 cm szélességben 90 fokkal elforgatva.

2.1.4. Környezet

A `\begin{itemize}`, `\end{itemize}` parancspárt *környezetnek* nevezzük, a kettő közötti rész pedig a *környezet belseje*. Ezen parancsok argumentumában kell a környezet nevét megadni. Például `\itemize` környezet alatt a `\begin{itemize}`, `\end{itemize}` parancspárt értjük, ami számosztan listát készít:

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

2.1.5. 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 akarja, 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 használhatja 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}
```

```
\end{itshape}
```

helyes, de helytelen a következő:

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

Tulajdonképpen egy paraméteres parancs kötelező argumentuma is blokk, pontosabban a parancs nevét követő (nem környezettel megadott) blokk tartalma lesz a parancs paramétere. minden 1 bájtos kódolású karakter blokknak számít, ezért például a

```
\textit szöveg
```

a „szöveg” szóban csak az s betűt szedi dőlten, de

```
\textit{szöveg}
```

esetén a teljes „szöveg” szó lesz dőlt. Ez csak a kötelező argumentum esetén van így. Opcionális argumentumnál kötelező megadni a szögletes zárójeleket a blokk határain.

2.1.6. Deklarációs parancs

Ha egy parancs önmagában nem jelenít meg semmit, nincs se kötelező se opcionális argumentuma, 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.

2.1.7. Komment

Ha a forrásállományba ún. *kommentet* akar elhelyezni, vagyis amit a L^AT_EX-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!
```

2.1.8. Dokumentumosztály, preambulum, dokumentumtest

A L^AT_EX forrásfájl szerkezete a következő séma szerint épül fel:

```
\documentclass[<opción>]{<dokumentumosztály>}
<preambulum>
```

```
\begin{document}
<dokumentumtest>
\end{document}
```

Elsőként egy *dokumentumosztályt* kell betölteni a `\documentclass` parancsal, 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 L^AT_EX-fordító figyelmen kívül hagyja.

2.1.9. Csomag

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

```
\usepackage[<opción>]{<csomag neve>}
```

parancsal. Például

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

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

```
\usepackage{listings}
```

esetén programkódokat tud megjeleníteni. Ha több opciót is betölt, 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 betölt, 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}
```

2.2. Fontosabb standard dokumentumosztályok

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

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

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

article Előadások, meghívók, kisebb jelentések, programdokumentációk, publikációk stb. készítéséhez. Főbb opciói:

10pt, 11pt, 12pt A dokumentum alap betűmérete. Alapopció: **10pt**

a4paper, a5paper, b5paper, letterpaper Lapméret. Alapopció az angoloknál szokványos levélpapír méret: **letterpaper**. Fontos, hogy bármelyik méretet is választja, a fizikai lapmóréret minden esetben A4 lesz, amennyiben az alapbeállításokkal telepítette a TeX-rendszert. Ezek az opciók csak a kiválasztott lapmóréretnek megfelelő margókat állítják be. Ha fizikailag is be akarja állítani a lapmóréretet, akkor a **geometry** csomagot kell használnia (lásd az 5.1. szakaszat).

oneside, twoside Egy- illetve kétoldalas szedés. Alapopció: **oneside**.

onecolumn, twocolumn Egy- illetve kéthasábos szedés. Alapopció: **onecolumn**.

notitlepage, titlepage Címlap nincs, van. Alapopció: **notitlepage**.

draft Jelzi a sorvégi túlcordulásokat és az ábráknak csak a doboza jelenik meg.

final Nem jelzi a sorvégi túlcordulásokat és az ábrákat megjeleníti. Ez alapopció.

report Beszámolók, értekezések, diplomamunkák készítéséhez használható. Az opciói ugyanazok, mint az **article** esetében. Alapértékek: **10pt, letterpaper, oneside, titlepage, final**. A részek és fejezetek ebben az osztályban minden új oldalon kezdődnek. Erre 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. Opciói megegyeznek a **report** dokumentumosztályéval. Alapértékek: **10pt, letterpaper, twoside, titlepage, openright, final**.

2.3. Az első dokumentum elkészítése

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

```
\documentclass{article}
\begin{document}
Hello World!
\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. Lépjön 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.

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

Az **article** dokumentumosztálynak az alap betűméretre három opciója van (**10pt, 11pt, 12pt**), a lapmórétre pedig többek között van egy **a4paper** opciója. Az előbb azért jelent meg a dokumentum 10 pt betűmóréttel, mert a **10pt** alapopció. Így ha például át

akar térfi A4 lapméretnek megfelelő margókra és 12 pt betűméretre, akkor az 1. sort egészítse ki az alábbi módon:

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

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

```
\documentclass[a4paper,12pt]{article}
\begin{document}
Sándor Petőfi is a famous Hungarian poet.
\end{document}
```

A TeXstudio alaphelyzetben UTF-8 kódolású fontokat használ, ugyanakkor 2018-tól a L^AT_EX alapesetben szintén UTF-8 kódolású fájlokat kezel. Emiatt az előző példa anélkül is tökéletesen működik, hogy feltüntette volna a forrásban a kódolás típusát. Azonban, 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 be kell tölteni `latin1` illetve `latin2` opcionál. Az `inputenc` használata nélkül ugyanazt a hatást érheti el, mintha betöltené azt `utf8` opcionál.

A továbbiakban mindig azt fogjuk feltételezni, hogy a forrásfájlok UTF-8 kódolással vannak elkészítve!



Ékezetes betű nemcsak 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. (Bővebben lásd a 4.1.2. alszakaszban.) Javítsa ki a 3. sort az alábbi módon:

```
\documentclass[a4paper,12pt]{article}
\begin{document}
S\'{a}ndor Pet\H{o}fi is a famous Hungarian poet.
\end{document}
```

Természetesen így nagyon körülményes ékezetes betűket begépelni, ráadásul a forrás is 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 `pdflatex` fordító az ékezetes betűket alapesetben két karakterként kezeli – alapbetű és a rátett ékezet –, ami néhány problémát fog okozni:

- Ékezetes betűket tartalmazó szótágok 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 szükség lesz a `fontenc` csomag betöltésére `T1` opcionálval, amely az úgynévezett T1 belső kódolást aktiválja (bővebben lásd a 24.2. szakaszban). Ezt írja a 2. sorba:

```
\documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\begin{document}
Sándor Petőfi is a famous Hungarian poet.
\end{document}
```

Ez még mindig nem elég a helyes elválasztás beállításához, hiszen a L^AT_EX nem tudja, hogy milyen nyelvű a dokumentum. Jelen esetben angol, amit a `babel` csomag `english` opciónjával kell a forrásban közölni (lásd a 3. sorban):

```
</> \documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\begin{document}
Sándor Petőfi is a famous Hungarian poet.
\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 [Fájl > Sablon készítése], 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[T1]{fontenc}
\usepackage[magyar]{babel}
\begin{document}
Magyar nyelvű szöveg.
\end{document}
```

A `babel` csomag `magyar` opciója betölti a `magyar.1df` fájlt, amely a magyar tipográfia megvalósításáért felelős. A `magyar.1df` első verzióját BÍRÓ ÁRPÁD és BÉRCES JÓZSEF készítették. A ma használatos jóval nagyobb tudású verziót SZABÓ PÉTER írta, de 2025-től TÓMÁCS TIBOR a fájl karbantartója. Az új `magyar.1df` úgy van beállítva kompatibilitási okok miatt, hogy alapesetben a régivel legyen egyenértékű néhány szükséges módosítást leszámítva (lásd [11]). Ahhoz, hogy az új elemek is érvényesülhessenek, a `babel` betöltése előtt át kell állítani a `magyar.1df` alapbeállításait (lásd a 3. sorban):

```
</> \documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.1df}
\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 egy hibaüzenetet kap, mert a `\Begin` parancs nincs definiálva:

Undefined control sequence. \Begin

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

Mielőtt bezárná a TeXstudiót, még egy feladatot el kell végezni. A munka elején megnyitott mappában a tex és pdf fájlokban kívül néhány munkafájl is létrejött. Többek között egy log kiterjesztésű naplófájl is, ami az esetlegesen rosszul begépelet

forráskódból származó hibákat is rögzíti. A munka végeztével ezeket érdemes törölni, amit TeXstudióból könnyen megtehet: [Eszközök](#) » [Segédfájlok törlése](#).

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



Videó: Az első L^AT_EX-dokumentum készítése

3. fejezet

A dokumentum nyelve

3.1. A babel csomag

A dokumentum nyelvét a `babel` csomag opciójaként lehet 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 ír, akkor a következő kód megfelelő:

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

Egy dokumentumon belül több nyelven is írhat. Ilyenkor a használt nyelveket a `babel` csomag opciójában vesszővel elválasztva kell felsorolni. Az utolsó lesz az alapértelmezett nyelv. Az alapnyelvről egy másik nyelvre a

```
\selectlanguage{<nyelv>}
```

parancsal térhet át. Ha csak pár bekezdés erejéig akar áttérni ideiglenesen egy másik nyelvre, akkor használja a

```
\begin{otherlanguage}{<nyelv>}
<szöveg>
\end{otherlanguage}
```

környezetet. Ha csak egy bekezdésre akar áttérni, akkor lehet használni a

```
\foreignlanguage{<nyelv>}{<szöveg>}
```

parancsot. Például a következő kódban az alapnyelv a német, amelybe beszúrtunk egy angol nyelvű részletet is:

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[english,ngerman]{babel}
\begin{document}
Deutsch Text
\foreignlanguage{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}
```

parancssal. A `defaults=hu-min` csak egy a `magyar.ldf` lehetséges opciói közül. Későbbiekben még foglalkozunk egyéb opciókkal is. Ha egyszerre több opciót használ, akkor azokat vesszővel kell elválasztani, továbbá a `defaults=hu-min` minden az első legyen a sorban. A `\PassOptionsToPackage{<opciónk>}{magyar.ldf}` helyett használható a

```
\def\magyarOptions{<opciónk>}
```

parancs is, de szintén csak a `babel` előtt. A két megoldás között annyi a különbség, hogy a `\PassOptionsToPackage{<opciónk>}{magyar.ldf}` parancsot többször kiadva, minden egyik opció érvényesülni fog, míg a második megoldást többször alkalmazva, csak az utolsónak beírt `\def\magyarOptions{<opciónk>}` opciói érvényesülnek.

3.2. A szavak elválasztása

A L^AT_EX alapból sorkizártan szedi a szöveget, így a sorvégi szavak elválasztása hosszabb szövegek esetén elkerülhetetlen. Amikor beállította a nyelvet, akkor a szavak nagy részét helyesen el fogja tudni választani a L^AT_EX, de teljesen ezt nem lehet automatizálni. Például a „karóra” szó esetében kétféle szótárolás is lehetséges, aszerint, hogy mit jelent: kar-ó-ra vagy ka-ró-ra.

Ebben a szakaszban leírtakat ne az összes forrásbeli szóra alkalmazza. Egyrészt feszleges, másrészt a forrást olvashatatlaná tenné. Csak a dokumentum megírásának legvégén nézze meg a sorvégi elválasztásokat, és a helytelen eseteket javítsa!



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

```
Már nem volt a szarkánál a kar\-\-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 szótárolása kar-ó-ra, akkor `kar\-\-ra` helyett ez is írható:

```
kar\_óra
```

Ekkor a `_` jel mutatja, hogy hol van a szóösszetétel határa, így a L^AT_EX helyesen tudja elválasztani.

Amennyiben a dokumentumban van egy többször is használatos szó, amit a L^AT_EX rosszul választ el, akkor célszerű még a preambulumban beállítani a helyes elválasztást, nem minden 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 szótárolását:

```
\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 téл-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 L^AT_EX 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 az **[AltGr]+ 7** gombokkal:

```
me`nnyi
```

Ekkor, ha a magyar nyelv aktív, a L^AT_EX 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 „időhoszsabítás” szó elválasztását adtuk meg. A helyes megoldás a következő:

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

Ezután pedig időho`ssabítás módon beírva mindenéppen jó elválasztást kapunk.

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

```
egyszer`-kétszer
```

Ekkor a kötőjelnél és minden szótagnál el tud választani. Amennyiben nem aktív a magyar nyelv, akkor az előző megoldás helyett használja a hyphenat csomag \hyp{} parancsát. Például

```
electromagnetic\hyp{}endoscopy
```

Ha kötőjelnél választ el, akkor a kötőjelet a következő sor elején nem ismétli meg. Ha mégis szükség van erre, mert ki akarja hangsúlyozni a kötőjel szerepét, akkor használja a ` | kódöt, ami az ún. fontos kötőjelet jelenti. De ez csak akkor fog működni, ha a magyar nyelv aktív. Például

```
nátrium` | klorid
```

A magyar.ldf alapbeállítás esetén egy szót csak akkor hajlandó elválasztani automatikusan egy lehetséges ponton, ha az elválasztás előtt és után is legalább 2 betű van a szóban. Például a „fáraó” szót csak „fá-raó” módon választja el, mert „fára-ó” esetén az elválasztás után csak egy betű van. Az „arany” eszerint soha nem lesz elválasztva, mert az egyetlen lehetséges „a-rany” elválasztás esetén az elválasztás előtt csak egy betű van. Ha ezt a korlátozást fel akarja oldani, akkor használja a magyar.ldf **hyphenmins=11** opciját:

```
\PassOptionsToPackage{defaults=hu-min,hyphenmins=11}{magyar.ldf}
```

Ebben az első 1 számjegy azt jelenti, hogy ennyi betűnek kell lenni legalább a szóban az elválasztás előtt, míg a második 1 számjegy azt jelenti, hogy ennyi betűnek kell lenni legalább a szóban az elválasztás után.

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 sorkizáras miatt nem lehet optimálisan tördelni a bekezdéseket!

Amennyiben a jobb olvashatóság kedvéért az oldalak utolsó sorában le akarja tiltani a szavak elválasztását (Pontosabban a szóelválasztással végződő sorok után le akarja tiltani az oldaltörést), akkor használja a következő parancsot:

```
\brokenpenalty10000
```

Ennek a módszernek a hátránya, hogy az oldaltörések optimalizálását nehezíti.

Ha szeretné megnézni, hogy egy szót hogyan tud elválasztani a L^AT_EX, akkor használja a

```
\showhyphens{\szólista}
```

parancsot, amelybe azoknak a szavaknak a listája kerül szóközökkel elválasztva, melyek elválasztási pontjaira kívánCSI. A fordítás után a naplófájlban nézheti meg az eredményt, illetve TeXstudio használata esetén a naplópanelen.

A szakasz elején szó volt róla, hogy csak a dokumentum megírása után kell ellenőrizni a sorvégi elválasztásokat, és a helytelen eseteknél kell korrigálni az ismertetett módszerekkel. Ezt a munkát – főleg nagy terjedelmű dokumentum esetén – jelentősen megkönnyíti a következő módszer. A preambulumba írja be a

```
\tracingparagraphs1
```

parancsot, majd fordítsa le a dokumentumot. Ezután, ha például a tex fájl neve dokumentum.tex, akkor futtassa a következőt parancssorban:

```
findhyph dokumentum.log
```

Ez létrehoz egy dokumentum.hyph fájlt, amiben listázásra kerülnek az elválasztott szavak az elválasztásuk módjával együtt.

3.3. Sorvégi túlcordulás

Ha a L^AT_EX nem tudja megoldani sorvégén egy szó elválasztását, akkor ún. sorvégi túlcordulás jöhet létre, melyre a következő üzenetet figyelmeztet a naplófájlban:

```
Overfull \hbox (...pt too wide) in paragraph at lines ...
```

TeXstudio használata esetén ez a naplópanelen látható. Ha a pdf-ben is be akarja jelölni a túlcordulási pontokat, akkor gépelje a következőt a preambulumba:

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

Túlcordulás akkor is létrejöhét, 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 túlcordulások úgy szűnnek meg, hogy az adott sorban a szóközök túl nagyok lesznek. Ez a megoldás leginkább csak keskeny oszlopokba íráskor indokolt. A \sloppy hatása a

```
\fussy
```

parancssal szüntethető meg.

3.4. A magyar.ldf aktív karakterei

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

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

„idézet” “idézet”



Később látni fogjuk, hogy a fordított aposztófjel aktív szerepe kikapcsolható. Ebben az esetben az előző megoldás a magyar nyitó idézőjelre nem fog működni! Ezért inkább a 4.3.7. alszakaszban leírtakat alkalmazza.

Magyar nyelv esetén is írhat közvetlenül (azaz az angol nyelv aktívvá tétele nélkül) angol nyitó idézőjelet ` ' módon (`AltGr`+ `7` és `Shift`+ `1`):

```
\documentclass{article}
\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 elő kell tenni a `\string` parancsot. Ezzel a megoldással is lehet magyar nyelv esetén közvetlenül angol nyitó idézőjelet írni:

```
\documentclass{article}
\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 kottaírásra 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 a kettőspont, pontosvessző, kérdőjel és felkiáltójel előtt rövid szóköz, ún. *spáciump* álljon. A forrásfájlban ezen jelek előtt nem kell külön gondoskodni a spáciumról, mert a `magyar.ldf` ezen jelek aktívvá tételevel ezt automatikusan megoldja. 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:

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

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

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

Az előző parancsokban a \string elhagyható, ha a ` jel nem aktív.

4.1.2. Ékezetes betűk

Korábban már volt róla szó, hogy a forrásállomány kódolásának beállítása után ékezetes betű közvetlenül a billentyűzetről is bevihető. De mi van akkor, ha olyan stílusfájlt ír, melyben a kódlap kiválasztását a felhasználóra bízza, vagy olyan ékezetes betűre van szükség, amely nincs a billentyűzeten? Ilyenkor használhatja az ékezet parancsokat.

ó \'{o}	ò `{o}	ó \={o}	ő \v{o}	ő \k{o}	őo \t{oo}
ő \H{o}	ô `^{o}	ó \.{o}	ő \rf{o}	ő \d{o}	
ö \"{o}	ő \~{o}	ő \u{o}	ő \c{o}	ő \b{o}	

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 érhet el: i, j. Ezzel már le tudja írni az í betűt \H{\i} módon. Ez alól a két gyakrabban előforduló í és ī 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: L ï d ï. A TeXstudióban minden ékezet parancs elérhető innen: [Oldalpanel > Szimbólumok > Speciális](#).

4.1.3. Speciális betűk

Œ \OE	Å \AA	æ \ae	Ø \Ø	í \i	Ł \L
œ \oe	Æ \AE	ß \ss	ø \o	ј \j	ł \l

További, csak T1 belső kódkészlet esetén használható speciális betűk:

D \DH	D \DJ	IJ \NG	P \TH
ð \dh	đ \dj	ŋ \ng	þ \th

4.1.4. Kerning

A *kerning* vagy más néven *alávágás* célja két egymás melletti betű igazítása oly módon, hogy a betűközök optikailag egyformának tűnjenekek. A következő példán keresztül ez érthetővé válik. Az első sor kerning segítségével készült, a második pedig anélkül:

AVATÁR
AVATÁR

A kerning a L^AT_EX-ben automatikusan működik. Ha le akarja tiltani adott helyen egy adott betűre, akkor azt tegye kapcsos zárójelek közé. Az előző példa második sora például így írható:

{A}V{A}T{Á}R

4.1.5. Ligatúrák

Ligatúra a betűknek a szokásosnál szorosabb összekötése. A legismertebbek a következő ún. f-ligatúrák:

ff fi fl ffi ffi

ff fi fl ffi ffi

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

4.1.6. 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 TeXstudióban [Súgó] > [Csomagleírások], ahol be kell írni a `symbols-a4` szót.

€	<code>\euro</code>	∈	<code>\eurosym</code>	؂	<code>\textvisiblespace</code>
£	<code>\pounds</code>	؄	<code>\textcent</code>	؅	<code>\textperthousand</code>
⌚	<code>\textcircledP</code>	⌚	<code>\textregistered</code>	؆	<code>\textpertenthousand</code>
⌚	<code>\textcopyright</code>	⌚	<code>\textcopyleft</code>	؇	<code>\textcelsius</code>
⌚	<code>\textbullet</code>	⌚	<code>\textasteriskcentered</code>	؈	<code>\textleaf</code>
†	<code>\dag</code>	⌚	<code>\textopenbullet</code>	؉	<code>\S</code>
‡	<code>\ddag</code>	⌚		؊	<code>\P</code>
*	<code>\textasteriskcentered</code>	⌚		؋	<code>\textnumero</code>
•	<code>\textbullet</code>	⌚		،	<code>\textreferencemark</code>
◦	<code>\textopenbullet</code>	⌚		؍	<code>\textcircled{}</code>

A `wasysym` csomag is sok szimbólumot tartalmaz. Ezek elérhetők a TeXstudióból is: [Oldalpanel] > [Szimbólumok] > [Egyéb szöveg] és [Oldalpanel] > [Szimbólumok] > `wasysym`. További lehetőség még többek között az `utfsym`, `fontawesome`, `fontawesome5`, `simpleicons` és `pifont` csomagok használata. Utóbbival úgynevezett PostScript jeleket írathat ki a

`\ding{kódszám}` ∈ `pifont`

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

﴿ 33	﴾ 34	﴿ 35	﴿ 36	﴿ 37	﴿ 38	﴿ 39	﴿ 40	﴿ 41	﴿ 42	﴿ 43	﴿ 44	﴿ 45	﴿ 46	﴿ 47	﴾ 48	﴾ 49	﴾ 50	﴾ 51	﴾ 52	﴾ 53	﴾ 54	﴾ 55	﴾ 56	﴾ 57	﴾ 58	﴾ 59	﴾ 60	﴾ 61	﴾ 62	﴾ 63	﴾ 64	﴾ 65	﴾ 66	﴾ 67	﴾ 68	﴾ 69	﴾ 70	﴾ 71	﴾ 72	﴾ 73	﴾ 74	﴾ 75	﴾ 76	﴾ 77	﴾ 78	﴾ 79	﴾ 80	﴾ 81	﴾ 82	﴾ 83	﴾ 84	﴾ 85	﴾ 86	﴾ 87	﴾ 88	﴾ 89	﴾ 90	﴾ 91	﴾ 92	﴾ 93	﴾ 94	﴾ 95	﴾ 96	﴾ 97	﴾ 98	﴾ 99	﴾ 100	﴾ 101	﴾ 102	﴾ 103	﴾ 104	﴾ 105	﴾ 106	﴾ 107	﴾ 108	﴾ 109	﴾ 110	﴾ 111	﴾ 112	﴾ 113	﴾ 114	﴾ 115	﴾ 116	﴾ 117	﴾ 118	﴾ 119	﴾ 120	﴾ 121	﴾ 122
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

• 123	③ 174	⑩ 191	⑦ 208	→ 225	» 243
• 124	④ 175	① 192	⑧ 209	➢ 226	↘ 244
“ 125	⑤ 176	② 193	⑨ 210	➢ 227	⤵ 245
” 126	⑥ 177	③ 194	⑩ 211	➢ 228	⤵ 246
⌚ 161	⑦ 178	④ 195	➔ 212	➡ 229	⤵ 247
⌚ 162	⑧ 179	⑤ 196	→ 213	➡ 230	⤵ 248
⌚ 163	⑨ 180	⑥ 197	↔ 214	▶ 231	⤵ 249
⌚ 164	⑩ 181	⑦ 198	↑ 215	➡ 232	⤵ 250
⌚ 165	① 182	⑧ 199	↘ 216	⇒ 233	⤵ 251
⌚ 166	② 183	⑨ 200	➔ 217	⇒ 234	⤵ 252
⌚ 167	③ 184	⑩ 201	⤵ 218	⇒ 235	⤵ 253
♣ 168	④ 185	① 202	➔ 219	⇒ 236	⤵ 254
♦ 169	⑤ 186	② 203	➔ 220	⇒ 237	
♥ 170	⑥ 187	③ 204	➔ 221	⇒ 238	
♠ 171	⑦ 188	④ 205	➔ 222	⇒ 239	
① 172	⑧ 189	⑤ 206	⤵ 223	⇒ 241	
② 173	⑨ 190	⑥ 207	⤵ 224	⇒ 242	

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
```

\TeXkézikönyv

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

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

\TeX kézikönyv, \TeX kézikönyv, \TeX 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

4.3.1. Pont, vessző, kettőspont, pontosvessző, kérdőjel, felkiáltójel

Ezek elő 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 L^AT_EX 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 elő 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.

4.3.2. 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;`

4.3.3. Nagykötőjel

A nagykötőjel (angolul en dash) forrásállományban -- (két kötőjel) módon 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 elő 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

4.3.4. Gondolatjel, kvírtmínusz

A gondolatjel forrásállományban -- (két kötőjel) módon 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.

Gondolatjelként az angolban a kvírtmínusz vagy más néven az „em dash” (—) jel is használható, de ez előtt és után nem szabad szóközt rakni. Magyarban tilos a használata. A kvírtmínusz forrásban --- (három kötőjel) módon írandó.

4.3.5. Zárójelek

Zárójelek előtt és után is szóköz áll, kivéve, ha írásjel követi.

4.3.6. Hármaspont

A hármaspont forrásállományban a \dots parancsal 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...)

4.3.7. 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 idézeten 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 szövegeben 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. Tölts be a `csquotes` csomagot `autostyle` opcióval. Ez a csomag a következő nyelvek idézőjeleit ismeri: croatian, danish, dutch, english, finnish, french, german, greek, italian, norwegian, polish, portuguese, russian, spanish, swedish. A magyart 2019. májusától ismeri (v5.2e), így ettől régebbi verzió használata esetén a `csquotes` csomag betöltése után a preambulumba gépelje a következőt:

```
\DeclareQuoteStyle{magyar}{,,}{''}{>>}{<<}
```

Ezután az

```
\enquote{<szöveg>} \enquote{<szöveg>} <szöveg> ∈ 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*{<szöveg>} ∈ csquotes
```

parancsot. Az `\enquote` és `\enquote*` parancsok argumentumában használható több bekezdésből álló szöveg is.

Minden típusú idézőjel elérhető parancsként is. Ezeket soroljuk fel a következőben, jelezve az alapvonalhoz viszonyított elhelyezkedését is.

<code>—, \quotesinglbase</code>	<code>__” \textquotedblright</code>	<code>___» \guillemotright</code>
<code>—' \textquoteright</code>	<code>__“ \textquotedblleft</code>	<code>___« \guillemotleft</code>
<code>—‘ \textquotyleft</code>	<code>__> \guilsinglright</code>	
<code>—,, \quotedblbase</code>	<code>__< \guilsinglleft</code>	

Tehát magyar idézőjel a következő módon is írható:

```
\quotedblbase{}szöveg\textquotedblright{}
```

4.4. Betűváltozatok

4.4.1. Osztályozás

A betűváltozatokat családjuk, testességük és alakjuk szerint osztályozzuk.

Család (family)

1. Antikva (roman)

```
\textrm{<szöveg>}
{\rmfamily <szöveg>}
```

2. Groteszk (sans serif)

```
\textsf{<szöveg>}
{\sffamily <szöveg>}
```

3. Írógép (typewriter)

```
\texttt{<szöveg>}
{\ttfamily <szöveg>}
```

Az írógép betűváltozat esetén a szavakat nem választja el a L^AT_EX, mert azt legtöbbször programkódok írására használják. Ez a korlátozás feloldható a `hyphenat` csomag `htt` opciójával vagy az írógép betűtípusra állítás után beírt

```
\hyphenchar\font=\defaulthyphenchar
```

kóddal. A családok jellemzői:

	talpas	vonalvastagság	betűszélesség
antikva	igen	változó	változó
groteszk	nem	állandó	változó
írógép	igen	állandó	állandó

Alapesetben az antikva család az alapértelmezett. Például

```
antikva \textrm{antikva} \textsf{groteszk} \texttt{írógép}
```

```
antikva antikva groteszk írógép
```

Testesség (series)

1. Normál (medium)

```
\textmd{<szöveg>}
{\mdseries <szöveg>}
```

2. Félkövér (boldface)

```
\textbf{<szöveg>}
{\bfseries <szöveg>}
```

Alapesetben a normál testesség az alapértelmezett. Például

```
normál \textmd{normál} \textbf{félkövér}
```

```
normál normál félkövér
```

Alak (shape)

1. Álló (upright)

```
\textup{<szöveg>}
{\upshape <szöveg>}
```

2. Kis/nagybetű (upper/lower case)

```
\textulc{<szöveg>}
{\ulcshape <szöveg>}
```

3. Döntött (slanted)

```
\textsl{<szöveg>}
{\slshape <szöveg>}
```

4. Dőlt (italics)

```
\textit{<szöveg>}
{\itshape <szöveg>}
```

5. Kiskapitalis (small caps)

```
\textsc{<szöveg>}
{\scshape <szöveg>}
```

Alapesetben az álló alak az alapértelmezett. Például

```
álló \textup{álló} \textsl{döntött} \textit{dőlt} \textsc{Kiskapitalis}
```

álló álló döntött dőlt KISKAPITÁLIS

Ha az aktuális fontkészlet támogatja, akkor két alak kombinálható is. Például, ha a T1 belső kódkészlet használata esetén alapértelmezett *European Computer Modern* fontkészletről áttérünk a *Latin Modern* fontkészletre az `\modern` csomag betöltésével, akkor elérhető a döntött kiskapitalis alak is:

```
{\slshape\scshape Döntött kiskapitalis}
```

DÖNTÖTT KISKAPITÁLIS

Fontos, hogy ebben az esetben az `\upshape` illetve `\textup` parancsok csak az álló alakot állítják vissza, a kiskapitalis alakot nem befolyásolják. Például

```
{\slshape\scshape Döntött kiskapitalis. \upshape Álló kiskapitalis.}
```

DÖNTÖTT KISKAPITÁLIS. ÁLLÓ KISKAPITÁLIS.

Ha az előző példában nem a döntött alakot akarja visszaállítani, hanem a kis/nagybetűs alakot, akkor használja a `\textulc` vagy `\ulcshape` parancsot. Például

```
{\slshape\scshape Döntött kiskapitalis. \ulcshape Döntött kis/nagybetüs.}
```

DÖNTÖTT KISKAPITÁLIS. Döntött kis/nagybetűs.

Korábban láttuk, hogy az `\upshape` illetve `\textup` parancsok a döntött vagy dőlt kiskapitalisból álló kiskapitalist generálnak. Ugyanakkor álló kiskapitalisból álló kis/nagybetűs alakot kapunk. Például

```
{\scshape Álló kiskapitalis. \upshape Álló kis/nagybetüs.}
```

ÁLLÓ KISKAPITÁLIS. Álló kis/nagybetűs.

Ha az alapértelmezett alakra akar visszatérni (álló kis/nagybetűs), akkor használja a `\normalshape`

parancsot, amely ekvivalens az `\upshape\ulcshape` kóddal. Például

```
\{ \slshape \scshape Döntött kiskapitális. \normalshape Álló kis/nagybetűs. }
```

DÖNTÖTT KISKAPITÁLIS. Álló kis/nagybetűs.

Nemcsak két alak, hanem a család, testesség és alak is kombinálható. Például

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

szöveg

Amikor vissza akar arra térti az alap betűváltozatra, akkor használja a

```
\textnormal{\textcolor{green}{szöveg}}
{\normalfont \textcolor{blue}{szöveg}}
```

parancsokat. A `\textup`, `\textsl`, `\textit` stb. parancsokat több bekezdésre nem lehet alkalmazni. Az `\upshape`, `\slshape`, `\itshape` stb. deklarációs parancsok, így használhatók környezetként is. Például

```
{\bfseries \textcolor{green}{szöveg}}
```

és

```
\begin{bfseries}\textcolor{green}{szöveg}\end{bfseries}
```

hatása ugyanaz. A fonttípusok beállításáról a 24. 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 rádőlné az állóra. Ezt nevezik *kurzív kiegyenlítésnek*. Ennek illusztrálására a következő mondatot először kurzív kiegyenlítés nélkül, majd pedig kurzív kiegyenlítéssel szedtük ki:

„Éhes zsíráf fogyasztja épp ízes uzsonnáját.”

„Éhes zsíráf fogyasztja épp ízes uzsonnáját.”

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

```
Éhes \textit{zsíráf} fogyasztja épp ízes uzsonnáját. \\
```

```
Éhes \textsl{zsíráf} fogyasztja épp ízes uzsonnáját.
```

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

```
Éhes {\itshape zsíráf\V} fogyasztja épp ízes uzsonnáját. \\
```

```
Éhes {\slshape zsíráf\V} fogyasztja épp ízes uzsonnáját.
```

4.4.3. Kiemelés

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

```
\emph{\textcolor{green}{szöveg}}, {\em \textcolor{green}{szöveg}}, \begin{em}\textcolor{green}{szöveg}\end{em}
```

parancsokat illetve környezetet. (Az első megoldás több bekezdésre nem használható.) Standard dokumentumosztályok esetén ezek figyelik az aktuális betűváltozatot, 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 `\V` parancssal nekünk kell megoldani. Például

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

Kiemelésre lehetőleg ne használja a félkövér típust, mert az a címekre van fenntartva. Az `\emph` illetve `\em` parancsok stílusát a következő parancssal lehet beállítani:

```
\DeclareEmphSequence{\langle 1. szint\rangle,\langle 2. szint\rangle,\langle 3. szint\rangle,...}
```

A szintek hatásai halmozódnak. Például

```
\DeclareEmphSequence{\sffamily,\slshape,\rmfamily}
\emph{Első, \emph{második, \emph{harmadik}}} szint.
```

Első, második, harmadik szint.

Az írógépek korábban betűritkítással emeltek ki. Ez L^AT_EX-ben is megoldható:

```
\so{\langle szöveg\rangle} \in soul
```

Ha a telepített `soul` csomagjának verziója 3.0-nál régebbi, akkor `soul` helyett használjon `soulutf8` csomagot. 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:

<u>szöveg</u>	<code>\underline{\langle szöveg\rangle}</code>	<u>szöveg</u>	<code>\xout{\langle szöveg\rangle} \in ulem</code>
<u>szöveg</u>	<code>\uline{\langle szöveg\rangle} \in ulem</code>	<u>szöveg</u>	<code>\cancel{\langle szöveg\rangle} \in cancel</code>
<u>szöveg</u>	<code>\uuline{\langle szöveg\rangle} \in ulem</code>	<u>szöveg</u>	<code>\bcancel{\langle szöveg\rangle} \in cancel</code>
<u>szöveg</u>	<code>\uwave{\langle szöveg\rangle} \in ulem</code>	<u>szöveg</u>	<code>\xcancel{\langle szöveg\rangle} \in cancel</code>
<u>szöveg</u>	<code>\sout{\langle szöveg\rangle} \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 nagybétűs szövegeben csupa kisbetűvel szedésük.

```
\MakeUppercase{\langle szöveg\rangle}
```

A `\langle szöveg\rangle`-et csupa nagybetűvel szedi ki.

```
\MakeLowercase{\langle szöveg\rangle}
```

A `\langle szöveg\rangle`-et csupa kisbetűvel szedi ki.

```
\MakeTextUppercase{\langle szöveg\rangle} \in textcase
```

A `\langle szöveg\rangle`-et csupa nagybetűvel szedi ki, de a matematikai képletek betűin nem változtat.

```
\MakeTextLowercase{\langle szöveg\rangle} \in textcase
```

A `\langle szöveg\rangle`-et csupa kisbetűvel szedi ki, de a matematikai képletek betűin nem változtat.

```
\NoCaseChange{\szöveg} ∈ textcase
```

Nem változtat a betűkön.

4.5. Betűméretek

A T1 belső kódkészlet alapból a *European Computer Modern* fontkészlet tölti be, melyben 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. Ez a korlátozás feloldható az `anyfontsize` csomaggal. Az `anyfontsize` csomagnál nagyobb tudású a `fix-cm`, de ennek betöltése még a `\documentclass` parancs előtt történik

```
\RequirePackage{fix-cm}
```

módon. A European Computer Modern fontkészlet helyett használhat másat is. Ehhez töltse be például az `lmodern`, `mlmodern`, `pxfonts`, `kpfonts`, `txfonts`, `newtxttext`, `times`, `lxfonts`, `bera`, `cyklop` csomagok valamelyikét a `fontenc` csomag előtt. Az ezekben található fontok minden méretben használhatók. Új fontok betöltéséről bővebben a 24. fejezetben olvashat.

4.5.1. Alapbetűméret

Az alapbetűméretet 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, akkor írja a preambulumba a `fontenc` csomag betöltése után, hogy

```
\usepackage[fontsize=\betűméret]{scrextend}
```

vagy

```
\usepackage[fontsize=\betűméret]{fontsize}
```

Ha az alapértelmezett European Computer Modern fontkészletet használja, akkor ne felejtse el feloldani a méretkorlátozt az `anyfontsize` csomaggal. A dokumentum tetszőleges pontján is át lehet állítani az alapbetűméretet:

```
\KOMAoptions[fontsize=\betűméret] ∈ scrextend
```

vagy

```
\changefontsizes{\betűméret} ∈ scrextend
```

vagy

```
\changefontsize{\betűméret} ∈ fontsize
```

Egy másik lehetőség tetszőleges alapbetűméret beállítására az 5.3. szakaszban tárgyalt `geometry` csomag `mag` opciója.

4.5.2. Betűméretet beállító deklarációs parancsok

A következő parancsok az alapbetűmérettől függően állítanak be betűméretet:

szöveg	<code>\tiny szöveg</code>
szöveg	<code>\scriptsize szöveg</code>
szöveg	<code>\footnotesize szöveg</code>
szöveg	<code>\small szöveg</code>

szöveg	<code>\normalsize szöveg</code>
szöveg	<code>\large szöveg</code>
SZÖVEG	<code>\Large szöveg</code>
SZÖVEG	<code>\LARGE szöveg</code>
SZÖVEG	<code>\huge szöveg</code>
SZÖVEG	<code>\Huge szöveg</code>

	alapbetűméret		
	10pt	11pt	12pt
<code>\tiny</code>	5 pt	6 pt	6 pt
<code>\scriptsize</code>	7 pt	8 pt	8 pt
<code>\footnotesize</code>	8 pt	9 pt	10 pt
<code>\small</code>	9 pt	10 pt	10.95 pt
<code>\normalsize</code>	10 pt	10.95 pt	12 pt
<code>\large</code>	12 pt	12 pt	14.4 pt
<code>\Large</code>	14.4 pt	14.4 pt	17.28 pt
<code>\LARGE</code>	17.28 pt	17.28 pt	20.74 pt
<code>\huge</code>	20.74 pt	20.74 pt	24.88 pt
<code>\Huge</code>	24.88 pt	24.88 pt	24.88 pt

Ezek deklarációs parancsok, így használhatók környezetként is. Például a

```
{\large szöveg}
```

és

```
\begin{large}szöveg\end{large}
```

kódok hatása ugyanaz.

4.5.3. Relatív betűméretek

Tetszőleges relatív betűméret is beállítható:

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

ahol az `<arányszám>` azt adja meg, hogy az alapbetűméretnek hányszorosát szeretné. Például

```
{\scalefont{2.5}szöveg}
```

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

4.5.4. Abszolút betűméretek

Abszolút betűméretet a következő parancssal érhet el:

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

Például 25 pontos szöveget 30 pontos sortávolsággal így lehet írni:

```
\fontsize{25}{30}\selectfont
```

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

Erre a példára többféle módszerrel is hivatkozhatunk, de ez a leggyakrabban használt.

Ha a sortávolságot meg akarja hagyni alapméréten, akkor `\the\baselineskip` helyére parancsot írja.

4.6. Térközök

A L^AT_EX minden nyomdászatban használatos mértékegységet ismer. Most csak néhányat sorolunk fel:

`pt` pont

`mm` milliméter

`cm` centiméter

`in` inch, 1 in = 25,4 mm = 72,27 pt

`ex` Az aktuális betűalakzatban az x betű magassága, amit általában függőleges méretek megadásához használnak.

`em` Az aktuális betűalakzat által definiált méret, amit általában vízszintes méretek megadásához használnak. (Régebben az aktuális M betű szélességét jelentette, de ez ma már nem igaz.)

4.6.1. Fix méretű vízszintes térközök

Vízszintes helykihagyás méretét a következő parancssal adhatja meg:

`\hspace{<térköz mérete>}`

A `\hspace{<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 hagyja ki az adott méretű helyet.

`\hspace*{<térköz mérete>}`

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

`\kern{<térköz mérete>}`

További vízszintes méretű helykihagyások:

<code>\,</code>	% = \hspace{0.33333em} (6/18em) □ a szóközt jelöli
<code>\enskip</code>	% = \hspace{0.5em}
<code>\quad</code>	% = \hspace{1em}
<code>\quad</code>	% = \hspace{2em}
<code>\, = \thinspace</code>	% = \kern0.16667em (3/18em)
<code>\! = \negthinspace</code>	% = \kern-0.16667em (-3/18em)
<code>\: = \medspace</code>	% = \kern0.2222em (4/18em)
<code>\negmedspace</code>	% = \kern-0.2222em (-4/18em)
<code>\; = \thickspace</code>	% = \kern0.2777em (5/18em)
<code>\negthickspace</code>	% = \kern-0.2777em (-5/18em)
<code>\,</code>	% = \kern0.33333em (6/18em)
<code>\enspace</code>	% = \kern0.5em

4.6.2. Rugalmas méretű vízszintes térközök

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

```
\stretch{<rugóerő>}
```

Ennek működése a következő példán érhetővé válik:

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

A

B

C

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

```
\fill % = \stretch{1}
```

```
\hfill % = \hspace{\fill}
```

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

```
A\hrulefill B
```

```
C\dotfill D
```

```
E\rightarrowfill F
```

```
G\leftarrowfill H
```

A

B C D E

→F G←

H

Az előbbiek általánosításaként, például = jelekkel a következő módon lehet kitölteni a térközt:

```
A\leaders\hbox{=}\hfill B
```

A

B

Rugalmas térköz a következő módon is megadható:

```
\hspace{<térköz mérete>} plus <plusz> minus <mínusz>}
```

Például

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

A 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 $12 - 2 = 10$ -től $12 + 4 = 16$ pontig.

4.6.3. Fix méretű függőleges térközök

Függőleges helykihagyás méretét a következő parancssal 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 kihagyja az adott méretű térközt.

Ha a `\vspace` illetve `\vspace*` parancsokban megadott *(térköz mérete)* nagyobb, mint amekkora hely még van a szövegtükör aljáig, akkor a különbség nem jelenik meg a következő oldal tetején. Amennyiben erre lenne szükség (pl. egy olyan feladatsor esetén, amelyben a feladatok után adott méretű helyet szeretnénk kihagyni a megoldásnak, akár oldaltöréssel is), akkor használja az

```
\xvspace{<terköz mérete>}
```

parancsot, amit a preambulumban kell definiálni a következő módon:

```
\newlength{\vspacex}
\newcommand{\xvspace}[1]{%
  \par
  \ifdim\dimexpr\pagegoal-\pagetotal<#1
    \setlength{\vspacex}{\dimexpr#1+\pagetotal-\pagegoal}%
    \pagebreak
    \vspace*{\vspacex}
  \else
    \vspace*{#1}
  \fi
}
```

További parancsok:

```
\lower{terköz mérete}\hbox{<szöveg>}
```

A *(térköz mérete)* azt adja meg, hogy a *(szöveg)* mennyivel legyen lejjebb, mint az alapvonal.

```
\textsuperscript{<szöveg>}
```

A *(szöveg)* felső indexbe kerül `\scriptsize` méretben.

```
\textsubscript{<szöveg>}
```

A *(szöveg)* alsó indexbe kerül `\scriptsize` méretben. Például

```
xxx\lower0.5ex\hbox{xxx}
17\textsuperscript{h}
H\textsubscript{2}O
```

xxxxxx 17^h H₂O

4.6.4. Rugalmas méretű függőleges térközök

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

```
\linespread{<szorzó>}
```

parancsot, ami az alapértelmezett sortávolságot megszorozza a *<szorzó>* értékével. Az írógépeknél használt másfeles illetve kettes sorközökhöz tartozó szorzó függ az alap be-tűmérettől:

	10 pt	11 pt	12 pt
másfeles	1.25	1.213	1.241
kettes	1.667	1.618	1.665

Másfeles sorköz a

```
\onehalfspacing \setspace
```

parancssal, illetve kettes sorköz a

```
\doublespace \setspace
```

parancssal állítható be. De pl. a normál sortávolság háromszorosa is beállítható a

```
\setstretch{3} \setspace
```

parancssal. A *setspace* és *hyperref* csomagok együttes használatánál a *setspace* előbb legyen betöltve.

4.7. Törések

4.7.1. Sortörések

A L^AT_EX automatikusan végzi a sortöréseket, de adott esetben ki is kényszerítheti azt:

```
\\"
```

Új sort kezd sorkizáras nélkül.

```
\\" [<méret>]
```

Ugyanaz mint a \\ de a következő sor távolsága *(méret)*-tel megnő. Például

```
\\" [2mm]
```

```
\\"*
```

Ugyanaz mint a \\ de nem enged meg oldaltörést.

```
\\"* [<méret>]
```

Ugyanaz mint \\ [<méret>] de nem enged meg oldaltörést.

```
\linebreak
```

Új sort kezd sorkizárással.

```
\nolinebreak
```

A sortörést letiltja az adott helyen.

4.7.2. Oldaltörések

A L^AT_EX maga végzi az oldaltöréseket. Ha azt akarja, hogy a telített oldalak alja egy-máshoz igazított legyen, akkor használja a

`\flushbottom`

parancsot. Ennek hatása a

`\raggedbottom`

parancssal 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`

A `\newpage` parancstól 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{<térköz>}`

Az aktuális oldal függőleges méretét `<térköz>`-zel megnöveli, de az előláb helyzetét nem igazítja hozzá. Például

`\enlargethispage{3mm}`

`\enlargethispage*{<térköz>}`

Ugyanaz mint * nélkül, 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 töltse 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.

További parancsok:

```
\indent
```

Kikényszeríti az adott bekezdés elején a behúzást.

```
\noindent
```

Letiltja az adott bekezdés elején a behúzást.

```
\setlength{\parindent}{<terköz>}
```

A bekezdés behúzásának mértékét átállítja `<terköz>` méretre.

```
\setlength{\parskip}{<terköz>}
```

Két bekezdés közötti térközt megnöveli `<terköz>` mérettel. Például

```
\setlength{\parskip}{5pt}
```

```
\hangindent=<terköz>
```

```
\hangafter=<szám>
```

Ez a két parancs csak a soron következő bekezdésre hat. Ha a `<terköz>` értéke pozitív, akkor a bekezdés bal margója `<terköz>` értékkel megnő. Ha a `<terköz>` értéke negatív, akkor a bekezdés jobb margója a `<terköz>` abszolút értékével nő meg. Ha a `<szám>` értéke 0, akkor a `\hangindent` parancs a bekezdés minden sorára vonatkozik. Ha a `<szám>` értéke pozitív egész, akkor a `\hangindent` parancs a bekezdés első `<szám>` darab sorára nem vonatkozik. Ha a `<szám>` értéke negatív egész, akkor a `\hangindent` parancs csak a bekezdés első $-\langle szám \rangle$ darab sorára vonatkozik. Tehát például

```
\hangindent=2cm
```

```
\hangafter=-3
```

esetén a következő bekezdés első 3 sorának bal margója megnő 2 cm-rel, illetve

```
\hangindent=-2cm
```

```
\hangafter=3
```

esetén a következő bekezdés jobb margója megnő 2 cm-rel, kivéve az első 3 sort.

```
\parfillskip=<terköz1> plus <terköz2>
```

A bekezdések utolsó sorának vége és a jobb margó közötti távolság minimum `<terköz1>`, maximum `<terköz1> + <terköz2>` lesz.

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

Képzeljen 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

Képzeljen 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. Például

</>

```
\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.''\\\n\\space*\{\fill\}(Rejtő Jenő)\\end{quotation}

„Örökös vigyora nemegyszer tévedésbe ejtette azokat, akik kissé könnyelműen, a külsejük után ítélik meg embertársaikat, és ezért a vi-gyorgó Jimmyt feliületesen 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 előttem, köd mögöttem,\nisten tudja, honnan jöttem,\nszél hozott, szél visz el,\nminek kérdjem: mért visz el?

Sose néztem, merre jártam,\na felhőknek kiabáltam,\nerdő jött: jaj, be szép!\n-- megcibáltam üstökét.

\end{verse}

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

Köd előttem, köd mögöttem,
isten tudja, honnan jöttem,
szél hozott, szél visz el,
minek kérdjem: 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 \in [magyar] babel

parancs. A kimondott szövegbe gondolatjelek között 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` fájl `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 érhetővé válik:

00000000000000	11111111111111
	11111111 2222222222
	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 egyel megnöveli a lábjegyzet sorszámát. Ha a

```
\footnote[\langle szám \rangle]{\lábjegyzet szövege}}
```

parancsot használja, akkor a lábjegyzet száma nem nő, hanem az a szám íródik ki, amit a *\langle szám \rangle* opcionálisan 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` opcionálisan a lábjegyzetek fölé nem tesz vízszintes vonalat. Ha mégis szeretne tenni, akkor írja be a következőt:

```
\footnotestyle{rule=fourth} \in [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} \in perpage
```

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

A lábjegyzetek számozását átállíthatja csillagosra a következő parancssal:

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

Visszaállítani arab számozásra így lehet:

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

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

```
\editorfootnote{\szerkesztő megjegyzése} \in [magyar]babel
```

Ez csillagos számozást használ és oldalanként újra indul.

Szintek (rész, fejezet, szakasz stb. lásd később) címében tipográfiaileg 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{\labbjegyzet 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 változatlanul 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{\labbjegyzet szövege}
```

Szöveget ír a lábjegyzetbe, de nem változtatja meg a lábjegyzet számát és az adott helyre nem teszi ki a lábjegyzet jelét.

```
\footnotetext[<szám>]{\labbjegyzet szövege}
```

Szöveget ír a lábjegyzetbe `<szám>` alatt, de nem változtatja meg a lábjegyzet számát és az adott helyre nem teszi ki a lábjegyzet jelét.

Ha egy lábjegyzetre többször is hivatkozni szeretne, akkor használja a [85.](#) oldalon található megoldást.

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 a következő parancsot:

```
\interfootnotelinepenalty=10000
```

4.11. Széljegyzetek

Széljegyzeteket a

```
\marginpar{\széljegyzet}
```

parancssal 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 minden 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
```

parancssal lehet.

Kétoldalas szedés esetén a széljegyzetek hol bal, hol jobb oldalon lesznek. Ha azt akarja, hogy a bal oldalra kerülve a széljegyzet más képpen 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 előtt kell tenni egy `\leavevmode` parancsot, különben a széljegyzet és a bekezdés első sora között szintküllönbség lép fel. Ez azért van így, mert a `\marginpar` nem kezdi új bekezdést.

A `\marginpar` parancs használata bizonyos esetekben nem lehetséges. Ilyen eset például a később ismertetett dobozokban való használata. Ekkor lehet megoldás a `\marginnote` parancs, amire ezek a korlátozások nem vonatkoznak. Ez a parancs ugyanúgy használható mint a `\marginpar`.

4.12. Színek kezelése

4.12.1. Színmodellek és paraméterek

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

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

`HTML` paramétere a szín hatjegyű hexadecimális kódja. (Lásd például [itt](#).) Az első két számjegy az `RGB` kód első paramétere 16-os számrendszerben, a következő két számjegy az `RGB` kód második paramétere 16-os számrendszerben, végül az utolsó két számjegy az `RGB` kód harmadik paramétere 16-os számrendszerben. Például, ha a szín `RGB` kódja 186,85,211, akkor a `HTML` kódja BA55D3.

`cmyk` 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.

`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.

Az `RGB`, `rgb` és `HTML` paletták lényegüket tekintve ugyanazt jelentik, csak a három alapszín mennyiségeinek megadási módja más-más. A monitor színkeverése az `RGB` paletta szerint történik, de a nyomdában `cmyk` palettát használnak. Ezért, ha a szerkesztett dokumentumot elektronikus publikációnak szánja, azaz monitoron kell majd elolvasni, akkor az `xcolor` csomag által kikevert színeket is ennek megfelelően kell előállítani. Ehhez használja az `xcolor` csomag `rgb` opcióját. Ha a dokumentum nyomdába kerül, akkor használja az `xcolor` csomag `cmyk` opcióját.

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 parancssal:

```
\definecolor{<színnév>}[<modell>]{<színpáraméter>} ∈ xcolor
```

Például

```
\definecolor{halvanyeszurke}{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 $\langle szám \rangle \% \langle színnév1 \rangle$ színhez $(100 - \langle szám \rangle) \% \langle színnév2 \rangle$ 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 % fehérét. Ezzel a technikával akár több adott nevű szín összekeveréséből is megadhat újabb színnevet. Például

```
green!30!yellow!20!black
```

esetén 30 % zöldhöz kevertünk 20 % sárgát, a maradék 50 % pedig fekete.

Színnevet definiálhat korábban definiált színnévvvel is a következő két kód valamelyikével, melyek ekvivalensek egymással:

```
\colorlet{új színnév}{régi színnév} ∈ xcolor
\definecolor{új színnév}{named}{régi színnév} ∈ xcolor
```

Például

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

Amennyiben egy színnév komplementerét szeretné megadni, akkor egy kötőjelet kell elő írni. Így például a `-yellow` a sárga komplementer színét jelenti.

4.12.3. Színes szöveg

Szövegek színezéséhez a következő parancsokat használhatja:

```
\textcolor{<modell>}{<színpáraméter>}{<egy bekezdés>} ∈ xcolor
\textcolor{<színnév>}{<egy bekezdés>} ∈ xcolor
{\color{<modell>}<színpáraméter>}{<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. Átlátszóság

Szövegek aktuális színének átlátszóságát a következő parancsokkal állíthatja be:

```
\texttransparent{<paraméter>}{<egy bekezdés>} ∈ transparent
\transparent{<paraméter>}{<több bekezdés>} ∈ transparent
```

A *<paraméter>* értéke egy 0 és 1 közötti szám. A 0 jelenti a teljesen átlátszót az 1 pedig a nem átlátszót. Például

```
\color{red}\texttransparent{0.5}{Szöveg}
```

4.12.5. Szöveg kiemelése színes háttérrel

Ehhez az **xcolor** csomag mellett használja a **soul** csomagot is. Ekkor a következő parancsokat használhatja:

```
\sethcolor{<színnév>} ∈ soul
\hl{<szöveg>} ∈ soul
```

Ha a telepített **soul** csomagjának verziója 3.0-nál régebbi, akkor **soul** helyett használjon **soulutf8** csomagot. A **\sethcolor** parancs megadja a kiemelés színét. Alapértelmezése **yellow**. A **\hl** színezi ki a *<szöveg>* hátterét, 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!

Ha **soulutf8** csomagot kénytelen használni és az **xcolor** csomagot valamelyen paletta opcióval töltötte be (**rgb**, **cmyk**, **stb.**), akkor az előző kód hibás eredményt ad. Ennek megoldásához a következő kódot írja be a **soulutf8** betöltése után (forrás [itt](#)):

```
\usepackage{regexpatch}
\makeatletter
\xpatchcmd*\{\SOUL@ulunderline}{\dimen@}{\SOUL@dimen}{}{}
\newdimen\SOUL@dimen
\makeatother
```

4.12.6. Szöveg kiemelése színes aláhúzással

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

```
\setul{<mélység>}{<vonavastagság>} ∈ soul
\setulcolor{<színnév>} ∈ soul
\ul{<szöveg>} ∈ soul
```

Ha a telepített `soul` csomagjának verziója 3.0-nál régebbi, akkor `soul` helyett használjon `soulutf8` csomagot. A `\setul` parancs `<mélység>` paramétere beállítja, hogy az aláhúzás mennyivel legyen az alapvonal alatt, a `<vonavastagság>` pedig, hogy milyen vastag legyen a vonal. A `\setulcolor` parancs `<színnév>` paramétere megadja az aláhúzás színét. Alapértelmezése `black`. Az `\ul` parancs húzza alá a `<szöveg>` részt, ami akár több sorból, vagy akár több bekezdésből is állhat. Például

```
</>   \setul{1pt}{1pt}
      \setulcolor{blue}
      \ul{Ez egy fontos szöveg, azért van aláhúzva kékkel!}
```

Ez egy fontos szöveg, azért van aláhúzva kékkel!

Ha `soulutf8` csomagot kénytelen használni, az `\ul` parancssal ugyanaz a probléma amit a `\hl` parancsnál ismertettünk, és a megoldása is ugyanaz.

4.12.7. Szöveg kiemelése színes áthúzással

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

```
\setstcolor{<színnév>} ∈ soul
\st{<szöveg>} ∈ soul
```

Ha a telepített `soul` csomagjának verziója 3.0-nál régebbi, akkor `soul` helyett használjon `soulutf8` csomagot. A `\setstcolor` parancs `<színnév>` paramétere megadja az áthúzás színét. Alapértelmezése `black`. Az `\st` parancs húzza át a `<szöveg>` részt, ami akár több sorból, vagy akár több bekezdésből is állhat. Például

```
</>   \setstcolor{blue}
      \st{Ez a szöveg át van húzva kékkel.}
```

Ez a szöveg át van húzva kékkel.

Ha `soulutf8` csomagot kénytelen használni, az `\st` parancssal ugyanaz a probléma amit a `\hl` parancsnál ismertettünk, és a megoldása is ugyanaz.

4.12.8. Színes lapok

A lap háttérszíne így adható meg:

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

Alapállapotba visszatérni a következő parancssal lehet:

```
\nopagecolor ∈ xcolor
```

4.13. Dátumtípusok

Ha a dokumentum fordításának dátuma 2025. október 4., akkor

2025	\number\year
10	\number\month
4	\number\day
2025. október 4.	\today (ha a magyar nyelv aktív)
October 4, 2025	\today (ha az angol nyelv aktív)
2025-10-04	\emitdate{a}{\today} ∈ [magyar]babel
2025. október 4.	\emitdate{b}{\today} ∈ [magyar]babel
2025. okt. 4.	\emitdate{c}{\today} ∈ [magyar]babel
2025. X. 4.	\emitdate{d}{\today} ∈ [magyar]babel
2025. 10. 04.	\emitdate{e}{\today} ∈ [magyar]babel
2025. október	\emitdate{f}{\today} ∈ [magyar]babel
2025. október 4	\emitdate{g}{\today} ∈ [magyar]babel
2025 október	\emitdate{h}{\today} ∈ [magyar]babel

Rögzített dátumok esetén:

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

A következő parancs kiírja, hogy a fordítás időpontja a hét melyik napjára esik.

\weekday ∈ eukdate

Csak angol verziója van. Ha magyarul akarja használni, akkor írja be a következőt a preambulumba az eukdate csomag betöltése után:

```
</> \makeatletter
      \renewcommand\weekday{%
        \ifcase\theeuk@date szombat\or vasárnap\or hétfő\or
        kedd\or szerda\or csütörtök\or péntek\fi}
      \makeatother
```

A következő parancs kiírja, hogy a fordítás dátumához képest *(nap)* múlva mi a dátum.

\DayAfter[*(nap)*] ∈ advdate

A *(nap)* alapértéke 1.

4.14. Számírás

Az 5 vagy annál több jegyű egész számokat ezres csoportosítással kell leírni. A csoporthoz a jobbról balra történik. Nem kell csoportosítani a 4 jegyű egész számokat, kivéve abban az esetben, ha egy táblázat olyan oszlopában található, amelyben szerepel 4-nél több jegyű egész szám is. Így lehet elérni, hogy a megfelelő számjegyek mindenkorral legyenek.

A csoportosító jel a magyarban a feles törhetetlen szóköz („,) vagy a pont, az angolban pedig a vessző. Tehát például

Helyesen 1\,234\,567 vagy 1.234.567, de 9999.

Helyesen 1 234 567 vagy 1.234.567, de 9999.

Az ezres csoportosítás automatizálható a

`\num{<szám>} ∈ siunitx`

parancssal. Például

</> `\num{1234567}`

1 234 567

A `\num` parancs csak a 4-nél több jegyű egész számokra alkalmazza az ezres csoportosítást. Ha az előzőekben leírt kivétel esetén szükséges a 4 jegyű számok ezres csoportosítása is, akkor alkalmazza az `siunitx` csomag `group-minimum-digits=4` opcióját. Ha ezt az opciót csak egy blokkon belül lokálisan szeretné bekapcsolni, akkor használja a következő parancsot a blokk elején:

`\sisetup{group-minimum-digits=4}`

Ha ezres csoportosítójelnek nem az alapértelmezett feles törhetetlen szóközt, hanem például a pontot szeretné, akkor használja az `siunitx` csomag `group-separator={.}` opcióját. Ekkor

</> `\num{1234567}`

1.234.567

Lehetőség van egész számok automatikus kibetűzésére is a

`\numspell{<szám>} ∈ numspell`
`\Numspell{<szám>} ∈ numspell`
`\anumspell{<szám>} ∈ numspell`
`\Anumspell{<szám>} ∈ numspell`

parancsokkal. Például

</> `\numspell{1234567} \\`
`\Numspell{1234567} \\`
`\anumspell{1234567} \\`
`\Anumspell{1234567}`

egymillió-kétszázharmincnégyezer-ötszázhatvanhét
Egymillió-kétszázharmincnégyezer-ötszázhatvanhét
az egymillió-kétszázharmincnégyezer-ötszázhatvanhét
Az egymillió-kétszázharmincnégyezer-ötszázhatvanhét

Ha sorszámot kell kibetűzni, akkor használja az

`\ordnumspell{<szám>} ∈ numspell`
`\Ordnumspell{<szám>} ∈ numspell`
`\aordnumspell{<szám>} ∈ numspell`
`\Aordnumspell{<szám>} ∈ numspell`

parancsokat. Például

</>

```
\ordnumspell{1234567} \\
\Ordnumspell{1234567} \\
\aordnumspell{1234567} \\
\Aordnumspell{1234567}
```

egymillió-kétszázharmincnégyezer-ötszázhatvanhetedik
 Egymillió-kétszázharmincnégyezer-ötszázhatvanhetedik
 az egymillió-kétszázharmincnégyezer-ötszázhatvanhetedik
 Az egymillió-kétszázharmincnégyezer-ötszázhatvanhetedik

A `\numspell`, `\Numspell`, `\ordnumspell` és `\Ordnumspell` parancsok ismerik az angol, német, francia, olasz és magyar nyelvet is. Ha a `babel` csomaggal ezen nyelvek valamelyikét töltötte be, akkor a számok betűzése is eszerint a nyelv szerint történik. Ellenkező esetben angolul. Például

</>

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[english,ngerman]{babel}
\usepackage{numspell}
\begin{document}
\numspell{1234567} \\
\selectlanguage{english}\numspell{1234567}
\end{document}
```

eine Million zweihundertvierunddreißigtausendfünfhundertsiebenundsechzig
 one million, two hundred and thirty-four thousand, five hundred and sixty-seven

4.15. Mértékegységek

Mértékszám és mértékegység közé minden feles törhetetlen szóközt kell tenni. Például

</>

```
123\,cm 1200\,km 50\,\%, 1000\,Ft 500\$ 10\,\AA
```

123 cm 1200 km 50 % 1000 Ft 500 \$ 10 Å

vagy

</>

```
20\,\textcelsius
1\,\textperthousand
5\,\textpertenthousand
```

20 °C 1 ‰ 5 ‰

Ez alól a szabály alól egy kivétel van, amikor szöveget írunk fokban, percben és másodpercben. Ekkor nincs a mértékszám után térköz. Ennek írását legkönnyebben az

`\ang{<fok>;<perc>;<másodperc>}` ∈ siunitx

parancssal oldhatja meg. Például

</>

```
\ang{1;;} \ang{2;3;} \ang{4;5;6}
```

1° 2°3' 4°5'6"

A mértékszám és mértékegység közötti térköz automatizálható az

`\qty[opción]{szám}{mértékegység} ∈ siunitx`

parancssal. Például

```
</> \qty{1}{\meter}
      \qty{2}{\centi\meter}
      \qty{3}{\angstrom}
      \qty{4}{\ohm}
      \qty{5}{\percent}
      \qty{6}{\degreeCelsius}
```

1 m 2 cm 3 Å 4 Ω 5 % 6 °C

Itt nemcsak a mértékegység jelét, hanem a nevét is be lehet írni. Azaz például `\angstrom` az `\AA` helyett, vagy `\percent` a `\%` helyett. Az elérhető mértékegységek nevei a `siunitx` csomag leírásában megtalálhatóak. Nézzünk még néhány példát:

```
</> \qty{1}{\centi\meter\tothe{2}}\\
      \qty{1}{\coulomb\per\mole}\\
      \qty[per-mode=symbol]{1}{\coulomb\per\mole}\\
      \qty[per-mode=fraction]{1}{\coulomb\per\mole}\\
      \qty[per-mode=fraction]{1}{\meter\per\second\tothe{2}}
```

1 cm²
 1 C mol⁻¹
 1 C/mol
 1 $\frac{\text{C}}{\text{mol}}$
 1 $\frac{\text{m}}{\text{s}^2}$

Az `siunitx` csomag használata esetén tizedes törtek beírásánál a tizedesvessző és a tizedes pont használata is megengedett, de mindenkorban a kimenetben tizedes pont lesz. Azaz például

```
</> \qty{10.7}{\centi\meter} = \qty{0,107}{\meter}
```

10.7 cm = 0.107 m

Ahhoz, hogy tizedesvessző legyen a kimenetben, használja az

```
</> output-decimal-marker={,}
```

opción. Ekkor az előző kód kimenete

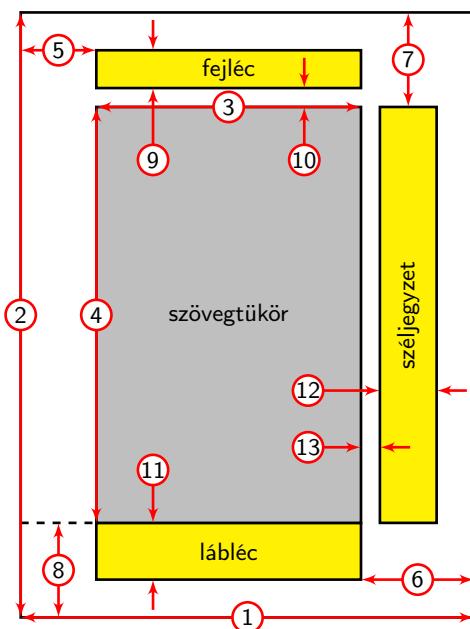
10,7 cm = 0,107 m

5. fejezet

Oldalak kinézete

5.1. Oldalak szerkezete és méretek

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



Az ábrán jelölt méreteket a `geometry` csomag opcióival állíthatja be:

- (1) `paperwidth=<méret>` Oldal szélessége.
- (2) `paperheight=<méret>` Oldal magassága.
- (3) `textwidth=<méret>` Szövegtükör szélessége.
- (4) `textheight=<méret>` Szövegtükör magassága.
- (5) `inner=<méret>` 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=<méret>` Külső margó (bel-ső margóval ellentétes oldalon) szélessége.
- (7) `top=<méret>` Felső margó magassága.
- (8) `bottom=<méret>` Alsó margó magassága.
- (9) `headheight=<méret>` Fejléc magassága.
- (10) `headsep=<méret>` Fejléc és szövegtükör távolsága.

- (11) `footskip=<méret>` Lábléc magassága.
- (12) `marginparwidth=<méret>` Széljegyzet területének szélessége.
- (13) `marginparsep=<méret>` Széljegyzet és szövegtükör távolsága.

Alapértelmezésben a szövegtükör méretei az oldal méreteinek 70 %-a, az alsó margó a felső másfélszerese, egyoldalas szedés esetén a belső és külső margó megegyezik, illetve kétoldalas szedés esetén a külső margó a belső másfélszerese.

Ha ISO szabvány méretet akar (A0–A6, B0–B6, C0–C6), akkor az `a0paper`, ..., `a6paper`, `b0paper`, ..., `b6paper`, `c0paper`, ..., `c6paper` opciók valamelyikét kell betölteni. A B sorozatban létezik JIS (Japanese Industrial Standards) szabvány is, ezek a `b0j`, ..., `b6j` opciókkal érhetők el. 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 opciót parancsban is meg lehet adni:

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

Például

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

Ha egy dokumentumon belül az oldal geometriáját néhány oldal erejéig át akarja állítani, akkor használja a

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

parancsot. Ezzel 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. Lapméret megváltoztatása dokumentumon belül

Említettük, hogy a lapméret nem állítható át dokumentumon belül a `\newgeometry` parancssal. Ha mégis szükség lenne rá, akkor használhatja a következőt az átállítási pont előtt:

```
\newstocksize{layoutsizes={<szélesség>,<magasság>},<egyéb opciók>} ∈ stocksize
```

Ezután a lapméret szélessége illetve magassága `<szélesség>` illetve `<magasság>` méretű lesz. A szélesség és magasság megadása helyett szabványos lapmóret is megadható. Például B5 méret esetén a `layoutsizes` opció helyett használja a

```
layout=b5paper
```

opción. Az `<egyéb opciók>` nélkül

- a szövegtükör mérete az oldalméret 70 %-a,
- a bal és jobb oldali margók egyenlőek egyoldalas szedés esetén,
- a külső margó másfélszerese a belső margónak kétoldalas szedés esetén,
- az alsó margó a felsőnek a másfélszerese,
- minden más paraméter megegyezik az eredeti oldal paramétereivel.

Ha ezek nem felelnek meg, akkor az `\restorestocksize` helyén a `geometry` csomag opcióival állíthatja be a megfelelő paramétereket. Visszatérni az eredeti lapmérethez a

```
\restoresize \stocksize
```

parancssal lehet.

5.3. 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 betölve, 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 hosszméret is megváltozik a `mag` értékének ezredszároson.

Ha valamely mértékegységgel megadott hosszméretet nem akarja, hogy a nagyítás során megváltozzon, akkor a mértékegység előre tegye a `true` szót (`truemm`, `truecm`, `truept`). Ha nagyításnál a beállított szabványos oldalméretet nem akarja, hogy változon, 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 sortávolság nélkül 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[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\begin{document}
Első sor.
\par\vspace{2truecm}
Második sor.
\end{document}
```

5.4. Többhasábos szedés

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

```
\begin{multicols}{<hasábszám>} ∈ multicol
<szöveg>
\end{multicols}
```

A `<hasábszám>` maximum 10 lehet. A hasábok közötti távolság 10 pt. Ennek átállítása például 1 cm-re:

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

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

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

5.5. 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 fej- 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 `pdflscape` 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, de a fej- és lábléc függőlegesen.

Ha a szövegtükör és széljegyzet tartalmával együtt a fej- és lábléc tartalmát is el akarja forgatni, továbbá szeretné a pdf nézőben az oldalt fizikailag is elforgatni, akkor továbbra is használja `landscape` környezetet, de az `lscape` vagy `pdflscape` csomagok betöltése helyett írja a következőket a preambulumba:

```
\usepackage{stocksizer}
\makeatletter
\newlength{\original@layoutwidth}
\newlength{\original@textwidth}
\setlength{\original@layoutwidth}{\Gm@layoutwidth}
\setlength{\original@textwidth}{\textwidth}
\newenvironment{landscape}{}{}
\AddToHook{env/landscape/before}{%
    \newstocksizer
    \layoutsize = {\Gm@layoutheight,\original@layoutwidth},
    \text      = {\textheight,\original@textwidth},
}
\AddToHook{env/landscape/after}{\restorestocksizer}
\makeatother
```

Ekkor az elforgatott oldalon a szövegtükör méretei felcserélődnek a normál oldalhoz képest. A bal és jobb oldali margók egyenlőek egyoldalas szedés esetén, illetve a külső margó másfélszerese a belső margónak kétoldalas szedés esetén, továbbá az alsó margó másfélszerese a felső margónak. minden más paraméter változatlan marad.

Ha azt szeretné, hogy az elforgatott oldal minden margója rendre megegyezzen a normál oldal margóival, akkor `text={...}` helyett használja a `keepmargins` opciót. Ekkor az elforgatott oldalon a szövegtükör ezekhez fog igazodni, ezért a méretei nem feltétlenül egyeznak meg a normál oldal szövegtükörének elforgatásával kapott méretekkel. Egyéb igény esetén a `text={...}` és `keepmargins` opciók helyett a margók és egyéb paraméterek beállítására a `geometry` csomag opciót használhatja.

5.6. Vágójelek nyomdai előkészítéshez

Egy pdf dokumentumot a nyomdában először a dokumentum méreténél nagyobb lapra nyomtatják ki, utána pedig levágják a széleit a megfelelő méretre. Ezzel lehet garantálni a margók illetve a függőleges és a vízszintes irányok pontosságát. Ehhez a nyomdai előkészítés során vágójeleket kell elhelyezni a lapra.

Ezt a munkát könnyen elvégezheti a `geometry` csomaggal. Például, ha azt akarja, hogy a vágás utáni méret 176 mm széles és 250 mm magas legyen, akkor a `geometry` csomagot a következő opciókkal töltse be (kiegészítve a margókra és egyéb méretekre vonatkozó opciókkal):

```
\usepackage[
    paperwidth = 206mm, % papír szélessége 176mm + 30mm
    paperheight = 280mm, % papír magassága 250mm + 30mm
    layoutwidth = 176mm, % vágás után a szélesség
    layoutheight = 250mm, % vágás után a magasság
    layoutoffset = 15mm,
    showcrop,           % mutassa a vágójeleket
    ...                % egyéb opciók helye
]{geometry}
```

A `layoutoffset=15mm` azt jelenti, hogy a pdf oldalon a felső vízszintes vágójelek a lap tetejétől, illetve a bal oldali függőleges vágójelek a lap bal szélétől 15 mm távolságokra helyezkedjenek el. Ahhoz, hogy az alsó és a jobb oldali vágójelek is raférjenek a lapra, a papír szélességének és magasságának a `layoutoffset` értékének legalább a duplájával nagyobbnak kell lenni, mint a vágás utáni méretnek. Ezért van megadva a papírméretek 30 mm-rel nagyobb érték, mint a vágás utáni méretek.

Az előbbi helyett standard méreteket is használhat. Például

```
\usepackage[
    paper = a4paper,      % nyomtatás A4-es papírra
    layout = b5paper,     % vágás után B5 lesz a lapméret
    layoutoffset = 15mm,
    showcrop,             % mutassa a vágójeleket
    ...                  % egyéb opciók helye
]{geometry}
```

Ha a pdf nem tartalmaz vágójeleket, és ennek pótlására a forrást nem szeretné vagy nem áll módjában szerkeszteni az előző módszerrel, akkor készítse el a következő tartalmú tex fájlt, tegye mellé a vágójelekkel ellátandó pdf fájlt, majd fordítsa le a tex fájlt.

```

3 \documentclass{minimal}
4 \usepackage{pdfpages,tikz}
5 \setlength{\paperwidth}{<pdf szélessége>+30mm}
6 \setlength{\paperheight}{<pdf magassága>+30mm}
7 \begin{document}
8 \AddToShipoutPictureFG{\begin{tikzpicture}[overlay]
9   \draw ([yshift= 15mm] current page.south west) -- +(10mm,0);
10  \draw ([xshift= 15mm] current page.south west) -- +(0,10mm);
11  \draw ([yshift= 15mm] current page.south east) -- +(-10mm,0);
12  \draw ([xshift=-15mm] current page.south east) -- +(0,10mm);
13  \draw ([yshift=-15mm] current page.north west) -- +(10mm,0);
14  \draw ([xshift= 15mm] current page.north west) -- +(0,-10mm);
15  \draw ([yshift=-15mm] current page.north east) -- +(-10mm,0);
16  \draw ([xshift=-15mm] current page.north east) -- +(0,-10mm);
17  \node at ([yshift=-5mm] current page.north) {<információ>};
18 \end{tikzpicture}}
19 \includepdf[noautoscale,pages=1-]{<pdf fájl>}
20 \end{document}

```

Fordítás után a kapott pdf megegyezik az eredetivel, de ki lesz bővítve vágójelekkel.

5.7. 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 pozíójába akar illet helyezni, akkor akkor tegye a következőt:

```

17 \documentclass{article}
18 \usepackage[type=none]{fgruler}
19 \begin{document}
20 \fgruler{upperleft}{<jobbra>}{{lefelé}}
21 \end{document}

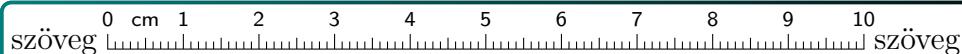
```

A **<jobbra>** helyére azt a távolságot kell beírni, amennyivel el akarja tolni jobbra a vonalzót a bal felső sarokhoz viszonyítva. A **<lefelé>** 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 vonalzók 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` opciójával. Ekkor minden oldal 5 mm-es közökkel rácsvosan 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

Ha több oldalból áll a dokumentum, akkor célszerű feltüntetni az oldalszámokat. De a fejezeteket, szakaszokat is számozzuk. Ez például a 6. fejezet, amely a 83. oldalon kezdődik. További számozott elemek: listák, ábrák, táblázatok, matematikai képletek és tételek, irodalomjegyzék elemei, stb. Az ilyen számozott elemekre sok esetben hivatkozunk. Ezek az ún. kereszthivatkozások. Ezeket nem érdemes a forrásban konkrétan beírni, hiszen egy ilyen szám a szerkesztés során változhat, így állandóan javítanunk kellene, ami egy idő után sok hibát eredményezne. Erre az a megoldás, hogy a L^AT_EX-re bízzuk a számozott elemknél és a kereszthivatkozásoknál a megfelelő számok beírását.

6.1. Címkék

Ha egy számozott elemre hivatkozni kell, akkor először meg kell címkézni a

```
\label{<címke>}
```

parancssal. A `<címke>` 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 ékezes 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` parancssal indítunk:

```
\begin{enumerate}
\item Ez egy listaelem.
\item Ez egy másik listaelem.
\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

Címkével ellátott elemre alapesetben a

```
\ref{<címke>}
```

parancsal tud hivatkozni. Az előző példát folytatva:

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

Lásd 2. listaelement.

Helyesebb a mondat, ha a sorszám elé határozott névelőt rakunk: „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 rómaítól különböző számozás esetén 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. Amennyiben betölti még a `huaz` csomagot is, akkor az előbbi két parancspár minden esetben ekvivalens. Például

 Lásd `\aref{lista-proba}.~listaelement.\`Aref{lista-proba}.~listaelementben olvasható.`

Lásd a 2. listaelement.

A 2. listaelementben olvasható.

Amikor címkézünk egy elemet, akkor nemcsak az adott sorszámot tudja a L^AT_EX, hanem azt is, hogy az adott elem melyik oldalon található. Adott címkéhez tartozó oldalszámot a következő parancssal írathatja ki:

```
\pageref{<címke>}
```

Ennek is vannak névelős verziói:

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

vagy rómaítól különböző oldalsámozás esetén az ezzel egyenértékű

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

Amennyiben betölti még a `huaz` csomagot is, akkor az előbbi két parancspár minden esetben ekvivalens. Például

 `\Aref{lista-proba}.~listaelement \pageref{lista-proba}.~oldalon találjuk.`

A 2. listaelement a 83. oldalon találjuk.

A `\pageref{<címke>}` eredménye az az oldalszám, ahol a `\label{<címke>}` parancs ki lett adva, míg a `\ref{<címke>}` parancs eredménye 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}
\Az{\ref{sec-nszt}} című szakaszban
```

1. Nagy számok törvénye

A „Nagy számok törvénye” című szakaszban

Ha egy lábjegyzetre többször is hivatkozni szeretne, akkor használja a

```
\footref{<címke>}
```

parancsot. Például

```
Szöveg\footnote{\label{footnote-proba}Lábjegyzet szövege.}
szöveg\footref{footnote-proba}
```

Szöveg¹ szöveg¹

¹ Lábjegyzet szövege.

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. Számoszatlan listák

Számoszatlan listákra az `\itemize` környezet használható. minden listaelemet `\item` parancs vezet be.

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

E környezetek négy szint mélysegig ágyazhatók egymásba. Például



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

Lista utáni szöveg.

A `magyar.ldf` fájl `defaults=hu-min` opciója megváltoztatja az alapértelmezett felsorolásjeleket. Ha ezt nem akarja, akkor használja a `labelitems=unchanged` opciót is:

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

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>] \in 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>} \in 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. Számoszatlan listák extra függőleges 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.
  \item Egy másik listaelem.
\end{compactitem}
Lista utáni szöveg.
```

- Lista előtti szöveg.
- Listaelem.
 - Egy másik listaelem.
- Lista utáni szöveg.

7.2. Leíró listák

A leíró listákra, azaz a szótárszerű felsorolásokra a `description` környezet való. minden listaelement `\item[\langle címke\rangle]` parancs előz meg. E környezetek hat szint mélységig ágyazhatók egymásba.

```
\begin{description}
  \item[⟨címke⟩] ⟨listaelem⟩
  \item[⟨címke⟩] ⟨listaelem⟩
\end{description}
```

Például

```
</> \begin{description}
  \item[Címke] szöveg szöveg szöveg szöveg szöveg szöveg
    szöveg szöveg szöveg szöveg szöveg
  \item[Másik címke] szöveg szöveg szöveg szöveg szöveg szöveg
    szöveg szöveg szöveg szöveg szöveg
\end{description}
```

Címke. szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg

Másik címke. szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg

Alaphelyzetben a címke félkövér betűtípusú, továbbá a magyarban még egy pont is kerül a címke után. Ezt átállíthatja például dőltre és kettőspontra a következő parancs beírásával:

```
\renewcommand{\descriptionlabel}[1]
  {\hspace{\labelsep}\normalfont\itshape#1:}
```

Alapesetben a címke után nincs semmilyen jel. Ha a `magyar.1df` fájl `defaults=hu-min` opciója esetén is ezt szeretné, akkor használja a `postdescription=unchanged` opciót is:

```
\PassOptionsToPackage{defaults=hu-min,postdescription=unchanged}{magyar.1df}
```

Ha a listaelemek között extra függőleges térközt meg akarja szüntetni, akkor használja a `paralist` csomag `compactdesc` környezetét a `description` helyett.

```
\begin{compactdesc} ∈ paralist
  \item[⟨címke⟩] ⟨listaelem⟩
  \item[⟨címke⟩] ⟨listaelem⟩
\end{compactdesc}
```

Ennek használata pontosan megegyezik a `description` környezettel. Például

```
</> \begin{compactdesc}
  \item[Címke] szöveg ...
  \item[Másik címke] szöveg ...
\end{compactdesc}
```

Címke. szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg

Másik címke. szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg szöveg

7.3. Számozott listák

Számozott listákra az `enumerate` környezet való. minden listaelementet `\item` parancs előz meg.

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

E környezetek négy szint mélységgel 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. Ha a `labelenums=unchanged` opciót használja, akkor a `magyar.ldf` nem változtatja meg az eredeti számozás stílusát.

7.3.1. Számozott listák számozási stílusának megváltoztatása

Ha számozott listában egy adott listaelement nem számozni, hanem csak egy jelleg szerepnél ellátni, akkor ezt az `\item` parancs opciójában adhatja meg:

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

Ekkor a számozás nem növekszik. Például

</>

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

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

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

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

A **(címke)** 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 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>}
```

Arab számmal (1, 2, 3, ...).

```
\roman{<számláló>}
```

Kis római számmal (i, ii, iii, ...).

```
\Roman{<számláló>}
```

Nagy római számmal (I, II, III, ...).

```
\alph{<számláló>}
```

Latin ábécé kisbetűivel (a, b, c, ...).

```
\Alpha{<számláló>}
```

Latin ábécé nagybetűivel (A, B, C, ...).

```
\greek{<számláló>} \in moreenum
```

Görög ábécé kisbetűivel (α , β , γ , ...).

```
\Greek{<számláló>} \in moreenum
```

Görög ábécé nagybetűivel (A , B , Γ , ...).

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}{\Alpha{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}
```

```
\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` opcionálval.

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 jele-níti 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.
 - ① Listaelem.
 - ② 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 nemcsak 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 függőleges 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.
    \end{compactenum}
\end{compactenum}
Lista utáni szöveg.
```

Lista előtti szöveg.
 1. Listaelem az első szinten.
 a) Listaelem a második 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
\end{inparaenum}
```

Szöveg 1. szöveg 2. szöveg

</>

```
Szöveg
\begin{inparaenum}[\itshape (a)]
  \item szöveg
  \item szöveg
\end{inparaenum}
```

Szöveg (a) szöveg (b) szöveg



Videó: Listák

7.4. Lista paramétereinek beállítása az enumitem csomaggal

Ezt a szakaszt akkor érdemes elolvasni, ha a listákra vonatkozó korábbi leírásokat már áttekintette. Ezekben többféle beállítási lehetőségről volt szó. Most az `enumitem` csomagot mutatjuk be, amellyel ezek a beállítások és még számos további beállítás kényelmesen megadható.

7.4.1. Korlátozások

Az `enumitem` csomag használatakor három korlátozást érdemes figyelembe venni.

1. A `paralist` csomaggal nem használható együtt.
2. A `magyar.ldf` fájl `defaults=hu-min` opciójának együttes használata esetén a számított listák 3. szintjén nem jelennek meg a címkék. Ennek az az oka, hogy ebben az esetben ezen a szinten a számláló a görög ábécé kisbetűi, amit nem kezel alapesetben az `enumitem`. Megoldásként az `enumitem` betöltése után írja be a következőket a preambulumba:

```
\usepackage{moreenum}
\setlist[enumerate,3]{label=\textit{\greek*)}}
```

Másik lehetőség a probléma megoldására, hogy a `defaults=hu-min` után használja még a `labelenum=unchanged` vagy `labelenum=hu-A` opciót is, melyek nem használnak számlálóhoz görög ábécét.

3. A `magyar.ldf` fájl `defaults=hu-min` opciójának együttes használata esetén a leíró listák címkéi körül elállítódnak a vízszintes térközök. Megoldásként az `enumitem` betöltése után írja be a következőt a preambulumba:

```
\renewcommand{\descriptionlabel}[1]{\normalfont\bfseries#1}.
```

Másik lehetőség a probléma megoldására, hogy a `defaults=hu-min` után használja még a `postdescription=unchanged` opciót is.

7.4.2. Listák globális formázása

Listák globális beállításaihoz használja a következő parancsot:

```
\setlist[<opcion>]{<formazas>} \in enumitem
```

Opciók nélkül használva, a megadott `<formázás>` minden lista minden szintjére vonatkozik. Ha a `<formázás>` csak bizonyos listakörnyezetek bizonyos szintjeire vonatkozik, akkor ezeket a környezetneveket és szintszámokat kell megadni opcióként. Például

```
\setlist[enumerate,itemize,1,2]{<formazas>}
```

Ha szintszámokat nem ad meg, akkor `<formázás>` a megadott listakörnyezetek minden szintjére vonatkozik. Például

```
\setlist[enumerate,itemize]{<formazas>}
```

Ha listakörnyezetet nem ad meg, akkor `\begin{formázás}` a megadott szintekre vonatkozik minden listakörnyezet esetén. Például

```
\setlist[1,2]{formázás}
```

Most rátérünk a `formázás`-ban megadható paramétereire.

Térközök megadása ♦ A 7.1. ábrán egy adott lista adott szintjének térközei megadó paraméterek vannak feltüntetve.



7.1. ábra. Lista paraméterek

Az ábrán a `partopsep` azért van zárójelben, mert annak értéke csak akkor adódik hozzá a `topsep` értékéhez, ha az adott szintet létrehozó környezet `\begin{parancsa}` előtt van egy üres sor a forrásban.

Például

```
\setlist[enumerate,1]{topsep=3pt,partopsep=0pt}
```

Ha nem adta meg az összes térközt beállító paramétert, akkor a hiányzókat az alapbeállítás vagy a korábbi beállítások alapján határozza meg.

A `leftmargin`, `itemindent`, `labelsep`, `labelwidth`, `labelindent` és `labelsep*` paraméterek nem függetlenek egymástól, hiszen

$$\text{labelsep} = \text{labelsep}* + \text{itemindent}$$

és

$$\text{leftmargin} = \text{labelindent} + \text{labelwidth} + \text{labelsep}.$$

Ezért ezeknek a paramétereknek (kivétel a `labelsep*`) értékként megadható a `!` jel, ami azt jelenti, hogy annak értékét a megadottak alapján számolja ki. Például

```
\setlist[enumerate,1]{labelindent=1cm,labelwidth=1cm,labelsep*=1cm,
itemindent=1cm,leftmargin!=!}
```

vagy

```
\setlist[enumerate,1]{labelindent=0cm,labelwidth=1cm,labelsep=1cm,
leftmargin=0cm,itemindent!=!}
```

Kompakt listához a következő két paraméterrel lehet nullázni a függőleges térközöket:

noitemsep Lenullázza a `parsep` és `itemsep` értékét. Például

```
\setlist[enumerate]{noitemsep}
```

nosep Lenullázza minden függőleges térköz méretét a listában. Például

```
\setlist{nosep}
```

Címkék formázása ♦ A következők a lista címkéinek beállítására szolgálnak.

align=*(irány)* A címke igazítása a címkének fenntartott `labelwidth` széles doboz bal vagy jobb oldalához. Ennek megfelelően lehet az *(irány)* `left` vagy `right`. Alapérték `right`.

start=*(szám)* Számoszott lista első elemének sorszáma. Alapértéke 1.

label=*(parancs)* Számoszott illetve számoszatlan lista címkéjének formázása. Például

```
\setlist[itemize,1]{label=\ddag}
\setlist[enumerate,1]{label=\textbf{\Roman{enumi}}.}}
```

A példában a `\Roman{enumi}` helyett írható `\Roman*` is, ugyanis a csillagos verzió mindenig az aktuális számlálóra vonatkozik. Természetesen `\Roman` helyett írható `\arabic`, `\alph`, stb. parancs is (lásd a 91. oldalon). Például

```
\setlist[enumerate,2]{label=\textit{(\alph*)}}
```

A `pifont` csomag karakterei (lásd a 46. oldalon) is felhasználhatók számlálónak. Ehhez az aktuális számláló értékét `\value{<számláló>}` módon lehet beírni a `\ding` parancsba. Például

```
\setlist[enumerate,1]{start=172,label=\ding{\value{enumi}}}
```

Azért 172 a kezdőérték, mert `\ding{172}` eredménye ①, `\ding{173}` eredménye ②, stb. A `\value` parancsnak is van csillagos verziója, ami az aktuális számlálót tartalmazza. Így az előző kód ekvivalens a következővel:

```
\setlist[enumerate,1]{start=172,label=\ding{\value*}}
```

label*=*(parancs)* Ugyanaz mint a `label`, de ebben az esetben a címke kiegészül az előző szint címkéjével. Például

```
\setlist[itemize]{label*=\textbullet}
\setlist[enumerate]{label*=\arabic*.}
```

esetén az `itemize` szintjeinek címkéi rendre •, ••, ••• és ••••, míg az `enumerate` szintjeinek címkéi rendre 1., 1.1., 1.1.1. és 1.1.1.1., ahol az 1 helyére a listaelemnek megfelelő szám kerül.

ref=*(parancs)* Az adott számoszott listaelemre történő kereszthivatkozás formázása. Például a `moreenum` csomag betöltése után

```
\setlist[enumerate,3]{label=\textit{\greek*},
ref=\theenumi\theenumii\greek*}
```

font=*(parancs)* A címke fonttípusát állítja be. Elsősorban leíró lista esetén használható. Például

```
\setlist[description]{font=\normalfont\itshape}
```

style=*(stílusnév)* Leíró lista címkéjének a stílusa. A *(stílusnév)* a következők lehetnek: `standard` Ez az alapbeállítás. Ekkor a címke csak egysoros lehet. A lista tartalma a címke sorában kezdődik.

`unboxed` Annyiban különbözik a `standard` értéktől, hogy ekkor a címke lehet többsoros is.

`nextline` Annyiban különbözik az `unboxed` értéktől, hogy ekkor a lista tartalma a címkét követő sorban kezdődik.

Kiegészítő beállítások ♦ Néhány további beállítási lehetőség:

`first=<kód>` Az adott listakörnyezetben az első `\item` parancs előtt fejti ki a `<kód>`-ot.
`after=<kód>` Az adott listakörnyezetben a listát lezáró `\end` parancs előtt fejti ki a `<kód>`-ot. Például

```
\setlist[enumerate,1]{first=\item[]\hrulefill,after=\item[]\hrulefill}
```

7.4.3. Listák lokális formázása

Egy adott lista helyi formázására a listakörnyezet opciójában van lehetőség:

```
\begin{listakörnyezet neve} [formázás]
\item ...
\end{listakörnyezet neve}
```

A `<formázás>` pontosan úgy használható, mint a globális beállításoknál. Például

```
\begin{itemize}[nosep,label=\textbullet]
\item szöveg
\end{itemize}
```

Címkék rövid megadása ♦ Az `enumitem` csomag `shortlabels` opciója lehetőséget ad a címkék rövid megadására. Ilyenkor számosztlan listák esetén például

```
\begin{itemize}[label=\ddag]
```

helyett írható ez is:

```
\begin{itemize}[\ddag]
```

Számosztott lista számosztása is rövidíthető. Például

```
\begin{enumerate}[label=\roman*)]
```

helyett használható a következő is:

```
\begin{enumerate}[i)]
```

A számosztott lista címkéjének rövid megadása bármilyen karaktert tartalmazhat, de ötnek a számosztási stílus beállítása a feladata:

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

Ezekből a betűkből a rövidített megadás csak egyet tartalmazhat. Ha ezen öt karakter valamelyikét nem számosztási stílus jeleként, hanem tényleges betűként akarja bevinni, akkor tegye kapcsos zárójelek közé. Például

```
</> \begin{enumerate}[\bfseries I. \axiom{..},align=left,nosep]
\item Listaelem.
\item Listaelem.
\end{enumerate}
```

I. axióma. Listaelem.

II. axióma. Listaelem.

</>

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

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

Folytatólagos számozott listák ♦ A `resume` paraméterrel lehetőség van arra, hogy az adott számozott lista első sorszáma folytatása legyen az előző számozott listának. Például

</>

```
\begin{enumerate}
    \item Listaelem
    \item Listaelem
\end{enumerate}
Szöveg
\begin{enumerate}[resume]
    \item Lista folytatása
\end{enumerate}
```

1. Listaelem
2. Listaelem

Szöveg

3. Lista folytatása

7.4.4. Sorfolytonos listák

Ha sorfolytonos listákat akar használni, akkor az `enumitem` csomagot `inline` opcionál töltse be. Ezután a sorfolytonos lista megjelenítése `enumerate*`, `itemize*` vagy `description*` környezettel lehetséges aszerint, hogy számozott, számozatlan vagy leíró listát szeretne. Például

</>

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

1. Listaelem
2. Listaelem

7.4.5. Új listakörnyezet definiálása

Saját listakörnyezet is definiálható a következő parancssal:

```
\newlist{\langle listanév\rangle}{\langle listatípus\rangle}{\langle maximális szintszám\rangle} \in enumitem
```

A *<listatípus>* lehet `enumerate`, `itemize` vagy `description` aszerint, hogy számozott, számozatlan vagy leíró listát szeretnénk definiálni. A *<maximális szintszám>* azt adja meg, hogy a definiált listát hány szint mélységgig lehessen egymásba ágyazni. Ezzel a parancssal *<maximális szintszám>* darab számláló is létrejön *<listanév>i*, *<listanév>ii*, *<listanév>iii*, *<listanév>iv*, *<listanév>v*, stb. néven, melyek a különböző szintek számlálói lesznek. Például

```
\newlist{steps}{enumerate}{2}
```

definiál egy `steps` nevű számozott listakörnyezetet, amely maximum két szint mélységgig ágyazható egymásba. Ezután ennek beállítása a `\setlist` parancssal pontosan úgy történhet, mint a többi listakörnyezet esetén. Például

```
\setlist[steps,1]{label=\textbf{\arabic*. lépés.},align=left}
\setlist[steps,2]{label=(\alph*)}
```

Az első szint számlálója `stepsi` a másodiké pedig `stepsii` lesz.

8. fejezet

Képek

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

Ha a forrásállomány fordítását a `latex` fordító végzi, akkor eps képeket kell használni. Ha `pdflatex` a fordító (TeXstudióban ez az alapbeállítás), akkor eps, jpg, png vagy pdf képeket is használhat. Ha eps képet tölt be, akkor azt a `pdflatex` fordító először pdf formátumra konvertálja, majd azt hívja meg.

8.1. Kép beillesztése

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

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

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

```
\includegraphics{abra}
```

Fontos, hogy ebben az esetben az `abra.jpg` fájlnak az aktuális mappában kell elhelyezkednie. Ha az aktuális mappa egy almappájában 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 felhasználót. 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 többek között 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=⟨arányszám⟩ Nagyítás/kicsinyítés mértéke (például **scale=2**).

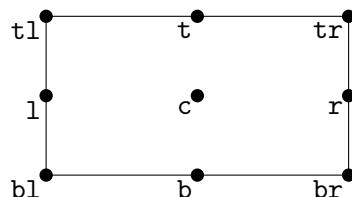
trim=⟨bal⟩ ⟨lent⟩ ⟨jobb⟩ ⟨fent⟩ A képen meghatároz egy keretet. Ezután a kép pozicionálása a keret levágása után maradó képre történik, de az egész kép meg fog jelenni, azaz tényleges vágás nem történik. Például, ha azt írjuk be, hogy **trim=10mm 11mm 12mm 13mm**, akkor a keret bal oldalon 10, alul 11,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

clip Ha a **trim** mellé ezt is betöltsük, 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 egy adott oldalon a `graphicx` után betöltött `eso-pic` csomaggal tud beilleszteni a következő parancssal:

```
\AddToShipoutPictureBG*{\langle háttér\rangle} ∈ eso-pic
```

Például

```
\AddToShipoutPictureBG*{%
    \AtPageLowerLeft{%
        \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 1 mm. 2020. októberé után telepített rendszerekben a `\put` parancsba közvetlenül is beírható hosszúság vagy hosszúságparancs, így a következő kód az előzővel ekvivalens lesz:

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

Ha a kép bal alsó sarkát az oldal bal alsó sarkába, azaz az origóba akarja helyezni, akkor a `\put` parancs felesleges:

```
\AddToShipoutPictureBG*{%
    \AtPageLowerLeft{%
        \includegraphics[width=15cm]{hatter}}%
}%
}
```

Az `\AtPageLowerLeft` parancs helyezi az origót az oldal bal alsó sarkához. Ehelyett a következő parancsokat használva az origó másikról is átrakható:

`\AtPageUpperLeft` origó az oldal bal felső sarkában,
`\AtTextUpperLeft` origó a szövegtükör bal felső sarkában,
`\AtTextLowerLeft` origó a szövegtükör bal alsó sarkában,
`\AtPageCenter` origó az oldal közepén.

Az előző kódok * nélkül az adott képet minden oldalon megjeleníti háttérként. Így lehet például vízjelet készíteni. Ennek hatása a

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

parancssal 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 külső pdf fájlból szeretne oldalakat beilleszteni, akkor használja az

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

parancsot. Például

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

a `doc.pdf` 3, 8, 9, 10, 11, 15 sorszámú oldalait szúrja be, ahol a `doc.pdf` ugyanabban a mappában van, ahol a L^AT_EX forrásfájl. Ha a pdf más mappában található, akkor meg kell adni a relatív útvonalat, hasonlóan mint a képeknél. 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.

Amennyiben az előbbi példában a `doc.pdf` belső vagy külső linkeket tartalmaz (keeszthivatkozások, url, stb.), akkor azt a forrásfájlba előző módon betöltve és lefordítva, a kapott pdf fájlban ezek a linkek nem fognak működni. A probléma megoldásához először készítsen egy tex kiterjesztésű fájlt ugyanabban a mappában, ahol az előbbi L^AT_EX forrásfájl is van. A tartalma legyen a következő:

```
\documentclass{article}
\directlua{require("newpax")}
\directlua{newpax.writenewpax("doc")}
\begin{document}
\end{document}
```

Fontos, hogy a kódban a `doc.pdf` fájlnak ne szerepeljen a kiterjesztése. Ha ennek a tex fájlnak például `foo.tex` a neve, akkor futtassa a következő parancsot:

```
lualatex foo.tex
```

Ez generálni fog egy `doc.newpax` fájlt, ami tartalmazza a `doc.pdf` linkjeinek adatait. Lehetőség van egyszerre több pdf linkjeinek az adatait is kinyerni. Például `doc1.pdf` és `doc2.pdf` esetén az előbbi `foo.tex` tartalma legyen a következő:

```
\documentclass{article}
\directlua{require("newpax")}
\directlua{
  newpax.writenewpax("doc1")
  newpax.writenewpax("doc2")}
\begin{document}
\end{document}
```

Ebből az előző eljárással megkapja a `doc1.newpax` és `doc2.newpax` fájlokat. A `newpax` kiterjesztésű fájlok birtokában már be lehet tölteni az eredeti L^AT_EX forrásba ezeket a pdf fájlokat az `\includepdf` parancssal ahogy korábban mutattuk, annyi kiegészítéssel,

hogy töltse még be a `hyperref` (lásd a 19.1. szakaszban) és a `newpax` csomagokat, továbbá a `\documentclass` parancs elé gépelje be a következőt:

```
\RequirePackage{pdfmanagement}
```

Az így kapott forrás úgy tölti be a pdf fájlokat, hogy a bennük található linkek az általunk készített pdf-ben is működni fognak. Például

```
\RequirePackage{pdfmanagement}
\documentclass{article}
\usepackage{pdfpages,hyperref,newpax}
\newpaxsetup{usefileattributes}
\begin{document}
\includepdf[pages=-]{doc1.pdf}
\includepdf[pages=-]{doc2.pdf}
\end{document}
```

Ebben a példában a

```
\newpaxsetup{usefileattributes}
```

parancs azt a célt szolgálja, hogy a linkek úgy jelenjenek meg, ahogy az az eredeti pdf-ben is voltak. Ha ehelyett

```
\newpaxsetup{usefileattributes=false}
```

szerepel, akkor a linkek a `hyperref` csomaggal beállított módon jelennek meg.

9. fejezet

Ábrák készítése

Az itt látható példák működéséhez töltse be a `pict2e` és `xcolor` csomagokat. Ha bonyolultabb képelemeket tartalmazó rajzokat szeretne készíteni L^AT_EX-hel, akkor nézze át a `curves`, `curve2e`, `xpicture`, `tikz` csomagok valamelyikének a leírását.

A legnagyobb tudású csomag ezek közül a `tikz`. A dokumentációjában is rengeteg példa van, de a következő oldalakon is számtalan példát találhat a használatára:

- <https://tikz.net/>
- <https://texample.net//tikz/examples/>

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{\langle vastagság \rangle }
```

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. és 7. sorban létrehoztunk egy 2 egység széles és 2 egység magas rajzfelületet, aminek a bal alsó sarkában lesz az origó.
- A 3. sorban a vonalak vastagságát állítottuk 2 pt-ra.
- 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}}
  \multiput(0,0)(0,0.5){5}{\line(1,0){2}}
\end{picture}
```



```
\polyline(<x1>,<y1>)(<x2>,<y2>)...(<xn>,<yn>)
```

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(<x1>,<y1>)(<x2>,<y2>)...(<xn>,<yn>)
```

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*(<x1>,<y1>)(<x2>,<y2>)...(<xn>,<yn>)
```

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}
```



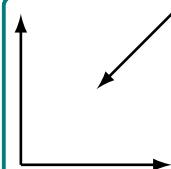
```
\vector(<x>,<y>){<v>}
```

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}
```



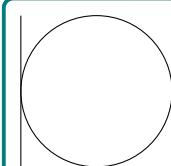
9.3. Körvonalak

```
\circle{<átmérő>}
```

Rajzol egy `<átmérő>` 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}
```

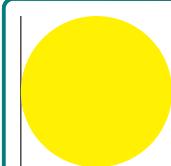


```
\circle*{<átmérő>}
```

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}
```

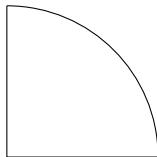


```
\arc[<szög1>,<szög2>]{<sugár>}
```

Rajzol egy $\langle \text{sugár} \rangle$ egység sugarú körívet $\langle \text{szög1} \rangle$ -től $\langle \text{szög2} \rangle$ -ig, melynek a referencia-pontja a középpontja. A $\langle \text{szög1} \rangle$ és $\langle \text{szög2} \rangle$ fokban 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}
```



$\text{\arc*}[\langle \text{szög1} \rangle, \langle \text{szög2} \rangle]\{\langle \text{sugár} \rangle\}$

Rajzol egy $\langle \text{sugár} \rangle$ egység sugarú telített körcikket $\langle \text{szög1} \rangle$ -től $\langle \text{szög2} \rangle$ -ig, melynek a referencia-pontja a középpontja. A $\langle \text{szög1} \rangle$ és $\langle \text{szög2} \rangle$ fokban 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

$\text{\oval}[\langle \text{sugár} \rangle](\langle x \rangle, \langle y \rangle)[\langle \text{rész} \rangle]$

Rajzol egy $\langle x \rangle$ egység széles és $\langle y \rangle$ egység magas lekerekített sarkú téglalapot, melynek sarkai negyed körök. A referencia-pontja a középpont. A $\langle \text{rész} \rangle$ 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 $\langle \text{sugár} \rangle$ -nál, melynek alapértéke 20pt. A $\langle \text{sugár} \rangle$ lehet egy szám, amikor is az értéke $\langle \text{sugár} \rangle$ egység, és lehet egy konkrét hossz is. Ha a $\langle \text{sugár} \rangle$ értéke 0, akkor normál téglalapot kapunk. A $\langle \text{sugár} \rangle$ nemcsak opcióban adható meg, hanem a

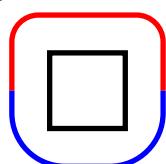
$\text{\renewcommand{\maxovalrad}{\langle \text{sugár} \rangle}}$

parancssal 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}
```



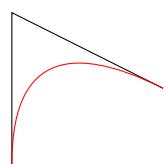
9.5. Bézier-görbék

```
\qbezier(<x1>,<y1>)(<x2>,<y2>)(<x3>,<y3>)
```

Rajzol egy másodfokú Bézier-görbét az adott koordinátájú kontrollpontokkal.

</>

```
\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}
```

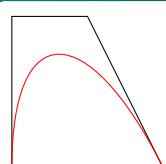


```
\cbezier(<x1>,<y1>)(<x2>,<y2>)(<x3>,<y3>)(<x4>,<y4>)
```

Rajzol egy harmadfokú Bézier-görbét az adott koordinátájú kontrollpontokkal.

</>

```
\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, körívekből és másodfokú Bézier-görbék ből áll.

```
\moveto(<x>,<y>)
```

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

```
\lineto(<x>,<y>)
```

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(<x2>,<y2>)(<x3>,<y3>)(<x4>,<y4>)
```

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[<n>]{<x>}{<y>}{<sugár>}{<szög1>}{<szög2>}
```

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.

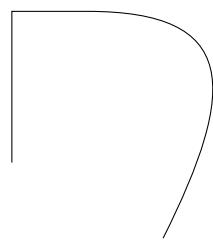
```
\strokepath
```

Megrajzolja a korábban meghatározott útvonalat.

```
\fillpath
```

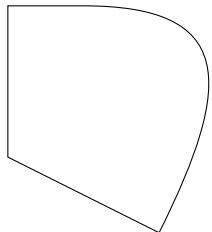
Kitölти az adott útvonallal határolt síkidomot.

```
</> \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}
```



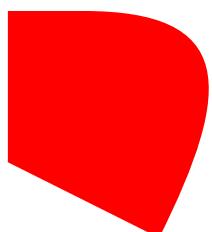
```
</> \setlength{\unitlength}{1cm}
\begin{picture}(3,3)
\moveto(0,1)
\lineto(0,3)
```

```
\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}
```



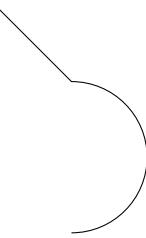
</>

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



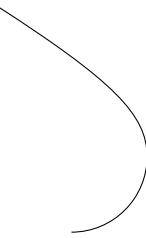
</>

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

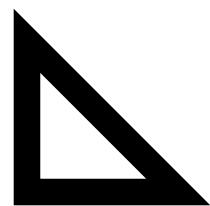


\miterjoin

Alapértelmezett csatlakozás.

</>

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

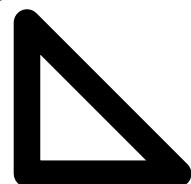


\roundjoin

Lekerekített csatlakozás.

</>

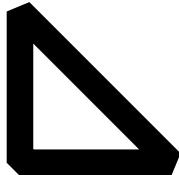
```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
    \linethickness{10pt}
    \roundjoin
    \moveto(0,0)
    \lineto(0,2)
    \lineto(2,0)
    \closepath
    \strokepath
\end{picture}
```



```
\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 árában

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

Egy *<x>* egység széles és *<y>* egység magas bekeretezett doboz jön létre, melynek a bal alsó sarkában található a referenciaipont. A *<szöveg>* ebben jelenik meg úgy pozicionálva, ahogy azt a *<pozíció>* opció megadja. A *<pozíció>* 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}
```

tl	t	tr
l	c	r
bl	b	br

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

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

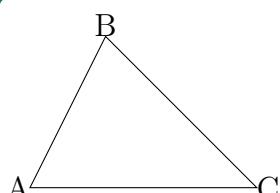
</>

```
\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

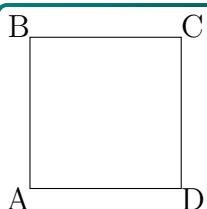
</>

```
\setlength{\unitlength}{1cm}
\begin{picture}(3,2)
\polygon(0,0)(1,2)(3,0)
\put(0,0){\makebox(0,0)[r]{A}}
\put(1,2){\makebox(0,0)[b]{B}}
\put(3,0){\makebox(0,0)[l]{C}}
\end{picture}
```



</>

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



Amint látható, az előző két példában túl közel vannak a betűk a sokszögek csúcsaihoz. Ezt a problémát a következő szakaszban mutatott módon lehet megoldani.

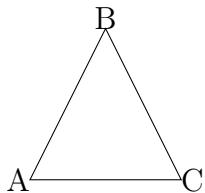
9.9. Koordináta megadása hosszmérettel

Az eddigiekben először megadtuk a koordináta-rendszerünkben az egység hosszát, majd minden koordináta ebben az egységben volt megadva. Ha valamiért szeretne konkrét hosszt is beírni, akkor 2020. október után telepített rendszer esetén ezt nyugodtan megteheti. Ezelőtt telepített rendszer esetén ehhez a `pict2e` és `xcolor` csomagok után még töltse be a `picture` csomagot is.

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

</>

```
\setlength{\unitlength}{1cm}
\begin{picture}(2,2)
    \polygon(0,0)(1,2)(2,0)
    \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 például toljuk el a betűket 2 pt távolságra a következőképpen:

</>

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



Ugyanezt kapnánk a következő módon is:

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

A 0-hoz azért nem írtunk mértékegységet, mert a jelentése alapesetben 0 pt, ami meggyezik 0 cm-rel.

10. fejezet

Táblázatok

A táblázatok elkészítése az egyik legbonyolultabb feladat a L^AT_EX-ben. Két szakaszban tárgyaljuk ennek menetét. Az elsőben a standard hagyományos eszközöket vesszük át, a másodikban pedig egy 2021. májusától létező eszközöt mutatunk be, a `tabulararray` csomagot.

10.1. Standard táblázatkezelési eszközök

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](#) és [LaTeX Complex Table Editor](#) weblapok sokat segítenek a táblázatok vizuális szerkesztésében.

10.1.1. 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 `\backslash` 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 például a 2. oszloptól a 4. oszlopig, akkor `\hline` helyett használjon

\cline{2-4}

parancsot. Több \cline is írható egymásután:

A teal icon representing a code editor or programming language, consisting of three slanted bars forming a bracket-like shape.

```
\begin{tabular}{|c|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:

A teal-colored icon representing code or programming, consisting of two slanted brackets (< and >) joined at their bases.

```
\begin{tabular}{|c|c|} \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 `\t@{...}` opciójában lehet megadni, hogy egy cella elő vagy után mi kerüljön. A `\t@{}` azt eredményezi, hogy nincs semmi, még térköz sem:

A teal-colored icon representing code or programming, consisting of two slanted brackets (< and >) joined at their bases.

```
\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 `>{...}` opciót is:

</>

```
\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>}
```

parancsal. 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{1}{c|}{\multicolumn{2}{c}{\Év}} & \\
\cline{2-3}
&\multicolumn{1}{l|}{} & \\
\multicolumn{1}{c|}{\multicolumn{2}{c}{2002}} & \\
\multicolumn{1}{c|}{\multicolumn{2}{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} ∈ array
```

Ezt írja az előző kód elő, 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 parancssal teheti meg:

```
\multirow{<cellaszám>}*{<szöveg>} ∈ multirow
\multirow{<cellaszám>}{<sélesség>}{<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	3
szöveg törve	4 5

A következő példában azt mutatjuk meg, hogyan lehet beállítani az egyes oszlopok sélességét.

```
</> \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\arraybackslash}p{2cm}
|>{\raggedleft\arraybackslash}p{2cm}
|>{\centering\arraybackslash}p{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.			
-------------------------------------	-------------------------------------	-------------------------------------	-------------------------------------

A következő kód a cella tartalmának függőleges középre igazítására egy példa. A kód az `array` csomag betöltése után működik.

```
</> \begin{tabular}{|c|c|c|c|} |>{\raggedright\arraybackslash}{16mm}
|>{\raggedleft\arraybackslash}{16mm}
|>{\centering\arraybackslash}{16mm}
|m{22mm}|}

\hline
szöveg szöveg szöveg &
szöveg szöveg &
szöveg &
sz ö v e g szöveg szöveg\\
\hline
\end{tabular}
```

szöveg	szöveg	szöveg	sz ö v e g szöveg szö- veg
--------	--------	--------	----------------------------------

Az előző kódban az oszloptípusokat előre definiálhatja a

`\newcolumntype{(jel)}{(definíció)} ∈ array`

parancssal, például így:

```
</> \newcolumntype{L}{\raggedright\arraybackslash}{16mm}
\newcolumntype{R}{\raggedleft\arraybackslash}{16mm}
\newcolumntype{C}{\centering\arraybackslash}{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 szélén.

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 színezhet 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 színezhet:

```
</> \begin{tabular}{ccc}
\rowcolor{cyan} egy & kettő & három\\
\end{tabular}
```

```
\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.

10.1.2. Hosszú táblázatok

Ha a táblázat olyan hosszú, hogy nem fér ki egy oldalon, akkor használja a `longtable` csomagot.

```
\begin{longtable}{<pozíció>}{<oszlopok>} \in longtable
\endfirsthead \in longtable
\endhead \in longtable
\endfoot \in longtable
\endlastfoot \in 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}
```

10.1.3. Kiadói minőségű táblázatok

Az előzőekben tárgyalt táblázatok hagyományos szerkezetűek voltak. Viszont az úgynevezett kiadói minőségű 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 @{} parancsal 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

10.1.4. Táblázatok alapvonalhoz igazítása

A `\begin{tabular}` és `\begin{tabular*}` környezeteknek nemcsak paramétereik, hanem opcióik is vannak. Ezekben lehet megadni az alapvonalhoz viszonyított pozíciójukat:

```
\begin{tabular}[<option>]{<parameters>}
\begin{tabular*}{<width>}[<option>]{<parameters>}
```

Opció nélkül (pontosabban alapopcionalval) az igazítás középre történik:

</>

```
szöveg
\begin{tabular}{|cc|}
\hline X&X\|X&X\|X&X\|X&X\\ \hline
\end{tabular}
szöveg
```

X	X
X	X
X	X
X	X

szöveg

Ha az `[<option>]` **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
```

X	X
X	X
X	X
X	X

szöveg

Ha az `[<option>]` **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
```

X	X
X	X
X	X
X	X

szöveg

Az utóbbi két illesztésnél zavaró lehet, hogy a szöveg alapvonala nem esik egybe a táblázat utolsó illetve első sorának alapvonálával. Ezen lehet segíteni a

```
\firsthline \in array
\lasthline \in array
```

parancsokkal:

</>

```
\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

10.2. A tabulararray csomag

A 2021. májusától létező `tabulararray` csomaggal lehetőség van arra, hogy a standard megoldásoktól eltérően a táblázat tulajdonságait és a tartalmát teljesen elkülönítve adjuk meg. Ezzel jelentősen megnövelhetjük a táblázat forrásának átláthatóságát és rugalmasságát.

10.2.1. Normál tabulararray táblázatok

Normál `tabulararray` táblázat készítéséhez a `tblr` környezetet lehet használni:

```
\begin{tblr}{<opción>} \in tabulararray
<cella 1,1> & <cella 1,2> & <cella 1,3> & ... & <cella 1,m> \\
<cella 2,1> & <cella 2,2> & <cella 2,3> & ... & <cella 2,m> \\
...
<cella n,1> & <cella n,2> & <cella n,3> & ... & <cella n,m> \\
\end{tblr}
```

A `<cella x,y>` a táblázatban az `x`-edik sor `y`-adik cellájának a tartalma. Egy cellán belül lehet sort is törni. Ilyenkor a cella tartalmát kapcsos zárójelek között kell megadni, a sortörést pedig a `\backslash` parancssal. Például

```
\begin{tblr}{hlines,vlines}
egy & kettő & {három\ négy} \\
öt & {hat\ hétf\ nyolc} & kilenc \\
\end{tblr}
```

egy	kettő	három négy
öt	hat hét nyolc	kilenc

Az `<opción>` a táblázat paramétereinek a beállítására szolgáló opciók listája vesszővel elválasztva. A lehetséges `<opción>`-at funkciójuk szerint csoportosítva adjuk meg.

Vízszintes vonalakat beállító opciók

Vízszintes vonalakat a következő opcióval lehet húzni és a tulajdonságokat beállítani.

hline{<számok>}=&{<szám>}{<oszlopszámok>}{<htul>}

<számok> Ezzel adjuk meg, hogy mely sorszámú vízszintes vonalakat húzza meg. Az 1. sorszámú vonal az 1. sor felett van, a 2. sorszámú vonal a 2. sor felett van, és így tovább, míg az utolsó sorszámú vonal az utolsó sor alatt van. A **<számok>** helyén lehet

- egyetlen szám;
- vesszővel elválasztott számok (pl. 1,3,4);
- „tól-ig” típusú (pl. 1-5);
- **X** kettővel az utolsó előtti;
- **Y** eggyel az utolsó előtti;
- **Z** utolsó;
- Ezek keverhetőek is (pl. 1,3,6-10,15-Z);
- 1-Z helyett írható – is;
- **odd** minden páratlan;
- **even** minden páros.

<szám> Adott sorszámú vonal helyén több vonal is húzható. Felülről az 1. megadásánál a **<szám>** értéke 1, felülről a 2. megadásánál a **<szám>** értéke 2, stb.

<oszlopszámok> Ezzel adjuk meg, hogy a vízszintes vonalak mely oszlopokat keresztezzék. Ennek szintaxisa megegyezik a **<számok>**-éval.

<htul> A meghúzott vízszintes vonalak tulajdonságainak a listája vesszővel elválasztva. A lehetséges tulajdonságok:

solid A vonalak folytonosak (alapértelmezett).

dashed A vonalak szaggatottak.

dotted A vonalak pontozottak.

text=<kód> A vonalak helyére **<kód>** kerül.

<vastagság> A vonal vastagsága (alapérték 0.4pt).

<színnév> A vonal színének a neve.

leftpos=<érték> Ha az **<érték>**

- 1** a vonalak a cellák tartalmának bal oldalától indulnak, a cellák bal margóját nem keresztezi;
- 0** az oszlopokat elválasztó esetlegesen több függőleges vonal esetén a vízszintes vonal az utolsó függőleges vonaltól indul;
- 1** az oszlopokat elválasztó esetlegesen több függőleges vonal esetén a vízszintes vonal az első függőleges vonaltól indul (alapérték).

rightpos=<érték> Ha az **<érték>**

- 1** a vonalak a cellák tartalmának jobb oldaláig tartanak, a cellák jobb margóját nem keresztezi;
- 0** az oszlopokat elválasztó esetlegesen több függőleges vonal esetén a vízszintes vonal az első függőleges vonalig tart;
- 1** az oszlopokat elválasztó esetlegesen több függőleges vonal esetén a vízszintes vonal az utolsó függőleges vonalig tart (alapérték).

endpos A **leftpos** és **rightpos** opciók nem cellánként hatnak, hanem csak az elsőre és az utolsóra. (Nem alapértelmezett.)

A **hline{<számok>}=&{<szám>}{<oszlopszámok>}{<htul>}** opcióból a következő rövidítések használhatók:

rövidítés

={1}{<oszlopszámok>}{{htul}}	={<oszlopszámok>}{{htul}}
={1}{-}{{htul}}	={{htul}}
={1}{-}{}{}	(elhagyható)
hline{-}	hlines

Példák

```
\begin{tblr}[hlines]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}[hlines={2pt,red}]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}[hline{1,3}={2pt,red},
           hline{2}]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}[hline{1,3}={2pt,red},
           hline{2}={leftpos=-1,rightpos=-1}]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}[hline{1,3}={2pt,red},
           hline{2}={leftpos=-1,rightpos=-1,endpos}]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

```
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}{hline{1,3}={1,3}{2pt,red},  
          hline{1,3}={2}{1,3}{blue,dashed}}  
egy & kettő & három \\  
négy & öt & hat \\  
\end{tblr}
```

egy	kettő	három
négy	öt	hat

Függőleges vonalakat beállító opciók

Függőleges vonalakat a következő opcióval lehet húzni és a tulajdonságokat beállítani.

`vline{<számok>}={<szám>}{<sorok számai>}{<vtul>}`

<számok> Ezzel adjuk meg, hogy mely sorszámú függőleges vonalakat húzza meg. Az 1. sorszámú vonal az 1. oszlop előtt van, a 2. sorszámú vonal a 2. oszlop előtt van, és így tovább, míg az utolsó sorszámú vonal az utolsó oszlop után van. A **<számok>** szintaxisa megegyezik vízszintes vonalaknál leírtakéval.

<szám> Adott sorszámú vonal helyén több vonal is húzható. Balról az 1. megadásánál a **<szám>** értéke 1, balról a 2. megadásánál a **<szám>** értéke 2, stb.

<sorok számai> Ezzel adjuk meg, hogy a függőleges vonalak mely sorokat keresztezzék. Ennek szintaxisa megegyezik a **<számok>**-éval.

<vtul> A meghúzott vízszintes vonalak tulajdonságainak a listája vesszővel elválasztva. A lehetséges tulajdonságok:

solid A vonalak folytonosak (alapértelmezett).

dashed A vonalak szaggatottak.

dotted A vonalak pontozottak.

text=<kód> A vonalak helyére **<kód>** kerül.

<vastagság> A vonal vastagsága (alapérték 0.4pt).

<színnév> A vonal színének a neve.

abovepos=<érték> Ha az **<érték>**

- 1 a vonalak a cellák tartalmának felső szélétől indulnak, a cellák felső margóját nem keresztezi;
- 0 a sorokat elválasztó esetlegesen több vízszintes vonal esetén a függőleges vonal a legalsó vízszintes vonaltól indul (alapérték);
- 1 a sorokat elválasztó esetlegesen több vízszintes vonal esetén a függőleges vonal a legfelső vízszintes vonaltól indul.

belowpos=<érték> Ha az **<érték>**

- 1 a vonalak a cellák tartalmának alsó széléig tartanak, a cellák alsó margóját nem keresztezi;
- 0 a sorokat elválasztó esetlegesen több vízszintes vonal esetén a függőleges vonal a legfelső vízszintes vonalig tart (alapérték);

- 1 a sorokat elválasztó esetlegesen több vízszintes vonal esetén a függőleges vonal a legalsó vízszintes vonalig tart.

A `vline{<számok>}={<szám>}{<sorok számai>}{<vtul>}` opcionálisan a következő rövidítések használhatók:

rövidítés	
={1}{<sorok számai>}{<vtul>}	={<sorok számai>}{<vtul>}
={1}{-}{<vtul>}	={<vtul>}
={1}{-}{}	(elhagyható)
vline{-}	vlines

Példák

```
\begin{tblr}{vlines}
egy & kettő & három \\
négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}{vlines={2pt,red}}
egy & kettő & három \\
négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}{vline{1,4}={2pt,red},vline{2,3}}
egy & kettő & három \\
négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}{vline{1,4}={2pt,red},vline{2,3}={abovepos=-1,belowpos=-1}}
egy & kettő & három \\
négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}{}
vline{1} = {2pt,red},
```

```

    vline{1} = {2}{2}{2pt,blue},
    vline{4} = {1}{2pt,blue},
    vline{4} = {2}{-}{2pt,red},
}
egy & kettő & három \\
négy & öt & hat \\
\end{tblr}

```

egy	kettő	három
négy	öt	hat

Cellákat beállító opciók

A cellák tulajdonságait a következő opcióval lehet beállítani.

`cellf{<sorok számai>}{|<oszlopok számai>}={<összevonás>}{|<ctul>}`

A `<sorok számai>`-val megadott sorok, illetve az `<oszlopok számai>`-val megadott oszlopok kereszteződéseiben álló cellák tulajdonságai. A `<sorok számai>`-nak illetve `<oszlopok számai>`-nak a szintaxisa megegyezik a vízszintes vonalaknál leírt `<számok>` szintaxisával.

`<összevonás>` Az adott cellát hány sorral és oszloppal vonjuk össze. Az értéke egy maximum két elemű lista, ahol a listaelemek:

`r=<szám>` Összevonandó sorok száma (alapérték 1).

`c=<szám>` Összevonandó oszlopok száma (alapérték 1).

`<ctul>` Az adott cellák tulajdonságainak a listája vesszővel elválasztva. A lehetséges tulajdonságok:

`l` A cella tartalma balra zárt.

`r` A cella tartalma jobbra zárt.

`c` A cella tartalma vízszintesen középre zárt.

`j` A cella tartalma sorkizárt (alapérték).

`t` A cella alapvonala a tartalma első sorának az alapvonala lesz (alapérték).

`b` A cella alapvonala a tartalma utolsó sorának az alapvonala lesz.

`m` A cella alapvonala a tartalmának a középvonala lesz.

`h` A cella tartalma a cella tetejére kerül.

`f` A cella tartalma a cella aljára kerül.

`<szélesség>` A cella szélessége margók nélkül (alapérték a cellatartalmának természetes szélessége.)

`<színnév>` A cella háttérszínének a neve.

`fg=<színnév>` A cella betűszínének a neve.

`font={<fonttípus>}` A cella fontjának a típusa. A `<fonttípus>` egy deklarációs parancs (pl. `font=\bfseries`).

`mode=<üzemmód>` minden cella alapértelmezetten olyan üzemmódban van, amilyen a környezete (szöveges/matematika). Az `<üzemmód>` lehetséges értékei:

`text` Átvált szöveges módra.

`math` Átvált sorközi matematikai módra.

`dmath` Átvált kiemelt matematikai módra.

`cmd=<parancs>` A `<parancs>` egy paraméteres parancs a paramétere nélkül. A cella tartalma ennek a parancsnak a paraméterébe íródik, és aszerint jelenik meg (pl. `cmd=\fbox`).

preto=*<szöveg>* A cella tartalma elő a *<szöveg>* lesz helyezve.

appto=*<szöveg>* A cella tartalma után a *<szöveg>* lesz helyezve.

A **cell{<sorok számai>}&{<oszlopok számai>}=&{<összevonás>}&{<ctul>}** opcióban a következő rövidítések használhatók:

	rövidítés
={}&{<ctul>}	={<ctul>}
cell{-}{-}	cells

Példák

```
\begin{tblr}[vlines,hlines,cells={c,font=\itshape}]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}[vlines,hlines,cell{-}{1,3}={c,font=\itshape}]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}[cells={c},cell{1}{odd}={yellow},cell{2}{2}={lime}]{}
  egy & kettő & három \\
  négy & öt & hat \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat

```
\begin{tblr}[vlines,hlines,cell{1}{2}={c=2,c,yellow}]{}
  egy & kettő & \\
  három & négy & öt \\
\end{tblr}
```

egy	kettő	
három	négy	öt

```
\begin{tblr}[vlines,hlines,cell{1}{2}={c=2,r=2,c,yellow}]{}
  egy & kettő & \\
  három & & \\
  négy & öt & hat \\
\end{tblr}
```

```
\end{tblr}
```

egy	kettő	
három		
négy	öt	hat

Sorokat beállító opciók

A sorok tulajdonságait a következő opcióval lehet beállítani.

```
row{<sorok számai>}={<stul>}
```

A *<sorok számai>*-val megadott sorok tulajdonságai. A *<sorok számai>*-nak a szintaxisa megegyezik a vízszintes vonalaknál leírt *<számok>* szintaxisával.

<stul> Az adott sorok tulajdonságainak a listája vesszővel elválasztva. A lehetséges tulajdonságok:

- l** A sor minden cellájának tartalma balra zárt.
- r** A sor minden cellájának tartalma jobbra zárt.
- c** A sor minden cellájának tartalma középre zárt.
- j** A sor minden cellájának tartalma sorkizárt (alapérték).
- t** A sor minden cellájának alapvonala a tartalmuk első sorának az alapvonala lesz (alapérték).
- b** A sor minden cellájának alapvonala a tartalmuk utolsó sorának az alapvonala lesz.
- m** A sor minden cellájának alapvonala a tartalmuk középvonala lesz.
- h** A sor minden cellájának tartalma a cella tetejére kerül.
- f** A sor minden cellájának tartalma a cella aljára kerül.

(magasság) A sorok magassága margók nélkül (alapérték a sorok tartalmának természetes magassága.)

(színnév) A sorok háttérszínének a neve.

fg=<színnév> A sorok betűszínének a neve.

font=<fonttípus> A sorok fontjának a típusa. A *<fonttípus>* egy deklarációs parancs (pl. **font=\bfseries**).

mode=<üzemmód> minden sor alapértelmezetten olyan üzemmódban van, amilyen a környezete (szöveges/matematika). Az *<üzemmód>* lehetséges értékei:

text Átvált szöveges módra.

math Átvált sorközi matematikai módra.

dmath Átvált kiemelt matematikai módra.

cmd=<parancs> A *<parancs>* egy paraméteres parancs a paramétere nélkül. A sorok celláinak tartalma ennek a parancsnak a paraméterébe íródik, és aszerint jelenik meg (pl. **cmd=\fbox**).

preto=<szöveg> A sorok celláinak tartalma elő a *<szöveg>* lesz helyezve.

appto=<szöveg> A sorok celláinak tartalma után a *<szöveg>* lesz helyezve.

abovesep=<távolság> A sorok felső margójának a magassága (alapérték 2pt).

abovesep+=<távolság> A sorok felső margója ennyivel megnő.

belowsep=<távolság> A sorok alsó margójának a magassága (alapérték 2pt).

belowsep+=<távolság> A sorok alsó margója ennyivel megnő.

rowsep=<távolság> A sorok alsó és felső margójának a magassága (alapérték 2pt).

rowsep+=⟨távolság⟩ A sorok alsó és felső margója ennyivel megnő.

A **row{⟨sorok számai⟩}={⟨stul⟩}** opcióban a következő rövidítés használható:

rövidítés
row{-} rows

Például

```
\begin{tblr}{row{odd}={lime},
           rows={c,fg=blue}}
  egy & kettő & három \\
  négy & öt & hat \\
  hét & nyolc & kilenc \\
\end{tblr}
```

egy kettő három
négy öt hat
hét nyolc kilenc

További rövidítések:

abovesep=⟨távolság⟩

ugyanaz, mint a **rows={abovesep=⟨távolság⟩}** opció.

belowsep=⟨távolság⟩

ugyanaz, mint a **rows={belowsep=⟨távolság⟩}** opció.

rowsep=⟨távolság⟩

ugyanaz, mint a **rows={rowsep=⟨távolság⟩}** opció.

Sor specifikációk ♦ A sorok és vízszintes vonalak beállítása sok esetben gyorsabban és egyszerűbben történhet az eddigiek nél a

rowspec={⟨specifikációk⟩}

opcional. A **⟨specifikációk⟩** egy felsorolás, amely minden sornak és vízszintes vonalnak megadja a tulajdonságait. Például

rowspec={|1|rr|}

azt jelenti, hogy a táblázat 3 sorból áll, az elsőnek balra (**1**), a másodiknak és a harmadiknak pedig jobbra (**r**) van igazítva a tartalma. Az első sor alatt és felett, illetve a harmadik sor alatt van vízszintes vonal (**|**). A lehetséges **⟨specifikációk⟩**:

Q[⟨stul⟩] Az adott sor cellái **⟨stul⟩** tulajdonságúak. A **⟨stul⟩** leírását lásd a **row** opciójánál.

Q Az adott sor cellái alaptulajdonságúak.

1 Ugyanaz, mint **Q[1]**.

r Ugyanaz, mint **Q[r]**.

c Ugyanaz, mint **Q[c]**.

t{⟨szélesség⟩} Ugyanaz, mint **Q[t,⟨szélesség⟩]**.

m{⟨szélesség⟩} Ugyanaz, mint **Q[m,⟨szélesség⟩]**.

b{⟨szélesség⟩} Ugyanaz, mint **Q[b,⟨szélesség⟩]**.

h{⟨szélesség⟩} Ugyanaz, mint **Q[h,⟨szélesség⟩]**.

f{⟨szélesség⟩} Ugyanaz, mint **Q[f,⟨szélesség⟩]**.

| [⟨htul⟩] Az adott vízszintes vonal ⟨htul⟩ tulajdonságú. A ⟨htul⟩ leírását lásd a vízszintes vonalaknál.

| Ugyanaz, mint | [] .

@{⟨kód⟩} a sor cellái alá vagy felé ⟨kód⟩ kerül. A @{ } azt eredményezi, hogy nincs semmi, még térköz sem.

Például

```
\begin{tblr}{rowspec={|cc| [2pt,red]Q[c,lime]@{}}}
  egy & kettő & három \\
  négy & öt & hat \\
  hétfő & nyolc & kilenc \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat
hétfő	nyolc	kilenc

A rowspec={⟨specifikációk⟩} opcióhoz új specifikáció a

\NewRowType

parancsal definiálható, melynek szintaxisa megegyezik a \newcommand szintaxisával, annyi különbséggel, hogy itt egy betű jelentését adjuk meg, nem egy parancsét. Például

```
\NewRowType{R}[1][] {Q[red,#1]}
```

után az R specifikáció ugyanaz lesz, mint Q[red] (sor celláinak háttérszíne piros), illetve az R[fg=white] specifikáció ugyanaz lesz, mint Q[red,fg=white] (sor celláinak háttérszíne piros, betűszíne fehér).

Oszlopokat beállító opciók

Az oszlopok tulajdonságait a következő opcióval lehet beállítani.

column{⟨oszlopok számai⟩}={⟨otul⟩}

Az ⟨oszlopok számai⟩-val megadott oszlopok tulajdonságai. Az ⟨oszlopok számai⟩-nak a szintaxisa megegyezik a vízszintes vonalaknál leírt ⟨számok⟩ szintaxisával.

⟨otul⟩ Az adott oszlopok tulajdonságainak a listája vesszővel elválasztva. A lehetséges tulajdonságok:

- l Az oszlop minden cellájának tartalma balra zárt.
 - r Az oszlop minden cellájának tartalma jobbra zárt.
 - c Az oszlop minden cellájának tartalma középre zárt.
 - j Az oszlop minden cellájának tartalma sorkizárt (alapérték).
 - t Az oszlop minden cellájának alapvonala a tartalmuk első sorának az alapvonala lesz (alapérték).
 - b Az oszlop minden cellájának alapvonala a tartalmuk utolsó sorának az alapvonala lesz.
 - m Az oszlop minden cellájának alapvonala a tartalmuk középvonala lesz.
 - h Az oszlop minden cellájának tartalma a cella tetejére kerül.
 - f Az oszlop minden cellájának tartalma a cella aljára kerül.
- ⟨szélesség⟩ Az oszlopok szélessége margók nélkül (alapérték az oszlopok tartalmának természetes szélessége.)

<színnév> Az oszlopok háttérszínének a neve.

fg=<színnév> Az oszlopok betűszínének a neve.

font={<fonttípus>} Az oszlopok fontjának a típusa. A *<fonttípus>* egy deklarációs parancs (pl. *font={\bfseries}*).

mode=<üzemmód> minden oszlop alapértelmezetten olyan üzemmódban van, amilyen a környezete (szöveges/matematika). Az *<üzemmód>* lehetséges értékei:

- text* Átvált szöveges módra.
- math* Átvált sorközi matematikai módra.
- dmath* Átvált kiemelt matematikai módra.

cmd=<parancs> A *<parancs>* egy paraméteres parancs a paramétere nélkül. Az oszlopok celláinak tartalma ennek a parancsnak a paraméterébe íródik, és aszerint jelenik meg (pl. *cmd=\fbox*).

preto=<szöveg> Az oszlopok celláinak tartalma elő a *<szöveg>* lesz helyezve.

appto=<szöveg> Az oszlopok celláinak tartalma után a *<szöveg>* lesz helyezve.

leftsep=<távolság> Az oszlopok bal margójának a szélessége (alapérték 6pt).

leftsep+=<távolság> Az oszlopok bal margója ennyivel megnő.

rightsep=<távolság> Az oszlopok jobb margójának a szélessége (alapérték 6pt).

rightsep+=<távolság> Az oszlopok jobb margója ennyivel megnő.

colsep=<távolság> Az oszlopok jobb és bal margójának a szélessége (alapérték 6pt).

colsep+=<távolság> Az oszlopok jobb és bal margója ennyivel megnő.

A *column{<oszlopok számai>}={<otul>}* opcióban a következő rövidítés használható:

	rövidítés
	<u>column{-} columns</u>

Például

```
\begin{tblr}{column{odd}={lime},
           columns={c,fg=blue}}
  egy & kettő & három \\
  négy & öt & hat \\
  hét & nyolc & kilenc \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat
hét	nyolc	kilenc

További rövidítések:

leftsep=<távolság>

ugyanaz, mint a *columns={leftsep=<távolság>}* opció.

rightsep=<távolság>

ugyanaz, mint a *columns={rightsep=<távolság>}* opció.

colsep=<távolság>

ugyanaz, mint a *columns={colsep=<távolság>}* opció.

Oszlop specifikációk ♦ Az oszlopok és függőleges vonalak beállítása sok esetben gyorsabban és egyszerűbben történhet az eddigieknel a

```
colspec={⟨specifikációk⟩}
```

opciónal. A ⟨specifikációk⟩ egy felsorolás, amely minden oszlopnak és függőleges vonalnak megadja a tulajdonságait. Például

```
colspec={|l|rr|}
```

azt jelenti, hogy a táblázat 3 oszloból áll, az elsőnek balra (l), a másodiknak és a harmadiknak pedig jobbra (r) van igazítva a tartalma. Az első oszlop mindkét oldalán, illetve a harmadik oszlop jobb oldalán van függőleges vonal (r). Amennyiben az ⟨opcionál⟩-ban csak a colspec={⟨specifikációk⟩} szerepel, akkor ehelyett írható röviden ⟨specifikációk⟩. Tehát például

```
\begin{tblr}{colspec={|l|rr|}}
```

írható

```
\begin{tblr}{|l|rr|}
```

módon is. A lehetséges ⟨specifikációk⟩:

Q[⟨otul⟩] Az adott oszlop cellái ⟨otul⟩ tulajdonságúak. Az ⟨otul⟩ leírását lásd a column opciónál.

Q Az adott oszlop cellái alaptulajdonságúak.

1 Ugyanaz, mint Q[1].

r Ugyanaz, mint Q[r].

c Ugyanaz, mint Q[c].

t{⟨szélesség⟩} Ugyanaz, mint Q[t,⟨szélesség⟩].

m{⟨szélesség⟩} Ugyanaz, mint Q[m,⟨szélesség⟩].

b{⟨szélesség⟩} Ugyanaz, mint Q[b,⟨szélesség⟩].

h{⟨szélesség⟩} Ugyanaz, mint Q[h,⟨szélesség⟩].

f{⟨szélesség⟩} Ugyanaz, mint Q[f,⟨szélesség⟩].

X[⟨otul*⟩] Az ⟨otul*⟩ ugyanaz, mint az ⟨otul⟩ az oszlopoknál, csak egy módosítással. Itt a ⟨szélesség⟩ helyett egy arányszámot kell beírni, amely alapesetben 1. Ekkor ez az oszlop olyan széles lesz, hogy a táblázat teljes szélessége a width (lásd később) opciónban megadott értékű legyen. Az arányszámnak akkor van jelentősége, ha több X típusú oszlopot adunk meg. Akkor az így megadott oszlopok szélességei az adott arányúak lesznek egymáshoz viszonyítva.

X Ugyanaz, mint X[1] vagy X[].

|[⟨vtul⟩] Az adott függőleges vonal ⟨vtul⟩ tulajdonságú. A ⟨vtul⟩ leírását lásd a függőleges vonalaknál.

| Ugyanaz, mint |[].

@{⟨kód⟩} az oszlop cellái elő vagy után ⟨kód⟩ kerül. A @{} azt eredményezi, hogy nincs semmi, még térköz sem.

Például

```
\begin{tblr}{colspec={|cc| [2pt,red]Q[c,lime]@{},  
cell{2}{2}=[red,fg=white]}  
egy & kettő & három \\  
négy & öt & hat \\  
hét & nyolc & kilenc \\  
\end{tblr}
```

egy	kettő	három
négy	öt	hat
hét	nyolc	kilenc

Amennyiben használ X oszlopspecifikációt, akkor a táblázat szélessége a

`width={<szélesség>}`

opciónal adható meg. Enélkül az értéke \linewidth, azaz annak a doboznak a szélessége, amelyben a táblázat van. Például

```
\begin{tblr}[colspec={|X[1]|X[2]|X[3]|},hlines]
  egy & kettő & három \\
  négy & öt & hat \\
  hét & nyolc & kilenc \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat
hét	nyolc	kilenc

```
\begin{tblr}[colspec={|X[1]|X[2]|X[3]|},width=12cm,hlines]
  egy & kettő & három \\
  négy & öt & hat \\
  hét & nyolc & kilenc \\
\end{tblr}
```

egy	kettő	három
négy	öt	hat
hét	nyolc	kilenc

A `colspec={<specifikációk>}` opcióhoz új oszlopspecifikáció a

`\NewColumnType`

parancsal definiálható, melynek szintaxisa megegyezik a `\newcommand` szintaxisával, annyi különbséggel, hogy itt egy betű jelentését adjuk meg, nem egy parancsét. Például

`\NewColumnType{R}[1][][Q[red,#1]]`

után az R specifikáció ugyanaz lesz, mint Q[red] (sor celláinak háttérszíne piros), illetve az R[fg=white] specifikáció ugyanaz lesz, mint Q[red,fg=white] (sor celláinak háttérszíne piros, betűszíne fehér).

Egyéb opciók

`rulesep={távolság}` Ha két oszlop vagy sor közé több vonalat húz, akkor ez lesz két vonal távolsága (alapérték 2pt).

`stretch={arányszám}` Függőleges széthúzás arányszáma (alapérték 1).

`verb` Ha cellákba verbatim szöveget is akar írni.

`baseline=<hová>` Hová kerüljön a táblázat alapvonala. A `<hová>` értékei:

`t` Táblázat teteje.

- T** Táblázat első sorának alapvonala.
- b** Táblázat alja.
- B** Táblázat utolsó sorának alapvonala.
- m** Táblázat közepe.
- szám*** Táblázat ***szám***-adik sorának alapvonala.

hspan=even Azok az oszlopok, amelyeknél ez lehetséges, egyenlő szélességűek lesznek.

hspan=minimal A táblázatot a lehető legkisebb szélességre állítja, pl. az összevont oszlopok tartalmának több sorba törésével.

vspan=even Azok a sorok, amelyeknél ez lehetséges, egyenlő magasságúak lesznek.

Alapopciók átállítása

Amennyiben az ***opciónk***-ban az alapopciókat meg akarja változtatni, akkor használja a

```
\SetTblrInner{\alapopciók}
```

lokális hatású parancsot. Ezután

```
\begin{tblr}{}
```

ugyanazt fogja jelenteni, mint

```
\begin{tblr}{\alapopciók}
```

Például

```
\SetTblrInner{vlines,hlines}
```

után a táblázatokban alapértelmezetten lesznek függőleges és vízszintes vonalak.

Táblázatadatok

A következő parancsokkal kiírathatjuk a táblázat adatait:

```
\therownum % az aktuális sor száma
\theclnum % az aktuális oszlop száma
\therowcount % a táblázat sorainak a száma
\theclcount % a táblázat oszlopainak a száma
```

Új táblázatkörnyezet definiálása

A **tblr** környezet helyett új táblázatkörnyezet is definiálható

```
\NewTblrEnviron{\környezetnév}
\SetTblrInner[\környezetnév]{\alapopciók}
```

módon. Ezután a

```
\begin{\környezetnév}{\opciónk} ... \end{\környezetnév}
```

ugyanazt fogja jelenteni, mint

```
\begin{tblr}{\alapopciók,\opciónk} ... \end{tblr}
```

10.2.2. Hosszú tabulararray táblázatok

Ha a táblázat olyan hosszú, hogy nem fér ki az adott oldalon, azaz szükség van oldaltörésre, akkor használja a **tabulararray** csomag **longtblr** környezetét. Ennek általános formája a következő:

```
\begin{longtblr}[\langle külső specifikációk\rangle]{\langle opciók\rangle} \in tabulararray
  \langle cella 1,1\rangle & \langle cella 1,2\rangle & \langle cella 1,3\rangle & ... & \langle cella 1,m\rangle \\
  \langle cella 2,1\rangle & \langle cella 2,2\rangle & \langle cella 2,3\rangle & ... & \langle cella 2,m\rangle \\
  ...
  \langle cella n,1\rangle & \langle cella n,2\rangle & \langle cella n,3\rangle & ... & \langle cella n,m\rangle \\
\end{longtblr}
```

Az *\langle opciók\rangle* ugyanazok, mint a `tblr` környezet esetében, de a lehetséges opciók sora kiegészül még a következőkkel:

`rowhead=\langle szám\rangle` Oldaltörés esetén a táblázat első *\langle szám\rangle* sora megismétlődik minden oldal tetején. Alapértéke 0.

`rowfoot=\langle szám\rangle` Oldaltörés esetén a táblázat utolsó *\langle szám\rangle* sora megismétlődik minden oldal alján. Alapértéke 0.

Mivel a hosszú táblázat több oldalra törhető, ezért az úsztatása (lásd a 11. fejezetben) indokolatlan. A táblázat feliratozása a *\langle külső specifikációk\rangle* opciólistában adható meg több más paraméterrel együtt. A *\langle külső specifikációk\rangle* opciólistában a listaelemeket vesszővel kell elválasztani. A lehetséges elemei:

`caption=\langle cím\rangle` A táblázat címe. Enélkül nincs cím, csak táblázatszámozás.

`entry=\langle cím a táblázatok jegyzékében\rangle` A táblázat rövid címe, ami a táblázatok jegyzékébe kerül. Enélkül ez megegyezik a *\langle cím\rangle*-mel.

`entry=none` Nem kerül be cím a táblázatok jegyzékébe.

`label=\langle címke\rangle` Kereszthivatkozás esetén ezzel a címkelvel utalhatunk a táblázatra.

`label=none` Nem lesz felirata és számozása a táblázatnak.

`headsep=\langle távolság\rangle` A táblázat címe és a táblázat teteje közötti távolság.

`footsep=\langle távolság\rangle` A táblázat alja és a `contfoot-text` (lásd később) közötti távolság.

`presep=\langle távolság\rangle` A táblázatot (és annak címét, ha van) megelőző szöveget követő függőleges térköz mérete.

`postsep=\langle távolság\rangle` A táblázat és az azt követő szöveg közötti függőleges térköz mérete.

A `longtblr` környezet használata előtt be tudunk állítani még néhány fontos paramétert.

A táblázatok címében a táblázat számozása nem a magyar tipográfia szerint történik, még a `babel` és a `caption` csomagok használata esetén sem. A következő két sorral ezt lehet helyreenni.

```
\DefTblrTemplate{caption-tag}{default}{\thetable.\~\tablename}
\DefTblrTemplate{caption-sep}{default}{.\enskip}
```

A következő kódokkal a cím, címke és az elválasztó fonttípusát állíthatja be:

```
\SetTblrStyle{caption-text}{font=\langle fonttípus\rangle} % cím fonttípusa
\SetTblrStyle{caption-tag}{font=\langle fonttípus\rangle} % címke fonttípusa
\SetTblrStyle{caption-sep}{font=\langle fonttípus\rangle} % elválasztó fonttípusa
```

Ha a táblázat több oldalra török, akkor minden megtört táblázatrész aljára érdemes egy figyelmeztető feliratot illeszteni a következő kóddal:

```
\DefTblrTemplate{contfoot-text}{default}{\langle szöveg\rangle}
\SetTblrStyle{contfoot}{\langle igazítás\rangle,\langle szín\rangle,font=\langle fonttípus\rangle}
```

A *\langle szöveg\rangle* lesz a figyelmeztető felirat (pl. „Folytatás a következő oldalon!”). Az *\langle igazítás\rangle* lehet `c`, `l` vagy `r` aszerint, hogy a középre, balra vagy jobbra akarja igazítani a *\langle szöveg\rangle*-et (alapérték `r`). A *\langle szín\rangle* a *\langle szöveg\rangle* színének a neve, illetve a *\langle fonttípus\rangle* a *\langle szöveg\rangle* fonttípusa. Ha nem akar ilyen figyelmeztetést, akkor írja ezt:

```
\DefTblrTemplate{contfoot-text}{default}{}{}
```

Alapesetben minden oldaltörés után a táblázat száma és címe ismét megjelenik a lap tetején. Ilyenkor a cím után érdemes egy figyelmeztetést elhelyezni, hogy ez a táblázat az előző oldal folytatása.

```
\DefTblrTemplate{conthead-text}{default}{<szöveg>}
\SetTblrStyle{conthead-text}{<szín>,font=<fonttípus>}}
```

A *<szöveg>* lesz a figyelmeztető felirat (pl. „(Folytatás)”). A *<szín>* a *<szöveg>* színének a neve, illetve a *<fonttípus>* a *<szöveg>* fonttípusa. Ha nem akar ilyen figyelmeztetést oldaltörésnél, csak címet, akkor írja ezt:

```
\DefTblrTemplate{conthead-text}{default}{}{}
```

Ha oldaltörésnél nem akarja megismételni a címet, csak a figyelmeztetést szeretné kiírni, akkor használja a következő kódot:

```
\DefTblrTemplate{conthead-text}{default}{<szöveg>}
\SetTblrStyle{conthead}{<igazítás>,<szín>,font=<fonttípus>}}
\DefTblrTemplate{middlehead}{default}{\UseTblrTemplate{conthead}{default}}
\DefTblrTemplate{lasthead}{default}{\UseTblrTemplate{conthead}{default}}
```

A *<szöveg>* lesz a figyelmeztető felirat (pl. „(Folytatás)”). Az *<igazítás>* lehet **c**, **l** vagy **r** aszerint, hogy a középre, balra vagy jobbra akarja igazítani a *<szöveg>*-et (alapérték **r**). A *<szín>* a *<szöveg>* színének a neve, illetve a *<fonttípus>* a *<szöveg>* fonttípusa.

Ha oldaltörésnél nem akarja megismételni a címet, és figyelmeztetést sem szeretne, akkor írja be a következő kódot.

```
\DefTblrTemplate{middlehead}{default}{}{}
\DefTblrTemplate{lasthead}{default}{}{}
```

Magyar nyelvű dokumentum esetén egy tipikus beállítás lehet a következő:

```
\DefTblrTemplate{caption-tag}{default}{\thetable.\~\tablename}
\DefTblrTemplate{caption-sep}{default}{.\enskip}
\SetTblrStyle{caption-tag}{font=\bfseries}
\SetTblrStyle{caption-sep}{font=\bfseries}
\DefTblrTemplate{contfoot-text}{default}{Folytatás a következő oldalon!}
\SetTblrStyle{contfoot}{font=\itshape}
\DefTblrTemplate{middlehead}{default}{}{}
\DefTblrTemplate{lasthead}{default}{}{}
```

Példa

Következik egy példa, ami jól mutatja az eddigiek használatát:

```
</> \DefTblrTemplate{caption-tag}{default}{\thetable.\~\tablename}
\DefTblrTemplate{caption-sep}{default}{.\enskip}
\SetTblrStyle{caption-tag}{font=\bfseries}
\SetTblrStyle{caption-sep}{font=\bfseries}
\DefTblrTemplate{contfoot-text}{default}{Folytatás a következő oldalon!}
\SetTblrStyle{contfoot}{l,font=\itshape}
\DefTblrTemplate{conthead-text}{default}{A táblázat folytatása\ldots}
\SetTblrStyle{conthead}{font=\itshape}
\DefTblrTemplate{middlehead}{default}{%
\UseTblrTemplate{conthead}{default}}
\DefTblrTemplate{lasthead}{default}{\UseTblrTemplate{conthead}{default}}
```

Lásd `\az{\ref{tblr-teszt}}.`-táblázatban.

```

\begin{longtblr}[
    caption = {A táblázat címe},
    label = {tblr-teszt},
]
{
    colspec = {XXX},
    width = 12cm,
    rowhead = 1,
    row{odd} = {gray},
    row{even} = {lightgray},
    row{1} = {purple,font=\bfseries},
}
\begin{tblr}{lll}
Oszlopcím 1 & Oszlopcím 2 & Oszlopcím 2 \\
Alpha & Beta & Gamma \\
Delta & Epsilon & Zeta \\
Eta & Theta & Iota \\
Kappa & Lambda & Mu \\
Nu & Xi & Omicron \\
Pi & Rho & Sigma \\
Tau & Upsilon & Phi \\
Chi & Psi & Omega \\
\end{tblr}
% és így tovább
\end{longtblr}

```

11. fejezet

Objektumok úsztatása

A táblázatok és képek beillesztésénél előfordulhat, hogy azok az adott oldalon már nem férnek el, és a következő oldalra való áthelyezésével az oldal alja telítetlen marad. Ennek megoldására született az úgynevezett „úsztatás”. Ez azt jelenti, hogy az objektum az optimálisabb oldaltördeles érdekében nem feltétlenül a forrásban megadott helyre kerül, hanem egy általunk engedélyezett pozícióba: az aktuális oldal tetejére, aljára vagy külön oldalra. (Bővebben lásd [9].)

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.
- ! Figyelmen kívül hagy minden korlátozást, amely a területre helyezhető úsztatások számával vagy a terület által elfoglalható maximális mérettel kapcsolatos.

Opcionak ezen jelek bármilyen kombinációja használható. A feldolgozás sorrendje minden esetben `!htbp`. Tehát mindegy, hogy például `tb` vagy `bt` van opcióként megadva, először mindenkorban az oldal tetején próbálja elhelyezni az objektumot és csak azután alulra. 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!} \in float
```

Előfordulhat, hogy egy úszó objektum a lap tetejére kerülve az előző téma sorai közé kerül, ami nem szerencsés. Ilyenkor használja a

```
\suppressfloats[<opción>]
```

parancsot. Az ezután következő úszó objektum nem jelenhet meg az oldal `<opción>` 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 \in 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` parancssal zárul, ami megjeleníti az addig még függőben maradt úsztatásokat.

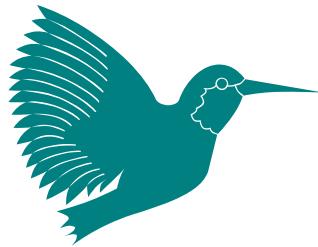
11.2. Úsztatás kéthasábos szedés esetén

Amikor az `article`, `report` vagy `book` standard dokumentumosztályokat `twocolumn` opcióval tölti be, akkor kéthasábos szedést kap. Ilyen esetben a `figure` és `table` úsztató környezetek az objektumot egy hasábban helyezik el. Az úsztató környezeteknek létezik csillagos verziója is, `figure*` és `table*`, melyeket pontosan úgy kell használni, mint a csillag nélküli verziókat, de ekkor az objektum a két hasábot keresztezi. Egyhasábos szedés, azaz `onecolumn` opció esetén a csillagos és csillag nélküli úsztató környezetek között nincs különbség.

Amennyiben a többlasábos szedést a `multicol` csomag `multicols` környezetével oldja meg, akkor az úsztató környezeteknek csak a csillagos verziója használható a `multicols` környezeten belül, továbbá annak opciójában nem szerepelhet `h` betű.

11.3. Úsztatott objektumok feliratozása

Ha a képekre, táblázatokra hivatkozni szeretnénk, célszerű azokat feliratozni, amely automatikus címkeszámot, címkenevet és címet tartalmaz. Például



1. ábra. A LATEX Project logója

A felirat részei az előző példában:



① címkeszám ② címkenév ③ címke ④ elválasztó ⑤ cím

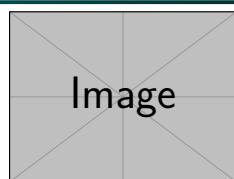
Ha ilyen objektumokból nagyon sok van, akkor az áttekinthetőség miatt célszerű ezen címeket táblázat- illetve ábrajegyzékben szerepelhetni oldalszám feltüntetésével, hasonlóan a tartalomjegyzékhez. Ezen feladatok elvégzésére szolgál a

`\caption[<jegyzékbbe kerülő cím]<{cím}]`

parancs. Az opció alapértéke megegyezik a `<cím>`-mel. Például

```
</> \begin{figure}[ht!]
    \centering
    \includegraphics[width=3cm]{example-image}
    \caption{Egy példa}\label{fig-pelda}
\end{figure}
\Aref{fig-pelda}.~ábrán látható \dots
```

Ennek hatására a képet megjeleníti középen és feliratozza. A címkenév aszerint lesz „ábra” illetve „táblázat” (illetve a babel csomag `english` opciója esetén „Figure” illetve „Table”), hogy `figure` vagy `table` környezetbe rakta a `\caption` parancsot. A `figure` környezet címkenevét a `\figurename`, a `table` környezet címkenevét a `\tablename` parancs tárolja. A címkeszám automatikus, amit a `figure` környezet esetén a `figure`, míg a `table` környezet esetén a `table` számláló tárol. (A számlálók kezelését lásd a 22.6. szakaszban.) A megadott cím bekerül a megfelelő jegyzékbe.



1. ábra. Egy példa

Az 1. ábrán látható ...

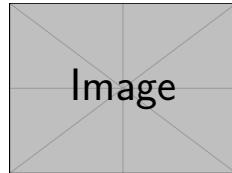
Ha nem akar számozást és címkenevet, csak címet, akkor használja a következőt:

`\caption*{\langle cím\rangle} \in \caption`

Ha nem akar címet adni, elég a számozás és a címkenév, akkor a `caption` csomag használata mellett tegye ezt:

</>

```
\begin{figure}[ht!]
\centering
\includegraphics[width=3cm]{example-image}
\caption{}\label{fig-pelda}
\end{figure}
```



1. ábra

Ha azt akarja, hogy a jegyzékbe más cím kerüljön mint a feliratba, akkor a jegyzékbe kerülő címet adja meg a \caption parancs opciójaként. Például

```
\caption[A \TeX szimbóluma]{A \TeX szimbóluma (tervezte Duane Bibby)}
```

A felirat az előző példában azért jelent meg a kép alatt, mert a \caption parancsot a kép betöltése után hívtuk meg. Ha elé írnánk, akkor a kép felett lenne a felirat. Ha a \caption parancs kiadásának helyétől függetlenül például a táblázatok esetében minden a táblázatok felett szeretné a feliratot, akkor használja a következő kódot:

```
\floatstyle{plaintop}\restylefloat{table} \in float
```

Lábjegyzet elhelyezése a \caption parancs argumentumában a szokott módon nem lehetséges. Azaz például a következő kód hibás:

```
\begin{figure}
\centering
\includegraphics[width=3cm]{example-image}
\caption{Kép címe\footnote{Lábjegyzet szövege.}}
\end{figure}
```

A megoldás a következő:

</>

```
\begin{figure}
\centering
\includegraphics[width=3cm]{example-image}
\caption[Kép címe jegyzékben]{Kép címe\footnotemark}
\end{figure}
```

vagy jegyzék használata nélkül

```
\begin{figure}
\centering
\includegraphics[width=3cm]{example-image}
\caption[]{}{Kép címe\footnotemark}
\end{figure}
```

Utóbbi esetben

```
\caption[]{}{Kép címe\footnotemark}
```

helyett a következő is írható:

```
\caption{Kép címe\protect\footnotemark}
```

Az eddigi módszerekkel a lábjegyzet szövege a normál lábjegyzetekhez hasonlóan a lap alján jelenik meg. Azonban a magyar tipográfiában elfogadottabb, ha közvetlenül a felirat alatt van. Ez a

```
\makeFootnotable{úsztató környezet} ∈ [magyar]babel
```

parancsal megoldható a következő példában található módon:

</>

```
\makeFootnotable{figure}
\begin{figure}
\centering
\includegraphics[width=3cm]{example-image}
\caption{Kép címe\protect\footnote{Lábjegyzet szövege.}}
\end{figure}
```

A `\caption` parancs argumentumába írt `\url` parancs (lásd a 14.4. szakaszban) elő mindig be kell írni a `\protect` parancsot, különben hibát kapunk. Például a `caption` és `url` csomagok betöltése után

</>

```
\begin{figure}
\centering
\includegraphics[width=3cm]{example-image}
\caption{Kép címe\ \
\protect\url{https://www.ctan.org/}}
\end{figure}
```



Videó: Képek és táblázatok

11.4. Úsztatott objektumok feliratainak testreszabása

A feliratok stílusának testreszabásához használja a következő parancsot:

```
\captionsetup[úsztató környezet]{opcion} ∈ caption
```

Az `úsztató környezet` lehet például `figure`, `table` vagy bármilyen úsztatott környezet neve, amit magunk definiálunk a későbbiekben ismertetett módszerrel. Ennek megadásával az `opcion` által megadott beállítások csak a `\captionsetup` parancs utáni `úsztató környezet` nevű úsztatott környezetekre vonatkoznak.

Ha az `úsztató környezet` nincs megadva, akkor az `opcion` által megadott beállítások az összes olyan úsztatott környezetre vonatkoznak, amely a `\captionsetup` parancs után van.

Több `\captionsetup` esetén a megadott `opcion` nem írják egymást felül, hanem összefűződnek, azaz minden egyik érvényesülni fog.

A `\captionsetup` hatása lokális, azaz ha például egy úsztatott környezetben van a `\caption` előtt, akkor az adott beállítás csak arra az egy `\caption`-re vonatkozik.

A következőkben az `opcion` közül sorolunk fel néhányat. A teljes leírást megtalálja a `caption` csomag dokumentációjában.

`format=`*formázásnév* A címke, elválasztó és cím elrendezését adja meg. Alapesetben két `formázásnév` van definiálva:

`plain` (alapopció) A felirat normál bekezdésként jelenik meg.

hang Többsoros felirat esetén a második sortól annyival lesz beljebb tolva, mint a címke és az elválasztó szélessége.

indentation=⟨hossz⟩ Többsoros felirat esetén a második sortól kezdve a sorok alapértelmezett behúzását a ⟨hossz⟩ értékével növeli meg. A ⟨hossz⟩ lehet negatív is. Emlékeztetőül, a második sortól kezdve a sorok alapértelmezett behúzása **format=plain** esetén 0pt, míg **format=hang** esetén a címke és az elválasztó szélessége.

labelformat=⟨formázásnév⟩ A címke felépítése. Több ⟨formázásnév⟩ is definiált alapból, most csak kettőt emlíünk meg:

original (alapopció) Ha a **babel** csomag **magyar** opciójával, a **magyar.ldf** pedig **defaults=hu-min** opciójával van betöltve, továbbá a címkenév nem üres, akkor a címke felépítése **címkészám**..**címkenév** (például 1. ábra).

Ha a **babel** csomag **english** opciójával van betöltve, továbbá a címkenév nem üres, akkor a címke felépítése **címkenév**..**címkészám** (például Figure 1).

Ha a címkenév üres, akkor nyelvtől függetlenül a címke felépítése **címkészám** lesz. (Ennek majd a **subcaption** csomagnál lesz jelentősége, ahol az alfeliratokban a címkenév üres.)

parens A címkészám zárójelben lesz.

empty A címke és az elválasztó nem jelenik meg.

További ⟨formázásnév⟩ is definiálható a **\DeclareCaptionLabelFormat** parancssal (lásd később).

labelsep=⟨elválasztásnév⟩ Az elválasztó típusa. Néhány előre definiált ⟨elválasztásnév⟩:

period Az elválasztó egy pont és utána egy szóköz. Ha a **babel** csomag **magyar** opciójával van betöltve, akkor ez az alapopció.

colon Az elválasztó egy kettőspont és utána egy szóköz. Ha a **babel** csomag **english** opciójával van betöltve, akkor ez az alapopció.

none Nincs elválasztó.

space Az elválasztó egy szóköz.

newline Az elválasztó egy sortörés. (A **format=hang** opciójával együtt nem használható.)

justification=⟨igazításnév⟩ A sorigazítás típusa. Néhány előre definiált ⟨igazításnév⟩:

justified (alapopció) A felirat sorkizárt.

centering A felirat középre zárt.

centerlast A felirat sorkizárt, kivéve a bekezdés utolsó sorát, ami középre zárt.

centerfirst A felirat sorkizárt, kivéve a bekezdés első sorát, ami középre zárt.

raggedright A felirat balra zárt.

raggedleft A felirat jobbra zárt.

singlelinecheck (alapopció) Ha a felirat egyetlen sorból áll, akkor az minden középre lesz igazítva, még akkor is, ha például a **justification=raggedright** opciót használjuk.

singlelinecheck=false Az egysoros feliratok is úgy lesznek igazítva, mint a többsorosak.

font={⟨fonttípus1⟩,⟨fonttípus2⟩,...} A felirat fonttípusa. Alapból definiált ⟨fonttípus⟩ értékek:

scriptsize A felirat előre **\scriptsize** parancsot illeszt be.

footnotesize A felirat előre **\footnotesize** parancsot illeszt be.

small A felirat előre **\small** parancsot illeszt be.

normalsize A felirat előre **\normalsize** parancsot illeszt be.

large A felirat előre **\large** parancsot illeszt be.

Large A felirat elő \Large parancsot illeszt be.

smaller Egy szinttel csökkenti a fontméretet. Ha például az aktuális beállítás font=small, akkor a font=smaller ugyanaz, mint a font=footnotesize.

larger Egy szinttel megnöveli a fontméretet. Ha például az aktuális beállítás font=small, akkor a font=larger ugyanaz, mint a font=normalsize.

normalfont A felirat elő \normalfont parancsot illeszt be.

up A felirat elő \upshape parancsot illeszt be.

it A felirat elő \itshape parancsot illeszt be.

sl A felirat elő \slshape parancsot illeszt be.

sc A felirat elő \scshape parancsot illeszt be.

md A felirat elő \mdseries parancsot illeszt be.

bf A felirat elő \bfseries parancsot illeszt be.

rm A felirat elő \rmfamily parancsot illeszt be.

sf A felirat elő \sfamily parancsot illeszt be.

tt A felirat elő \ttfamily parancsot illeszt be.

normalcolor A felirat elő \normalcolor ∈ **xcolor** parancsot illeszt be.

color=<színnév> A felirat elő \color{<színnév>} ∈ **xcolor** parancsot illeszt be.

singlespacing A felirat elő \singlespacing ∈ **setspace** parancsot illeszt be.

onehalfspacing A felirat elő \onehalfspacing ∈ **setspace** parancsot illeszt be.

doublespacing A felirat elő \doublespacing ∈ **setspace** parancsot illeszt be.

stretch=<sorköz> A felirat elő \setstretch{<sorköz>} ∈ **setspace** parancsot illeszt be.

normal (alapopció) \normalfont\normalsize parancsokat illeszt be a felirat elő.

Ha az **xcolor** csomag be van töltve, akkor még a \normalcolor parancsot is beilleszti. Ha a **setspace** csomag be van töltve, akkor még a \singlespacing parancsot is beilleszti.

font+={<fonttípus1>,<fonttípus2>,...} A font opcióhoz hasonlóan működik, de ez nem írja felül a korábbi fontbeállítást, hanem hozzáadódik.

labelfont={<fonttípus1>,<fonttípus2>,...} A címke és az elválasztó fonttípusa. A definiált <fonttípus> értékek ugyanazok, mint a font={<fonttípus1>,<fonttípus2>,...} esetében.

labelfont+={<fonttípus1>,<fonttípus2>,...} A labelfont opcióhoz hasonlóan működik, de ez nem írja felül a korábbi fontbeállítást, hanem hozzáadódik.

textfont={<fonttípus1>,<fonttípus2>,...} A cím fonttípusa. A definiált <fonttípus> értékek ugyanazok, mint a font={<fonttípus1>,<fonttípus2>,...} esetében.

textfont+={<fonttípus1>,<fonttípus2>,...} A textfont opcióhoz hasonlóan működik, de ez nem írja felül a korábbi fontbeállítást, hanem hozzáadódik.

width=<szélesség> A felirat dobozának szélessége <szélesség> lesz, továbbá a doboz középre lesz igazítva. Például width=10cm.

margin=<margó> A felirat dobozának bal széle a szövegtükör bal szélétől, illetve a felirat dobozának jobb széle a szövegtükör jobb szélétől <margó> távolságra lesz. Például margin=1cm.

margin={<balmargó>,<jobbmargó>} A felirat dobozának bal széle a szövegtükör bal szélétől <balmargó> távolságra lesz, illetve a felirat dobozának jobb széle a szövegtükör jobb szélétől <jobbmargó> távolságra lesz. Például margin={1cm,2cm}.

twoside (alapopció) Kétoldalas szedés esetén a margin={<balmargó>,<jobbmargó>} opcióban a páros oldalakon a bal és jobb margó felcserélődik. Más szavakkal, ekkor a <balmargó> a belső margót, a <jobbmargó> pedig a külső margót jelenti.

oneside Kétoldalas szedés esetén a `margin={⟨balmargó⟩,⟨jobbmargó⟩}` opcióban a páros oldalakon a bal és jobb margó nem cserélődik fel. Más szavakkal, ekkor a `⟨balmargó⟩` egy- és kétoldalas szedés esetén is minden oldalon a bal margót, a `⟨jobbmargó⟩` pedig a jobb margót jelenti.

skip=⟨hossz⟩ Az úsztatott objektum és a felirat függőleges távolsága `⟨hossz⟩` lesz, melynek alapértéke `10pt`.

list=⟨logikai érték⟩ A `⟨logikai érték⟩` alapesetben `true`. Ekkor a `⟨jegyzékbe kerülő cím⟩` bekerül a megfelelő jegyzékhez tartozó fájlba, amennyiben a jegyzéket megjelenítő parancs ki van adva (lásd a 17.6. szakaszban). Hogy ténylegesen is megjelenik-e a jegyzékbен, azt még az is befolyásolja, hogy a jegyzék szintmélysége nem kisebb-e mint az adott bejegyzés szintje. Ha a `⟨logikai érték⟩` `false`, akkor a `⟨jegyzékbe kerülő cím⟩` nem kerül be a megfelelő jegyzékhez tartozó fájlba.

listformat=⟨formázásnév⟩ Azt szabályozza, hogy a címkeszám milyen formátumban kerüljön be a megfelelő jegyzékhez tartozó fájlba, azon belül is a `\numberline` parancsba (lásd a 17.6. szakaszban). Ha a `babel` csomag `magyar` opciójával van betöltve, és például az ábra címkeszáma 1, akkor alapesetben a következő lesz beírva a lof kiterjesztésű fájl megfelelő pontján:

```
\numberline{1.}
```

Érdemes tudni, hogy a `magyar.lfd defaults=hu-min` opciója átdefiniálja a standard `\numberline` parancsot úgy, hogy amennyiben az argumentuma nem pontra végződik, akkor még egy pontot is rak a végére, ezzel biztosítva azt a magyar szabályt, hogy sorszám után pont kell. Mivel ebben az esetben a `caption` csomag tesz pontot a szám után, ezért a `magyar.lfd` ezen nem változtat, azaz az ábrák jegyzékeben „1.” formátumban jelenik meg a címkeszám.

A `⟨formázásnév⟩` lehetséges értékei:

subsimple (angol nyelvű dokumentumokban alapopció) Ekkor csak a címkeszám kerül a `\numberline` argumentumába. A `magyar.lfd defaults=hu-min` opciója esetén az átdefiniált `\numberline` miatt, mivel most nincs pont az argumentum végén, ezért még ez kiegészül egy ponttal, így a jegyzékben lesz pont a címkeszám után.

simple Annyiban különbözik a `subsimple` opciótól, hogy a címkeszám előtt berakja még az ún. prefixét is. A prefixet a

```
\p@⟨úsztató környezet⟩
```

parancs tárolja, amely általában üres. Ha `subcaption` csomagot használ (lásd később), akkor az alfeliratok címkeszámainak prefixe a

```
\p@sub⟨úsztató környezet⟩
```

parancsban található.

subparens A címkeszám kerek zárójelbe rakva kerül a `\numberline` argumentumába. A `magyar.lfd defaults=hu-min` opciója esetén az átdefiniált `\numberline` miatt, mivel most nincs pont az argumentum végén, ezért még ez kiegészül egy ponttal. Így például amikor a címkeszám 1, a jegyzékben „(1).” fog megjelenni „(1)” helyett. A probléma a `subparens` átdefiniálásával megoldható (lásd később).

parens Annyiban különbözik a `subparens` opciótól, hogy a kerek zárójelbe tett címkeszám előtt berakja még a prefixet. Magyar nyelvű dokumentum esetén hasonló a gond mint a `subparens` esetében. A probléma a `parens` átdefiniálásával megoldható (lásd később).

empty Nem rak semmit a `\numberline` argumentumába. A `magyar.1df` használata esetén `defaults=hu-min` opcióval, az átdefiniált `\numberline` miatt, mivel most nincs pont az argumentum végén, ezért még ez kiegészül egy ponttal, azaz a címkeszám helyén a jegyzékben egy pont fog megjelenni. A probléma az `empty` átdefiniálásával megoldható (lásd később).

További `(formázásnév)` is definiálható a `\DeclareCaptionListFormat` parancssal (lásd később).

További stílusbeállító parancsok:

```
\DeclareCaptionLabelFormat{\formázásnév}{\formázáskód} ∈ caption
```

A `\captionsetup` parancs `labelformat=\formázásnév` opciójában deklarálható ezzel egy új `\formázásnév`, aminek a hatását a `\formázáskód` adja meg. A `\formázáskód`-ban a címkenévre a #1 kóddal, míg a címkeszámra a #2 kóddal kell utalni. Ez a parancs csak preambulumban adható ki. Például

```
\DeclareCaptionLabelFormat{fbox}{\fbox{\#2.\~{\#1}}}
```

esetén a `\captionsetup` parancs `labelformat=fbox` opciója a címkét bekeretezi.

A következő két parancs abban segít, hogy egy formázás másképpen viselkedjen, ha vagy a címkenév vagy a címkeszám hiányzik.

```
\bothIfFirst{\arg1}{\arg2} ∈ caption
```

Ha `\arg1` nem üres, akkor kifejti először az `\arg1`-et, majd az `\arg2`-t. Ha `\arg1` üres, akkor nem csinál semmit.

```
\bothIfSecond{\arg1}{\arg2} ∈ caption
```

Ha `\arg2` nem üres, akkor kifejti először az `\arg1`-et, majd az `\arg2`-t. Ha `\arg2` üres, akkor nem csinál semmit. Például magyar nyelvű dokumentum esetén a `\captionsetup` parancs `labelformat=original` opciója a következő módon van deklarálva:

```
\DeclareCaptionLabelFormat{original}{\#2\bothIfSecond{.~}{\#1}}
```

```
\DeclareCaptionListFormat{\formázásnév}{\formázáskód} ∈ caption
```

A `\captionsetup` parancs `listformat=\formázásnév` opciójában deklarálható ezzel egy új `\formázásnév`, aminek a hatását a `\formázáskód` adja meg. A `\formázáskód`-ban a címkeszám prefixére a #1 kóddal, míg a címkeszámra a #2 kóddal kell utalni. Ez a parancs csak preambulumban adható ki. Például

```
\DeclareCaptionListFormat{brace}{[\#2]}
```

hatására a `\captionsetup` parancs `labelformat=brace` opciója a címkeszámot szögletes zárójelbe rakva írja a `\numberline` argumentumába. A `magyar.1df` `defaults=hu-min` opciója esetén még ez kiegészül egy ponttal. Így például amikor a címkeszám 1, a jegyzékben „[1].” fog megjelenni „[1]” helyett. Ezt a következő kóddal lehet megoldani:

```
\makeatletter
\newcommand\hudot{.}
\def\magyar@numberline#1\vfuzz{\hb@xt@{\tempdima{#1\hudot\hfil}}}
\makeatother
```

Ha a később ismertetett `tocloft` csomagot is használja, akkor az előző helyett a következőket írja be:

```
\makeatletter
\newcommand\hudot{.}
\def\magyar@numberline#1\vfuzz{%
```

```
\hb@xt@{\tempdima{`@cftbsnum #1`@cftasnum\hudot\hfil}`@cftasnumb}
\makeatother
```

Ezután az előbbi brace formázásnevet így kell definiálni:

```
\DeclareCaptionListFormat{brace}{[#2]\protect\def\protect\hudot{}}
```

Ezzel a technikával lehetővé válik a `subparens`, `parens` és `empty` formázásnevek javítása `magyar.ldf` használata esetén. Ehhez a következőt kell beírni az opciók megadása előtt:

```
\DeclareCaptionListFormat{subparens}{(#2)\protect\def\protect\hudot{}}
\DeclareCaptionListFormat{parens}{#1(#2)\protect\def\protect\hudot{}}
\DeclareCaptionListFormat{empty}{\protect\def\protect\hudot{}}
```

11.5. Saját úsztatott objektumok létrehozása

Alaphelyzetben a táblázatokat és az ábrákat lehet úsztatni saját címkével és jegyzékkel. De saját úsztató környezetet is definiálhat. Például ha azt szeretné, hogy az úsztató környezet neve legyen `graph`, a címkenév legyen „grafikon” és a jegyzék címe legyen „Grafikonok jegyzéke”, akkor a következő írja a preambulumba:

```
\DeclareCaptionType{graph}[grafikon][Grafikonok jegyzéke] \in caption
```

ami ezzel egyenértékű:

```
\DeclareFloatingEnvironment{graph}[grafikon][Grafikonok jegyzéke] \in newfloat
```

Ezután pontosan úgy használhatók a `graph` és `graph*` környezetek, mint a `figure` és `figure*` illetve `table` és `table*`. A `graph` környezet címkenevét a `\graphname` parancs, a címkeszámot pedig a `graph` számláló tárolja.

Arra ügyelni kell, hogy olyan környezetnevet válasszon, ami még nem foglalt. Például az előző esetben, ha `graph` helyett `graf` környezetet definiál, akkor hibát kap. Ennek az a magyarázata, hogy a `graf` környezet definiálásánál egy `\endgraf` parancs is létrejön, ami az `\end{graf}` kiadásakor fejtődik ki. De `\endgraf` parancs már létezik, így hibás lesz a fordítás. Ha illet tapasztal, akkor válasszon másik környezetnevet.

11.6. Úsztatás mellőzése

Ha egy objektum helyét az úsztatás kikapcsolásával a felhasználó szeretné „kisakkozni”, akkor az úsztató környezetnek használja a `H` opcióját, amely a `float` csomag betöltésével válik elérhetővé. Például

```
\begin{figure}[H]
\centering
\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 akar úsztatni, akkor miért kell mégis úsztató környezetbe tenni? A válasz az, hogy a `\caption` parancs az úsztató környezetből tudja, hogy milyen címkenevet és -számot kell adnia. Másik lehetőség a

```
\captionof{<környezet>}[<cím jegyzékbén>]{<cím>} \in caption
```

használata, amit nem kell úsztatott környezetbe rakni, mert a címkenevét és -számot a `\környezet` megadása miatt tudja. Például az előző kóddal azonos hatású a következő:

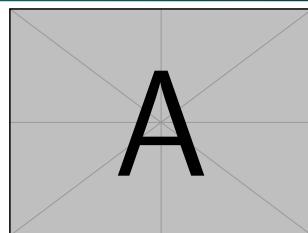
```
\begin{center}
\begin{minipage}{\linewidth}
\centering
\includegraphics{fig}
\captionof{figure}{2002-es statisztika}\label{fig-2002stat}
\end{minipage}
\end{center}
```

A `minipage` környezet azért kell, hogy a kép és a felirat között megakadályozza az oldaltörést. A `\centering` parancs középre igazít, ezért a `center` környezet csak azért kell, hogy a függőleges térközöket beállítsa.

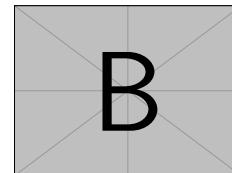
11.7. Több objektum egy úsztató környezetben

Ha egy úsztató környezetben két ábrát akar elhelyezni, mindenkorral saját felirattal, akkor ezt megteheti a következő minta alapján:

```
\begin{figure}
\centering
\begin{minipage}[t]{5cm}
\centering
\includegraphics[width=4cm]{example-image-a}
\caption{Egyik ábra címe}\label{fig-pelda-a}
\end{minipage}
\begin{minipage}[t]{5cm}
\centering
\includegraphics[width=3cm]{example-image-b}
\caption{Másik ábra címe}\label{fig-pelda-b}
\end{minipage}
\end{figure}
```



1. ábra. Egyik ábra címe

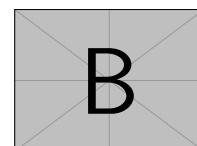


2. ábra. Másik ábra címe

Egy úsztató környezetben több objektumot csoporthozhat úgy is, hogy legyen egy főfelirata és mindegyik objektumnak egy alfelirata. Például



(a) Egyik ábra címe



(b) Másik ábra címe

1. ábra. A két ábra főcíme

Ehhez töltse be a `subcaption` csomagot (ami egyúttal betölti a `caption` csomagot is), majd az úsztató környezetben az egyes objektumokat illessze a következő parancsba:

```
\subcaptionbox[\langle jegyzék cím\rangle]{\langle cím\rangle}[\langle szélesség\rangle][\langle poz.\rangle]{\langle objektum\rangle} \in subcaption
\subcaptionbox*{\langle cím\rangle}[\langle szélesség\rangle][\langle poz.\rangle]{\langle objektum\rangle} \in subcaption
```

A csillagos verzió esetén a cím nem kerül be a jegyzékbe, továbbá számozása sem lesz az objektumnak.

`\langle jegyzék cím\rangle` A jegyzékbe bekerülő cím. Alapértéke a `\langle cím\rangle`.

`\langle cím\rangle` Az `\langle objektum\rangle` címe, ami az `\langle objektum\rangle` alatt fog megjelenni. A `\langle cím\rangle` előtt a feliratban csak a címkeszám jelenik meg, címkenév nincs.

`\langle szélesség\rangle` Az `\langle objektum\rangle` és az alatta elhelyezkedő felirata részére fenntartott doboz szélessége. Alapértéke az `\langle objektum\rangle` szélessége.

`\langle poz.\rangle` Az `\langle objektum\rangle` pozíciója a részére fenntartott dobozban. Alapértéke `c`, ami középre illeszt. Balra illesztés esetén `l`, jobbra illesztés esetén `r` az értéke.

Hivatkozás esetén a `\langle cím\rangle` paraméter végére kell beírni a `\label{\langle hivatkozás név\rangle}` parancsot. Erre hivatkozni a `\ref{\langle hivatkozás név\rangle}` vagy

```
\subref{\langle hivatkozás név\rangle} \in subcaption
```

parancsokkal lehet. Utóbbi parancs az objektum (alapesetben kis alfanumerikus) címkeszámát eredményezi. Természetesen `\subcaptionbox*` esetén értelmetlen a `\subref` használata. Az éppen aktuális objektum címkeszámát a

```
sub\langle úsztató környezet neve\rangle \in subcaption
```

számláló tárolja, azaz például a `figure` környezet esetén a `subfigure`.

A `\ref{\langle hivatkozás név\rangle}` esetén az objektum címkeszáma előtt megjelenik a prefixe is, ami alapesetben az úsztató objektum címkeszáma. A prefixet a

```
\p@sub\langle úsztató környezet neve\rangle \in subcaption
```

parancs tárolja, azaz például a `figure` környezet esetén a `\p@subfigure`.

A `subcaption` csomagot az `amsmath` vagy `mathtools` csomaggal együtt használva a `\subref` parancs nem működik ennek a könyvnek a megírásakor aktuális verziókkal (lásd <https://tex.stackexchange.com/questions/746029/new-conflict-of-subcaption-and-amsmath>). A hiba orvoslásáig használja a következő javító kódot:

```
\usepackage{etoolbox}
\makeatletter
\patchcmd\caption@subtypehook{\let\label\subcaption@label}
{\let\label\subcaption@label\let\ltx@label\subcaption@label}{}{\fail}
\makeatother
```

Ha a feliratok stílusát szeretné testre szabni, akkor a következő parancs pontosan úgy használható, mint a 11.4. szakaszban:

```
\captionsetup[\langle hatókör\rangle]{\langle opciók\rangle} \in subcaption
```

- Ha a `\langle hatókör\rangle` nincs megadva, akkor minden úsztató környezet fő- és alfelirataira vonatkozik. Például

```
\captionsetup{labelfont=bf}
```

- Ha a `\langle hatókör\rangle` egy úsztató környezet neve, akkor csak azon úsztató környezet fő- és alfelirataira vonatkozik. Például

```
\captionsetup[figure]{labelfont=bf}
```

- Ha a `\captionsetup[hatókör]{sub}{labelfont=bf}` helyére `sub` van írva, akkor minden úsztató környezet alfelirataira vonatkozik. Például

```
\captionsetup[sub]{labelfont=bf}
```

- Ha a `\captionsetup[hatókör]{úsztató környezet}{sub}{labelfont=bf}` helyére `sub` van írva, akkor csak az `\captionsetup[úsztató környezet]{sub}{labelfont=bf}` nevű úsztató környezet alfelirataira vonatkozik. Például

```
\captionsetup[subfigure]{labelfont=bf}
```

Több `\captionsetup` esetén a megadott opciók összefűzése nem a kiadás, hanem a `\captionsetup[hatókör]` típusa szerinti sorrendben történik a következő módon:

1. A `caption` csomag alapbeállításai.

2. `\captionsetup{opciónk}`

3. `\captionsetup[úsztató környezet]{opciónk}`

4. A `subcaption` csomag alapbeállításai, amely minden úsztató környezet alfelirataira vonatkozik:

```
\captionsetup[sub]{margin=0pt, font+=smaller, labelformat=parens,
labelsep=space, skip=6pt, list=false}
```

5. `\captionsetup[sub]{opciónk}`

6. `\captionsetup[sub[úsztató környezet]]{opciónk}`

Tehát például

```
\captionsetup[sub]{opciónk1}
\captionsetup{opciónk2}
```

után az opciók sorrendje

- főfeliratokra: `caption` csomag alapbeállításai, `\captionsetup{opciónk2}`
- alfeliratokra: `caption` csomag alapbeállításai, `\captionsetup{opciónk2}`, `subcaption` csomag alapbeállításai, `\captionsetup{opciónk1}`.

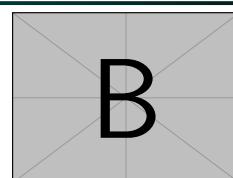
Példák

```
</> \begin{figure}[ht!]
\centering
\subcaptionbox{Egyik ábra címe\label{abra-egyik}}[5cm]{\includegraphics[width=3cm]{example-image-a}}%
\subcaptionbox{Másik ábra címe\label{abra-masik}}[5cm]{\includegraphics[width=3cm]{example-image-b}}
\caption{A két ábra főcíme}\label{abra-mindketto}
\end{figure}

Lásd \az{\ref{abra-egyik}}.~ábrán.
Lásd \az{\ref{abra-mindketto}}.~ábra \subref{abra-masik} részét.
```



(a) Egyik ábra címe



(b) Másik ábra címe

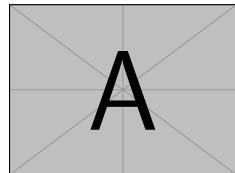
1. ábra. A két ábra főcímé

Lásd az 1a. ábrán. Lásd az 1. ábra b részét.

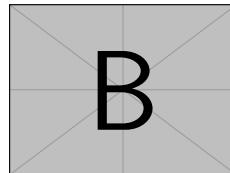
</>

```
\begin{figure}[ht!]
\centering
\subcaptionbox*{Egyik ábra címe}
[5cm]{\includegraphics[width=3cm]{example-image-a}}%
\subcaptionbox*{Másik ábra címe}
[5cm]{\includegraphics[width=3cm]{example-image-b}}
\caption{A két ábra főcíme}\label{abra-mindketto}
\end{figure}
```

Lásd \az{\ref{abra-mindketto}}.~ábrán.



Egyik ábra címe



Másik ábra címe

1. ábra. A két ábra főcíme

Lásd az 1. ábrán.

A következő példában először állítsuk át a `figure` környezetbeli objektumok címkeszámozását nagy alfanumerikusra, továbbá a prefixet úgy, hogy az ábra címkeszámá után egy perjel álljon:

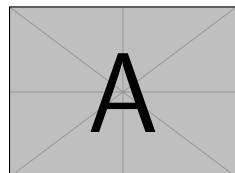
</>

```
\renewcommand{\thesubfigure}{\Alph{subfigure}}
\makeatletter
\renewcommand{\p@subfigure}{\thefigure/}
\makeatother
```

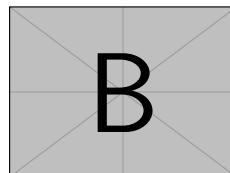
Ezután

```
\begin{figure}[ht!]
\centering
\subcaptionbox{Egyik ábra címe\label{abra-egyik}}
[5cm]{\includegraphics[width=3cm]{example-image-a}}%
\subcaptionbox{Másik ábra címe\label{abra-masik}}
[5cm]{\includegraphics[width=3cm]{example-image-b}}
\caption{A két ábra főcíme}\label{abra-mindketto}
\end{figure}
```

Lásd \az{\ref{abra-egyik}}.~ábrán.



(A) Egyik ábra címe



(B) Másik ábra címe

1. ábra. A két ábra főcíme

Lásd az 1/A. ábrán.

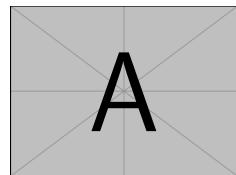
A következő példa nagyon hasonló eredményt ad az előzőhez, csak a hivatkozás formája lesz más.

```
</> \renewcommand{\thesubfigure}{(\Alph{subfigure})}
\captionsetup[sub]{labelformat=original}
```

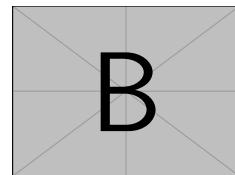
Ezután

```
\begin{figure}[ht!]
\centering
\subcaptionbox{Egyik ábra címe\label{abra-egyik}}[5cm]{\includegraphics[width=3cm]{example-image-a}}%
\subcaptionbox{Másik ábra címe\label{abra-masik}}[5cm]{\includegraphics[width=3cm]{example-image-b}}
\caption{A két ábra főcíme}\label{abra-mindketto}
\end{figure}
```

Lásd `\az{\ref{abra-egyik}}`-ábrán.



(A) Egyik ábra címe



(B) Másik ábra címe

1. ábra. A két ábra főcíme

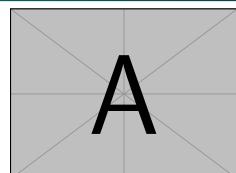
Lásd az 1(A) ábrán.

A következő esetben először azt állítjuk be, hogy minden úsztató környezet fő- és alfeliratainak címkeneve félkövér legyen, továbbá minden úsztató környezet alfeliratainak címkeneve körül eltűnjön a zárójel és utána az elválasztó egy pont és egy szóköz legyen.

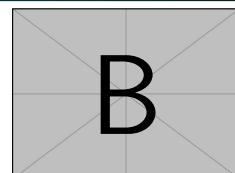
```
</> \captionsetup{labelfont=bf}
\captionsetup[sub]{labelformat=original,labelsep=period}
```

Ezután

```
\begin{figure}[ht!]
\centering
\subcaptionbox{Egyik ábra címe\label{abra-egyik}}[5cm]{\includegraphics[width=3cm]{example-image-a}}%
\subcaptionbox{Másik ábra címe\label{abra-masik}}[5cm]{\includegraphics[width=3cm]{example-image-b}}
\caption{A két ábra főcíme}\label{abra-mindketto}
\end{figure}
```



a. Egyik ábra címe



b. Másik ábra címe

1. ábra. A két ábra főcíme

12. fejezet

Objektumok körbefuttatása

Objektumok szöveggel történő körbefuttatására több lehetőség is van. Itt három megoldást mutatunk a `wrapstuff`, `wrapfig2` és a `floatflit` csomagokkal.

12.1. A `wrapstuff` csomag

```
\begin{wrapstuff}[opción] ∈ wrapstuff  
<objektum>  
\end{wrapstuff}
```

vagy

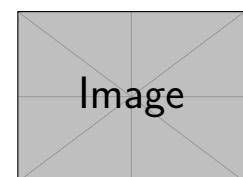
```
\wrapstuffset{opción} ∈ wrapstuff  
\begin{wrapstuff}  
<objektum>  
\end{wrapstuff}
```

Ez új bekezdést nyit, majd az ezután beírt szöveggel lesz az *<objektum>* körbefuttatva.
Például

</>

```
\begin{wrapstuff}  
\includegraphics[width=3cm]{example-image.pdf}  
\end{wrapstuff}  
, „Hirdetését olvastam és közlöm önnel, hogy ...
```

„Hirdetését olvastam és közlöm önnel, hogy elveszett életkedvének becsületes megtalálója vagyok. Hat évig Bostonban mint a villamosszék áramelindítója működtem, amíg egy alkalommal kaucsukkézelőm miatt rövidzárat történt, és az elítélt áramhiány következtében nem szenderült jobblétre, illetve a rosszabb létféleben, a szender mellőzésével életben maradt. Sajnos, kivégzés nem lehetséges áldozat nélkül, és így nyugalmas állásomból nyomban röpítettek. Legyen szabad ennyit a kaucsukmanzsettámrol.” (Rejtő Jenő: Piszkos Fred közelép, részlet)



A `wrapstuff` környezet és az azt követő szöveg közé üres sor is rakható, az nem befolyásolja az eredményt. A lehetséges *<opción>*:

- r Az *<objektum>* a jobb margónál helyezkedik el (alapopció).
- l Az *<objektum>* a bal margónál helyezkedik el.
- i Az *<objektum>* a belső margónál helyezkedik el.

o Az `<objektum>` a külső margónál helyezkedik el.

c Az `<objektum>` vízszintesen középen helyezkedik el.

ratio=⟨arányszám⟩ Az `<arányszám>` egy 0 és 1 közötti szám, amely azt határozza meg, hogy az `<objektum>` középpontja vízszintesen a bal margótól milyen arányban helyezkedjen el a szövegtükör szélességéhez képest. A `c` a `ratio=0.5` rövidítése. Az `i` illetve `o` opcióval együtt használva a belső illetve a külső margóhoz viszonyít a bal margó helyett.

top=⟨szám⟩ A `<szám>` határozza meg, hogy a körbefuttatás során hány sor legyen az `<objektum>` fölött. Alapértéke 0.

lines=⟨szám⟩ A `<szám>` határozza meg, hogy a körbefuttatás során hány sor legyen az `<objektum>` mellett. Alapesetben a `wrapstuff` csomag automatikusan határozza meg a `<szám>` értékét.

width=⟨szélesség⟩ Ha az `<objektum>` egysoros, akkor a `<szélesség>` az `<objektum>` természetes szélessége lesz, ilyenkor ezt nem kell megadni. Ha az `<objektum>` többsoros, akkor ezt az opciót mindenképpen be kell állítani.

height=⟨magasság⟩ Alapesetben a `<magasság>` az `<objektum>` természetes magassága. Ettől eltérni ezzel az opcióval lehet.

column Ha a körbefuttatás során két oszlopba kell rendezni a szöveget (például `c` opció esetén), akkor az oszlopok úgy lesznek elrendezve, mint a többhasábos szedés esetén. Azaz az első oszlop is sorfolytonosan olvasható és a második is. (Alapopció.)

column=false Ha a körbefuttatás során két oszlopba kell rendezni a szöveget (például `c` opció esetén), akkor az oszlopok nem úgy lesznek elrendezve, mint a többhasábos szedés esetén, hanem csak meg lesznek szakítva a sorok az `<objektum>` által.

column=par Ha a körbefuttatás során két oszlopba kell rendezni a szöveget (például `c` opció esetén), akkor az oszlopok bekezdésenként úgy lesznek elrendezve, mint a többhasábos szedés esetén. Azaz az első oszlop is sorfolytonosan olvasható és a második is minden bekezdésen belül.

leftsep=⟨hossz⟩ Az `<objektum>`-nak fenntartott hely (a `width` opciójával) és a körbefutó szöveg távolsága az `<objektum>` bal oldalán. A `<hossz>` alapértéke `1em`.

rightsep=⟨hossz⟩ Az `<objektum>`-nak fenntartott hely (a `width` opciójával) és a körbefutó szöveg távolsága az `<objektum>` jobb oldalán. A `<hossz>` alapértéke `1em`.

hsep=⟨hossz⟩ A `leftsep` és `rightsep` értékét is ugyanarra a `<hossz>` értékre állítja be.

abovesep=⟨hossz⟩ Az `<objektum>`-nak fenntartott hely (a `height` opciójával) és a körbefutó szöveg távolsága (a sortávolsággal megnövelve) az `<objektum>` felett. A `<hossz>` alapértéke `0pt`.

belowsep=⟨hossz⟩ Az `<objektum>`-nak fenntartott hely (a `height` opciójával) és a körbefutó szöveg távolsága (a sortávolsággal megnövelve) az `<objektum>` alatt. A `<hossz>` alapértéke `0pt`.

vsep=⟨hossz⟩ Az `abovesep` és `belowsep` értékét is ugyanarra a `<hossz>` értékre állítja be.

hoffset=⟨hossz⟩ Az `<objektum>` ennyivel lesz eltolva vízszintesen a margóra, feltéve, hogy az valamelyik oldalsó margónál helyezkedik el. (Tehát például `c` opció esetén hatástalan.) A `<hossz>` alapértéke `0pt`.

voffset=⟨hossz⟩ Az `<objektum>` ennyivel lesz eltolva felfelé. A `<hossz>` alapértéke `0pt`.

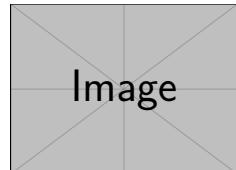
type=⟨úsztató környezet neve⟩ Az `<úsztató környezet neve>` lehet a standard `figure` vagy `table`, de bármilyen más általunk definiált úsztató környezet név is beírható. Ekkor a `wrapstuff` környezetben használható a `\caption` parancs, mint az úsztató környezetekben. Mivel ekkor az `<objektum>` többsoros lesz, így ebben az esetben meg kell adni a `width` opciót értékét is.

Példák

</>

```
\begin{wrapstuff}[c,top=2,column=false]
\includegraphics[width=3cm]{example-image.pdf}
\end{wrapstuff}
,,Hirdetését olvastam és ajánlatom a következő: ...
```

„Hirdetését olvastam és ajánlatom a következő: Ön világhírű lehet, ha örökkbefogad és nekem ajándékozza vagyonát, azután gyalog, vagy úgynevezett roller nevű játékgörgőn elzarándokol Kalazt hirdeti, hogy a nyerskosztnak híre menne, nevét megiskában, mivel így szólna az emtesítzen, hogy melyik vonattal kuttába és útközben mindenhol csodákat művel! E marhaságmernék a világ mind a négy sárberiség: »No lám csak...«. Érvár egy negyvenöt éves nyugalmazott ítéletvégrehajtót (aki úgy érzi, hogy még tartogat számára valamit az élet), és hetvenkét szabályszerűen végrehajtott, igazolt kivégzést mutathat fel. Az ön embere vagyok!” (Rejtő Jenő: Piszkos Fred közbelép, részlet)



</>

```
\begin{wrapstuff}[l,width=3cm]
\centering
\includegraphics[width=3cm]{example-image.pdf} \\
Cím
\end{wrapstuff}
,,Uram! Világhírűvé lenni gyerekjáték! ...
```

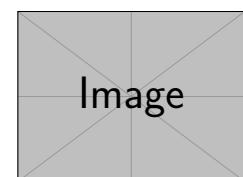


„Uram! Világhírűvé lenni gyerekjáték! Feltaláltam a »Gyermekeid játszva gyilkolnak« nevű csörgőre, kereplőre és kisvasútra alkalmazható tetszetős gyerekjátékot, mely egyben légyölő. Két légy súlyától a szerkezet működésbe jön és lecsap! Finanszírozzon Ön, és nevemet Edisoné mellett, Önét az én nevem mellett emlegeti az emberiség e találmány révén, amely két legyet pusztít el és egy csapásra világhírűvé teszi...” (Rejtő Jenő: Piszkos Fred közbelép, részlet)

</>

```
\begin{wrapstuff}[width=3cm,type=figure]
\includegraphics[width=3cm]{example-image.pdf}
\caption{Cím}
\end{wrapstuff}
,,Dear M.~Felséges! ...
```

„Dear M. Felséges! Továbá tiszttel kirájná őnagysága és a szerevet tiszttel császári öreganyja felségednek akit én is úgy szeretek mint magam privát mamáját ki pedig szegény régen elhunya, de asz nagy hazudság, hogy csak miattami bánódástól mivel hőn tiszttel atyámra is büszke lehetek. Hállás köszönést vagyok bátor amiért a Holkongból érkezett rendőri átirásra felsék nem hatyta eksz kartárcsát beszárni és ügyemben ovadékot lehelyezett. Azóta nekem elutaszni is szabad. Lábrahelyeztek. Eszt mekközönöm kiválló tiszteettel, vagyok bátor. Szíves érdeklődést van szerencséje felsékednek hoty írjag töpször és őszintén mivel szereti eszeket.” (Rejtő Jenő: Piszkos Fred közbelép, részlet)



1. ábra. Cím

12.2. A wrapfig2 csomag

```
\begin{wrapfloat}{\környezetnélv}{\sorszám}{\hely}{\kinyúlás}{\szélesség}
{\objektum}
\end{wrapfloat}
```

\környezetnélv Az *\objektum* típusának megfelelő úsztató környezet neve, mint a `figure` vagy `table`, de lehet bármilyen magunk által definiált úsztató környezetnélv is (lásd a 11.5. szakaszban). A `wrapfloat` környezetben elhelyezett `\caption` parancs ennek megfelelően írja ki a címet.

\sorszám A `wrapfig2` kiszámolja, hogy az *\objektum* magassága hány sornak felel meg és ennek megfelelően történik a szöveg körbefuttatása. Ha valamiért ezt rosszul számolja ki, akkor opcionálisan itt lehet azt megadni a *\sorszám* segítségével, hogy hány sort húzzon beljebb, ahol az *\objektum* el fog helyezkedni.

\hely Ez határozza meg, hogy hol helyezkedjen el az *\objektum*. A lehetséges értékek:

- l** (left) Bal margónál, úsztatás nélküli.
- L** (left) Bal margónál, úsztatással.
- r** (right) Jobb margónál, úsztatás nélküli.
- R** (right) Jobb margónál, úsztatással.
- i** (inner) Belső margónál, úsztatás nélküli. A belső margó egyoldalas szedés esetén minden a bal margó, míg kétoldalas szedés esetén a páratlan oldalakon a bal, a páros oldalakon pedig a jobb margó.
- I** (inner) Belső margónál, úsztatással.
- o** (outer) Külső margónál, úsztatás nélküli. A külső margó egyoldalas szedés esetén minden a jobb margó, míg kétoldalas szedés esetén a páratlan oldalakon a jobb, a páros oldalakon pedig a bal margó.
- O** (outer) Külső margónál, úsztatással.

\kinyúlás Opcionálisan ezzel lehet megadni, hogy az *\objektum* mennyire lógjon ki a margóra (pl. `1cm`). Negatív hossz is megadható. Az alapértéke `0pt`.

\szélesség Az *\objektum* részére lefoglalt terület szélessége a `\columnsep` hosszúságparancs aktuális értékével megnövelve.

A körbefuttatás első sora a `wrapfloat` környezet utáni szöveg első sora lesz. Például

```
\begin{wrapfloat}{figure}{R}{4cm}
\centering
\includegraphics[width=3cm]{example-image}
\caption{Példa}
\end{wrapfloat}
```

A `wrapfloat` környezetnek létezik egy csillagos verziója is:

```
\begin{wrapfloat}{\környezetnélv}{\sorszám}{\hely}{\kinyúlás}{\szélesség}*
```

Ezt akkor lehet használni, ha a *\sorszám* opciót is megadjuk. Ekkor a `wrapfig2` által kiszámolt sorok számát, ami az *\objektum* magasságának felel meg, megnöveli a *\sorszám* értékével, ami lehet negatív is. Az így kapott szám lesz azon sorok száma, amiket beljebb húz az *\objektum* elhelyezéséhez. A többi paramétert pontosan úgy kell használni, mint a csillag nélküli esetben.

A két standard úsztató környezet, a `figure` és `table` esetén van egy rövidített verzió is:

```
\begin{wrapfloat}{figure}{...}
<objektum>
\end{wrapfloat}
```

egyenértékű a következővel:

```
\begin{wrapfigure}{...}
<objektum>
\end{wrapfigure}
```

illetve

```
\begin{wrapfloat}{table}{...}
<objektum>
\end{wrapfloat}
```

egyenértékű a következővel:

```
\begin{wraptable}{...}
<objektum>
\end{wraptable}
```

12.3. A floatflt csomag

A `floatflt` csomag `floatingfigure` környezete ábrákra lett kitalálva, de a `caption` csomag `\captionof` parancsával táblázatokra, vagy bármely saját úsztatott objektumra is alkalmazható. Például

```
\begin{floatingfigure}[r]{4cm}
\centering
\includegraphics[width=3cm]{example-image}
\caption{Egy példa}\label{fig-pelda}
\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` opciójával töltse be. Ha azt akarja, hogy az alapopció `l` legyen, akkor a `floatflt` csomagot `lflt` opciójával töltse 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\oddpagestrue\fi
\centering
\includegraphics[width=3cm]{example-image}
\caption{Egy példa}\label{fig-pelda}
\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 adjon ki a

```
\setlength{\fingutter}{5mm} \in 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 elő `\noindent` parancsot.

A `floatflt` csomagnak létezik három hibája, amire érdemes odafigyelni. Bizonyos esetekben a `floatingfigure` környezetbe zárt objektum függőlegesen nem jól pozicionál. Ilyenkor a `floatingfigure` környezet elő 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. Ezkel 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.

13. 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 a képek, táblázatok, az úszó objektumok, de doboz például egy betű vagy egy vonal is. Háromféle dobozt fogunk megkülönböztetni:

- 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.

13.1. 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 (alapció), ha **l**, akkor balra, ha **r**, akkor jobbra és **s** esetén széthúzza/összenyomja a teljes dobozsélességre. 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 parancssal:

```
\framebox[<dozók 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 parancssal állítható be például 1 pt-ra:

```
\setlength{\fboxrule}{1pt}
```

A keret és a szöveg távolsága, mely alapesetben 3 pt, a következő parancssal állítható be például 2 pt-ra:

```
\setlength{\fboxsep}{2pt}
```

Ha a `\makebox` parancsot opciók nélkül használja, akkor elég csak

```
\mbox{<szöveg>}
```

parancsot írni. Hasonlóan, ha a `\framebox` parancsot opciók nélkül használja, akkor csak

```
\fbox{<szöveg>}
```

parancsot kell írni.

Színes egysoros dobozok is előállíthatók. Ezeket néhány példán mutatjuk meg:

```
</> \colorbox{red}{szöveg} \in xcolor
```

```
szöveg
```

```
\colorbox[RGB]{128,0,128}{szöveg} \in xcolor
```

```
szöveg
```

```
</> \fcolorbox{red}{yellow}{szöveg} \in xcolor
```

```
szöveg
```

```
\fcolorbox[RGB]{0,64,128}{192,192,192}{szöveg} \in xcolor
```

```
szöveg
```

Ezknél a keretet pontosan úgy lehet beállítani, mint a `\framebox` esetén.

A következő parancs egy olyan egysoros doboz készít, amely az alapvonaltól magasabban/alicsonyabban helyezkedik el:

```
\raisebox{\emelés}{\szöveg}
```

Például

```
</> AAA\raisebox{4pt}{BBB}CCC\raisebox{-4pt}{DDD}
```

Az *\emelés*-ben használhatók még a

```
\width  
\height  
\depth  
\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}
```

13.2. Bekezdésdobozok

Bekezdésdobozokba akár többsoros vagy több bekezdésnyi szöveget is rakhatal a következő parancssal 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 egységnak, doboznak számít:



```

SZÖVEG
\fbox{\begin{minipage}[t][2cm][s]{5cm}}
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 szöveg 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 veszi 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 szöveg szöveg

Itt már megfelelő méretűek a szóközök az első sorban, de ennek érdekében a doboz szélességét 6 cm-ről csökkenteni kellett valamennyivel. A `varwidth` környezet akkor is

használható, ha a töréspontokat mi adjuk meg, így a doboz szélessége nem ismert. Ekkor a `(szélesség)` helyére írja a `\textwidth` parancsot. Például

```
</> \fbox{\begin{varwidth}{\textwidth}
szöveg\\
szöveg szöveg\\
szöveg szöveg szöveg
\end{varwidth}}
```

```
szöveg
szöveg szöveg
szöveg szöveg szöveg
```

13.3. Vonaldobozok

Vonaldobozokat a következő parancssal készíthet:

```
\rule[<emelés>]{<szélesség>}[<magasság>]
```

Ez egy `(szélesség)` szélességű és `(magasság)` magasságú téglalapot rajzol, melynek az alja az alapvonaltól az `(emelés)` mértékével lesz feljebb. Például

```
xxxxx\rule[1ex]{2cm}{2mm}
```

```
xxxxx
```

```
x\rule[0.5ex]{3cm}{1pt}x
```

```
x-----x
```

```
x\rule[-0.5ex]{3cm}{1pt}x
```

```
x-----x
```

13.4. Dobozok nyújtása, tükrözése

Dobozok nyújtása a következő parancssal oldható meg:

```
\scalebox{<x>}{<y>}{<doboz>} ∈ graphicx
```

`(doboz)` A nyújtandó doboz. Ennek helyére egyszerű szöveg is kerülhet, amit ekkor dobozként kezel.

`(x)` A vízszintes nyújtás szorzó (lehet negatív is).

`(y)` A függőleges nyújtás szorzó (lehet negatív is), melynek alapértéke `(x)`.

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}
```

szöveg **SZÖVEG** szöveg گویندگی متن

Amint látjuk ezzel tükrözni is tudunk. A függőleges tengelyű tükrözésre külön parancs is létezik:

`\reflectbox{<dozó>} ∈ graphicx`

amely egyenértékű a `\scalebox{-1}{1}{<dozó>}` parancssal.

13.5. Dobozok átméretezése

`\resizebox{<szélesség>}{<magasság>}{<dozó>} ∈ graphicx`
`\resizebox*{<szélesség>}{<magasság>}{<dozó>} ∈ graphicx`

<dozó> Az átméretezett doboz. Ennek helyére egyszerű szöveg is kerülhet, amit ekkor dobozként kezel.

<szélesség> Az átméretezett doboz szélessége.

<magasság> Az átméretezett doboz magassága. Ez a `\resizebox` esetén az alapvonaltól mért magasságot, míg `\resizebox*` esetén a teljes magasságot jelenti.

Ha *<szélesség>* vagy *<magasság>* helyén ! jel van, akkor azt a méretet a másikhoz arányosan állítja be. Például

</>

```
szöveg
\resizebox{!}{0.5cm}{szöveg}
\resizebox*{!}{0.5cm}{szöveg}
\resizebox{3cm}{0.5cm}{szöveg}
```

szöveg SZÖVEG szöveg SZÖVEG

13.6. Dobozok forgatása

Dobozokat a következő parancssal forgathat:

`\rotatebox[origin=<centrum>]{<sörg>}{<dozó>} ∈ graphicx`

<dozó> Elforgatandó doboz. Ennek helyére egyszerű szöveg is kerülhet, amit ekkor dobozként kezel.

<sörg> Forgatás szöge fokban. Pozitív érték esetén az óra járásával ellentétes irányban forgat.

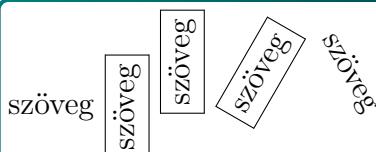
<centrum> Forgatás középpontja, ami a **t1**, **t**, **tr**, **l**, **c**, **r**, **Bl**, **B**, **Br**, **bl**, **b**, **br** értékekkel veheti fel (alapérték **Bl**). Ezek magyarázata a következő ábrán található:



Például



```
szöveg
\rotatebox [origin=c]{90}{\fbox{szöveg}}
\rotatebox{90}{\fbox{szöveg}}
\rotatebox [origin=bl]{60}{\fbox{szöveg}}
\rotatebox [origin=Br]{-60}{szöveg}
```



13.7. Dobozok transzformálása a pdf-trans csomaggal

A `pdf-trans` csomagot rendhagyó módon nem a `\usepackage`, hanem az `\input` paranccsal kell betölteni, azaz

```
\input{pdf-trans}
```

módon. Ezzel rengeteg transzformálási módot elérhetünk. A csomag hátránya, hogy csak `pdflatex` fordítóval működik.

13.7.1. Tükrözés

```
\boxflipx\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` függőleges szimmetriatengelyére történő tükrözése. A doboz magassága, mélysége, szélessége nem változik. Például

```
szöveg \boxflipx\hbox{szöveg}
```

```
\boxflipy\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` vízszintes szimmetriatengelyére történő tükrözése. A doboz magassága, mélysége, szélessége nem változik. Például

```
szöveg \boxflipy\hbox{szöveg}
```

```
\boxflipxy\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` szimmetria-középpontjára történő tükrözése. A doboz magassága, mélysége, szélessége nem változik. Például

```
szöveg \boxflipxy\hbox{szöveg}
```

```
\boxflipbase\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` alapvonalára történő tükrözése. A magasság és mélység értéke felcserélődik, a szélesség változatlan marad. Például

```
szöveg \boxflipbase\hbox{szöveg}
```

szöveg szöveg

13.7.2. Eltolás

```
\boxbaselineat{<százalék>}\hbox{<dozob>}
```

A *<dozob>* mélysége a teljes magasságnak a *<százalék>* % lesz. A teljes magasság és a szélesség nem változik. Például

```
szöveg
\boxbaselineat{0}\hbox{szöveg}
\boxbaselineat{50}\hbox{szöveg}
\boxbaselineat{100}\hbox{szöveg}
```

szöveg szöveg szöveg szöveg

```
\boxtranslate{<jobbra>}{<fel felé>}\hbox{<dozob>}
```

A *<dozob>* tartalmának eltolása a megadott mértékben. A doboz méretei nem változnak. Emiatt a tartalom ki fog „lógni” a dobozból. Például

```
\setlength{\fboxsep}{0pt}
\fbox{szöveg}
\fbox{\boxtranslate{0mm}{2mm}\hbox{szöveg}}
\fbox{\boxtranslate{2mm}{0mm}\hbox{szöveg}}
\fbox{\boxtranslate{2mm}{2mm}\hbox{szöveg}}
\fbox{\boxtranslate{-2mm}{-2mm}\hbox{szöveg}}
```

szöveg szöveg szöveg szöveg

```
\boxmoveright{<jobbra>}\hbox{<dozob>}
```

A *<dozob>* tartalmának eltolása jobbra a megadott mértékben. A doboz szélessége az eltolás mértékével nő, a többi méret nem változik. Emiatt, ha a *<jobbra>* értéke negatív, akkor a tartalom ki fog „lógni” a dobozból. Például

```
\setlength{\fboxsep}{0pt}
\fbox{szöveg}
\fbox{\boxmoveright{5pt}\hbox{szöveg}}
\fbox{\boxmoveright{-5pt}\hbox{szöveg}}
```

szöveg szövegszöveg

```
\boxraise{<fel felé>}\hbox{<dozob>}
```

A *<dozob>* tartalmának eltolása felfelé. A doboz szélessége nem változik. A doboz magassága az eredeti magasság és a *<fel felé>* összege, kivéve, ha ez az összeg negatív. Ekkor a magasság 0pt lesz. A doboz mélysége az eredeti mélység és a *<fel felé>* különbsége, kivéve, ha ez a különbség negatív. Ekkor a mélység 0pt lesz. Például

```
\setlength{\fboxsep}{0pt}
\fbox{szöveg}
```

```
\fbox{\boxraise{1pt}\hbox{szöveg}}
\fbox{\boxraise{5mm}\hbox{szöveg}}
\fbox{\boxraise{-5pt}\hbox{szöveg}}
\fbox{\boxraise{-5mm}\hbox{szöveg}}
```



13.7.3. Döntés

```
\boxslantbb{<szög1>}{<szög2>}\hbox{<doboz>}
```

A `<doboz>` függőleges tengelyének `<szög1>` fokkal történő jobbra irányuló döntése, a vízszintes tengelyének pedig `<szög2>` fokkal történő felfelé irányuló döntése. Az eredeti doboz alapvonalának bal végpontja marad az alapvonalon. A doboz méretei igazodnak a tartalomhoz. Például

```
szöveg \boxslantbb{45}{-15}\hbox{szöveg}
```

```
\boxslantbr{<szög1>}{<szög2>}\hbox{<doboz>}
```

Ugyanaz mint az előbb, de most az eredeti doboz alapvonalának jobb végpontja marad az alapvonalon. Például

```
szöveg \boxslantbr{45}{-15}\hbox{szöveg}
```

```
\boxslantx{<szög>}\hbox{<doboz>}
```

A `<doboz>` függőleges tengelyének `<szög>` fokkal történő jobbra irányuló döntése. A doboz méretei változatlanok maradnak, nem igazodik a tartalomhoz. Emiatt a tartalom ki fog „lógni” a dobozból. Például

```
szöveg \boxslantx{45}\hbox{szöveg} szöveg
```

```
\setlength{\fboxsep}{0pt}
\fbox{\boxslantx{45}\hbox{szöveg}}
```

```
\boxslanty{<szög>}\hbox{<doboz>}
```

A `<doboz>` vízszintes tengelyének `<szög>` fokkal történő felfelé irányuló döntése. A doboz méretei változatlanok maradnak, nem igazodik a tartalomhoz. Emiatt a tartalom ki fog „lógni” a dobozból. Például

```
\setlength{\fboxsep}{0pt}
\fbox{\boxslanty{20}\hbox{szöveg}}
```

A rotated box containing the word "szöveg". The box is rotated 90 degrees counter-clockwise.

13.7.4. Forgatás

```
\boxrevolveleft\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` forgatása 90° -kal az óra járásával ellentétes irányban. A forgatás középpontja az alapvonala bal végpontja, így a mélység `0pt` lesz. A többi méret igazodik a tartalomhoz. Például

A rotated box containing the word "szöveg". The box is rotated 90 degrees counter-clockwise.

```
\boxrevolveright\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` forgatása 90° -kal az óra járásával megegyező irányban. A forgatás középpontja az alapvonala jobb végpontja, így a mélység `0pt` lesz. A többi méret igazodik a tartalomhoz. Például

A rotated box containing the word "szöveg". The box is rotated 90 degrees clockwise.

```
\boxrotatebbl{\langle szög\rangle}\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` forgatása `\langle szög\rangle` fokkal az óra járásával megegyező irányban. A forgatás középpontja az alapvonala bal végpontja. A doboz méretei igazodnak a tartalomhoz. Például

A rotated box containing the word "szöveg". The box is rotated 20 degrees clockwise.

A rotated box containing the word "szöveg". The box is rotated 20 degrees counter-clockwise.

```
\boxrotatebbr{\langle szög\rangle}\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` forgatása `\langle szög\rangle` fokkal az óra járásával megegyező irányban. A forgatás középpontja az alapvonala jobb végpontja. A doboz méretei igazodnak a tartalomhoz. Például

A rotated box containing the word "szöveg". The box is rotated 20 degrees counter-clockwise.

A rotated box containing the word "szöveg". The box is rotated 20 degrees counter-clockwise.

```
\boxrotate{\langle szög\rangle}\hbox{\langle doboz\rangle}
```

A `\langle doboz\rangle` forgatása `\langle szög\rangle` fokkal az óra járásával megegyező irányban. A forgatás középpontja az alapvonala bal végpontja. A doboz méretei változatlanok maradnak, nem igazodik a tartalomhoz. Emiatt a tartalom ki fog „lógni” a dobozból. Például

```
\setlength{\fboxsep}{0pt}
szöveg
\fbox{szöveg}
\fbox{\boxrotate{20}\hbox{szöveg}}
```

szöveg szöveg **szöveg**

```
\boxrotatec{szög}\hbox{doboz}
```

A `(doboz)` forgatása `(szög)` fokkal az óra járásával megegyező irányban. A forgatás középpontja a `(doboz)` szimmetria-középpontja. A doboz méretei változatlanok maradnak, nem igazodik a tartalomhoz. Emiatt a tartalom ki fog „lógni” a dobozból. Például

```
\setlength{\fboxsep}{0pt}
szöveg
\fbox{szöveg}
\fbox{\boxrotatec{20}\hbox{szöveg}}
```

szöveg szöveg **Szöveg**

```
\boxrotatexy{szög}{x}{y}\hbox{doboz}
```

A `(doboz)` forgatása `(szög)` fokkal az óra járásával megegyező irányban. A forgatás középpontja a `(doboz)` (`x`, `y`) koordinátájú pontja, ahol az origó az alapvonal bal végpontja. A doboz méretei változatlanok maradnak, nem igazodik a tartalomhoz. Emiatt a tartalom ki fog „lógni” a dobozból. Például

```
\setlength{\fboxsep}{0pt}
szöveg
\fbox{szöveg}
\fbox{\boxrotatexy{20}{10mm}{5mm}\hbox{szöveg}}
```

szöveg szöveg **Szöveg**

13.7.5. Nyújtás

```
\boxscale{x}{százalék}\hbox{doboz}
```

A `(doboz)` szélességének változtatása `(százalék)` % mértékben. A magasság és mélység változatlan marad. Például

```
szöveg \boxscale{300}\hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxscaleto{méret}\hbox{doboz}
```

A `(doboz)` szélességének változtatása `(méret)`-re. A magasság és mélység változatlan marad. Például

```
szöveg \boxscaleto{8cm}\hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxuniscalexto{<méret>}\hbox{<doboz>}
```

A *<doboz>* szélességének változtatása *<méret>*-re. A magasság és mélység arányosan változik. Például

```
szöveg \boxuniscalexto{3cm}\hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxscaley{<százalék>}\hbox{<doboz>}
```

A *<doboz>* teljes magasságának változtatása *<százalék>* % mértékben. A szélesség változatlan marad. Például

```
szöveg \boxscaley{200}\hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxscaleyto{<méret>}\hbox{<doboz>}
```

A *<doboz>* teljes magasságának változtatása *<méret>*-re. A szélesség változatlan marad. Például

```
szöveg \boxscaleyto{8mm}\hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxuniscaleyto{<méret>}\hbox{<doboz>}
```

A *<doboz>* teljes magasságának változtatása *<méret>*-re. A szélesség arányosan változik.

```
\boxscalehtto{<méret>}\hbox{<doboz>}
```

A *<doboz>* magasságának változtatása *<méret>*-re. A mélység arányosan változik, a szélesség változatlan marad.

```
\boxuniscalehtto{<méret>}\hbox{<doboz>}
```

A *<doboz>* magasságának változtatása *<méret>*-re. A mélység és szélesség arányosan változik.

```
\boxscaledpto{<méret>}\hbox{<doboz>}
```

A *<doboz>* mélységének változtatása *<méret>*-re. A magasság arányosan változik, a szélesség változatlan marad.

```
\boxuniscaledpto{<méret>}\hbox{<doboz>}
```

A *<doboz>* mélységének változtatása *<méret>*-re. A magasság és szélesség arányosan változik.

```
\boxscalexy{<százalék1>}{<százalék2>}\hbox{<doboz>}
```

A *<doboz>* szélességének *<százalék1>* % illetve a teljes magasságának *<százalék2>* % mértékben történő változtatása. Például

```
szöveg \boxscalexy{300}{200}\hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxscale{<méret1>}{<méret2>} \hbox{<doboz>}
```

A `<doboz>` szélességének `<méret1>` illetve a teljes magasságának `<méret2>` méretre történő változtatása. Például

```
szöveg \boxscale{8cm}{8mm} \hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxextscale{<szélesség>}{<magasság>}{<mélység>} \hbox{<doboz>}
```

A `<doboz>` szélessége `<szélesség>`-gel, magassága `<magasság>`-gal, mélysége `<mélység>`-gel nő meg.

```
\boxextscaleto{<szélesség>}{<magasság>}{<mélység>} \hbox{<doboz>}
```

A `<doboz>` szélessége `<szélesség>`, magassága `<magasság>`, mélysége `<mélység>` méretű lesz.

```
\boxscale{<százalék>} \hbox{<doboz>}
```

A `<doboz>` minden méretének `<százalék>` % mértékben történő megváltoztatása. Például

```
szöveg \boxscale{200} \hbox{szöveg}
```

szöveg **SZÖVEG**

```
\boxexts{<jobbra>}{<balra>}{<felfelé>}{<lefelé>} \hbox{<doboz>}
```

A doboz tartalmának megnyújtása az adott irányokba. A doboz eredeti méretei nem változnak. Emiatt a tartalom ki fog „lógni” a dobozból. Például

```
\setlength{\fboxsep}{0pt}
\fbox{\boxexts{0mm}{0mm}{3mm}{0mm} \hbox{szöveg}}
\fbox{\boxexts{0mm}{0mm}{0mm}{3mm} \hbox{szöveg}}
\fbox{\boxexts{3mm}{0mm}{0mm}{0mm} \hbox{szöveg}}
\fbox{\boxexts{0mm}{3mm}{0mm}{0mm} \hbox{szöveg}}
```

SZÖVEG **SZÖVEG** **SZÖVEG** **SZÖVEG**

13.7.6. Doboz átméretezése

```
\boxresizeto{<szélesség>}{<magasság>}{<mélység>} \hbox{<doboz>}
```

A doboz méretei az adott értékek lesznek. A doboz tartalmának méretei nem változnak. Például

```
\setlength{\fboxsep}{0pt}
szöveg
\fbox{szöveg}
\fbox{\boxresizeto{20mm}{10mm}{5mm} \hbox{szöveg}}
```

szöveg **szöveg** szöveg

```
\boxextents{\langle bal \rangle}{\langle jobb \rangle}{\langle fent \rangle}{\langle lent \rangle}\hbox{\langle doboz \rangle}
```

A doboz mélysége 0pt lesz. A doboz szélessége az eredeti szélesség + $\langle bal \rangle$ + $\langle jobb \rangle$, ha ez az érték pozitív, ellenkező esetben 0pt lesz. A doboz magassága az eredeti teljes magasság + $\langle fent \rangle$ + $\langle lent \rangle$, ha ez az érték pozitív, ellenkező esetben 0pt lesz. A doboz tartalmának méretei nem változnak, a bal, jobb, felső, alsó margók rendre $\langle bal \rangle$, $\langle jobb \rangle$, $\langle fent \rangle$, $\langle lent \rangle$ lesznek. Például

```
\setlength{\fboxsep}{0pt}
szöveg \fbox{szöveg}
\fbox{\boxextents{20mm}{15mm}{10mm}{5mm}\hbox{szöveg}}
```



13.7.7. Egyebek

```
\boxclip\hbox{\langle doboz \rangle}
```

A doboz tartalmának a dobozon kívüli részeit levágja. Például

```
\setlength{\fboxsep}{0pt}
\fbox{\boxslantx{45}\hbox{szöveg}}
\boxclip\hbox{\fbox{\boxslantx{45}\hbox{szöveg}}}}
```

```
\boxsh\hbox{\langle doboz \rangle}
```

Megrajzolja kék színnel a doboz határvonalait és az alapvonalát. Például

```
\boxsh\hbox{\Huge szöveg}
```

```
\boxshow{\langle w \rangle w \langle r \rangle \langle g \rangle \langle b \rangle RG}{[\langle x \rangle \langle y \rangle]0 d}{[]}\hbox{\langle doboz \rangle}
```

Ugyanazt csinálja, mint az előző parancs, de ekkor a vonal színének rgb kódja megadható az $\langle r \rangle$, $\langle g \rangle$ illetve $\langle b \rangle$ paraméterekkel. Az alapvonalat jelző szaggatott vonalban a vonalak hossza $\langle x \rangle$ pt a szünetek hossza pedig $\langle y \rangle$ pt lesz. A vonalak vastagsága $\langle w \rangle$ pt lesz. Például

```
\boxshow{.4 w 1 .5 .5 RG}{[10 5]0 d}{[]}\hbox{\Huge szöveg}
```

```
\wd\transbox
\dp\transbox
\ht\transbox
```

Az eddigi parancsokban a méretek megadásánál ezekkel hivatkozhatunk a $\langle doboz \rangle$ eredeti szélességére, mélységére illetve magasságára.

\boxinfo\hbox{\langle doboz \rangle}

Kiírja a *<doboz>* méreteit. Például

	\hbox	
gggggggggggggg	wd 246.68976pt	gggggggggggggg
	ht 5.1654pt	
	dp 2.33276pt	

13.8. Doboz méreteinek nullázása

\smash{\langle szöveg \rangle} ∈ mathtools

A létrehozott dobozt megjeleníti, de annak teljes magasságát 0 pt-nak tekinti. Például

```
\setlength{\fboxsep}{0pt}  
\fbox{g} \fbox{\smash{g}}
```

g g

\smash[t]{\langle szöveg \rangle} ∈ mathtools

A létrehozott dobozt megjeleníti, de úgy kezeli, mintha annak magassága 0 pt lenne.
Például

```
\setlength{\fboxsep}{0pt}  
\fbox{g} \fbox{\smash[t]{g}}
```

g g

\smash[b]{\langle szöveg \rangle} \in \mathtt{mathtools}

A létrehozott dobozt megjeleníti, de úgy kezeli, mintha annak mélysége 0 pt lenne.
Például

```
\setlength{\fboxsep}{0pt}  
\fbox[\textcolor{gray}{g}]\fbox{\smash[b]{\textcolor{gray}{g}}}
```

g g

\llap{\langle szöveg \rangle}

A létrehozott dobozt úgy kezeli, mintha annak szélessége 0 pt lenne. A doboz tartalmát a doboz bal oldalára illeszti. Tehát

doboz1 } \llap{ } \langle szöveg \rangle } \langle doboz2 \rangle

esetén a *doboz1* után közvetlenül a *doboz2* van, hiszen a *szöveg* doboz szélessége 0 pt. A *szöveg* doboz úgy jelenik meg, hogy annak jobb oldala a *doboz1* jobb oldalához van illesztve. Például:

```
xxxxxxxxxxxxxxxxxxxxyy
```

A `\mathllap` ∈ `mathtools` parancs ugyanúgy használható, mint az `\llap`, csak matematikai módban (lásd később).

```
\rlap{<szöveg>}
```

A létrehozott dobozt úgy kezeli, mintha annak szélessége 0 pt lenne. A doboz tartalmát a doboz jobb oldalára illeszti. Tehát

```
<doz1>\rlap{<szöveg>}<doz2>
```

esetén a `<doz1>` után közvetlenül a `<doz2>` van, hiszen a `<szöveg>` doboz szélessége 0 pt. A `<szöveg>` doboz úgy jelenik meg, hogy annak bal oldala a `<doz2>` bal oldalához van illesztve. Például

```
xxxxxxxxxxxxxxxx\rlap{\rule[0.5ex]{2cm}{0.4pt}}yyyyy
```

```
xxxxxxxxxxxxxxxxxxxxyy
```

A `\mathrlap` ∈ `mathtools` parancs ugyanúgy használható, mint az `\rlap`, csak matematikai módban (lásd később).

```
\clap{<szöveg>}
```

A létrehozott dobozt úgy kezeli, mintha annak szélessége 0 pt lenne. A doboz tartalmát a doboz közepére illeszti. Tehát

```
<doz1>\clap{<szöveg>}<doz2>
```

esetén a `<doz1>` után közvetlenül a `<doz2>` van, hiszen a `<szöveg>` doboz szélessége 0 pt. A `<szöveg>` doboz úgy jelenik meg, hogy annak közepe a `<doz1>` jobb oldalához van illesztve. Például

```
xxxxxxxxxxxxxxxx\clap{\rule[0.5ex]{2cm}{0.4pt}}yyyyy
```

```
xxxxxxxxxxxxxxxxxxxxyy
```

Megjegyezzük, hogy a `\clap` parancs 2020. október előtt telepített TeX-rendszerök esetén csak a `mathtools` csomag betöltésével érhető el.

A `\mathclap` ∈ `mathtools` parancs ugyanúgy használható, mint az `\clap`, csak matematikai módban (lásd később).

13.9. Láthatatlan dobozok

```
\phantom{<szöveg>}
```

A létrehozott doboz úgy viselkedik, mintha láthatatlan betűkkel íródott volna. Például

```
</> \noindent Ez most látszik,\ \
\phantom{most nem látszik,}
most megint látszik.
```

```
Ez most látszik,
most megint látszik.
```

```
\vphantom{<szöveg>}
```

Létrehoz egy `<szöveg>` teljes magasságával megegyező teljes magasságú, 0 pt szélességű ún. *gyámfát*. A `<szöveg>` nem jelenik meg. Például

```
\setlength{\fboxsep}{0pt}
\fbox{g} \fbox{\vphantom{g}}
```

g |

\phantom{<szöveg>}

Létrehoz egy *<szöveg>* szélességével megegyező szélességű, 0 pt magasságú és 0 pt mélységű dobozt. A *<szöveg>* nem jelenik meg. Például

```
\setlength{\fboxsep}{0pt}
\fbox{g} \fbox{\phantom{g}}
```

g —

13.10. Dobozok halmozása

Dobozok halmozására a **stackengine** csomagot használhatjuk.

\Shortstack[<igazítás>]{<dobozlista>}

A *<dobozlista>* elemeit egymás alá helyezi úgy, hogy a dobozok függőlegesen egyforma távolságokra vannak egymástól (alapesetben 3pt), továbbá az utolsó listaelem helyezkedik el a környező szöveg alapvonalán. A *<dobozlista>* elemeit szóközökkel kell elválasztani. Ha egy listaelem szóközöt vagy parancsot tartalmaz, akkor kapcsos zárójelek közé kell tenni. Az *<igazítás>* aszerint lehet **c**, **l** vagy **r**, hogy a dobozokat középre, balra vagy jobbra akarjuk igazítani (alapértéke **c**). Például

```
xxxxx
\Shortstack[r]{a bb ccc dddd yyyy} xxxxxx
\Shortstack{a bb ccc dddd yyyy} xxxxxx
\Shortstack[l]{a bb ccc dddd yyyy} xxxxxx
```

a bb ccc dddd xxxxx	a bb ccc dddd yyyyy	a bb ccc dddd yyyyy

xxxxx \Shortstack{{\LaTeX} {Plain \TeX}} xxxxxx

\LaTeX xxxxx Plain \TeX xxxxxx

\Shortunderstack[<igazítás>]{<dobozlista>}

Úgy működik, mint a \Shortstack, de ekkor az első listaelem helyezkedik el a környező szöveg alapvonalán. Például

xxxxx \Shortunderstack{{\LaTeX} {Plain \TeX}} xxxxxx

\LaTeX xxxxxx Plain \TeX

```
\def\Sstackgap{<távolság>}
```

A `\Shortstack` és `\Shortunderstack` parancsokban a dobozok alapesetben 3pt távolságra vannak egymástól. Ezzel a parancssal ez átállítható `<távolság>` értékre. Például

```
\def\Sstackgap{5pt}
xxxxx \Shortstack{a bb ccc dddd yyyy} xxxx
```

a
bb
ccc
ddd
xxxxx yyyy xxxx

```
\Longstack[<igazítás>]{<dobozlista>}
```

Úgy működik, mint a `\Shortstack`, de ekkor a dobozok alapvonalai lesznek egymástól azonos távolságra (alapértéke `\normalbaselineskip`). Például

```
xxxxx
\Longstack[r]{a bb ccc dddd yyyy} xxxx
\Longstack{a bb ccc dddd yyyy} xxxx
\Longstack[l]{a bb ccc dddd yyyy} xxxx
```

a	a	a
bb	bb	bb
ccc	ccc	ccc
ddd	ddd	ddd
xxxxx yyyy xxxx	yyyy xxxx	yyyy xxxx

```
\Longunderstack[<igazítás>]{<dobozlista>}
```

Úgy működik, mint a `\Longstack`, de ekkor az első listaelem helyezkedik el a környező szöveg alapvonalán. Például

```
xxxxx \Longunderstack{{\LaTeX} {Plain \TeX}} xxxx
```

xxxxx	\LaTeX	xxxxx
Plain \TeX		

```
\def\Lstackgap{<távolság>}
```

A `\Longstack` és `\Longunderstack` parancsokban a dobozok alapvonalai alapesetben `\normalbaselineskip` távolságra vannak egymástól. Ezzel a parancssal ez átállítható `<távolság>` értékre. Például

```
\def\Lstackgap{.5\normalbaselineskip}
xxxxx \Longstack{a bb ccc dddd yyyy} xxxx
```

a
bb
ccc
ddd
xxxxx yyyy xxxx

```
\def\useanchorwidth{T}
```

Alapesetben a halmozott dobozokat tartalmazó doboz szélessége a legszélesebb doboz elem szélessége lesz. Ezzel a parancssal a környező szöveg alapvonalán elhelyezkedő dobozelem szélességét veszi fel. Például

```
xxxxx \Shortunderstack{{\LaTeX} {Plain \TeX}} xxxx
\def\useanchorwidth{T}%
\Shortunderstack{{\LaTeX} {Plain \TeX}} xxxx
```

```
xxxxx LATEX xxxx LATEX xxxx
Plain TEX Plain TEX
```

14. fejezet

Verbatim, programkód, URL

14.1. 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 következő parancsot:

```
\verb|<verbatim szöveg>|  
\verb*|<verbatim szöveg>|
```

A | határolójel lehet bármely más, szóköztől és *-tól különböző ASCII karakter, ami nem szerepel a verbatim szövegben. Ha ASCII betű a határolójel, akkor a \verb és ezen betű közé tegyen szóközt. Például

```
</> \verb|\LaTeX\ könyv|\\  
\verb+|\LaTeX\ kód+
```

```
\LaTeX\ könyv  
\LaTeX\ kód
```

Ha \verb helyett \verb* parancsot ír, akkor az eredményben a szóközök 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 ír be 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 minden megtalál az Olvasó. Csak egy példán illusztráljuk a tudását (az `xcolor` csomag használata mellett):

```
</> \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
```

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[<opcionális>]{<név>} ∈ fancyvrb
```

Itt a | határolóra hasonló a szabály, mint a `\verb` parancsnál. Az `<opcionális>` 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|}
```

A `fancyvrb` csomagnak van egy kiterjesztése is, az `fvextra` csomag, amely számos új opciót és parancsot definiál, például a `breaklines` opciót, amely lehetővé teszi a túl hosszú verbatim sorok megtörését. Az `fvextra` csomag automatikusan betölти a `fancyvrb` csomagot is.

14.2. Verbatim szöveg kiírása fájlba

Ha azt akarja, hogy fordítás közben egy L^AT_EX-kód ne értelmeződjön, hanem egy fájlba legyen elmentve, akkor használja a `filecontents*` környezetet `overwrite` opcióval:

```
\begin{filecontents*}[overwrite]{<fájlnév>}
```

Például

```
\begin{filecontents*}[overwrite]{pelda.tex}
\LaTeX
\end{filecontents*}
```

hatására a létrejön egy `pelda.tex` fájl az aktuális mappában, aminek a tartalma

```
\LaTeX
```

lesz. Ha már korábban létezett a `pelda.tex` fájl, akkor annak tartalmát felülírja és erről egy figyelmeztetést is kap a konzolon. Ha nem akarja ezt a figyelmeztetést, akkor az `overwrite` mellett használja a `nowarn` opciót is.

További lehetőségeket tartalmaz erre a feladatra a `newfile` és az `answers` csomagok.

14.2.1. A `newfile` csomag

```
\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 kimenti a \texttt{minta.tex}-be,
\end{writeverbatim}
BBB
\begin{writeverbatim}{proba}
majd ezt hozzáfűzi.
\end{writeverbatim}
CCC
\closeoutputstream{proba}
```

Ebben a mentés folyamatának `proba` nevet adtunk, amely a `minta.tex` fájlba menti el verbatim szövegként azon `writeverbatim` környezetek tartalmát, melyek argu-

mentumában `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 kimenti a \texttt{\texttt{minta.tex}}-be,  
majd ezt hozzáfűzi.
```

A következő paranccsal egy adott szöveget úgy menthet el egy fájlban, hogy a benne található parancsokat kifejtse:

```
\addtostream{\langle streamnév\rangle}{\langle szöveg\rangle} \in newfile
```

Például

```
\newoutputstream{proba}  
\openoutputfile{minta.tex}{proba}  
\addtostream{proba}{\thepage}  
\closeoutputstream{proba}
```

esetén a `minta.tex` fájlba bekerül az éppen aktuális oldal száma. Ha az `\addtostream` parancsban a `\langle szöveg\rangle` által tartalmazott parancsok valamelyikét nem akarja kifejteni, akkor tegyen elé `\protect` parancsot. Például

```
\newoutputstream{proba}  
\openoutputfile{minta.tex}{proba}  
\addtostream{proba}{  
  \protect\section{Cím}  
  Aktuális szakaszszám: \thesection}  
\closeoutputstream{proba}
```

esetén a `minta.tex` fájl tartalma a következő lesz:

```
\section{Cím}  
Aktuális szakaszszám: 14.2
```

Ha létrehoz egy `\fajlnév` fájlt az előző módon, akkor azt egy adott ponton be is töltheti a forrásfájlba a következő paranccsal:

```
\input{\fajlnév}
```

Például

```
\newoutputstream{proba}  
\openoutputfile{minta.tex}{proba}  
\begin{writeverbatim}{proba}  
BBB  
\end{writeverbatim}  
\closeoutputstream{proba}  
AAA  
\input{minta.tex}
```

eredménye

```
AAA BBB
```

Az előző kódban az `\input{minta.tex}` csak a `\closeoutputstream{proba}` után állhat. Ha már korábban szükség van a fájl betöltésére, akkor a következő kód használható:

```
\newinputstream{\langle streamnév\rangle} ∈ newfile
\openinputfile{\langle fájlnév\rangle}{\langle streamnév\rangle} ∈ newfile
\readstream{\langle streamnév\rangle} ∈ newfile
\closeinputstream{\langle streamnév\rangle} ∈ newfile
```

ahol a `\langle streamnév\rangle` nem ugyanaz, mint output esetén, de a `\langle fájlnév\rangle` igen. Például

```
\newinputstream{probainput}
\openinputfile{minta.tex}{probainput}
\readstream{probainput}
\closeinputstream{probainput}
AAA
\newoutputstream{proba}
\openoutputfile{minta.tex}{proba}
\begin{writeverbatim}{proba}
BBB
\end{writeverbatim}
\closeoutputstream{proba}
```

eredménye

```
BBB
AAA
```

Ennek a megoldásnak a hátránya, hogy az első fordításnál hibát fog jelezni, mert ekkor még nem találja a `minta.tex` fájlt, de ez a második fordításnál megszűnik. Ha az első fordításnál sem szeretne hibát kapni, akkor gépelje be a következőt a preambulum elejére:

```
\usepackage{silence}
\ErrorsOff*
```

Az előbb leírtak jól használhatók például a következő esetben. Az a feladata, hogy írjon egy példatárat úgy, hogy a megoldások külön kötetben szerepeljenek. A feladatok számozását automatikusra kell állítani (lásd később), hiszen előfordulhat, hogy már begépelt két feladat közé kell beékelni egy harmadikat. Ilyenkor a számozások elcsúsznak. A probléma az, hogy ilyen beékelések esetén a megoldásoknál is meg kell keresni a beszúrási pontot. Ez gyakorlatilag átláthatatlan káoszt és sok hibát 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.

14.2.2. Az answers csomag

```
\Opensolutionfile{\langle streamnév\rangle}{\langle fájlnév\rangle} ∈ answers
\Closesolutionfile{\langle streamnév\rangle} ∈ answers
\begin{Filesave}{\langle streamnév\rangle} ∈ answers
\Newassociation{\langle verbatim környezet\rangle}{\langle környezet\rangle}{\langle streamnév\rangle} ∈ answers
```

Ha a `\fajlnév` 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 kimenti a \kodx{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` paranccsal 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

14.3. Programkódok

Különböző programnyelvek kódjainak megjelenítésére alkalmas a `listings` és `minted` csomagok.

14.3.1. A `listings` csomag

```
\lstinline[opcion](<határoló>) <kód>(<határoló>) ∈ listings
\begin{lstlisting}[opcion] ∈ listings
<kód>
\end{lstlisting}
\lstinputlisting[opcion]{{<kódot tartalmazó fájl>}} ∈ listings
```

Az `\lstinline` sorközi kód esetén alkalmazható, melyben a `<határoló>` hasonlóan adható meg, mint a `\verb` esetén. Ugyanakkor `<határoló>`-ként kapcsos zárójel is használható, mint a paraméteres parancsokban, kivéve táblázatokban.

Az `<opcion>` a következő parancsban is megadhatók:

```
\lstset{opcion} ∈ 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{literate={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{Ü}{{\H{U}}}1{É}{{'E}}1{Á}{{'A}}1
{Í}{{'I}}1}
```

Ez a magyar ékezetes betűket parancs formátumra konvertálja, ami megoldja a problémát.

Opciók ♦ Tekintsük át az előbbi parancsok opciót. 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 akarja, 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 esetén

`postbreak=\hbox{\textcolor{red}{\rightarrowfill}}`

`prebreak=\hbox{⟨jel⟩}` Ugyanaz, mint előbb, csak a sorok törése elő 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 `\listinline` parancsban hatástan.

`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.

`linewidth=⟨hossz⟩` Szövegtükör bal széle és a kód jobb széle közötti távolság.

`showspaces` Szóköz ↞ módon jelölve.

`showtabs` Tabulátort jelöli.

`tabsize=⟨szóközszerződés⟩` Tabulátor mérete `⟨szóközszerződés⟩` 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. A számlálója **lstnumber**. Például

```
numberstyle=\tiny
```

numbersep=⟨hossz⟩ A sorszám és a kód távolsága.

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). Lehetséges még **shadowbox** és **none** is az érték. Például, ha fent és bal oldalon akarunk vonalat húzni, akkor

```
frame=tl
```

frameround=⟨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

```
frameround=tftf
```

framerule=⟨hossz⟩ Keret vonalának vastagsága.

framesep=⟨hossz⟩ Keret és kód közötti távolság.

rulesep=⟨hossz⟩ Keret dupla vonalai közötti távolság.

rulecolor=⟨szín⟩ Keret vonalának színe. Például

```
rulecolor=\color{red}
```

rulesepcolor=⟨szín⟩ A dupla keretvonalak közötti területnek a színét ezzel állíthatjuk be.

fillcolor=⟨szín⟩ Keret és kód közötti szín.

literate={⟨mit⟩}{⟨mire⟩}{⟨szám⟩} A programkódban található ⟨mit⟩ helyére a ⟨mire⟩ L^AT_EX-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 található

```
(* \pounds *)
```

helyén a végeredményben £ lesz, azaz (* és *) jelek közötti L^AT_EX-parancs ki lesz fejtve. Ez az opció nem használható **\listinline** esetén. Ezt a problémát úgy lehet megoldani, hogy a **listings** (vagy **listingsutf8**) csomag betöltése után a következőket írja a preambulumba:

```
\usepackage{etoolbox}
\makeatletter
\patchcmd{\lst@TextStyle}{\let\lst@DefEsc\empty}{\def\lst@DefEsc{\empty}}{}{}
```

`language=⟨programnyelv⟩` Programnyelv kulcsszavainak, megjegyzéseinek a kiemelését tölti be. A definiált nyelvek listája megtalálható a csomag leírásában. Például

`language=Delphi`

Saját nyelvet így lehet definiálni:

```
\lstdefinelanguage{⟨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 is beállít kiemelést, akkor azt a `language` opció után tegye, 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. Ezzel az opcióval a korábban beállított megjegyzés stílusok törlődnek. Ha `comment` helyett `morecomment` opciót használ, 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=[1][⟨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=[1][\itshape\color{red}]{//}`

esetén a `//` jelről az adott sor az adott stílusú lesz, beleértve a `//` jelet is.

`commentstyle=⟨stílus⟩` Például, ha

`comment=[1]{//}`

módon definiált megjegyzést, akkor alapból dőlt betűvel fog megjelenni. Ezt a stílust utólag ezzel az opcióval módosíthatja. Például

```
commentstyle=\itshape\color{green}
```

delim=[s] [*stílus*] {*ettől*} H*(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*} H*(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={\\}**

opcionálisan. Ezután a

```
morekeywords={\\chapter,\\section}
```

opcionálisan definiálhatja a kulcsszavakat.

style=⟨stílusnév⟩ Előre definiált stílust hív meg. Stílus definiálása a következő parancssal történik:

```
\lstdefinestyle{⟨stílusnév⟩}{⟨opcionálisan⟩}
```

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 töltse 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 Számoszt kód ne kerüljön be a kódok jegyzékébe.

numberbychapter=false A kódok számosztása ne a fejezetszámmal együtt történjen.

label={⟨címke⟩} Kereszthivatkozás címkéje. Ezt a **\label** parancs helyett kell használni.

A parancs azért nem használható, mert azt a L^AT_EX már a programkód részének tekintené.

Példák ♦



```
\begin{lstlisting}[language=Delphi,basicstyle=\footnotesize]
function Trim(s: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;

```

A következő példához be kell tölteni az `xcolor` és `caption` csomagokat is.

</>

```

\def\lstlistingname{kód}
\lstset{language=Delphi,basicstyle=\footnotesize,
keywordstyle=\bfseries\color{blue},numbers=left,
frame=tBBl,frameround=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 ...

A `magyar.ldf` és a `listings` csomag együttes használata esetén, ha aktív karakter (`:;!?`) van egy címben (`\title`, `\author`, `\chapter`, `\section`, stb.) vagy egy `\label`-ben, akkor a 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 legszerencsébb megoldás, mert ezzel a magyar tipográfiában kötelező kis szóköz a `:!?` jelek előtt nem fog megjelenni. Szerencsébb (bár körülményesebb) megoldás, hogy a címben szereplő `:!?` karakterek elé `\kern.1em\string` parancsot, míg ``` előle `\shu` parancsot teszünk. A `\label` parancsban megadott címkében az aktív karaktereket törölje vagy cserélje például kötőjelre.

14.3.2. A `minted` csomag

Programkódok kiíratására a `minted` csomag is használható, de ez a `listings` csomagtól eltérően egy Python könyvtárat vesz igénybe. Ezért a használatához először telepíteni kell a `Python` rendszert. Amennyiben `TeXfireplace` keretrendszer használ, akkor erre nincs szükség, mert az alapból tartalmazza. Szintén nincs szükség kiegészítésre `Overleaf` használata esetén.

Ha a `babel` csomag `magyar` opciónját használja, akkor lehetséges, hogy a ``` jel aktív, ami a `minted` csomag esetén bizonyos esetekben problémát jelenthet. Ennek megoldására használja a `magyar.ldf active=onlycs` opciónját (lásd a 3.4. szakaszt).

A programkódok kiíratása a következő parancsokkal történhet:

```
\mintinline[<opción>]{<nyelv>}<határoló><kód><határoló> ∈ minted
\begin{minted}[<opción>]{<nyelv>} ∈ minted
<kód>
\end{minted}
\inputminted[<opción>]{<nyelv>}=<kódot tartalmazó fájl> ∈ minted
```

Az `\mintinline` sorközi kód esetén alkalmazható, melyben a `<határoló>` hasonlóan adható meg, mint a `\verb` esetén. Ugyanakkor `<határoló>`-ként kapcsos zárójel is használható, mint a paraméteres parancsokban. Ez utóbbit kell használni, ha törékeny argumentumba írjuk, például `\section` parancsba.

A `<nyelv>` a `<kód>` programnyelvének a neve. Ezek listája elérhető a `Pygments` oldalán. Például

szöveg `\mintinline{python}{print(x**2)}` szöveg

szöveg `print(x**2)` szöveg

A `<kódot tartalmazó fáj>` neve egy parancssorban végrehajtott parancsot generál, így a kellemetlen helyzeteket elkerülendő érdemes a fájlnévben a szóközöt mellőzni. A hordozhatóság miatt ékezetes betűket sem érdemes a fájlnévben használni.

A lehetséges `<opción>` közül felsorolunk néhányat:

`style=<stílusnév>` Ez határozza meg a kiemelés színösszetételét. Az alapértelmezett `<stílusnév>` `default`. A definiált `<stílusnév>` lista elérhető a `Pygments` oldalán. Ha egy `<stílusnév>` néhány elemét szeretné a L^AT_EX-forrásban átdefiniálni, akkor az `xstring` csomag betöltése után ezt például a következő módon teheti meg.

```
\makeatletter
\AddToHook{cmd/minted@patch@PygmentsStyleDef/after}{%
  \saveexpandmode%
  \fullexpandarg%
  \IfStrEq{\mintedtexoptvalueof{style}}{staroffice}{%
    \def\PYG@tok@k{\let\PYG@bf\textbf{%
      \def\PYG@tc##1{\textcolor{red}{##1}}}}%
  }{%
    \restoreexpandmode%
  }
\makeatother
```

Ekkor a `staroffice` stílusnév esetén a kulcsszavak pirossal és félkövéren lesznek kiszedve. A további átdefiniálási lehetőségek a forrásfájl mappájában generált `_minted` mappában található `.minted` és `.style.minted` kiterjesztésű fájlok elemzésével fedezhetők fel.

formatcom=*<formázási parancsok>* A *<formázási parancsok>* a kód elején lesznek kifejtve. Ezzel a kód betűtípusát is be lehet állítani. Például

```
formatcom=\footnotesize\itshape
```

fontfamily=*<érték>* Beállítja a fontcsaládot. Lehetséges értékek: **tt**, **courier**, **helvetica**. Alapértéke **tt**.

autogobble Ha a kód minden sora szóközökkel kezdődik, akkor a soreleji felesleges szóközöket eltávolítja.

gobble=*<szám>* minden sor elején *<szám>* darab karaktert eltávolít.

baselinestretch=*<szorzó>* A sortávolságot megszorozza a *<szorzó>* értékével.

linenos Sorok számozása. Ennek számlálója **FancyVerbLine**, így a sorszámok formázása a **\theFancyVerbLine** parancs átdefiniálásával lehetséges. Például

```
\def\theFancyVerbLine{\textcolor{red}{\tiny\arabic{FancyVerbLine}}}
```

firstline=*<szám>* A bemeneti kód hanyadik sorától jelenjen meg a kimenetben.

lastline=*<szám>* A bemeneti kód hanyadik soráig jelenjen meg a kimenetben.

firstnumber=*<szám>* Az első sorszám értéke.

numberblanklines=false Az üres sorok nem lesznek számozva.

numbersep=*<távolság>* A sorszámok és a sorok eleje közötti távolság. Alapértéke **12pt**.

showspaces A szóközök helyét megjelöli.

space=*<szóköz jele>* A **showspaces** opció esetén ez lesz a szóköz jele. Alapértéke **,**, azaz

```
space=\textvisiblespace
```

tabsize=*<szám>* A tabulátorot *<szám>* darab szóközzel helyettesíti. Alapértéke **8**.

xleftmargin=*<távolság>* A kód bal margójának nagysága.

xrightmargin=*<távolság>* A kód jobb margójának nagysága.

frame=single Bekeretezi a kódot.

framerule=*<vastagság>* A keret vastagsága (alapértéke **0.4pt**).

framesep=*<távolság>* A keret és a kód közötti távolság (alapértéke **\fboxsep**).

rulecolor=*<színnév>* A keret színének a neve.

highlightlines={<sorszámlista>} Kiemeli a megadott sorszámú sorokat. Például

```
highlightlines={1,3-5}
```

highlightcolor=*<színnév>* A kiemelt sorok színének a neve (alapértéke **LightCyan**).

bgcolor=*<színnév>* Háttér színe. Például

```
bgcolor=lightgray
```

breaklines Lehetővé teszi a kód túl hosszú sorainak a megtörését bármely szóköznél.

breaksymbolright=\hbox{<törésjel>} Ezt a jelet használja a **breaklines** opció a sortörés jobb oldalán. Alapértéke üres.

breaksymbolleft=\hbox{<törésjel>} Ezt a jelet használja a **breaklines** opció a sortörés bal oldalán. A *<törésjel>* alapértéke **\hookrightarrow**, aminek a kódja

```
\tiny\ensuremath{\hookrightarrow}
```

breakanywhere A **breaklines** opcióval együtt használva bármely karakternél megtörhető a sor, nem csak szóköznél. Ha a kód ékezes betűket is tartalmaz, akkor **pdflatex** fordító esetén nem biztos, hogy működni fog ez az opció. Ha **xelatex** vagy **lualatex** fordítót használ, akkor nincs ilyen gond.

breakanywheresymbolpre=\hbox{<törésjel>} Ezt a jelet használja a **breakanywhere** opció a sortörés bal oldalán. A *<törésjel>* alapértéke **\rfloor**, aminek a kódja

```
\,\footnotesize\rfloor
```

breakanywheresymbolpost=\hbox{*törésjel*} Ezt a jelet használja a **breakanywhere** opció a sortörés jobb oldalán. Alapértéke üres.

breakafter=*karakterek* A **breaklines** opciójával együtt használva nemcsak szóköznél, hanem a felsorolt *karakterek* bármelyikénél is megtörhető a sor. Speciális karakterek (foglalt karakter illetve az opciót jelölő szögletes zárójelek) esetén egy \ jelet kell elé tenni. Például

```
breakafter=-/\#
```

esetén a - / # jelek bármelyike és a szóköz után lehetséges sortörés.

breakafterinrun A **breakafter** opciójával együtt használva, ha a megadott karakterekből több is előfordul közvetlenül egymás után egy csoportban, akkor csak a csoport utolsó karaktere után lehetséges sortörés.

breakaftersymbolpre=\hbox{*törésjel*} Ezt a jelet használja a **breakafter** opció a sortörés bal oldalán. A *törésjel* alapértéke „, aminek a kódja

```
\,\footnotesize\ensuremath{\lfloor}
```

breakaftersymbolpost=\hbox{*törésjel*} Ezt a jelet használja a **breakafter** opció a sortörés jobb oldalán. Alapértéke üres.

breakbefore=*karakterek* Hasonló mint a **breakafter**, de ekkor nem a megadott karakterek után, hanem előttük lehetséges sortörés.

breakbeforeinrun A **breakbefore** opciójával együtt használva, ha a megadott karakterekből több is előfordul közvetlenül egymás után egy csoportban, akkor csak a csoport első karaktere előtt lehetséges sortörés.

breakbeforesymbolpre=\hbox{*törésjel*} Ezt a jelet használja a **breakbefore** opció a sortörés bal oldalán. A *törésjel* alapértéke „, aminek a kódja

```
\,\footnotesize\ensuremath{\rfloor}
```

breakbeforesymbolpost=\hbox{*törésjel*} Ezt a jelet használja a **breakbefore** opció a sortörés jobb oldalán. Alapértéke üres.

escapeinside=*karakter1**karakter2* A megadott karakterek között a L^AT_EX-kódok kifejtődnek. Speciális karakterek (foglalt karakter illetve az opciót jelölő szögletes zárójelek) esetén egy \ jelet kell elé tenni. Például

```
escapeinside=\#\%
```

A megadott két karakter lehet azonos is, például

```
escapeinside=||
```

Az **escapeinside** opciótól nincs hatása sztringeken és kommenteken belül.

texcomments A kommentekben a L^AT_EX-kódok kifejtődnek.

mathescape A kommentekben a matematikai mód elérhető lesz.

vspace=*hossz* A **minted** környezet és az **\inputminted** parancs által megjelenített kiemelt kódok előtti és utáni függőleges térközök mérete. Alapértéke **\topsep**.

További parancsok ♦ Az *opciók* globálisan is megadhatók a következő parancssal:

```
\setminted[nyelv]{opciók} \minted
```

Ha nem adunk meg *nyelv*-et, akkor minden programnyelvre vonatkoznak az *opciók*.

```
\setmintedinline[nyelv]{opciók} \minted
```

Hasonló, mint a **\setminted**, de csak a **\minted** parancsra hat.

```
\usemintedstyle[nyelv]{stílusnév} \minted
```

Ezzel globálisan is megadható a `\begin{minted}[<környezetnév>]{<nyelv>}{<opción>}`. Ha nem adunk meg `<nyelv>`-et, akkor minden programnyelvre vonatkozik.

```
\begin{minted}[<környezetnév>]{<nyelv>}{<opción>} ∈ minted
```

Ennek hatására a

```
\begin{minted}[<környezetnév>]
<kód>
\end{minted}
```

ugyanazt eredményezi, mint

```
\begin{minted}[<opción>]{<nyelv>}
<kód>
\end{minted}
```

Másrészről

```
\begin{minted}[<környezetnév>*]{<további opciók>}
<kód>
\end{minted}
```

ugyanazt eredményezi, mint

```
\begin{minted}[<opción>, <további opciók>]{<nyelv>}
<kód>
\end{minted}
```

A `<környezetnév>` alapértéke `<nyelv>code` lesz.

```
\begin{mintedinline}[<parancsnév>]{<nyelv>}{<opción>} ∈ minted
```

Ennek hatására a

```
\begin{minted}[<parancsnév>]{<határoló><kód><határoló>}
```

ugyanazt eredményezi, mint

```
\begin{mintedinline}[<opción>]{<nyelv>}{<határoló><kód><határoló>}
```

A `<parancsnév>` alapértéke `<nyelv>inline` lesz.

```
\begin{mintedfile}[<parancsnév>]{<nyelv>}{<opción>} ∈ minted
```

Ennek hatására a

```
\begin{minted}[<parancsnév>]{<kódot tartalmazó fájl>}
```

ugyanazt eredményezi, mint

```
\begin{inputminted}[<opción>]{<nyelv>}{<kódot tartalmazó fájl>}
```

A `<parancsnév>` alapértéke `<nyelv>file` lesz.

Kódok feliratozása ♦ A programkódok feliratozásához használhatja a `minted` csomag `newfloat` opcióját. Ekkor betöltődik a `newfloat` csomag is, melynek a segítségével létrejön egy `listing` úsztatott környezet. Ezzel a szokott módon feliratozhat. Ha magyar nyelvű dokumentumot ír, akkor még töltse be a `caption` csomagot is. Például

```
\begin{listing}[ht!]
\caption{Egy példakód}\label{lst-pelda}
\begin{minted}[vspace=0pt]{python}
def boring(args = None):
    pass
\end{minted}
```

```
\end{listing}
```

Az előbbi kódban a `vspace=0pt` opció azért kell, hogy a `minted` környezet ne növelje meg a `listing` környezet által beállított függőleges térközöket.

A `listing` környezet használatának a hátránya, hogy megakadályozza az oldaltörést, amikor a kód nem fér ki az adott oldalra. Ekkor használhatja a `caption` csomag betöltése után a `\captionof` parancsot. Például

```
\begingroup
\captionof{listing}{Egy példakód}\label{lst-pelda}
\begin{minted}[python]
def boring(args = None):
    pass
\end{minted}
\endgroup
```

A `\captionof` csak blokkban használható, ezért vannak a `\begingroup` és `\endgroup` parancsok.

A felirat címkéje „Listing” lesz, ami átdefiniálható így:

```
\SetupFloatingEnvironment{listing}{name=<címke>}
```

A `minted` illetve `listings` segítségével megjelenített kódok feliratozására és keretezésére jóval több lehetőséget biztosít a `tcolorbox` csomag, de ezt itt nem részletezzük.

A `minted` csomag opciói ♦ A `minted` csomag betöltésekor a következő opciók használhatók többek között:

`highlightmode=immediate` A `minted` csomag 2.9-nél újabb verziója esetén az első fordítás után a programkódok helyén a pdf-ben egy **<MINTED>** felirat jelenik meg. A végeredmény csak a második fordítás után lesz látható. Ez kereszthivatkozások esetén nem jelent gondot, hiszen ekkor amúgy is többször kell fordítani. Ha mégis szükség lenne az eredmény megjelenésére az első fordítás után, akkor használja ezt az opciót. Kereszthivatkozások esetén kerülje ezt az opciót, mert jelentősen megnövelheti a fordítási időt.

`newfloat` A kódok feliratozásához célszerű betölteni, amely a `newfloat` csomag segítségével definiál egy `listing` nevű úsztatott környezetet.

`chapter` A kódlisták száma új fejezet nyitásakor nullázódik, továbbá a szám elő bekerül a fejezet száma is egy ponttal elválasztva.

`section` A kódlisták száma új szakasz nyitásakor nullázódik, továbbá a szám elő bekerül a szakasz száma is egy ponttal elválasztva.

`frozencache` A szerkesztés során a `minted` gyorsítótárat használ. Amikor már nem kell változtatni a megjelenítendő programkódokon, akkor ezzel az opcióval lehet véglegesíteni a gyorsítótárat. Ezzel a programkódok tartalma már nem változtatható, viszont a fordítás már működni fog Python nélkül is egyetlen futtatással. Ennek akkor van haszna, ha olyan felhasználóhoz kerül a L^AT_EX-forrás, akinek a gépére nincs telepítve Python. Ezzel a fordítási idő is jelentősen lecsökken.

Példa ♦ Most tekintsünk egy teljes példát.



```
\documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min,active=onlycs}{magyar.ldf}
\usepackage[magyar]{babel}
```

```
\usepackage{caption}
\usepackage[newfloat]{minted}
\SetupFloatingEnvironment{listing}{name=kód}

\begin{document}

\begin{listing}[ht!]
\caption{Egy példakód}\label{lst-pelda}
\begin{minted}[vspace=0pt,linenos,frame=single]{Delphi}
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{minted}
\end{listing}

Lásd \az{\ref{lst-pelda}}.~kódot.

\end{document}
```

1. kód. Egy példakód

```
1 function Trim(s:string):string;
2 var i:integer;
3 begin
4   result := '';
5   for i := 1 to length(s) do
6     if s[i] <> ' ' then result := result + s[i];
7 end;
```

Lásd az 1. kódot.

14.4. URL címek megadása

Az internetcímek is verbatimnak tekinthetők. Ezeket

```
\url{<URL cím>} ∈ url
```

módon lehet megadni, amely meg tudja törni a sor végén az URL címet. A lehetséges töréspontok megadhatók az `\UrlBreaks` parancssal. Például

```
\def\UrlBreaks{\do\.\do@\do\\\do\/\do!\do\_do\\\do\\\do;\do\>\do]\do\{-}
```

Ha bármelyik karakternél meg akarja engedni a törést, akkor nincs szükség az előbbi megoldásra, csak `ur1` helyett az `xurl` csomagot kell használni.

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}
```

Ha az `\url` parancs olyan URL címet tartalmaz, melyben % jel is van, továbbá ez az `\url` parancs `\caption`, `\footnote` vagy valamilyen bekezdésdobozban helyezkedik el, akkor a fordítás hibás lesz. Ennek megoldására használja az

```
\urldef{\parancs}\url{\URL cím}
```

kódot a dobozon kívül, majd a dobozba az `\url{\URL cím}` helyett a `\parancs`-ot kell írni. Például a

```
szöveg\footnote{\url{www.google.com/search?client=firefox-b-d&q=%25}}
```

kód hibával fordulna, de

```
\urldef{\myurl}\url{www.google.com/search?client=firefox-b-d&q=%25}  
szöveg\footnote{\myurl}
```

hatására a % jelet tartalmazó URL cím hiba nélkül megjelenik a lábjegyzetben.

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
```

Ha bármelyik karakternél meg akarja engedni a törést, akkor a `hyperref` előtt az `xurl` csomagot is töltse be. Amennyiben az `xurl` és a `biblatax` csomagokat együtt használja, akkor az `xurl`-t a `biblatax` után töltse be.

A `hyperref` csomag `latex` fordító esetén nem töri meg a linkekét. Ekkor töltse még be a `hyperref` után a `breakurl` csomagot is.

15. fejezet

Képletek

15.1. Matematikai mód

Ha matematikai képletet akar szerkeszteni, akkor használja a `mathtools` és `amssymb` csomagokat. *Ha egy parancs csak ezen két csomag valamelyikének betöltésével érhető el, akkor azt a továbbiakban már nem fogjuk külön jelezni!*

A `mathtools` az `amsmath` csomag kiterjesztése. Ezt úgy éri el, hogy a `mathtools` először betölti az `amsmath` csomagot, majd további funkciókkal bővíti, illetve néhány hibát kijavít. A továbbiakban arra nem fogunk kitérni, hogy az egyes parancsokat az `amsmath` vagy a `mathtools` definiálja. Ehhez olvassa el a csomagok leírásait.

Amennyiben a `hyperref` csomagot is használja (lásd a 19.1. szakaszban), akkor azt a `mathtools`, `amsmath`, `amssymb` csomagok után töltse be. A `mathtools` és `amsmath` csomagok a standard dokumentumosztályok opciójában megadott lapméretet fizikailag is beállítják.

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 L^AT_EX-hel tudatnunk kell, hogy képlet következik.

Ha egy képlet kb. akkora mint egy szó, akkor azt a szövegbe illesztjük, mint például $\sin^2 \alpha + \cos^2 \alpha = 1$ esetén. Ez az ún. *szövegközi 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*.

Szövegközi matematikai mód megadása a következő sorok bármelyikével lehetséges:

```
$<math>\langle\text{képlet}\rangle$  
\langle\text{képlet}\rangle\\  
\begin{math}\langle\text{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:

```
\[<math>\langle\text{képlet}\rangle\]
\begin{displaymath}\langle\text{képlet}\rangle\end{displaymath}
\begin{equation*}\langle\text{képlet}\rangle\end{equation*}
```

Például

 Bármit is teszünk, `\[2+2=4.\]`

Bármit is teszünk,

$$2 + 2 = 4.$$

Sokan helytelenül a L^AT_EX-ben a kiemelt matematikai mód megadására a `$$\langle\text{képlet}\rangle$$` kódot használják, ami első ránézésre jónak tűnik. Ugyanakkor több problémát is okoz, amiért kerülni kell a használatát (lásd például <https://texfaq.org/FAQ-dollardoll>).

A kiemelt matematikai képletek alaphelyzetben középre igazítva jelennek meg. Azonban a `mathtools` csomag `fleqn` opciójával elérhető, hogy balra legyen igazítva úgy, hogy a képlet a bal margótól 2,5 em távolságra kezdődjön. Ez az érték átállítható a következő parancssal:

```
\setlength{\mathindent}{\terköz}
```

A `fleqn` opció globális hatású. Ha lokálisan akarja használni egy adott kiemelt képletre, akkor azt

```
\begin{fleqn}[\bal{margó}] \in nccmath
\langle\text{kiemelt}\text{ }\text{képlet}\rangle
\end{fleqn}
```

környezetbe kell zárni. A `\bal{margó}` alapértéke 0pt. Például

```
\begin{fleqn}
\[2+2=4\]
\end{fleqn}
```

A következő parancs makrók írásánál hasznos:

```
\ensuremath{\langle\text{képlet}\rangle}
```

Ez függetlenül attól, hogy az `\ensuremath` matematikai vagy szöveg módban lett aktíválva, 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 lépnünk a matematikai módból a

```
\text{\szöveg}
```

parancssal. 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 matematikai módban a szóközöket

a L^AT_EX felülbírálja, mert a nagyon speciális tipográfia miatt ezt nem bízza a szerzőre. Ilyenkor az a megoldás, hogy a szóközöket szöveg módban kell kiadni:

```
\[1+1=2\text{ és }2+2=4\]
```

$$1 + 1 = 2 \text{ és } 2 + 2 = 4$$

De ez még mindig nem tökéletes, mert 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

```
$\frac{31}{54}$  
\[\frac{31}{54}\]
```

$$\frac{31}{54}$$

Amint látjuk a méret sem egyforma. Ha kiemelt matematikai módban olyan betűmérettel és stílusban szeretne valamit megjeleníteni, mintha az szövegközi matematikai módban lenne, akkor használja a

`\textstyle`

parancsot, fordított esetben pedig a

`\displaystyle`

parancsot. Például

```
$\frac{31}{54} \ \text{\displaystyle\frac{31}{54}}$  
\[\frac{31}{54} \ \text{\textstyle\frac{31}{54}}\]
```

$$\frac{31}{54}$$

Az indexek minden matematikai módban ugyanakkorák, hasonlóképpen az index indexe is. Ha nem indexbe ír, de index stílusra akar váltani, akkor használja a

`\scriptstyle`

parancsot. Index indexe stílusra átváltani a következő paranccsal lehet:

`\scriptscriptstyle`

Például

```
$2^x \ \text{\scriptstyle 2^x} \ \text{\scriptscriptstyle 2^x}$
```

$$2^x_{2^x_{2^x}}$$

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: $a^2+b^2=c^2$}
```

$$\text{Pitagorasz-tétel: } a^2 + b^2 = c^2$$

$$\text{Pitagorasz-tétel: } a^2 + b^2 = c^2$$

A `mathtools` csomaggal ez akkor is működik, 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 a `mathtools` csomaggal együtt sem. Ezen segít a `fixcmex` csomag, melyet az `lmodern` után kell betölteni.

15.2. Matematikai betűváltozatok

Matematikai módban a betűk közötti távolság és a szóközök kezelése másképpen történik, mint szövegmódban. Ezért a betűtípusokat nem a `\textit`, `\textrm` stb. paranccsal választjuk ki, hanem

```
\mathit{<karakterek>}
\mathrm{<karakterek>}
\mathbf{<karakterek>}
\mathsf{<karakterek>}
\mathbf{<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 minden félkövéren akar szedni, mint a következő esetben:

```
\section*{A $\mathbf{\sum \frac{1}{n}}$ sor tulajdonságai}
```

$$A \sum \frac{1}{n} \text{ sor tulajdonságai}$$

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}
```

$$A \sum \frac{1}{n} \text{ sor tulajdonságai}$$

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

```
\[\pmb{\alpha}\]
```

$$\alpha$$

A `\pmb` parancshoz hasonlóan működik a

```
\bm{\langle karakterek\rangle} \in \bm
```

parancs is, ami sokszor jóval szébb eredményt ad. Sok matematikai szimbólumnak és betűnek van félkövér verziója, ami a

```
\boldsymbol{\langle karakterek\rangle}
```

parancsal jelenik meg. Néhány jelnek, mint a szumma vagy integrál, nincs félkövér verziója. Ilyenkor hatástanlan a `\boldsymbol`. Ebben az esetben használja a következő kódot a preambulumban:

```
\usepackage{pdfrender}
\def\boldsymbolx#1{\textpdfrender{TextRenderingMode=2,
  LineJoinStyle=1,LineWidth=.3pt}{#1}}
```

Ezután a `\boldsymbolx` hasonlóan használható mint a `\boldsymbol`.

15.3. Kalligrafikus, dupla szárú betűk és fraktúrák

```
\mathcal{\langle karakterek\rangle}
\mathscr{\langle karakterek\rangle} \in \mathfrak{mathrsfs}
\mathbb{\langle karakterek\rangle}
\mathds{\langle karakterek\rangle} \in \mathfrak{dsfont}
\mathds{\langle karakterek\rangle} \in \mathfrak{[sans]dsfont}
\mathfrak{\langle karakterek\rangle}
```

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Ha a `dsfont` csomagot `sans` opciójával töltöttem be, akkor

```
$\mathds{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
```

ABCDEFGHIJKLMNOPQRSTUVWXYZ

15.4. Görög betűk

α \alpha	ζ \zeta	λ \lambda	ρ \rho
β \beta	η \eta	μ \mu	σ \sigma
γ \gamma	θ \theta	ν \nu	τ \tau
δ \delta	ι \iota	ξ \xi	υ \upsilon
ϵ \epsilon	κ \kappa	π \pi	ϕ \phi

χ	<code>\chi</code>	ς	<code>\varsigma</code>	Υ	<code>\Upsilon</code>	\varPi	<code>\varPi</code>
ψ	<code>\psi</code>	φ	<code>\varphi</code>	Φ	<code>\Phi</code>	\varSigma	<code>\varSigma</code>
ω	<code>\omega</code>	Γ	<code>\Gamma</code>	Ψ	<code>\Psi</code>	\varUpsilon	<code>\varUpsilon</code>
\digamma	<code>\digamma</code>	Δ	<code>\Delta</code>	Ω	<code>\Omega</code>	\varPhi	<code>\varPhi</code>
ε	<code>\varepsilon</code>	Θ	<code>\Theta</code>	\varGamma	<code>\varGamma</code>	\varPsi	<code>\varPsi</code>
ϑ	<code>\vartheta</code>	Λ	<code>\Lambda</code>	\varDelta	<code>\varDelta</code>	\varOmega	<code>\varOmega</code>
\varkappa	<code>\varkappa</code>	Ξ	<code>\Xi</code>	\varTheta	<code>\varTheta</code>		
ϖ	<code>\varpi</code>	Π	<code>\Pi</code>	\varLambda	<code>\varLambda</code>		
ϱ	<code>\varrho</code>	Σ	<code>\Sigma</code>	\varXi	<code>\varXi</code>		

Az `upgreek` csomaggal a következő alakok is elérhetők:

α	<code>\upalpha</code>	λ	<code>\uplambda</code>	χ	<code>\upchi</code>	Δ	<code>\Updelta</code>
β	<code>\upbeta</code>	μ	<code>\upmu</code>	ψ	<code>\uppsi</code>	Θ	<code>\Uptheta</code>
γ	<code>\upgamma</code>	ν	<code>\upnu</code>	ω	<code>\upomega</code>	Λ	<code>\Uplambda</code>
δ	<code>\updelta</code>	ξ	<code>\upxi</code>	ε	<code>\upvarepsilon</code>	Ξ	<code>\Upxi</code>
ϵ	<code>\upepsilon</code>	π	<code>\uppi</code>	ϑ	<code>\upvartheta</code>	Π	<code>\Uppi</code>
ζ	<code>\upzeta</code>	ρ	<code>\uprho</code>	ϖ	<code>\upvarpi</code>	Σ	<code>\Upsigma</code>
η	<code>\upeta</code>	σ	<code>\upsigma</code>	ϱ	<code>\upvarrho</code>	Υ	<code>\Upupsilon</code>
θ	<code>\uptheta</code>	τ	<code>\uptau</code>	σ	<code>\upvarsigma</code>	Φ	<code>\Upphi</code>
ι	<code>\upiota</code>	υ	<code>\upupsilon</code>	φ	<code>\upvarphi</code>	Ψ	<code>\Uppsi</code>
κ	<code>\upkappa</code>	ϕ	<code>\upphi</code>	Γ	<code>\Upgamma</code>	Ω	<code>\Upomega</code>

Ha az `upgreek` csomagot `Symbol` opcióval tölti be, akkor az előbbi parancsok Adobe Symbol fontokat használnak.

15.5. 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>	$\ddot{\dot{a}}$	<code>\ddot{\dot{a}}</code>	\mathring{a}	<code>\mathring{a}</code>
\bar{a}	<code>\bar{a}</code>	\breve{a}	<code>\breve{a}</code>	$\ddot{\dot{a}}$	<code>\ddot{\dot{a}}</code>		
\vec{a}	<code>\vec{a}</code>	\check{a}	<code>\check{a}</code>				

Ha az i és j jelekre akar matematikai módban ékezetet tenni, akkor ne a korábban ismertetett `\i` és `\j`, hanem az

```
\imath  
\jmath
```

parancsokat használja. Például

```
$\check{\imath}\check{\jmath}$
```

15.6. Közönséges matematikai jelek

$\%$	<code>\%</code>	\neg	<code>\neg</code>	\nexists	<code>\nexists</code>	∇	<code>\nabla</code>
\bot	<code>\bot</code>	\forall	<code>\forall</code>	\Re	<code>\Re</code>	∂	<code>\partial</code>
\top	<code>\top</code>	\exists	<code>\exists</code>	\Im	<code>\Im</code>	\eth	<code>\eth</code>

\emptyset	<code>\emptyset</code>	\angle	<code>\angle</code>	\natural	<code>\natural</code>	$ $	<code> </code> vagy <code>\vert</code>
∞	<code>\infty</code>	\measuredangle	<code>\measuredangle</code>	$\#$	<code>\#</code>	\parallel	<code>\parallel</code> vagy <code>\Vert</code>
\triangle	<code>\triangle</code>	\sphericalangle	<code>\sphericalangle</code>	1°	<code>1^\circ</code>	$1'$	<code>1'</code> (1° \prime)
\square	<code>\square</code>	\flat	<code>\flat</code>	$1''$	<code>1''</code>		
\blacksquare	<code>\blacksquare</code>	\sharp	<code>\sharp</code>				

15.7. Műveleti jelek

$+$	<code>+</code>	\times	<code>\times</code>	\vee	<code>\vee</code>	\ominus	<code>\ominus</code>
$-$	<code>-</code>	\div	<code>\div</code>	\star	<code>\star</code>	\odot	<code>\odot</code>
$/$	<code>/</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 elő és után is térköz kell. Ezt a L^AT_EX 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

`$a\mathbin{\dag}ab$`

`a † ab`

A magyarban tipográfiai szabály, hogy a $+$ vagy $-$ jel a sor végére kerülve a következő sor elején megismétlődjön. Ezt a `magyar.lfd` automatikusan megoldja. Ha olyan műveleti jelre is szeretné ezt a hatást elérni, amit a `magyar.lfd` nem kezel, akkor az adott pontra a `<műveleti jel>` helyett a következő parancsot írja:

`\MathBrk{<műveleti jel>} ∈ [magyar]babel`

15.8. Három pont

$1, \dots, n$	<code>1,\ldots,n</code>	$\int \cdots \int$	<code>\int\ldots\int</code>
$1 + \dots + n$	<code>1+\cdots+n</code>	\vdots	<code>\vdots</code>
$1 \cdots n$	<code>1\cdots n</code>	\ddots	<code>\ddots</code>

Az `\ldots` az alapvonalra, míg a `\cdots` illetve integrálok esetén a `\dotsi` függőlegesen középre igazítja a három pontot. Több esetben ez automatizálható a

`\dots`

parancssal. Például

$\$1,\dots,n \quad 1 + \dots + n \quad \int \cdots \int$

ugyanazt az eredményt adják, de

$\$1\dots n\$$ (rossz!)

1 ... n (rossz!)

esetén nem jó helyen lesznek a pontok.

15.9. Relációjelek

= =	::≈ \Colonapprox	≈ \cong	» \gg
:= \coloneqq	≈ \colonsim	< <	∈ \in
::= \Coloneqq	::≈ \Colonsim	> >	∋ \ni
:= \coloneq	:: \dblcolon	≤ \leq	⊂ \subset
::= \Coloneq	: : (arányjel)	≥ \geq	⊃ \supset
=: \eqqcolon	÷ \doteq	≤\leq \leqq	⊆ \subsetreq
=:: \Eqqcolon	≡ \equiv	≥\geq \geqq	⊇ \supsetreq
=: \eqcolon	~ \sim	≤\leqslant \leqslant	\mid
=:: \Eqcolon	≈ \simeq	≥\geqslant \geqslant	\parallel
≈ \colonapprox	≈ \approx	≤\ll \ll	⊥ \perp

Ekvivalencia reláció esetén még a modulus jelölése is kell:

$\$a \bmod m\$\\$
 $\$a \equiv b \pmod{m}\$\\$
 $\$a \equiv b \mod{m}\$\\$
 $\$a \equiv b \pod{m}\$$

$a \bmod m$
 $a \equiv b \pmod{m}$
 $a \equiv b \mod{m}$
 $a \equiv b \pod{m}$

A nyilak is a relációjelek közé tartoznak:

← \leftarrow	↔ \Leftrightarrow	↔ \updownarrow	\updownarrow \updownarrowarrow
⟵ \longleftarrow	⟺ \Longleftrightarrow	↑ \Uparrow	\Uparrow \Uparrow
→ \rightarrow	↪ \mapsto	↓ \Downarrow	\Downarrow \Downarrow
⟹ \longrightarrow	↦ \longmapsto	↔ \Updownarrow	\Updownarrow \Updownarrow
⤠ \leftrightarrow	⤠ \leftharpoonup	↗ \nearrow	\nearrow \nearrow
⤠ \longleftrightarrow	⤠ \leftharpoondown	↘ \searrow	\searrow \searrow
⤠ \Leftarrow	⤠ \rightharpoonup	↖ \swarrow	\swarrow \swarrow
⤠ \Longleftarrow	⤠ \rightharpoondown	↙ \nwarrow	\nwarrow \nwarrow
⤠ \Rightarrow	↑ \uparrow		
⤠ \Longrightarrow	↓ \downarrow		

A magyarban tipográfiai szabály, hogy az = jel a sor végére kerülve a következő sor elején megismétlődjön. Ezt a `magyar.1df` automatikusan megoldja. Ha olyan relációjelre

is szeretné ezt a hatást elérni, amit a `magyar.ldf` nem kezel, akkor az adott pontra a `\langle relációjel \rangle` helyett a

```
\MathBrk{\langle relációjel \rangle} \in [magyar] babel
```

parancsot írja. Relációjeleket negálni (áthúzni) a

```
\not
```

parancssal lehet. Például

```
$a\not=b$
```

 $a \neq b$

Néhány esetben ez nem ad megfelelő eredményt:

```
$\not\mid \quad \not\parallel \quad \not\downarrow \quad \not\uparrow$
```

 $\not\mid \not\parallel \not\downarrow \not\uparrow$

Ezek helyett külön tervezésű negált reláció jelet kell használni:

```
$\nmid \quad \nparallel \quad \ndownarrow \quad \nuparrow$
```

 $\nmid \nparallel \nmid \nparallel$

A relációjelek körüli térközökre ugyanaz a szabály, mint a műveleti jelekre:

```
$a=b$
```

 $a = b$

A L^AT_EX bármit relációjelnek tekint, amit a

```
\mathrel{\langle karakterek \rangle}
```

parancsba írunk. Például

```
$a|b\$ \\ $a\mathrel{|}b$
```

 $a|b$
 $a \mid b$

Jelek egymásra helyezésével is készíthet relációjelet a

```
\stackrel{\langle felül \rangle}{\langle alul \rangle}
```

parancssal. Például

```
$A\stackrel{f}{\longrightarrow}B$
```

 $A \xrightarrow{f} B$

15.10. Matematikai zárójelek

Bal oldali (nyitó) zárójelek

((\langle	$\backslash langle$	\lceil	$\backslash ulcorner$	\parallel	$\backslash lVert$
[[vagy $\backslash lbrack$	\lceil	$\backslash lceil$	\lfloor	$\backslash llcorner$		
{	{ vagy $\backslash lbrace$	\lfloor	$\backslash lfloor$	\lvert	$\backslash lvert$		

Jobb oldali (csukó) zárójelek

))	\rangle	\rangle	\urangle	\urcorner	\urVert
]] vagy \rbrack]	\rceil	\lrcorner	\lrcorner	\lVert
}	\} vagy \rbrace]	\rfloor	\rvert	\rvert	

Például

```
$\lvert -1 \rvert$
```

$|-1|$

Közönséges matematikai jeleket ne használjon zárójelként. Például

```
$|-1|$
```

$| -1 |$

rossz eredményt ad, mert a program ezt úgy értelmezi, hogy a $|$ jelből kivonjuk az 1-et, így a $-$ jel körül térközöket hagy.

Amennyiben egy jelet nyitó zárójelként akar értelmezni, akkor tegye elé a

`\mathopen`

parancsot. Például a `\mathopen|` ekvivalens az `\lvert` parancssal.

Amennyiben egy jelet csukó zárójelként akar értelmezni, akkor tegye elé a

`\mathclose`

parancsot. Például a `\mathclose|` ekvivalens az `\rvert` parancssal.

A `\mathopen` és `\mathclose` parancsok használatára tekintsük a következő esetet:

```
$]-1, 1[\setminus\{0\}$
```

$]-1, 1[\setminus\{0\}$

Az eredmény rossz, hiszen a $]$ jel csukó zárójelként van értelmezve, így az utána található $-$ jelet kivonásként értelmezi, másrészt a $[$ nyitó zárójelként van értelmezve, így az utána található \backslash jelet nem tekinti relációknak. A megoldás az, hogy ideiglenesen a $]$ jelet nyitó, a $[$ jelet pedig csukó zárójelként értelmezzük:

```
$\mathopen]-1, 1[\mathclose\setminus\{0\}$
```

$]-1, 1[\setminus\{0\}$

Természetesen, ha egy jel nyitó zárójelként van értelmezve és annak is használja, illetve, ha egy jel csukó zárójelként van értelmezve és annak is használja, akkor az előbbi megoldásra nincs szükség. Például a következő kód helyes eredményt ad:

```
$[-1, 1]\setminus\{0\}$
```

$[-1, 1] \setminus \{0\}$

Az előbbi zárójelek mérete nem igazodik a képlethez. Például

```
\left[f\left(\frac{1}{2}\right)=0\right]
```

$$f\left(\frac{1}{2}\right) = 0$$

Ilyen esetben a nyitó zárójel elő tegye a

`\left`

míg a csukó zárójel elő

`\right`

parancsot. Például

```
\[f\left(\frac{1}{2}\right)=0\]
```

$$f\left(\frac{1}{2}\right) = 0$$

A `\left` és `\right` parancsok használatával már azt is megadjuk, hogy melyik a nyitó és melyik a csukó zárójel, így ezekkel nem szabad együtt használni a `\mathopen` és `\mathclose` parancsokat. Tehát például a következő kód helyes eredményt ad:

```
\[\left|-\frac{1}{2}\right|
```

$$\left|-\frac{1}{2}\right|$$

A `\left` és `\right` parancsok nemcsak a méretre hatnak, hanem a zárójeleket övező térközöket is megnövelik. Ha ezt nem szeretné, akkor a `\left` és `\right` helyett használja az

`\mleft \mleftright`

`\mright \mleftright`

parancsokat. Például

```
\[f\mleft(\frac{1}{2}\mright)=0\]
```

$$f\left(\frac{1}{2}\right) = 0$$

Amennyiben ugyanezt a hatást szeretné elérni a `\left` és `\right` parancsok esetében is, akkor adja ki a preambulumban a

`\mleftright \mleftright`

parancsot.

Ha egyetlen zárójelre van szükség, melynek méretben igazodni kell a képlethez, míg a zárójel páját 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 a

`\left. \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}, 1, 2, \ldots, \right.\$\\
\$ \left.\left.n-1, n\right)\$
```

$$\dots \left(\frac{1}{1+\frac{1}{2}}, 1, 2, \dots, n-1, n \right)$$

Megoldás

```
\dotfill$\left(\frac{1+\frac{1}{2}}{1}, 2, \ldots, \right.$\\
$\left.n-1, n\,\vphantom{\frac{1+\frac{1}{2}}{1}}\right)$
```

$$\dots \left(n-1, n \right) \left(\frac{1}{1+\frac{1}{2}}, 1, 2, \dots \right)$$

Néhány esetben nem ad megfelelő eredményt a zárójelek automatikus méretezése. Például

$$\left\{ \{a,b\}, \{c,d\} \right\}$$

$$\{\{a,b\}, \{c,d\}\}$$

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:

$$\$\\Biggl(\\biggl(\\Bigl(\\bigl(\\bigl(\\cdots\\bigr)\\Bigr)\\biggr)\\Biggr)\\Biggr)$$

$$\left(\left(\left(\left((\cdot) \right) \right) \right) \right)$$

Például az előző képlet a következő módon oldható meg helyesen:

$\$\\bigl\\{\\{a,b\\},\\{c,d\\}\\bigr\\} \$$

$$\left\{ \{a,b\}, \{c,d\} \right\}$$

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

\bigm \Bigm \biggm \Biggm

parancsokat. Például

```

$$\left\{ \frac{2^n}{n^2} \mid n \text{ egész} \right\}$$

```

Az előbbi rögzített méretű zárójelek a következő séma szerint vannak definiálva:

```
\makeatletter
\def\langle#1\rangle{\bBigg@{#1}}
\def\langle#1\rangle_m{\mathrel{\langle#1\rangle}}
\def\langle#1\rangle_l{\mathopen{\langle#1\rangle}}
\def\langle#1\rangle_r{\mathclose{\langle#1\rangle}}
\makeatother
```

ahol

$\langle#1\rangle$	big	Big	bigg	Bigg
$\langle#1\rangle_m$	1	1.5	2	2.5

A 3 illetve 3.5 méretek már nincsenek definiálva, de ezt megteheti a következő kóddal:

```
\makeatletter
\def\biggg{\bBigg@{3}}
\def\bigggm{\mathrel{\biggg}}
\def\bigggl{\mathopen{\biggg}}
\def\bigggr{\mathclose{\biggg}}
\def\Biggg{\bBigg@{3.5}}
\def\Bigggm{\mathrel{\Biggg}}
\def\Bigggl{\mathopen{\Biggg}}
\def\Bigggr{\mathclose{\Biggg}}
\makeatother
```

Ezután a `\biggg`, `\bigggm`, `\bigggl`, `\bigggr`, `\Biggg`, `\Bigggm`, `\Bigggl`, `\Bigggr` parancsok hasonlóan használhatók, ahogyan azt a korábbiakban ismertettük.

Definiálhat olyan paraméteres parancsot, mely a paramétert a kiválasztott zárójelbe teszi. Ehhez használja a

```
\DeclarePairedDelimiter{\parancs}{\langle}{\rangle}
```

parancsot. Például

```
\DeclarePairedDelimiter{\abs}{\lvert}{\rvert}
\abs{\frac{a}{b}} \quad
\abs*{\frac{a}{b}} \quad
\abs[\Bigg]{\frac{a}{b}}
```

$$\left| \frac{a}{b} \right| \quad \left| \frac{a}{b} \right| \quad \left| \frac{a}{b} \right|$$

15.11. 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
\	\backslash	\setminus		
:			:	\colon
			\mid	\left \right
	\		\parallel	\left\ \right\
<			<	\left\langle \right.
>			>	\left. \right\rangle
\bot	\bot		\perp	
\dag	\dag	\dagger		
\ddag	\ddag	\ddagger		

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` opciójával töltötte be, akkor a vessző matematikai üzemmódban két szám között tizedesvesszőként értelmezett, de egyéb esetben megmarad az eredeti szerepe. Például

```
$2,5\cdot 2=5\$\\
$a,b,c$
```

$2,5 \cdot 2 = 5$
 a, b, c

Ha két szám között a vesszőt nem tizedesvesszőként használja, akkor a vessző után tegyen egy szóközt:

```
$1, 2, 3,\dots$
```

$1, 2, 3, \dots$

Ha nem a magyar nyelv van beállítva, akkor a vesszőnek nincs kettős szerepköre. Ha magyar nyelv esetén sem akarja ezt a kettős szerepkört, akkor töltse még be a `mathhucomma=unchanged` opciót is a `defaults=hu-min` után. Ekkor tizedesvesszőt így kell írni:

```
$2{,}5\cdot 2=5$
```

$2,5 \cdot 2 = 5$

15.12. Esetek szétválasztása

A korábbiakban szóba került az esetek szétválasztása, amikor egy zárójel nem zárójel-ként funkcionál. Erre többek között a `cases` környezet használható. Például

</>

```
\[f(x)=
\begin{cases}
0, & \text{ha } x \in \mathbb{Q}, \\
1, & \text{különben.}
\end{cases}\]
```

$$f(x) = \begin{cases} 0, & \text{ha } x \in \mathbb{Q}, \\ 1, & \text{különben.} \end{cases}$$

A `cases` környezet úgy működik, mint egy két oszlopból álló táblázat, melynek minden két oszlopában matematikai üzemmódban vagyunk `\textstyle` stílusban. Ezért volt szükség a második oszlopban a szöveget `\text` parancsba írni. További környezetek:

`dcases` A `cases`-től annyiban különbözik, hogy minden két oszlopban matematikai üzemmódban vagyunk `\displaystyle` stílusban.

`rcases` A `cases`-től annyiban különbözik, hogy a kapcsos zárójel jobb oldalon van.

`drcases` A `dcases`-től annyiban különbözik, hogy a kapcsos zárójel jobb oldalon van.

Minden esetszétválasztó környezetnek van csillagos verziója is, amely annyiban különbözik a normál verziótól, hogy a második oszlop szöveg üzemmódban van. Például

</>

```
\[f(x)=
\begin{cases*}
0, & \text{ha } x \text{ racionális,} \\
1, & \text{ha } x \text{ irracionális.}
\end{cases*}\]
```

$$f(x) = \begin{cases} 0, & \text{ha } x \text{ racionális,} \\ 1, & \text{ha } x \text{ irracionális.} \end{cases}$$

15.13. Változó hosszúságú vízszintes jelek

\widehat{xyz}	<code>\widehat{xyz}</code>	\overleftarrow{xyz}	<code>\underleftarrow{xyz}</code>
\widetilde{xyz}	<code>\widetilde{xyz}</code>	\overrightarrow{xyz}	<code>\overrightarrow{xyz}</code>
\overline{xyz}	<code>\overline{xyz}</code>	$\overleftarrow{\overrightarrow{xyz}}$	<code>\underrightarrow{xyz}</code>
\underline{xyz}	<code>\underline{xyz}</code>	$\overleftarrow{\overrightarrow{xyz}}$	<code>\overleftarrow{\overrightarrow{xyz}}</code>
\overleftarrow{xyz}	<code>\overleftarrow{xyz}</code>	$\overleftarrow{\overleftarrow{xyz}}$	<code>\underleftarrow{\overleftarrow{xyz}}</code>
$\overleftarrow{\overleftarrow{xyz}}$			
$\xleftarrow[fent]{len}{t}$	<code>\xleftarrow[fent]{len}{t}</code>	$\xLeftarrow[fent]{len}{t}$	<code>\xLeftarrow[fent]{len}{t}</code>
$\xrightarrow[fent]{len}{t}$	<code>\xrightarrow[fent]{len}{t}</code>	$\xrightarrow[fent]{len}{t}$	<code>\xhookleftarrow[fent]{len}{t}</code>
$\xleftrightsquigarrow[fent]{len}{t}$	<code>\xleftrightsquigarrow[fent]{len}{t}</code>	$\xleftrightsquigarrow[fent]{len}{t}$	<code>\xhookrightarrow[fent]{len}{t}</code>
$\xLeftarrow[fent]{len}{t}$	<code>\xLeftarrow[fent]{len}{t}</code>	$\xLeftarrow[fent]{len}{t}$	<code>\xmapsto[fent]{len}{t}</code>
$\xRightarrow[fent]{len}{t}$	<code>\xRightarrow[fent]{len}{t}</code>	$\xRightarrow[fent]{len}{t}$	<code>\xleftharpoondown[fent]{len}{t}</code>

$\frac{fent}{lent}$	<code>\xleftharpoonup[lent]{fent}</code>	$\frac{fent}{lent}$	<code>\xrightleftharpoons[lent]{fent}</code>
$\frac{fent}{lent}$	<code>\xightharpoondown[lent]{fent}</code>	$\frac{fent}{lent}$	<code>\xleftrightharpoons[lent]{fent}</code>
$\frac{fent}{lent}$	<code>\xightharpoonup[lent]{fent}</code>		
\overbrace{xxxxxx}^n	<code>\underbrace{xxxxxx}</code>		
\overbrace{xxxxxx}^n	<code>\underbrace{xxxxxx}_{n}</code>		
\overbrace{xxxxxx}^n	<code>\underbracket{xxxxxx}</code>		
\overbrace{xxxxxx}^n	<code>\underbracket{xxxxxx}_{n}</code>		
\overbrace{xxxxxx}^n	<code>\underbracket[0.4pt][7pt]{xxxxxx}</code>		
\overbrace{xxxxxx}^n	<code>\underbracket[0.4pt][7pt]{xxxxxx}_{n}</code>		
\overbrace{xxxxxx}^n	<code>\overbrace{xxxxxx}</code>		
\overbrace{xxxxxx}^n	<code>\overbrace{xxxxxx}^n</code>		
\overbrace{xxxxxx}^n	<code>\overbracket{xxxxxx}</code>		
\overbrace{xxxxxx}^n	<code>\overbracket{xxxxxx}^n</code>		
\overbrace{xxxxxx}^n	<code>\overbracket[0.4pt][7pt]{xxxxxx}</code>		
\overbrace{xxxxxx}^n	<code>\overbracket[0.4pt][7pt]{xxxxxx}^n</code>		

15.14. Gyökvonás

Az $\langle n \rangle$ -edik gyök $\langle x \rangle$ kiírása:

```
\sqrt[n]{\langle x \rangle}
```

Az opció elhagyásával négyzetgyököt kapunk. Például

```
$\sqrt{2}\sqrt[3]{5}$
```

Lehetőség van az $\langle n \rangle$ igazítására is:

```
\uproot{\langle fel \rangle}
\leftroot{\langle balra \rangle}
```

$\langle fel \rangle$ egész szám, hatására n felcsúszik $\frac{\langle fel \rangle}{18}$ em-mel.

$\langle balra \rangle$ egész szám, hatására n balra csúszik $\frac{\langle balra \rangle}{18}$ 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}}\\$$

$\sqrt{x} + \sqrt{y}$

A `\smash[b]` parancs az y mélységét 0-nak veszi, így a két gyökjel mérete egyforma lesz. Ugyanennek a problémának egy másik lehetséges megoldása:

$\$\\sqrt{x\\mathstrut}+\\sqrt{y\\mathstrut}\\$$

$\sqrt{x} + \sqrt{y}$

A `\mathstrut` olyan 0 pt szélességű doboz (gyámfá), melynek a magassága és mélysége megegyezik a zárójel magasságával és mélységével.

Amennyiben a kézíráshoz hasonlóan a gyökjelet lezárt véggel, azaz $\sqrt{}$ alakban szeretné használni, akkor írja a preambulumba a `mathtools` csomag betöltése után a következőket:

```
\usepackage{letltxmacro}
\makeatletter
\let\oldr@t\@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
```

15.15. Mátrixok

```
\begin{mátrix környezet}
elem & elem & elem & ... \\
elem & elem & elem & ... \\
...
elem & elem & elem & ... \\
elem & elem & elem & ...
\end{mátrix környezet}
```

ahol a *\langle mátrix környezet \rangle* lehetséges értékei:

- matrix** Határoló zárójel nincs.
- pmatrix** A határoló zárójel () alakú.
- bmatrix** A határoló zárójel [] alakú.
- Bmatrix** A határoló zárójel { } alakú.
- vmatrix** A határoló zárójel | | alakú.

Vmatrix A határoló zárójel $\| \ |$ alakú.

Például

```
</> \begin{pmatrix} a & b \\ c & d \end{pmatrix}
\begin{bmatrix} a & b \\ c & d \end{bmatrix}
\begin{vmatrix} a & b \\ c & d \end{vmatrix}\]
```

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

Ezen mátrixok oszlopai középre vannak igazítva, továbbá az oszlopok maximális száma 10, amit a következő parancssal lehet átállítani például 20-ra:

```
\setcounter{MaxMatrixCols}{20}
```

Szöveg közben az $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ helyett szébb az $(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix})$ mátrix. Ehhez minden előző mátrix környezetnek van egy **small** verziója:

smallmatrix Határoló zárójel nincs.

psmallmatrix A határoló zárójel () alakú.

bsmallmatrix A határoló zárójel [] alakú.

Bsmallmatrix A határoló zárójel { } alakú.

vsmallmatrix A határoló zárójel || alakú.

Vsmallmatrix A határoló zárójel ||| alakú.

Például

```
$\begin{psmallmatrix} a & b \\ c & d \end{psmallmatrix}$
```

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Az eddigi mátrix környezetnek van csillagos verziója is, amely esetén opcióban lehet megadni az oszlopok igazítását jobbra (**r**) vagy balra (**l**). Például

```
</> \begin{bmatrix}[r] a & b \\ c & -d \end{bmatrix}
\begin{vmatrix}[l] 1.2 & 1 \\ 2 & 2.23 \end{vmatrix}\]
```

$$\begin{bmatrix}[r] -a & b \\ c & -d \end{bmatrix} \begin{vmatrix}[l] 1.2 & 1 \\ 2 & 2.23 \end{vmatrix}$$

Három pont helyett hosszabb pontsorozatok is kiírhatók

```
\hdotsfor[<sűrűség>]{<oszlopok>}
```

parancssal, ahol **<sűrűség>** a pontsor sűrűsége (alapérték 1) és **<oszlopok>** a keresztezett oszlopok száma. Például

```
</> \begin{pmatrix}
1 & 2 & 3 & \hdotsfor{2} & 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 & \dots & n \\ 0 & 1 & 2 & \dots & n-1 \\ \hdotsfor[0.5]{5} \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}
```

$$\begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 0 & 1 & 2 & \dots & n-1 \\ \dots & & & & \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

A `\hdotsfor` parancs nem ad jó eredményt, ha a `colortbl` csomagot is használja. Ebben az esetben a `mathtools` csomag betöltése után a preambulumba írja a következőket:

```
\usepackage{etoolbox}
\makeatletter
\patchcmd{\hdotsfor}{\hfill}{\hskip\z@\@plus 1filll}\relax
\makeatother
```

15.16. Matematikai jelek egymásra helyezése

Ezekre már láttunk korábban lehetőségeket a változó hosszúságú vízszintes jeleknél és a relációjeleknél. Az ott leírt parancsok mindegyike reláció típushoz eredményez. Most két másik parancsot ismertetünk:

```
\overset{\langle amit \rangle}{\underset{\langle amire \rangle}{\langle amit \rangle}}
```

Mindkettő típusa meg fog egyezni a második argumentumba írt jel típusával. Például

```
$a\overset{*}{+}b\underset{\mathrm{d}}{=}c$
```

$$a \overset{*}{+} b = c$$

Egy másik összetettebb példa:

```
</> \begin{aligned} & P(A) + P(\overline{A}) \\ & \underset{\Omega}{\underbrace{\underbrace{\mathclap{\text{additivitás}}}_{\Updownarrow}}}_{\Downarrow} = \\ & P(\Omega) \\ & \overset{5. \text{ axióma}}{\overbrace{\mathclap{5. \text{ axióma}}}_{\Downarrow}} = \\ & 1 \\ & \Longrightarrow P(\overline{A}) = 1 - P(A) \end{aligned}
```

$$P(A) + P(\overline{A}) = P(\Omega) \stackrel{\substack{\Downarrow \\ \text{additivitás}}}{=} 1 \stackrel{\substack{\Downarrow \\ 5. \text{ axióma}}}{=} P(\overline{A}) = 1 - P(A)$$

A `\mathclap` leírását lásd a 13.8. szakaszban.

Ha közönséges matematikai jelet szeretne építeni `\scriptstyle` stílusban, akár ket-tónél több sorban, akkor használja a

```
\substack{\langle sor1 \rangle \\ \langle sor2 \rangle \\ ...}
```

parancsot, vagy a

```
\begin{subarray}{(igazítás)}
<sor1>\<sor2>\<...>
\end{subarray}
```

környezetet. Az *(igazítás)* lehet **c** (középre) **l** (balra) és **r** (jobbra). Például

</> $\sum_{\substack{i=1, \dots \\ j \in \mathbb{Z} \\ k=j, \dots}} a_{ijk}$

$$\sum_{\substack{i=1, \dots \\ j \in \mathbb{Z} \\ k=j, \dots}} a_{ijk}$$

</> $\sum_{\begin{array}{l} i=1, \dots \\ j \in \mathbb{Z} \\ k=j, \dots \end{array}} a_{ijk}$

$$\sum_{\substack{i=1, \dots \\ j \in \mathbb{Z} \\ k=j, \dots}} a_{ijk}$$

15.17. Matematikai indexek

```
jel_{<alsó>} vagy jel\sb{<alsó>}
jel^{<felső>} vagy jel\sp{<felső>}
jel_{<alsó>}^{<felső>} vagy jel\sb{<alsó>}\sp{<felső>}
```

Például

\$x_{n+1}, x^{n+1}, x_k^{n+1}\$

$$x_{n+1}, x^{n+1}, x_k^{n+1}$$

A felső indexek lejjebb kerülnek a **\cramped** parancs használatával. Például

\$\cramped{{x^2}^x} {x^2}^x\$

$$x^{2x} x^{2x}$$

A később tárgyalt ún. operátorok minden a négy sarkába, vagy alá és fölé is lehet indexet.

```
\sideset{_<bal alsó>}^{<bal felső>}{_<jobb alsó>}^{<jobb felső>}{<operátor>}
<operátor>\limits_{<alul>}^{<felül>}
```

Például

\$\sideset{_a^b}{_c^d}{\prod} \text{ és } \prod_a^b \prod_c^d

$$\prod_a^b \prod_c^d \text{ és } \prod_1^2$$

Ha ugyanezt nem operátorral szeretné csinálni, akkor az indexelendő jelet átmenetileg operátorrá kell tenni a

`\mathop{⟨jel⟩}`

Például

\$\sideset{_a^b}{_c^d}\mathop{X}\$ és \$\mathop{X}\limits_{1}^{2}\$

$${}_a^b X {}_c^d \text{ és } X {}_1^2$$

Bal alsó és felső index a következő módon is írható:

`\prescript{⟨bal felső⟩}{⟨bal alsó⟩}{⟨jel⟩}`

Például

\$\prescript{14}{2}{C}\$

15.18. Törtek, binomiális együtthatók

`\frac{⟨számláló⟩}{⟨nevező⟩}`
`\dfrac{⟨számláló⟩}{⟨nevező⟩} = {\displaystyle \frac{⟨számláló⟩}{⟨nevező⟩}}`
`\tfrac{⟨számláló⟩}{⟨nevező⟩} = {\textstyle \frac{⟨számláló⟩}{⟨nevező⟩}}`

Például

\$\frac{x^2}{x+1}\$

`\binom{⟨fent⟩}{⟨lent⟩}`
`\dbinom{⟨fent⟩}{⟨lent⟩} = {\displaystyle \binom{⟨fent⟩}{⟨lent⟩}}`
`\tbinom{⟨fent⟩}{⟨lent⟩} = {\textstyle \binom{⟨fent⟩}{⟨lent⟩}}`

Például

\$\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

\$\left[\genfrac{}{}{1pt}{}{n+1}{m+1}\right]\genfrac{}{}{0pt}{}{1}{\binom{n+1}{m+1}}

$$\left\{ \frac{n+1}{m+1} \right\}_{m+1}^{n+1}$$

Lánctörtek a következő paranccsal írhatók:

```
\cfrac{<igazítás>}{<számláló>}{<nevező>}
```

ahol az *<igazítás>* lehet **l** (balra), **r** (jobbra) és **c** (középre, alapérték). Például

```
\[\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}\]
```

$$\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}$$

```
\[\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}\]
```

$$\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}$$

Ha egy tört számlálója vagy nevezője túl hosszú, akkor két sorba lehet törni a

```
\splitfrac{<1. sor>}{<2. sor>}
```

paranccsal. Például

```
\[a=\frac{\splitfrac{xy+xy+xy+xy+xy+}{xy+xy+xy+xy}}{z}\]
```

$$a = \frac{xy + xy + xy + xy + xy +}{z}$$

A *\splittfrac* hasonló a *\splitfrac*-hoz, de a két sor közötti távolság nagyobb.

15.19. Operátorok, függvények

15.19.1. Nagy operátorok

\sum	<code>\sum</code>	\int	<code>\int</code>	\oint	<code>\oint</code>	\bigcup	<code>\biguplus</code>
\prod	<code>\prod</code>	\iint	<code>\iint</code>	\iiint	<code>\iiint</code>	\times	<code>\bigtimes</code>
\coprod	<code>\coprod</code>	\bigcap	<code>\bigcap</code>	\bigwedge	<code>\bigwedge</code>		
\bigoplus	<code>\bigoplus</code>	\bigcup	<code>\bigcup</code>	\bigvee	<code>\bigvee</code>		
\bigsqcup	<code>\bigsqcup</code>	\odot	<code>\odot</code>	\otimes	<code>\otimes</code>		

Magyar nyelvű dokumentumokban az integrál jelét nem dőlten, hanem álló alakban szokás írni. Ehhez European Computer Modern vagy Latin Modern fontok használata

esetén töltse be a `cmupoint` csomagot. Vannak olyan fontváltó csomagok, amelyekben ez automatikusan vagy opcionálisan megvalósul. Ilyenek például a `pxfonts`, `newtxmath`, `newpxmath` csomagok.

A `\bigtimes` operátornak van egy nagyobb verziója is:

~~\times~~ `\varprod` \in `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}{pxex}{m}{n}
\DeclareMathSymbol{\varprod}{\mathop}{largesymbolsA}{16}
```

A nagy operátorok más méretben jelennek meg szövegközi illetve kiemelt matematikai módban. Például

`\sum` $\backslash[\sum]$

\sum

15.19.2. „Nolimits” függvények

<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>	<code>\lim</code>	<code>\varliminf</code>
<code>arctan</code>	<code>\arctan</code>	<code>deg</code>	<code>\deg</code>	<code>log</code>	<code>\log</code>	<code>\lim</code>	<code>\varlimsup</code>
<code>arg</code>	<code>\arg</code>	<code>dim</code>	<code>\dim</code>	<code>sec</code>	<code>\sec</code>	<code>\lim_{\rightarrow}</code>	<code>\varinjlim</code>
<code>cos</code>	<code>\cos</code>	<code>exp</code>	<code>\exp</code>	<code>sin</code>	<code>\sin</code>	<code>\lim_{\leftarrow}</code>	<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>		

A „nolimits” függvények indexei mindig a függvény neve mellett jelennek meg. Például

`$\log_2 x$` $\backslash[\log_2 x]$

$\log_2 x$

`\int_a^b`

\int_a^b

A L^AT_EX bármit „nolimits” függvénynek tekint, amit az

`\operatorname{<karakterek>}`

parancsba írunk. Például

`$\operatorname{tg}(x)$`

$\operatorname{tg}(x)$

15.19.3. „Limits” függvények

det	<code>\det</code>	inj lim	<code>\injlim</code>	lim sup	<code>\limsup</code>	proj lim	<code>\projlim</code>
gcd	<code>\gcd</code>	lim	<code>\lim</code>	max	<code>\max</code>	Pr	<code>\Pr</code>
inf	<code>\inf</code>	lim inf	<code>\liminf</code>	min	<code>\min</code>	sup	<code>\sup</code>

A nagy operátorok (az integráljel kivételével) és a „limits” függvények indexei szövegközi matematikai módban mellette jelennek meg, de kiemelt matematikai módban alatta és fölötté. Például

`$\sum_{n=1}^{\infty} a_n$ $\sum_{n=1}^{\infty} a_n$`

$$\sum_{n=1}^{\infty} a_n$$

$$\sum_{n=1}^{\infty} a_n$$

`$\lim_{n \rightarrow \infty} a_n$ $\lim_{n \rightarrow \infty} a_n$`

$$\lim_{n \rightarrow \infty} a_n$$

$$\lim_{n \rightarrow \infty} a_n$$

Ha ezen egy adott helyen változtatni akar, akkor a `\limits` és `\nolimits` parancsokkal teheti meg. Például

`$\sum\limits_{n=1}^{\infty} a_n$ $\sum\limits_{n=1}^{\infty} a_n$`

$$\sum_{n=1}^{\infty} a_n$$

$$\sum_{n=1}^{\infty} a_n$$

A `\limits` parancs nincs hatással a „nolimits” függvényekre, kivéve az integráljelet:

`$\log\limits_2 x$ $\log\limits_2 x$`

$$\log_2 x$$

$$\log_2 x$$

`$\int\limits_a^b a^b$ $\int_a^b a^b$`

$$\int_a^b$$

$$\int_a^b$$

A integráljel „limits” függvényé tehető a `mathtools` csomag `intlimits` opciójával. Ezután már az integrál is pontosan úgy viselkedik, mint bármelyik más nagy operátorjel.

A `LATEX` bármit „limits” függvénynek tekint, amit az

`\operatorname*{<karakterek>}`

parancsba írunk. Például

`$\operatorname*{Min}_k$ $\operatorname*{Min}_k$`

Min_k

Min_k

A „limits” függvények körüli térközök hosszú indexek esetén túl naggyá válhatnak.
Például

```
\[x=\sum_{1\leq i\leq j\leq n}V_{ij}\]
```

$$X = \sum_{1 \leq i \leq j \leq n} V_{ij}$$

Erre ad megoldást a `\smashoperator` parancs:

```
\[x=\smashoperator{\sum_{1\leq i\leq j\leq n}}V_{ij}\]
```

$$X = \sum_{1 \leq i \leq j \leq n} V_{ij}$$

Az `r` illetve `l` opcióval csak a jobb illetve bal oldalon szűnik meg a túl nagy térköz.

Például

```
\[x=\smashoperator[r]{\sum_{1\leq i\leq j\leq n}}V_{ij}\quad x=\smashoperator[l]{\sum_{1\leq i\leq j\leq n}}V_{ij}\]
```

$$X = \sum_{1 \leq i \leq j \leq n} V_{ij} \quad X = \sum_{1 \leq i \leq j \leq n} V_{ij}$$

A „limits” függvények a következő esetben sem adnak tökéletes eredményt:

```
\[\limsup_{n\rightarrow\infty}\max_{p\geq n}\]
```

$$\limsup_{n \rightarrow \infty} \max_{p \geq n}$$

A két függvény indexei azért nincsenek egy szintben, mert a függvények dobozának mélysége különböző. Ezen a problémán segít az `\adjustlimits` parancs. Például

```
\[\adjustlimits\limsup_{n\rightarrow\infty}\max_{p\geq n}\]
```

$$\limsup_{n \rightarrow \infty} \max_{p \geq n}$$

15.19.4. 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.

Új „nolimits” függvény definiálása ♦ Ez a következő parancsokkal lehetséges:

```
\newcommand{\parancs}{\mathop{\mathrm{jel}}}\nolimits
```

vagy

```
\DeclareMathOperator{\parancs}{jel} % Ez csak preambleumba írható!
```

Például

```
\newcommand{\tg}{\mathop{\mathrm{tg}}}\nolimits
```

vagy

```
\DeclareMathOperator{\tg}{tg}
```

után

```
$\tg^2x$ \[\tg^2x\]
```

$$\begin{array}{c} \text{tg}^2 x \\ \text{tg}^2 x \end{array}$$

Egy már létező „nolimits” függvény át is definiálható.

```
\renewcommand{\parancs}{\mathop{\mathrm{parancs}}}\nolimits
```

Például

```
\renewcommand{\tan}{\mathop{\mathrm{tan}}}\nolimits
```

után

```
$\tan^2x$ \[\tan^2x\]
```

$$\begin{array}{c} \text{tg}^2 x \\ \text{tg}^2 x \end{array}$$

Új „limits” függvény definiálása ♦ Ez a következő parancsokkal lehetséges:

```
\newcommand{\parancs}{\mathop{\mathrm{parancs}}}
```

vagy

```
\DeclareMathOperator*{\parancs}{\mathrm{parancs}} % Ez csak preambleumban írható!
```

Például

```
\newcommand{\Min}{\mathop{\mathrm{Min}}}
```

vagy

```
</> \DeclareMathOperator*{\Min}{Min}
```

után

```
$\Min_{k \in \mathbb{N}}$ \[\Min_{k \in \mathbb{N}}\]
```

$$\begin{array}{c} \text{Min}_{k \in \mathbb{N}} \\ \text{Min}_{k \in \mathbb{N}} \end{array}$$

Egy már létező „limits” függvény át is definiálható.

```
\renewcommand{\parancs}{\mathop{\mathrm{parancs}}}
```

Például

```
\renewcommand{\min}{\mathop{\mathrm{min}}}
```

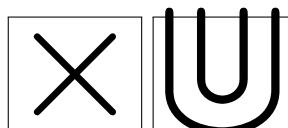
```
$\min_{k \in \mathbb{N}}$ \[\min_{k \in \mathbb{N}}\]
```

$$\begin{array}{c} \text{Min}_{k \in \mathbb{N}} \\ \text{Min}_{k \in \mathbb{N}} \end{array}$$

Új nagy operátor definiálása ♦ A rendelkezésre álló nagy operátorok (\sum , \prod , stb.) a matematikai fontkészletekben külön vannak megtervezve a szövegközi illetve a kiemelt matematikai üzemmód esetére. Ez felhasználói szinten korántsem egyszerű feladat, így ehelyett meglévő jeleket alakítunk át úgy, hogy az a lehető legjobban utánozza a nagy operátorok tulajdonságait.

Az alapötlet az, hogy a kiválasztott jelet kinagyítjuk úgy, hogy annak magassága megegyezzen az éppen aktuális méretű \prod jel magasságával. Ennek azonban van két nehézsége.

Az egyik abból fakad, hogy egy karakter dobozának méretei nem feltétlenül egyezik meg a karakter látható méreteivel. Például a \times és \cup jeleket egyforma magasságra kinagyítva és bejelölve a dobozuk határát, a következőt láthatjuk:



Az ábra szerint a \times doboza fehér margókat tartalmaz minden oldalon, míg az \cup doboza a bal és jobb oldalon margókat tartalmaz de alul és felül kilóg a dobozból.

A másik nehézség, hogy a \prod doboza másképp viselkedik szövegközi és kiemelt matematikai módban. Ennek bemutatására mindenki esetben ugyanarra a magasságra nagyítva és jelölve a doboz határait a következőt láthatjuk:



Az ábra szerint szövegközi módban csak bal és jobb oldalon van margó, de kiemelt módban minden oldalon.

Emiatt, ha például a \times és \cup jeleket csak simán kinagyítanánk az éppen aktuális \prod magasságára, akkor egymás után kiírva őket, a következőt kapnánk szövegközi módban:

$$\prod \times \cup$$

Míg kiemelt módban:

$$\prod \times \cup$$

De nemcsak a magassággal van gond, hanem az oldalmargók különbözősége miatt a jelet övező vízszintes térközök sem lesznek egyezők.

Tehát minden jel esetén egyedi mértékben kell egyeztetni a doboz margóját az aktuális \prod dobozával, ráadásul külön a szöveg és külön a kiemelt módban.

Ehhez a preambleumban a `mathtools` és `amssymb` csomagok betöltése után írja be a következőket:

```
\usepackage{scalerel,trimclip}
\DeclareRobustCommand{\bigopscale}[5]{%
\scalerel*{\trimbox{%
{#1\width}{%
{#2\totalheight}{%
{#3\width}{%
{#4\totalheight}{%
{$\#5$}}}{\prod}}}}{\DeclarerelBigOperator}[3][]{%
```

```
\DeclareMathOperator*{\#2}{%
  \begingroup%
  \ DeclareKeys{%
    DL.store = \bigopDL, DR.store = \bigopDR,
    DB.store = \bigopDB, DT.store = \bigopDT,
    TL.store = \bigopTL, TR.store = \bigopTR,
    TB.store = \bigopTB, TT.store = \bigopTT}%
  \SetKeys{DL=0, DB=0, DR=0, DT=0, TL=0, TR=0, TB=0, TT=0, #1}%
  \mathchoice
  {\bigopscale{\bigopDL}{\bigopDB}{\bigopDR}{\bigopDT}\{#3\}}
  {\bigopscale{\bigopTL}{\bigopTB}{\bigopTR}{\bigopTT}\{#3\}}
  {\bigopscale{\bigopTL}{\bigopTB}{\bigopTR}{\bigopTT}\{#3\}}
  {\bigopscale{\bigopTL}{\bigopTB}{\bigopTR}{\bigopTT}\{#3\}}%
  \endgroup}%
}
```

Ezután a következő módon definiálhatjuk a `\times` nagy operátor megfelelőjét például `\BigTimes` néven:

```
\DeclareMathBigOperator
[DL=0.165,DR=0.165,DB=0.09,DT=0.09,TL=0.17,TR=0.17,TB=0.136,TT=0.136]
{\BigTimes}{\times}
```

Az opcióban szereplő kétbetűs kulcsok első betűje aszerint **D** vagy **T**, hogy kiemelt (Display) vagy szöveg (Text) üzemmódra vonatkozik. A második betű aszerint **L**, **R**, **B** vagy **T**, hogy bal (Left), jobb (Right), alsó (Bottom) vagy felső (Top) margóra vonatkozik. minden kulcs alapértéke 0. Például a `DL=0.165` azt jelenti, hogy kiemelt üzemmódban a bal margóból vágjuk le a teljes szélesség 0,165-nyi részét, míg a `TB=0.136` azt jelenti, hogy szöveg üzemmódban az alsó margóból vágjuk le a teljes magasság 0,136-nyi részét.

Sajnos ezeket csak kísérletezéssel tudjuk megállapítani. Az ellenőrzéshez a dokumentumtestbe írja a következőket:

```
{\setlength{\fboxsep}{0pt}
\fbox{$\displaystyle\prod$}
\fbox{$\displaystyle\BigTimes$}
\fbox{$\displaystyle\prod$}
\fbox{$\displaystyle\BigTimes$}}
```

$$\prod \Bigtimes \prod \Bigtimes$$

Amint látható, a `\prod` és `\BigTimes` jelek a dobozaikban hasonlóan helyezkednek el minden üzemmódban, tehát a kompenzációs paraméterek megfelelők. Az ellenőrzés után az előző kód törölhető.

Ezután már a nagy operátoroknak megfelelően használhatjuk a `\BigTimes` jelet. Például

```
</> $ \prod_{i=1}^n A_i \BigTimes_{i=1}^n A_i $\\ $ [\prod_{i=1}^n A_i \BigTimes_{i=1}^n A_i] $
```

$$\prod_{i=1}^n A_i \Bigtimes_{i=1}^n A_i$$

$$\prod_{i=1}^n A_i \Bigtimes_{i=1}^n A_i$$

Másik példaként tekintsük az \uplus ($\backslash\text{\cup}$) nagy operátor megfelelőjét. Definiáljuk ezt például $\backslash\text{\BigCup}$ néven a következő módon:

```
\fbox{\setlength{\fboxsep}{0pt}
\fbox{$\displaystyle\prod$}
\fbox{$\displaystyle\BigCup$}
\fbox{$\displaystyle\prod$}
\fbox{$\displaystyle\BigCup$}}
```

Az értékek azért negatívak, mert a jel „kilóg” a neki fenntartott dobozból. A paraméterek ellenőrzése:

```
\fbox{\setlength{\fboxsep}{0pt}
\fbox{$\displaystyle\prod_{i=1}^n A_i \BigCup_{i=1}^n A_i$}
\fbox{$[\prod_{i=1}^n A_i \BigCup_{i=1}^n A_i]$}}
```

$$\prod \bigcup \prod \bigcup$$

Használata:

```
$\prod_{i=1}^n A_i \bigcup_{i=1}^n A_i$  
$[\prod_{i=1}^n A_i \bigcup_{i=1}^n A_i]$
```

$$\prod_{i=1}^n A_i \bigcup_{i=1}^n A_i$$

$$\prod_{i=1}^n A_i \bigcup_{i=1}^n A_i$$

15.19.5. Differenciál operátor, differenciálás

$f'(x), f''(x) \quad \mathbf{f}'(\mathbf{x}), \mathbf{f}''(\mathbf{x})$ (' az aposztrófjel **Shift** + **1**)

$\frac{\partial f(x, y)}{\partial y} \quad \frac{\partial f(x, y)}{\partial y}$

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}
```

$$\int f(x) dx \quad \text{és} \quad \frac{df(x)}{dx}$$

15.20. 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}\]
```

$$\begin{array}{ccccc}
 A & \longrightarrow & B & \longleftarrow & C \\
 \downarrow & & \uparrow & & \parallel \\
 D & \xrightarrow[l]{f} & E & = = = & F \\
 b \downarrow j & & b \uparrow j & & \\
 G & \xleftarrow[l]{f} & H & &
 \end{array}$$

Ettől többet tud az `xy` csomag, amit itt nem részletezünk.

15.21. Színek használata képletekben

Képletekben a színezéshez ne használjon `\color` vagy `\textcolor` parancsot, mert nem lesznek jók a térközök. Ehelyett a

```
\mathcolor[⟨modell⟩]{⟨színparaméter⟩}{⟨képlet⟩} ∈ xcolor
\mathcolor{⟨színnév⟩}{⟨képlet⟩} ∈ xcolor
```

használata javasolt. Például

```
\[ x = \mathcolor{red}{\sum_{i=1}^n x_i} \]
```

$$X = \sum_{i=1}^n x_i$$

15.22. 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{$\sum_{n=1}^\infty$}
\[ \colorbox{red}{$\displaystyle\sum_{n=1}^\infty$}
\quad\text{és}\quad
\fbox{$\displaystyle\sum_{n=1}^\infty$} \]
```

$$\sum_{n=1}^\infty$$

$$\sum_{n=1}^\infty$$

Létezik egy kifejezetten képlet bekeretezésére alkalmas `\boxed` parancs, melynek a belséjében 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}}
```

$$\sum_{n=1}^{\infty}$$

$$\sum_{n=1}^{\infty}$$

$$\sum_{n=1}^{\infty}$$

15.23. Kiemelt képletek sorszámozása

A kiemelt képletek sorszámozására használja az `equation` környezetet. Hivatkozás esetén `\ref` helyett az `\eqref` parancs használható:

```
\begin{equation}\label{<címke>}
<képlet>
\end{equation}
\eqref{<címke>}
```

A `magyar.ldf`-ben az `\eqref` előtt automatikus határozott névelő is rakható:

```
\Az{\eqref{<címke>}} \in [magyar]babel
\az{\eqref{<címke>}} \in [magyar]babel
```

Amennyiben a képletek római számozásúak, akkor ez nem ad helyes névelőt. Ez a probléma a `huaz` csomag betöltésével megoldható, ugyanakkor ebben az esetben a következő parancspár is használható, ami az előzővel egyenértékű:

```
\Aeqref{<címke>} \in huaz
\aeqref{<címke>} \in huaz
```

A `magyar.ldf` szerzője az előbbiek helyett az

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

megoldást javasolja, de ez nem feltétlenül helyes. Ugyanis az `\eqref` parancs eredménye minden álló betű lesz, még dőlt betűs környezetben is. Ezt viszont az `\aref{...}` és `\Aref{...}` parancsok esetén nem teljesül.

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 a `mathtools` csomagot `leqno` opcióval töltse be.

Az előbbi számozást `article` osztályban kapjuk. Ekkor az egész dokumentumban folytonos a számozás, azaz új szakasz nyitásakor 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 nyitásakor 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 a

```
\tag
\tag*
```

parancsok segítségével. 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*` parancssal számozott, akkor arra ne az `\eqref` parancssal hivatkozzon, mert az zárójelbe teszi a képletszámot. Helyette a `\refeq` parancsot alkalmazza.

Láttuk, hogy az alapértelmezett sorszámozás normál betűtípusossal zárójelben jelenik meg. Ha ezen változtatni akar, akkor használja a

```
\newtagform{\langle név\rangle}{\langle formázás\rangle}{\langle bal oldali zárójel\rangle}{\langle jobb oldali zárójel\rangle}
```

parancsot. Ahonnan ezt a beállítást aktiválni szeretné, oda írja be a

```
\usetagform{\nev}
```

parancsot. Például

```
\newtagform{brackets}[\textbf{[}{]}\{}[\textbf{]}{]}
\usetagform{brackets}
\begin{equation}\label{egyenlet-Einstein}
E=mc^2
\end{equation}
Lásd \eqref{egyenlet-Einstein}
```

$$E = mc^2$$

[1]

Lásd [1]

Visszatérni az alapbeállításhoz a következő parancssal lehet:

```
\usetagform{default}
```

Amennyiben `hyperref` csomagot is használ, akkor a `\tag` illetve `\tag*` parancsok használata `equation` környezetben, a fordítás során figyelmeztető üzeneteket eredményezhet (itt nem részletezett okok miatt). Ha ezt el akarja kerülni, akkor a `\tag` illetve `\tag*` parancsokat `equation` helyett használja `equation*` környezetben.

15.24. Képletek eltörése

Ha egy képlet nem fér ki egy sorban, akkor meg is lehet törni a `multiline` környezettel.

```
\begin{multiline}\label{címke}
<képlet 1. sora>\\
<képlet 2. sora>\\
...
<képlet n. sora>
\end{multiline}
```

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 a `mathtools` csomagot `fleqn` opcióval töltötte be, hogy a kiemelt képletek balra legyenek igazítva, akkor a középre igazított sorok a bal oldalra igazodnak.

Ha a `mathtools` 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 `multiline*` környezetet használja. Például



```
</> \begin{multiline}\label{egyenlet-pelda}
1+8+27+64=\\
=1+3+5+7+\{\}\\
+9+11+13+\{\}\\

```

```
+15+17+19
\end{multiline}
```

$$\begin{aligned}
 1 + 8 + 27 + 64 &= \\
 &= 1 + 3 + 5 + 7 + \\
 &\quad + 9 + 11 + 13 + \\
 &\quad + 15 + 17 + 19 \quad (1)
 \end{aligned}$$

</>

```
\begin{multiline}\label{egyenlet-pelda}
1+8+27+64=\\
\shoveleft{=1+3+5+7+{}\\\\
+9+11+13+{}\\\
+15+17+19
\end{multiline}
```

$$\begin{aligned}
 1 + 8 + 27 + 64 &= \\
 &= 1 + 3 + 5 + 7 + \\
 &\quad + 9 + 11 + 13 + \\
 &\quad + 15 + 17 + 19 \quad (1)
 \end{aligned}$$

</>

```
\begin{multiline}\label{egyenlet-pelda}
1+8+27+64=\\
=1+3+5+7+{}\\\\
\shoveright{+9+11+13+{}\\\\
+15+17+19
\end{multiline}
```

$$\begin{aligned}
 1 + 8 + 27 + 64 &= \\
 &= 1 + 3 + 5 + 7 + \\
 &\quad + 9 + 11 + 13 + \\
 &\quad + 15 + 17 + 19 \quad (1)
 \end{aligned}$$

Ha a megtört képletet adott pontokon illeszteni is szeretné egymáshoz, akkor használható a `split` környezet.

```
\begin{equation}\label{címke}
\begin{split}
\langle \text{képlet 1. sora} \rangle & \& \langle \text{képlet 1. sora} \rangle \\
\langle \text{képlet 2. sora} \rangle & \& \langle \text{képlet 2. sora} \rangle \\
... \\
\langle \text{képlet } n. \text{ sora} \rangle & \& \langle \text{képlet } n. \text{ sora} \rangle
\end{split}
\end{equation}
```

A `split` hasonlóan működik, mint egy táblázat. A tabulálást itt is a `&` jelrel, míg a sortörést a `\&` parancssal végezze. A `multiline` környezettel ellentétben ez nem biztosít kiemelt matematika környezetet, így erről külön kell gondoskodni. Ezért van az előző kódban `equation` környezetbe zárva. De természetesen lehetett volna `equation*`

környezetbe is tenni, amivel számozás nélküli esetet kapunk. Az egyenlet számozása függőlegesen középre lesz igazítva. Ha a `mathtools` csomagot `tbtags` opcióval tölti be, akkor a képletszám az utolsó sorban jelenik meg. Ha még a `leqno` opciót is használja, hogy a számozás a bal oldalon legyen, akkor a képletszámozás az első sor bal oldalán lesz. Például

```
</> \begin{equation}\label{egyenlet-pelda}
\begin{split}
100 &= 1+8+27+64=\\
&= 1+3+5+7+9+\boxed{\phantom{000}}\\
&\phantom{100=1+8+27+64=}+11+13+15+17+19
\end{split}
\end{equation}
```

$$\begin{aligned} 100 &= 1 + 8 + 27 + 64 = \\ &= 1 + 3 + 5 + 7 + 9 + \\ &\quad + 11 + 13 + 15 + 17 + 19 \end{aligned} \tag{1}$$

Ha a megtört képletben egy olyan részt akar bekeretezni, amely tabulátorjelet tartalmaz, akkor a korábban ismertetett `\boxed` parancs helyett az `\Aboxed` parancsot használja. Például

```
</> \begin{equation*}
\begin{split}
\Aboxed{100 &= 1+8+27+64}=\\
&= 1+3+5+7+9+\boxed{\phantom{000}}\\
&\phantom{100=1+8+27+64=}+11+13+15+17+19
\end{split}
\end{equation*}
```

$$\begin{aligned} \boxed{100} &= 1 + 8 + 27 + 64 = \\ &= 1 + 3 + 5 + 7 + 9 + \\ &\quad + 11 + 13 + 15 + 17 + 19 \end{aligned}$$

15.25. 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. képlet \rangle \label{címke 1} \\
\langle 2. képlet \rangle \label{címke 2} \\
...
\langle n. képlet \rangle \label{címke n}
\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 \label{egyenlet-pelda-a}\\
x^2+xy+y^2 \label{egyenlet-pelda-b}
\end{gather}
```

$$x + y \quad (1)$$

$$x^2 + xy + y^2 \quad (2)$$

```
</> \begin{gather}
x+y \notag\\
x^2+xy+y^2 \label{egyenlet-pelda}
\end{gather}
```

$$\left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\} \quad (1)$$

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.\}]
```

$$\left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\}$$

```
</> \begin{equation}\label{egyenlet-pelda}
\left.\begin{gathered}
x+y\\
x^2+xy+y^2
\end{gathered}\right.
\end{equation}
```

$$\left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\} \quad (1)$$

```
</> [\left.\begin{gathered}
x+y\\
x^2+xy+y^2
\end{gathered}\right.\}
\quad\text{\'es}\quad
\left.\begin{gathered}
\end{gathered}\right.]
```

```
2x+y\\
x^2+3xy+y^2
\end{gathered}\right.\}]
```

$$\left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\} \quad \text{és} \quad \left. \begin{array}{c} 2x + y \\ x^2 + 3xy + y^2 \end{array} \right\}$$



```
</> szöveg
$ \begin{gathered}
x+y\\
x^2+xy+y^2
\end{gathered} $
szöveg
```

$$\text{szöveg } \left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\} \text{ szöveg}$$

```
szöveg
$ \begin{gathered}[t]
x+y\\
x^2+xy+y^2
\end{gathered} $
szöveg
```

$$\text{szöveg } \left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\} \text{ szöveg}$$

```
szöveg
$ \begin{gathered}[b]
x+y\\
x^2+xy+y^2
\end{gathered} $
szöveg
```

$$\text{szöveg } \left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\} \text{ szöveg}$$

A `gathered` helyett használhatók még az `lgathered` illetve `rgathered` környezetek is, melyek csak annyiban különböznek a `gathered`-től, hogy a sorok nem középre, hanem balra illetve jobbra igazítottak. Például



```
\left[ \begin{array}{l}
x+y\\
x^2+xy+y^2
\end{array} \right]
```

$$\left. \begin{array}{c} x + y \\ x^2 + xy + y^2 \end{array} \right\}$$

15.26. Több képlet egymás alatt illesztéssel

Egymás alatti képletekben lehetnek olyan elemek, amelyeket egymáshoz kell illeszteni. Erre több környezet is lehetőséget ad. Az `align` környezetben az igazítás a táblázatoknál tanultak szerinti `r@{ }1r@{ }1...`, ahol az első oszlop előtti, utolsó oszlop utáni, illetve az 1 és r oszlopok közötti távolságok egyenletesen oszlanak el.

```
\begin{align}
<1. sor jobbra>&<balra> & <jobbra>&<balra> & <jobbra> ... \label{<címke 1>}\\
<2. sor jobbra>&<balra> & <jobbra>&<balra> & <jobbra> ... \label{<címke 2>}\\
...
<n. sor jobbra>&<balra> & <jobbra>&<balra> & <jobbra> ... \label{<címke n>}\\
\end{align}
```

Például

```
</> \begin{align}
x&=y+z & y&=bd & z&=bc \label{egyenlet-pelda-a}\\
b&=10 & 2c&=56 & d&=44 \label{egyenlet-pelda-b}\\
\end{align}
```

$x = y + z$	$y = bd$	$z = bc$	(1)
$b = 10$	$2c = 56$	$d = 44$	(2)

A `\tag`, `\tag*`, `\notag` parancsok itt is ugyanúgy használhatók, mint a `gather` környezetben. Az `align*` környezet pontosan azt csinálja, mint az `align`, de nem tesz ki képletszámokat.

```
</> \begin{align*}
x&=y+z=&\\
&=bd+bc=&\\
&=1000\\
\end{align*}
```

$$\begin{aligned} x &= y + z = \\ &= bd + bc = \\ &= 1000 \end{aligned}$$

```
</> \begin{align*}
x&=y+z= \quad \& \text{a definícióból}\\
&=bd+bc= \quad \& \text{mivel } ac=b\\
&=1000 \quad \& \text{behelyettesítve}\\
\end{align*}
```

$$\begin{aligned} x &= y + z = && \text{a definícióból} \\ &= bd + bc = && \text{mivel } ac = b \\ &= 1000 && \text{behelyettesítve} \end{aligned}$$

```
</> \begin{align*}
x&=y+z= \quad \& \text{a definícióból}\\
&=bd+bc= \quad \& \text{mivel } ac=b\\
\end{align*}
```

```
&=1000 & \text{behelyettesítve}
\end{aligned*}
```

$$\begin{aligned} x &= y + z = && \text{a definícióból} \\ &= bd + bc = && \text{mivel } ac = b \\ &= 1000 && \text{behelyettesítve} \end{aligned}$$

Az `align*` környezet ún. részformulaképző változata az `aligned` környezet. Ez azt jelenti, hogy úgy működik mint az `align*`, de szövegközi matematikai módba, `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

$$\begin{aligned} \text{szöveg } x &= y + z & y &= bd & z &= bc \\ &b = 10 & 2c &= 56 & d &= 44 \end{aligned} \text{ szöveg}$$

```
szöveg
$\begin{aligned}[t]
x&=y+z & y&=bd & z&=bc \\
b&=10 & 2c&=56 & d&=44
\end{aligned}$
```

szöveg

$$\begin{aligned} \text{szöveg } x &= y + z & y &= bd & z &= bc \text{ szöveg} \\
&b = 10 & 2c &= 56 & d &= 44 \end{aligned}$$

```
szöveg
$\begin{aligned}[b]
x&=y+z & y&=bd & z&=bc \\
b&=10 & 2c&=56 & d&=44
\end{aligned}$
```

szöveg

$$\begin{aligned} x &= y + z & y &= bd & z &= bc \\
\text{szöveg } b &= 10 & 2c &= 56 & d &= 44 \end{aligned} \text{ szöveg}$$

```
\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}
```

$$\left. \begin{array}{l} x = y + z \\ b = 10 \end{array} \quad \begin{array}{l} y = bd \\ 2c = 56 \end{array} \quad \begin{array}{l} z = bc \\ d = 44 \end{array} \right\} \quad (1)$$

```
</> \[ \left. \begin{aligned} x&=y+z \\ b&=10 \\ \end{aligned} \right\} \quad \left. \begin{aligned} \text{\'es} \\ y&=bd \\ 2c&=56 \\ \end{aligned} \right\} \]
```

$$\left. \begin{array}{l} x = y + z \\ b = 10 \end{array} \right\} \quad \text{és} \quad \left. \begin{array}{l} y = bd \\ 2c = 56 \end{array} \right\}$$

```
</> \vspace{\baselineskip}
\[\left.\begin{aligned}
\noalign{\vspace{-\baselineskip}}\\
v&=s\\
v'_{x}&=1\\
v'_{y}&=2\\
\end{aligned}\right.\]
```

$$\begin{cases} v'_x = 1 \\ v'_y = 2 \end{cases}$$

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@{}1 r@{}1 ...` 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@{}1` oszloppár van. Ezt úgy lehet kiszámolni, hogy a & tabulátorjelek számához hozzáadunk 1-et, majd osztjuk 2-vel. 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 \text{\label{egyenlet-pelda-a}}\\
3&=3 & 3&=1+2& \quad 3&=1+1+1 \quad \text{\label{egyenlet-pelda-b}}
```

```
\end{alignat}
```

$$1 = 1 \quad 2 = 2 \quad 2 = 1 + 1 \quad (1)$$

$$3 = 3 \quad 3 = 1 + 2 \quad 3 = 1 + 1 + 1 \quad (2)$$

Ezzel lineáris egyenletrendszer felírása is megoldható. Például

</>

```
\begin{alignat*}{3}
13&x+{} & 4&y & =9\\
3&x-{} & 12&y+{} & 23z&=14
\end{alignat*}
```

$$13x + 4y = 9$$

$$3x - 12y + 23z = 14$$

</>

```
\begin{alignat*}{4}
13&x+{} & 4&y & ={} & 9&\\
3&x-{} & 12&y+{} & 23z&={}& 14&
\end{alignat*}
```

$$13x + 4y = 9$$

$$3x - 12y + 23z = 14$$

Lineáris egyenletrendszer jóval könnyebben is megvalósítható a `\systeme` parancs segítségével. Ennek általános leírását lásd a `systeme` csomag leírásában, most csak néhány példával illusztráljuk a működését.

</>

```
\[\systeme{2a-3b+4c=2, a+8b+5c=8, -a+2b+c=-5}\]
```

$$\begin{cases} 2a - 3b + 4c = 2 \\ a + 8b + 5c = 8 \\ -a + 2b + c = -5 \end{cases}$$

</>

```
\[\systeme[delim={.,\rbrace}]{2a-3b+4c=2, a+8b+5c=8, -a+2b+c=-5}\]
```

$$\left. \begin{array}{l} 2a - 3b + 4c = 2 \\ a + 8b + 5c = 8 \\ -a + 2b + c = -5 \end{array} \right\}$$

</>

```
\[\systeme[eq sep = ;]{1,5x-0,45y=0,7; x-0,8y=1,4}\]
```

$$\left. \begin{array}{l} 1,5x - 0,45y = 0,7 \\ x - 0,8y = 1,4 \end{array} \right\}$$

</>

```
\[\systeme{(2+\sqrt{2})x-(1-\sqrt{2})y=1, x+(1+\sqrt{2})y=-1}\]
```

$$\begin{cases} (2 + \sqrt{2})x - (1 - \sqrt{2})y = 1 \\ x + (1 + \sqrt{2})y = -1 \end{cases}$$

</> `\[\systeme[sort={x,y}]{mx-y=3, x-m^2y=1}\]`

$$\begin{cases} mx - y = 3 \\ x - m^2y = 1 \end{cases}$$

</> `\[\systeme{x+y=125@(L_1), x-y=12@(L_2)}\]`

$$\begin{cases} x + y = 125 & (L_1) \\ x - y = 12 & (L_2) \end{cases}$$

Visszatérve az `alignat*` környezetre, annak most egy ún. részformulaképző változatát, az `alignedat` környezetet ismertetjük. Ez azt jelenti, hogy úgy működik mint az `alignat*`, de szövegközi 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}`

$$\begin{cases} 11x - 4y = 7 \\ x - y = 0 \end{cases} \quad (1)$$

</> `\[\left.\begin{alignedat}{2}`
`11&x-{} & 4y&=7\\`
`&x-{} & y&=0`
`\end{alignedat}\right.\!`
`\Rightarrow`
`x=y=1\]`

$$\begin{cases} 11x - 4y = 7 \\ x - y = 0 \end{cases} \Rightarrow x = y = 1$$

</> `szöveg`
`$\begin{alignedat}{2}`
`11&x-{} & 4y&=7\\`
`&x-{} & y&=0`
`\end{alignedat}$`
`szöveg`

szöveg
$$\begin{aligned} 11x - 4y &= 7 \\ x - y &= 0 \end{aligned}$$
 szöveg

```
szöveg
$\begin{alignedat}{2}
11&x-{} & 4y&=7\\
&x-{} & y&=0
\end{alignedat}$
szöveg
```

szöveg $11x - 4y = 7$ szöveg
 $x - y = 0$

```
szöveg
$\begin{alignedat}{2}
11&x-{} & 4y&=7\\
&x-{} & y&=0
\end{alignedat}$
szöveg
```

$11x - 4y = 7$
szöveg $x - y = 0$ szöveg

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}
```

$$\begin{aligned} 2x &= 2y \Rightarrow x = y \\ 6z &= 600 \Rightarrow z = 100 \end{aligned} \tag{1}$$

Ezt a környezetet gyakran használják tévesen a következő esetben:

```
\begin{eqnarray*}
1+3 &=& 4 \\
1+3+5 &=& 9
\end{eqnarray*}
```

$$\begin{aligned} 1 + 3 &= 4 && \text{Ez rossz példa!} \\
1 + 3 + 5 &= 9 && \text{Így soha!}
\end{aligned}$$

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*}
```

$$\begin{aligned} 1 + 3 &= 4 \\ 1 + 3 + 5 &= 9 \end{aligned}$$

Általánosságban elmondható, hogy az `eqnarray` környezet használatát kerülni kell (lásd <https://tug.org/TUGboat/tb33-1/tb103madsen.pdf>).

Ha az illesztett képletek száma nyílt, azaz csak függőlegesen elhelyezett három ponttal lehet jelölni, akkor a következő példákban látható megoldásokat alkalmazza:

</>

```
\begin{align*}
a_1 &= b_1 \\
&\vdots \\
a_n &= b_n
\end{align*}
```

$$\begin{aligned} a_1 &= b_1 \\ &\vdots \\ a_n &= b_n \end{aligned}$$

</>

```
\begin{alignat*}{3}
A &+ B &= C &+ D \\
&\vdots && \\
&\vdots && \\
C &+ D &= Y &+ K
\end{alignat*}
```

$$\begin{aligned} A + B &= C + D \\ &\vdots && \\ &\vdots && \\ C + D &= Y + K \end{aligned}$$

Szöveget is elhelyezhet illesztett képletek sorai között. Erre az

```
\intertext{\textcolor{green}{(szöveg)}}
\shortintertext{\textcolor{green}{(szöveg)}}
```

parancsok használhatók, melyek a következő környezetekben működnek: `align`, `align*`, `flalign`, `flalign*`, `alignat`, `alignat*`. Például

</>

```
\begin{align*}
f(x) &= \int 4x \ln x \, dx, \mathbf{d} x = \\
&\shortintertext{parciális integrálás után} \\
&= 2x^2 \ln x - x^2 + C = \\
&= x^2 \left( 2 \ln x - 1 \right) + C
\end{align*}
```

$$f(x) = \int 4x \ln x \, dx =$$

parciális integrálás után

$$= 2x^2 \ln x - x^2 + C = \\ = x^2 (2 \ln x - 1) + C$$

Az `\intertext` annyiban különbözik a `\shortintertext` parancstól, hogy ott nagyobb a képletek és szöveg közötti térköz.

15.27. 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{align}
x &= ac + bc \quad \text{\label{reszformula-pelda-a}} \\
y &> dc \quad \text{\label{reszformula-pelda-b}}
\end{align}
\end{subequations}
```

$$x = ac + bc \tag{1a}$$

$$y > dc \tag{1b}$$

Ha a részformulák számozásának stílusát át akarja állítani például (1/a) alakúra, akkor a `mathtools` csomag betöltése után másolja be a következő kódot:

```
</> \usepackage{etoolbox}
\patchcmd{\subequations}{\alph}{\alpha}{}{}
```

15.28. Oldaltörés többsoros képletekben

Alapértelmezésben a többsoros képletek közben nem megengedett az oldaltörés. Ezt a preambulumba írt

```
\allowdisplaybreaks
```

parancsal 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.

15.29. Táblázat matematikai módban

Táblázatot matematikai módban `tabular` helyett `array` környezettel kell készíteni, ami-nek használata megegyezik a `tabular` környezettel, de a cellák tartalmát nem kell külön matematikai módba rakni. Hasonlóan használható a `tabarray` csomag `tblr` környezete is (lásd a 10.2. szakaszban.) 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}
\alpha & \beta \\
\gamma & \delta \\
\hline
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9 \\
\hline
\end{array}\]
```

$$\begin{array}{|c|c|c|c|} \hline & \alpha & \beta & \\ \hline \gamma & & \delta & \\ \hline 1 & 2 & 3 & \\ 4 & 5 & 6 & \\ 7 & 8 & 9 & \\ \hline \end{array}$$

15.30. Számolás L^AT_EX segítségével

Számolni is lehetséges L^AT_EX-ben az

```
\fpeval{\langle keplet\rangle}
```

parancs segítségével, amelynek a neve a „*floating point evaluation*” azaz „lebegő pontos értékelés” kifejezésre utal. Ez a parancs 2022. júniusa után telepített rendszerekben alapból elérhető, míg korábbi verziókban az `xfp` csomag betöltésével használható. A `\langle keplet\rangle` a következőket tartalmazhatja:

`+ - * / ^ ()` műveleti jelek, hatványozás, zárójelek. Például $2^3 \cdot \frac{5+2,2}{2}$

```
\fpeval{(2^3)*(5+2.2)/2}
```

28.8

`\langle x\rangle e\langle n\rangle` normálalak. Például $2,2 \cdot 10^{-1} + 3,3 \cdot 10^2$

```
\fpeval{2.2e-1+3.3e2}
```

330.22

`sign($\langle x \rangle$) sqrt($\langle x \rangle$) abs($\langle x \rangle$) exp($\langle x \rangle$) ln($\langle x \rangle$)` előjel, négyzetgyök, abszolút érték, természetes alapú exponenciális illetve logaritmus függvények. Például $\sqrt{\ln(2)}$

`\fpeval{sqrt(ln(2))}`

0.8325546111576978

`fact($\langle n \rangle$)` faktoriális. Például 5!

`\fpeval{fact(5)}`

120

`sin($\langle x \rangle$) cos($\langle x \rangle$) tan($\langle x \rangle$) cot($\langle x \rangle$) sec($\langle x \rangle$) csc($\langle x \rangle$)` trigonometrikus függvények, ahol az $\langle x \rangle$ radiánban van megadva. Például $\sin 2$

`\fpeval{sin(2)}`

0.9092974268256817

`sind($\langle x \rangle$) cosd($\langle x \rangle$) tand($\langle x \rangle$) cotd($\langle x \rangle$) secd($\langle x \rangle$) cscd($\langle x \rangle$)` trigonometrikus függvények, ahol az $\langle x \rangle$ fokban van megadva. Például $\sin 20^\circ$

`\fpeval{sind(20)}`

0.3420201433256687

`asin($\langle x \rangle$) acos($\langle x \rangle$) atan($\langle x \rangle$) acot($\langle x \rangle$) asec($\langle x \rangle$) acsc($\langle x \rangle$)` inverz trigonometrikus függvények, ahol az eredményt radiánban kapjuk. Például $\arcsin(-0,2)$

`\fpeval{asin(-.2)}`

-0.2013579207903308

`asind($\langle x \rangle$)acosd($\langle x \rangle$)atand($\langle x \rangle$)acotd($\langle x \rangle$)asecd($\langle x \rangle$)acscd($\langle x \rangle$)` inverz trigonometrikus függvények, ahol az eredményt fokban kapjuk. Például $\arcsin(-0,2)$

`\fpeval{asind(-.2)}` (fok)

-11.53695903281549 (fok)

`min($\langle x1 \rangle, \langle x2 \rangle, \dots$)` számok minimuma. Például $\min\{5, 2, 7\}$

`\fpeval{min(5,2,7)}`

2

`max($\langle x1 \rangle, \langle x2 \rangle, \dots$)` számok maximuma. Például $\max\{5, 2, 7\}$

`\fpeval{max(5,2,7)}`

7

`trunc($\langle x \rangle$)` levágja a tizedesjegyeket (azaz 0 felé egészre kerekít). Például

`\fpeval{trunc(5.21)}, \fpeval{trunc(-5.21)}`

5, -5

`trunc($\langle x \rangle, \langle n \rangle$)` az első $\langle n \rangle$ darab tizedesjegyet meghagyja, a többöt levágja (azaz 0 felé $\langle n \rangle$ tizedesjegyre kerekít). Például

`\fpeval{trunc(5.21,1)}, \fpeval{trunc(-5.21,1)}`

5.2, -5.2

`floor(<x>)` lefelé egészre kerekít (azaz az egészrész függvény). Például

`\fpeval{floor(5.21)}, \fpeval{floor(-5.21)}`

5, -6

`floor(<x>, <n>)` lefelé `<n>` tizedesjegyre kerekít. Például

`\fpeval{floor(5.21,1)}, \fpeval{floor(-5.21,1)}`

5.2, -5.3

`ceil(<x>)` felfelé egészre kerekít. Például

`\fpeval{ceil(5.21)}, \fpeval{ceil(-5.21)}`

6, -5

`ceil(<x>, <n>)` felfelé `<n>` tizedesjegyre kerekít. Például

`\fpeval{ceil(5.21,1)}, \fpeval{ceil(-5.21,1)}`

5.3, -5.2

`round(<x>)` a legközelebbi egészre kerekít. Például

`\fpeval{round(5.51)}, \fpeval{round(-5.5)}`

6, -6

`round(<x>, <n>)` a legközelebbi `<n>` tizedesjegyre kerekít. Például

`\fpeval{round(5.56,1)}`

5.6

`rand()` (pszeudo) véletlen szám 0 és 1 között egyenletes eloszlás szerint (azaz a $[0, 1]$ intervallum tetszőleges p hosszúságú részintervallumába p valószínűséggel eshet). Például

`\fpeval{rand()}`

0.7790313711999643

`randint(<m>, <n>)` (pszeudo) véletlen egész szám, amely egyenlő valószínűséggel bár-milyen érték lehet `<m>` és `<n>` között (beleértve a határokat is). Például

`\fpeval{randint(1,6)}`

4

`pi deg` a π értéke illetve 1° értéke radiánban.

`mm cm em ex ...` az adott mértékegység pontban kifejezve. Például

`\fpeval{mm}, \fpeval{2cm}`

2.845275590551181, 56.90551181102362

Ez hosszúságparancsok esetén is működik, azaz az aktuális hosszúságot kiírja pontban. Például

```
\fpeval{\textwidth}
```

403.1072692871094

ami így számolható át centiméterbe:

```
\fpeval{\textwidth/cm}
```

14.16760016589536

$\langle x_1 \rangle, \langle x_2 \rangle, \dots, \langle x_n \rangle$ vektorok kezelése. Például

```
\fpeval{(1,2,3)*2.1 + (1,1,1)}
```

(3.1, 5.2, 7.3)

= != < <= > >= relációk. Ha a relációval megadott logikai kifejezés igaz, akkor 1, ellenkező esetben pedig 0 értékkel tér vissza. Például

```
\fpeval{2<2.1}, \fpeval{2>2.1}
```

1, 0

! && || Logikai műveletek: negálás, és, vagy. Például

```
\fpeval{!(2<2.1)}, \fpeval{(2<2.1)&&(2>2.1)}, \fpeval{(2<2.1)|| (2>2.1)}
```

0, 0, 1

$\langle \text{logikai kifejezés} \rangle ? \langle \text{ha igaz} \rangle : \langle \text{ha hamis} \rangle$ feltétel vizsgálata. Például

```
\fpeval{\sin(10)>0 ? \sin(10) : 10*\sin(10)}
```

-5.440211108893698

Amennyiben a számolt értéket meg akarja jeleníteni a dokumentumban, akkor magyar nyelv esetén felmerül a következő probléma. Például

```
\frac{\sin(3,5)}{2} + 2\cdot 10^{-3}
= \fpeval{\sin(3.5)/2 + 2e-3}\]
```

$$\frac{\sin(3,5)}{2} + 2 \cdot 10^{-3} = -0.1733916138448099$$

Amint látható, az eredményben tizedesvessző helyett tizedespont van. Ezt a következő módon lehet megoldani. Először definiálunk egy `\tocomma` parancsot, ami a pontot vesszőre cseréli ([forrás](#)):

```
\ExplSyntaxOn
\cs_new:Npn \mich_convert_tocomma:n #1
{
    \exp_args:Ne \tl_to_str:n
        { \str_map_function:nN {#1} \__mich_convert_tocomma:n }
}
\cs_new:Npn \__mich_convert_tocomma:n #1
{
```

```
\int_compare:nNnTF { `#1 } = { `. }
{ ,
{ #1 }
}
\cs_generate_variant:Nn \mich_convert_tocomma:n {e}
\cs_set_eq:NN \tocomma \mich_convert_tocomma:e
\ExplSyntaxOff
```

Ezután

```
\[ [\frac{\sin(3,5)}{2} + 2\cdot 10^{-3}]
= \tocomma{\fpeval{\sin(3.5)/2 + 2e-3}}\]
```

$$\frac{\sin(3,5)}{2} + 2 \cdot 10^{-3} = -0,1733916138448099$$

Ugyanez a hatás az `xstring` csomag `\StrSubstitute` parancsának használatával egyszerűbben is elérhető:

```
\[ [\frac{\sin(3,5)}{2} + 2\cdot 10^{-3}]
= \StrSubstitute{\fpeval{\sin(3.5)/2 + 2e-3}}{.}{,}\]
```

15.31. HTML-ben képletek írása L^AT_EX szintaxissal

Erre alkalmas a `MathJax` nevű JavaScript. Egy példát is megnézhet itt: [kattintson ide!](#)

16. fejezet

További formai elemek

16.1. Görög betűk

Görög betűkre legtöbbször képletek írásakor van szükség. Ezt az esetet a 15.4. szakaszban tárgyaljuk. Ha latin betűs környezetben szeretne görög betűket írni, de nem képletben, akkor ehhez a T1 belső kódkészlet előtt tölts be az LGR-t is:

```
\usepackage[LGR,T1]{fontenc}
```

Ezután használja a következő kódot:

```
{\fontencoding{LGR}\selectfont <görög betűk>}
```

Például

```
</> {\fontencoding{LGR}\selectfont abcdefghijklmnopqrstuvwxyz}
```

αβγδεφηιθκλμνοπχρστυξψ

A következő kóddal egyéb fontkészletet is használhat:

```
{\fontencoding{LGR}\fontfamily{<font>}\selectfont <görög betűk>}
```

ahol a ** értékei a következők lehetnek például: *artemisia*, *gfsbaskerville*, *bodoni*, *complutum*, *udidot*, *neohellenic*, *porson*, *solomos*, *txr*, *mak*, *llcmss*. Például

```
</> {\fontencoding{LGR}\fontfamily{gfsbaskerville}\selectfont abcdefghijklmnopqrstuvwxyz} \\
{\fontencoding{LGR}\fontfamily{solomos}\selectfont abcdefghijklmnopqrstuvwxyz}
```

αβγδεφηιθκλμνοπχρστυξψ
αβγδεφηιθκλμνοπχρστυξψ

A *textgreek* csomaggal görög betű a neve alapján is kiíratható:

α \textalpha	ι \textiota	π \textpi	ω \textomega
β \textbeta	κ \textkappa	ρ \textrho	\Alpha \textAlpha
γ \textgamma	λ \textlambda	σ \textsigma	\Beta \textBeta
δ \textdelta	μ \textmu	τ \texttau	Γ \textGamma
ϵ \textepsilon	ν \textnu	υ \textupsilon	Δ \textDelta
ζ \textzeta	ξ \textxi	ϕ \textphi	\Epsilon \textEpsilon
η \texteta	\omicron \textomikron	χ \textchi	\Zeta \textZeta
θ \texttheta		ψ \textpsi	\Eta \textEta

Θ \textTheta	Ξ \textXi	Υ \textUpsilon	ϕ \straightphi
I \textIota	O \textOmicron	Φ \textPhi	ϑ \scripttheta
K \textKappa	Π \textPi	X \textChi	θ \straighttheta
Λ \textLambda	P \textRho	Ψ \textPsi	ϵ \straightepsilon
M \textMu	Σ \textSigma	Ω \textOmega	
N \textNu	T \textTau	ς \textvarsigma	

16.2. Cirill betűk

Latin betűs szövegben néha szükség lehet cirill betűkre is. Ehhez a T1 belső kódkészlet előtöltse be a T2C-t is:

```
\usepackage[T2C,T1]{fontenc}
```

Ekkor a T1 lesz az alapértelmezett. A T2C cirill betűi:

А \CYRA	Ы \CYRERY	҃ \cyrc	҂ \CYRSEMISFTSN
Б \CYRB	Ӧ \CYRSFTSN	҄ \cyrch	҇ \CYRSCHWA
В \CYRV	Ӧ \CYREREV	҅ \cyrsh	҈ \CYRII
Г \CYRG	Ҍ \CYRYU	҆ \cyrshch	҉ \CYRJE
҂ \CYRD	ҍ \CYRYA	Ҋ \cyrhrdsn	Ҋ \CYRpalochnka
҃ \CYRE	Ҏ \cyra	ҋ \cyrery	Ҍ \cyrabch
҄ \CYRYO	ҏ \cyrb	Ҍ \cyrsftsn	Ҕ \cyrabchdsc
҅ \CYRZH	Ґ \cyrv	Ҏ \cyrerev	Җ \cyrabhdze
҇ \CYRZ	ґ \cyrg	ҏ \cyryu	҈ \cyrkhcrs
҈ \CYRI	Ғ \cyrd	ғ \cyrya	҉ \cyrkdsc
҉ \CYRISHRT	Ҕ \cyre	Ҕ \CYRABHCH	Ҋ \cyrmdsc
Ҋ \CYRK	ҕ \cyryo	Ҕ \CYRABHDSC	ҋ \cyrndsc
ҋ \CYRL	ҏ \cyrzh	Җ \CYRABHDZE	Ҍ \cyrotld
Ҍ \CYRM	Ґ \cyrz	҈ \CYRKHCRS	ҍ \cyrphk
ҍ \CYRN	ґ \cyri	҉ \CYRKDSC	Ҏ \cyrrtick
Ҏ \CYRO	Ғ \cyrishrt	Ҋ \CYRMDSC	ҏ \cyrtdsc
ҏ \CYRP	Ҕ \cyrk	ҋ \CYRNDSC	Ҍ \cyrshha
Ґ \CYRR	ҏ \cyrl	Ҍ \CYROTLD	҅ \cyrghk
ґ \CYRS	Ґ \cyrm	Ҕ \CYRPHK	҈ \cyrhdsc
Ғ \CYRT	ґ \cyrn	Җ \CYRRTICK	҉ \cyrdzhe
ғ \CYRU	Ҕ \cyro	҈ \CYRTDSC	Ҋ \cyrdze
Ҕ \CYRF	ҏ \cyrp	҉ \CYRSHHA	ҋ \cyretse
ҕ \CYRH	Ґ \cyrr	Ҋ \CYRGHK	Ҍ \cyrchrds
ҏ \CYRC	ґ \cyrs	ҋ \CYRHDSC	Ҕ \cyrsemisftsn
Ґ \CYRCH	Ғ \cyrt	Ҍ \CYRDZHE	Ҏ \cryscha
ґ \CYRSH	Ҕ \cyru	҄ \CYRDZE	Ҍ \cyrii
Ғ \CYRSHCH	ҏ \cyrf	҅ \CYRETSE	҉ \cyrje
ғ \CYRHRDSN	Ґ \cyrh	҆ \CYRCHRDSC	Ҕ \cyrabha

Az T2C cirill betűit az alábbi parancssal jelenítheti meg:

```
{\fontencoding{T2C}\selectfont cirill betűk}
```

Például

```
</> {\fontencoding{T2C}\selectfont
```

```
\CYRR\'\cyr\cyrs\cyrs\cyrk\cyri\cyrishrt\ \cyrt\cyre\cyrk\cyrs\cyrt\
```

Русский текст

Ezek elérhetők TeXstudióból is: [Oldalpanel] > Szimbólumok > Cirill.

16.3. 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 \dots 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.

16.4. Iniciálék

16.4.1. Latin iniciálé

Írja a következőket a preambulumba

```
</> \usepackage{anyfontsize,lettrine}  
\setcounter{DefaultLines}{4}
```

majd a dokumentumtestbe

```
\lettrine{K}{ezdetben} teremté 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.
```

KEZDET BEN teremté 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.

Ha ékezes betűt használ iniciálénak, akkor azt parancsként írja be. Például

```
\lettrine{\{E\}}{s} monda Isten
```

16.4.2. 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 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 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.

Ha ékezes betűt használ iniciálénak, akkor azt parancsként írja be. Például

```
\lettrine{\'{E}}{s} monda Isten
```

A **Zallman** mintázat helyére a következő mintázatok is beírhatók: **Acorn**, **AnnSton**, **ArtNouv**, **ArtNouvc**, **Carrickc**, **Eichenla**, **Eileen**, **EileenBl**, **Elzevier**, **GotIn**, **GoudyIn**, **Kinigcap**, **Konanur**, **Kramer**, **MorrisIn**, **Nouveaud**, **Romantik**, **Rothdn**, **RoyalIn**, **Sanremo**, **Starburst**, **Typocaps**.

16.4.3. Gótikus iniciálé

Írja a következőket a preambulumba

</>

```
\usepackage{yfonts,anyfontsize,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 \dots 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 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.

16.4.4. Díszes gótikus iniciálé

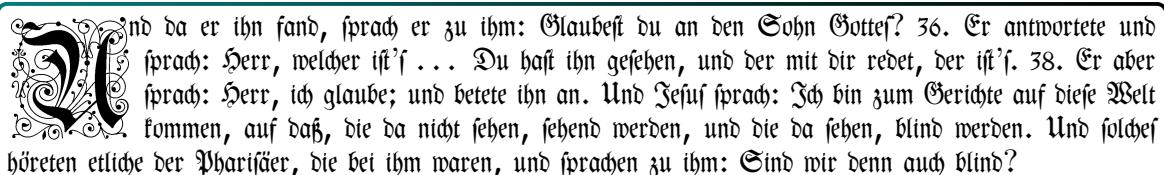
Írja a következőket a preambulumba

</>

```
\usepackage{yfonts,anyfontsize,lettrine}
\setcounter{DefaultLines}{4}
\renewcommand{\LettrineTextFont}{}
\renewcommand{\LettrineFontHook}{\usefont{U}{yinit}{m}{n}}
```

majd a dokumentumtestbe

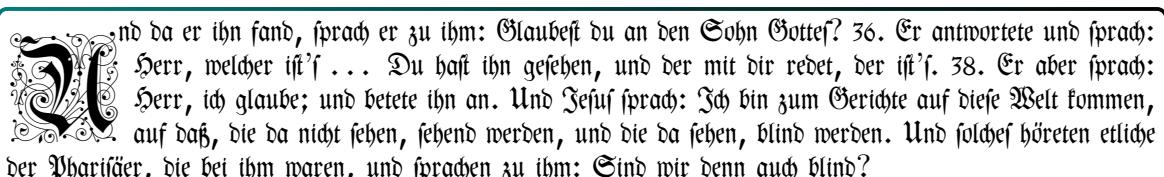
```
{\frakfamily\fraklines\lettrine{U}nd da er ihn fand, sprach er zu ihm:
Glaubest du an den Sohn Gottes? 36.~Er antwortete und sprach:
Herr, welcher ist's \dots Du hast ihn gesehen, und der mit dir
redet, der ist's. 38.~Er aber sprach: Herr, ich glaube; und
betete ihn an. Und Jesus sprach: Ich bin 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
Pharis\"{a}er, die bei ihm waren, und sprachen zu ihm: Sind wir denn
auch blind?\par}
```

And 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 *yinit* fontcsalád csak bittérképes (nem skálázható) verzióban érhető el, így az eredmény kinagyítva recés szélű iniciálét eredményez. Amennyiben *pdflatex* helyett *xelatex* vagy *lualatex* fordítót használ (lásd a 25. fejezetet), akkor elérhető a vektorgrafikus (skálázható) verzió is:

</>

```
\documentclass[a4paper,12pt]{article}
\usepackage[german]{babel}
\usepackage{fontspec}
\usepackage{yfonts,lettrine}
\setcounter{DefaultLines}{4}
\renewcommand{\LettrineTextFont}{}
\newfontface\Yinit{Yinit}
\renewcommand{\LettrineFontHook}{\Yinit}
\begin{document}
{\frakfamily\fraklines\lettrine{U}nd da er ihn fand, ... \par}
\end{document}
```

And 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?

Ez a vektorgrafikus iniciálé közvetlenül nem érhető el *pdflatex* fordítóval, de ha nem fontként, hanem képként tölti be, akkor megoldható a probléma. Az iniciálé képként történő mentéséhez készítse el a következő például *yinit-U.tex* néven:

```
\documentclass[border={-.28pt .05pt .05pt .05pt}]{standalone}
\usepackage{fontspec}
```

```
\newfontface\Yinit{Yinit}
\begin{document}
\Yinit U
\end{document}
```

Fordítsa le a fájlt a következő parancssorral

```
lualatex yinit-U.tex
```

vagy TeXstudióban [Eszközök] > [Parancsok] > [LuaLaTeX]. A kapott yinit-U.pdf fájlt tegye a forrásfájl mappájába, majd az alszakasz elején mutatott példában a \lettrine{U} helyére írja a következőt:

```
\lettrine[image]{yinit-U}
```

16.5. Betűk kontúrozása és árnyékolása

A betűk kontúrozására a `contour` csomag használható `outline` opcióval. Ez automatikusan betölti a `xcolor` csomagot is, ami az `xcolor`-nak egy kisebb tudású verziója. Ha ki akarja használni az `xcolor` lehetőségeit, akkor azt is töltse be. A

```
\contourlength{<méret>} ∈ contour
```

parancsal 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 árnyékolásához használja a

```
\shadowtext{<szöveg>} ∈ shadowtext
```

parancsot. Például

```
</> \shadowtext{Árnyékolt szöveg}
```

A korábban ismertetett `xcolor` csomaggal az árnyék színe is beállítható:

```
\shadowcolor{<szín>} ∈ shadowtext
```

Például

```
</> \shadowcolor{blue!40!white}
\shadowtext{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

```
</> \shadowoffx{4pt}
\shadowoffy{2pt}
\shadowtext{Árnyékolt szöveg}
```

Árnyékolt szöveg

16.6. Alá- és föléhúzás egyszerre

```
</> \overunderline{Egy érdekes kiemelés}
```

Egy érdekes kiemelés

Az `\overunderline` parancs alapból 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}%
\setlength{\ruleht}{0.4pt}%
\setlength{\rulesep}{1.2pt}%
\rlap{\rlap{\rule[-\rulesep-\ruleht]{\wd1}{\ruleht}}%
\rule[\ht2+\rulesep]{\wd1}{\ruleht}}%
\contour{white}{\copy1}%
\end{group}
}
```

16.7. Díszítő elemek

A `pgfornament` csomag sok olyan díszítő elemet tartalmaz, amit az oldalak illetve szövegrészletek díszítésére használhatunk. Itt csak egy példát mutatunk, a csomag útmutatójában részletes leírást és sok további példát találhat.

```
</> \begin{center}
\pgfornament[ydelta=-.5em,color=cyan]{97}
{\Large\bfseries Cím}
\pgfornament[symmetry=v,ydelta=-.5em,color=cyan]{97}
\end{center}
```



A pgfornament csomag v1.3-nál korábbi verzióiban (2024. augusztus 14. előtti telepítés) a \pgfornament parancs egy felesleges szöközt generál a díszítés után. Ez korrigálható a következő kóddal:

```
\AddToHook{cmd/pgf@ornament/after}{\unskip}
```

16.8. 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\vsiz+\baselineskip{%
          \hrule
          \leaders\vbox to\baselineskip{\hrule width\hsize\vfill}\vfill
        }%
      }%
    }%
  }%
}
```

16.9. 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}
}
```

16.10. Az oldal két pontjának összekötése vonallal

Ehhez töltse 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{\langle név\rangle}{\langle szöveg\rangle}
\Draw[\langle nyíl típusa\rangle]{\langle vonal alakja\rangle}{\langle név1\rangle}{\langle név2\rangle}
```

A `\Node` kijelöli a pontokat, a `\Draw` pedig összekötő őket. A `\langle vonal alakja\rangle` 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=\langle szög\rangle, in=\langle szög2\rangle]` Íves vonal, amely `\langle szög1\rangle` fokos szögben indul és `\langle szög2\rangle` 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 `\langle nyíl típusa\rangle` opciók (alapértelmezésben nincs nyíl, csak vonal):

`->` A nyíl `\langle név2\rangle` felé mutat.

`<-` A nyíl `\langle név1\rangle` felé mutat.

<-> A nyíl *(név1)* és *(név2)* felé is mutat.
color=(szín) A vonal színe.
>=(nyílvég) A nyílvég alakja. (Pl. *>=latex*, *>=stealth*, stb. Bővebben láasd 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.

16.11. 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
\makeatletter
\def\hourminute{\number\hour:\two@digits{\minute}}
\makeatother
```

Tegyük fel, hogy a dokumentum fordításának időpontja 463 perc, azaz 7 óra 43 perc. Ekkor

```
463 \number\time
7 \number\hour
43 \number\minute
7:43 \hourminute
```

Az előbb definiált *\hourminute* parancs helyett használható a következő is:

```
\currenttime ∈ datetime
```

16.12. 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=[SZÖ]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=[SZÖVEG],
        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=[SZÖVEG],
        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}
```



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=[SZÖVEG],
        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.

16.13. 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 elkezdi 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 legszebbek 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 „vak” 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 opcionálisan alapértéke: 1-7.

\lipsum[*szám*] ∈ lipsum

Ekkor a lorem ipsum *szám*-adik 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 L^AT_EX-ben. A csomag betöltése után a `\hulipsum` parancs ugyanúgy használható, mint az előbb ismertetett `\lipsum`. Ha egy komplett strukturált dokumentumot (lásd a 17. fejezetet) szeretne létrehozni címmel, szerzővel, tartalomjegyzékkel, fejezettel, szakasszal, alszakasszal, al-alszakasszal, paragrafussal, alparagrafussal, listákkal, ábrával és irodalomjegyzékkel, mindeneket a *Lórum ipse* szövegével, akkor használja a

\hulipsumdocument \in hulipsum

parancsot.

Az angol verzió a `kantlipsum` csomaggal érhető el. Ekkor a `\kant` parancs használható hasonló opciókkal, mint a `\lipsum`. A `blindtext` angol, német, francia és latin nyelven is ad „vak” szöveget a `babel` csomag opciójának megfelelően, a

\Blindtext [*szám*] ∈ blindtext

parancsal, ahol a ***(szám)*** a megjelenítendő bekezdések száma (alapérték 5). Ezzel a csomaggal is lehet komplett strukturált dokumentumot létrehozni:

\Blinddocument \in blindtext

16.14. Vízjel

Írja a preambulumba a következőket:

A teal-colored icon representing code or programming, consisting of three slanted bars forming a bracket-like shape.

```
\usepackage{draftwatermark,xcolor,anyfontsize}
\SetWatermarkText{\textcolor{blue}{\textit{szöveg}}}
\SetWatermarkColor{\textcolor{red}{\textit{színnév}}}
\SetWatermarkFontSize{\textcolor{green}{\textit{magasság}}}
\SetWatermarkAngle{\textcolor{brown}{\textit{szög}}}
```

Ezután a dokumentum minden oldalának közepén, a háttérben megjelenik a *(szöveg)*, amely *(magasság)* magas, *(színnév)* színű és el van forgatva *(szög)* fokkal. Ugyanerre a hatásra mutatunk még két lehetséges megoldást:

```
\usepackage{eso-pic,tikz,graphicx}
\AddToShipoutPictureBG{%
\begin{tikzpicture}[overlay]
\node[rotate=<szög>,color=<színnév>] at (current page.center)
{\resizebox{!}{<magasság>}{<szöveg>}};
\end{tikzpicture}}}
```

vagy

```
\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}}}}}
```

16.15. 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 ö p q r s t u ú ü ü v z x y}
```

```

◊
a á b
c d e é f g
h i í j k l m n
o ó ö ö p q r s
t u ú ü ü v
z x y
◊
```

A `shapepar` csomag segítségével más formájú bekezdések is készíthetők, sőt magunk is tervezhetünk újakat.

Különleges bekezdések készíthetők a `fancypar` csomaggal is. Példaként megmutatunk egy részletet Fülig Jimmy leveléből az uralkodóhoz (Rejtő Jenő: Piszkos Fred, a kapitány):

```
</> \NotebookPar{%
\usefont{T1}{frc}{m}{sl}
~\\
Igen tiszta kiráj úr, kedves mamája és T. neje őfelsége!\\\\\\
Mai nappal keltfent levelét vettem és kibontám.
Ezzel 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:\\
\hspace*{\fill}az eksz Fülig Jimmy, sajátulag.\}}
```

Igen tisztelet kíráj úr, kedve
 Mai nappal keltfent levelet vettetem é
 felelek tisztelettel érte
 udvarára tele
 talánít, van szerenc
 nekem nagy strapa a tétlenség. Én váro
 ha nem is be
 (...)
 Ezzer zárom soraimat alanti tisztelettel fel
 gi barát
 uralkodó őnac
 Kelt mo
 az eksz Fülig Jimmy, sajátulag.

16.16. 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. 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}
```



Ha az előző kódhoz `pdflatex` fordítót használ, akkor még az `auto-pst-pdf` csomagot is töltse be, továbbá használja a `-shell-escape` kapcsolót a fordításnál. Ha a forrásállomány például a `dokumentum.tex`, akkor parancssorba írja be, hogy

```
pdflatex -shell-escape dokumentum.tex
```

majd `Enter`. Ha kereszthivatkozásokat is használ, akkor célszerűbb a `latexmk` program használata `-shell-escape` kapcsolóval:

```
latexmk -pdf -shell-escape dokumentum
```

majd `Enter`. TeXstudióból történő fordításhoz alkalmazza az 1.10. szakasz 2. pontjának beállítását. Ezután `Eszközök` → `Parancsok` → `Latexmk`.

16.17. QR-kód

QR-kód könnyen generálható a `qrcode` csomaggal:

```
\qrcode[height=<magasság>]{<URL cím>} ∈ qrcode
```

Például

```
</> \qrcode[height=25mm]{https://www.ctan.org}
```



16.18. Dátumtípusok automatikus toldalékolása

Tegyük fel, hogy a dokumentum fordításának dátuma 2025. október 4. Ekkor

2025. október 4-én `\ontoday ∈ [magyar]babel`

2025. október 4-én `\ondatemagyar ∈ [magyar]babel`

2025. október 4-én \emitdate[a+an]{g}{\today} ∈ [magyar]babel
 2025. október 4-e \emitdate[e]{g}{\today} ∈ [magyar]babel

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

16.19. Számok automatikus toldalékolása

Ehhez használja a

\told ∈ [magyar]babel

parancsot. Ha automatikus névelőt is akar elé tenni, akkor pedig az

\atold ∈ [magyar]babel
 \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} eredménye 3 és \ref{eq-c} eredménye 1. Az utolsó sorban nem használható a \told\eqref{eq-c}+at parancs!

16.20. Automatikus magyar határozott névelő

Magyar nyelvű dokumentum készítésekor nem csak kereszthivatkozások (lásd a 6. fejezetben) esetén lehet hasznos a határozott névelő automatizálása. Például formanyomtatványokban bizonyos szavakhoz, kifejezésekhez is célszerű ez az eljárás. Ehhez használja az

\az{\<szó>} ∈ [magyar]babel
 \Az{\<szó>} ∈ [magyar]babel

parancsokat, melyek kiírják a *<szó>*-t és előtte a megfelelő határozott névelőt. Az \Az parancs annyiban különbözik az \az parancstól, hogy a névelő kezdőbetűje nagy. Például

```
\documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\newcommand{\egyetemA}{Eszterházy Károly Katolikus Egyetem}
\newcommand{\egyetemB}{Debreceni Egyetem}
```

```
\begin{document}
\Az{\egyetemA} és \az{\egyetemB} hallgatóinak a száma összesen: \dots
\end{document}
```

Az Eszterházy Károly Katolikus Egyetem és a Debreceni Egyetem hallgatóinak a száma összesen: ...

Megzavarhatja a működést, ha a `\langle szó\rangle`-ban a makrók kifejtése után kapcsos zárójelek vagy kifejthető tokenek vannak. Például helytelen eredményt ad a következő:

```
\documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\usepackage{xcolor}
\newcommand{\egyetem}{Eszterházy Károly Katolikus Egyetem}
\begin{document}
\Az{\textcolor{blue}{\egyetem}} hallgatóinak a száma: \dots % ROSSZ!
\end{document}
```

A Eszterházy Károly Katolikus Egyetem hallgatóinak a száma: ...

Ezen a problémán lehet segíteni például az

```
\az*\langle szó\rangle \in [magyar]babel
\Az*\langle szó\rangle \in [magyar]babel
```

parancsokkal, melyek a `\langle szó\rangle`-nak megfelelő névelőt írják ki (kis illetve nagy kezdőbetűvel) de utána nem jelenik meg a `\langle szó\rangle`. Például

```
\documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\usepackage{xcolor}
\newcommand{\egyetem}{Eszterházy Károly Katolikus Egyetem}
\begin{document}
\Az*\egyetem \textcolor{blue}{\egyetem} hallgatóinak a száma: \dots
\end{document}
```

Az Eszterházy Károly Katolikus Egyetem hallgatóinak a száma: ...

Megemlíjtjük, hogy a `huaz` csomag betöltésével közvetlenül az

```
\Az{\textcolor{blue}{\egyetem}}
```

kód is helyes eredményt ad.

Vannak olyan mássalhangzók, melyek ha magukban állnak, akkor nem „az” hanem „a” névelőt kell előjük tenni. Ezek a következők: f, l, ly, m, n, ny, s, sz, r, x, y. Az `\az` illetve `\Az` parancsok ezt is megoldják:

```
\az{F betű}, \az{fal}
```

az F betű, a fal

UTF-8 kódolású dokumentum esetén, ha egy szó kezdőbetűje f, l, ly, m, n, ny, s, sz, r, x vagy y (azaz az előbb említett kivételes mássalhangzók), ugyanakkor a második

betűje ékezetes (pl. *száz*, *nyúl*, *főnök*), akkor azt `\az` illetve `\Az` parancsba rakva hibás eredményt kapunk. Hasonló a gond azokkal a szavakkal, melyeknek az első betűje ékezetes az nem az előbb említett kivételes mássalhangzók valamelyike követi (pl. *ágy*). Ezt a gondot a `huaz` csomag betöltése megoldja.

A `magyar.ldf` `hunnewlabel=yes` alapértelmezett opciója valósítja meg, hogy római számozás esetén is helyes névelőt kapjunk egy kereszthivatkozásnál. Azonban ekkor az oldalszámozást nem lehet átállítani alfanumerikusra (bár ez egy nagyon ritkán felmerülő igény). Ha mégis szükség van erre, akkor a `defaults=hu-min` opció után töltse be a `hunnewlabel=no` opciót is. Ekkor azonban a római számozás előtti automatikus névelő használat nem működik. Ezt a gondot a `huaz` csomag betöltése megoldja.

16.21. TeX-hel kapcsolatos logók

```
\TeX, \LaTeX, \LaTeXe  
\AmS ∈ amsmath  
\METAFONT, \METAPOST ∈ hvlogos  
\XeTeX, \XeLaTeX, \LuaTeX, \LuaLaTeX ∈ hvlogos  
\BibLaTeX, \LaTeXIII ∈ hvlogos
```

Ha az `amsmath` és `hvlogos` csomagokat együtt használja, akkor az `amsmath` előbb legyen betöltve.

17. 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 tételeszerű bekezdések
- függelék
- bibliográfia
- tárgymutató

17.1. Főcím, címlap, kivonat

A mű címét, szerzőjét és dátumot a következő parancsokkal adhatja meg.

```
\title{<cím>}\author{<szerző>}\date{<dátum>}\maketitle
```

A `\title`, `\author` és `\date` parancsok írhatók preambleumban, de a `\maketitle` csak a dokumentumtestbe. A `<dátum>` alapértéke

```
\today
```

ami a fordításkori aktuális dátumot jelenti. Ezen parancsok argumentumaiba lábjegyzetek is írhatók a

```
\thanks{<szöveg>}
```

parancssal. Ez a `\footnote`-tól függetlenül számos. A `\maketitle` a rendelkezésre álló adatokból elkészíti a címet. Alapesetben a `report` és `book` dokumentumosztályok esetén a cím külön oldalra kerül, míg `article` esetén nem. Amennyiben a `titlepage` opcióval

tölti be az `article` dokumentumosztályt, akkor ebben az esetben is külön oldalra kerül a cím.

Ezután nyithat egy `abstract` környezetet (kvízére a `book` osztályt), melybe a mű rövid kivonatát írhatja.

17.2. 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 fejlécben is más-más címet akar kiírni, akkor használja a 391. vagy a 396. oldalon található megoldások valamelyikét.

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 sorszámat, 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
```

opciónnal 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 ahogyan 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, melyben a fejezetcímét középre igazítjuk. A dokumentumtestbe írja be a következőt:

```
</> \titleformat{\chapter}[display]{\normalfont\bfseries\filcenter}
{\huge\thechapter.\~\chapertitlename}{20pt}{\Huge} \in 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 a dokumentumtestben:

```
\begin{document}
<cím>
\frontmatter
<jegyzékek, előszó, bevezetés>
\mainmatter
<szöveg fő része, függelék>
\backmatter
<bibliográfia, tárgymutató>
\end{document}
```

A magyar tipográfiában `\frontmatter` esetén az oldalsámozás nagy római számokkal történik, míg az angolban kis római számokkal. Ezt a `magyar.1df` nem kezeli. A hibát úgy tudja javítani, hogy a `\frontmatter` után kiadja a

```
\pagenumbering{Roman}
```

parancsot is.

17.3. Fattyúsorok

A tipográfiában a hosszú dokumentumok tördelésének súlyos hibája az úgynevezett fattyúsor. Két előfordulása van:

özvegysor 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.

17.4. Fej- és láblécek

17.4.1. 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{\langle infó1 \rangle}{\langle infó2 \rangle}
```

- lábléc üres
- oldalszám a fejléc külső margójánál
- `\langle infó1 \rangle` a fejlécben
 - egyoldalas szedésnél nincs
 - kétoldalas szedésnél páros oldalon a belső margónál
- `\langle infó2 \rangle` a fejlécben
 - egyoldalas szedésnél minden oldalon a belső margónál
 - kétoldalas szedésnél páratlan oldalon a belső margónál
- `\langle infó1 \rangle` és `\langle infó2 \rangle` bármikor megváltoztatható a `\markboth` parancssal. Külön csak az `\langle infó2 \rangle` is megadható a

```
\markright{\langle infó2 \rangle}
```

parancssal.

Ha egy konkrét oldalra vonatkozóan meg akarja az oldalstílust változtatni, akkor az adott szövegrészhez gépelje be:

```
\thispagestyle{\langle stílus \rangle}
```

ahol a `\langle stílus \rangle`: `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 töltse be az `emptypage` csomagot. Ha mi kényszerítettünk ki üres oldalt, akkor oda írja be a következő parancsot:

```
\thispagestyle{empty}
```

17.4.2. 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”.

```
\leftmark
```

Ennek eredménye az aktuális oldalon és a korábban kiadott

```
\markboth{\langle bal \rangle}{\langle jobb \rangle}
```

parancsok közül az utolsónak a `\langle bal \rangle` argumentuma.

```
\firstleftmark \in extramarks
```

Ennek eredménye az aktuális oldalon kiadott `\markboth{\langle bal \rangle}{\langle jobb \rangle}` parancsok közül az elsőnek a `\langle bal \rangle` argumentuma. Ha nincs ilyen, akkor az utolsónak kiadott ilyen parancs `\langle bal \rangle` argumentuma.

```
\rightmark
```

Ennek eredménye az aktuális oldalon kiadott `\markboth{\langle bal \rangle}{\langle jobb \rangle}` illetve

```
\markright{\langle jobb \rangle}
```

parancsok közül az elsőnek a `\langle jobb \rangle` argumentuma. Ha nincs ilyen, akkor az utolsó ilyen parancs `\langle jobb \rangle` argumentuma.

```
\lastrightmark \in extramarks
```

Ennek eredménye az aktuális oldalon és a korábban kiadott `\markboth{\langle bal \rangle}{\langle jobb \rangle}` illetve `\markright{\langle jobb \rangle}` parancsok közül az utolsónak a `\langle jobb \rangle` argumentuma.

Alapesetben a `\rightmark` és `\leftmark` parancsok az aktuális szint és alszint információit 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	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`

Ennek tartalma kerül a láblécbe egyoldalas nyomtatás esetén minden oldalon, kétoldalas nyomtatás esetén a páratlan oldalon.

`\@evenfoot`

Ennek tartalma kerül a láblécbe kétoldalas nyomtatás esetén a páros oldalon.

`\@oddhead`

Ennek tartalma kerül a fejlécbe egyoldalas nyomtatás esetén minden oldalon, kétoldalas nyomtatás esetén a páratlan oldalon.

`\@evenhead`

Ennek 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 `sajat` nevű stílust, melynek a

`\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 beírva 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` paranccsot. 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\csname MakeUppercase \endcsname\relax#1}}
\newcommand{\HeadRule}[1]
  {\lower-1ex\hbox{\makebox[\textwidth]{#1}}%
   \rlap{\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:

Arab számozás:

```
\pagenumbering{arabic}
```

Kis római számozás:

```
\pagenumbering{roman}
```

Nagy római számozás:

```
\pagenumbering{Roman}
```

Angol ábécé kis betűivel számozódnak:

```
\pagenumbering{alpha}
```

Angol ábécé nagy betűivel számozódnak:

```
\pagenumbering{Alpha}
```

Ezek nemcsak az oldalszámozás stílusát változtatják meg, hanem annak értékét is visszaállítják 1-re. Ha nem akar oldalszámozást, akkor adja ki a következő parancsot:

```
\pagenumbering{gobble}
```

Amikor `report` vagy `book` osztály esetén a `magyar.ldf` fájl `chapternumber=Huordinal` opcióval van betöltve, akkor a fejlecben 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 fejlecben. A következő kód visszaállítja a fejlecben 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 valamelyen információt, akkor a fejlecben rossz címek jelennek meg. Ilyen esetekben a `\leftmark` és `\rightmark` tartalmát át kell definiálni a `\markboth` illetve `\markright` parancsokkal. Például, ha a számozatlan fejezetben nem akar szintinformációt, akkor a `\chapter*{Számozatlan fejezet címe}` után gépelje be a `\markboth{}{}` parancsot.



```
\documentclass{book}
\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}
\markboth{}{}

szöveg\newpage szöveg
\end{document}
```

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
\leftmark	szakasz	fejezet
\rightmark	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 adjon ki a

```
\fancyhf{}
```

parancsot, mely a korábban definiált fej- és lábléc beállításokat törli. Lehetőség van a főszöveget elválasztani egy vonallal, az ún. léniával, a fejléctől és lábléctől. Ezeknek a vonalaknak a vastagságát a következő parancsokkal állíthatja be:

```
\renewcommand{\headrulewidth}{<vastagság>}
```

Fejléc alatti lénia vastagsága. Alapértéke **0.4pt**.

```
\renewcommand{\footrulewidth}{<vastagság>}
```

Lábléc feletti lénia vastagsága. Alapértéke **0pt**.

Egy létező stílust átdefiniálhat, vagy egy újat létrehozhat a következő parancssal:

```
\fancypagestyle{<stílusnév>}{{<stílus>}}
```

A következő kódöt, 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 oldalsámozás. Amikor fejezetkezdő oldalra érünk, akkor a `report` és `book` osztály `plain` stílusra vált, így az oldalsámlámpákerül 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 oldalsámozást szeretne, ahol m az aktuális oldalsámlámpá, n pedig az utolsó oldalé, akkor töltse 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 nemcsak oldalsámlámpák és szintinformációk jeleníthetők meg, hanem saját információk is. Ezek betűtípusait tetszőlegesen beállíthatja. Használhatja a korábban már megismert `\nouppercase{<söveg>}` parancsot is, amit a `fancyhdr` csomag alapból definiál, így ezt nem nekünk kell megtenni, mint tettük ezt korábban.

17.5. Tételszerű bekezdések

Sokszor lehet szükség olyan bekezdések írására, melyeknek típuscímét vagy sorszámat kell adni. Ilyen például a matematikában a tételek, bizonyítás, definíció, vagy a törvénykönyvben a paragrafusok stb. Ezek az ún. tételszerű bekezdések, melyeket a `\newtheorem` parancssal definiált környezetekkel hozhat létre.

```
\newtheorem{<téteknév>}{{<tételecím>}}
\newtheorem{<téteknév>}{{<tételecím>}}[<számláló>]
\newtheorem{<téteknév>}[<együtnév>]{{<tételecím>}}
```

`<téteknév>` Létrejön egy `<téteknév>` környezet és egy `<téteknév>` számláló, mely minden újabb ilyen környezet megnyitásakor növekszik eggyel.

`<tételecím>` Ez lesz a tételszerű bekezdés típuscíme (definíció, megjegyzés stb.). Ezen cím mellett megjelenik a `<téteknév>` számláló aktuális értéke is.

`<számláló>` Egy már korábban definiált számláló, általában valamelyik szint számlálója (`chapter`, `section` stb.). Ennek változásakor a `<téteknév>` nevű számláló lenullázódik. A `<számláló>` és a `<téteknév>` számláló együtt jelenik meg (például 2.1. tétele).

(együtnév) Egy másik térelszerű környezet neve. A *(téternév)* és *(együtnév)* környezetek számlálói együtt fognak növekedni.

A létrehozott térelszerű környezetet az alábbi módon használhatja:

```
\begin{tételnév}[(egyedi cím)]
<A bekezdés szövege>
\end{tételnév}
```

Az *(egyedi cím)* megadása esetén, az a *(tétcím)* után jelenik meg zárójelben. Hivatkozni a térelszerű 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 térelszerű bekezdésekben a cím félkövéren, a szöveg pedig dőlten jelenik meg. A térelszerű bekezdések stílusait magunk is beállíthatjuk az *ntheorem* vagy az *amsthm* csomaggal. Mi most csak az *amsthm* csomaggal foglalkozunk. Ennek használatánál ügyelni kell arra, hogy az *amsmath* illetve *mathtools* csomagok után legyen betölve. A stílus beállítása a következő parancsal lehetséges:

```
\theoremstyle{<stílusnév>} \in 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 parancssal:

```
\newtheoremstyle{<sn>}{<fe>}{{<le>}}{<sz>}{<be>}{<cf>}{<cp>}{<ct>}{<cd>} \in 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 beállítás-hoz 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 téretszerű környezetnek nem akar számozást, akkor használja a következőt:

```
\newtheorem*{<téTELnév>}{<téTELcím>} \in 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 \square 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 \square vagy ■ módon jelölnek, 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.

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}
    ...
\end{array}^{\wedge}\mathrm{i}\pi+1=0.\quad\text{\qedhere}\]
\end{proof}
```

$$e^{i\pi} + 1 = 0.$$

□

Ez a megoldás természetesen nem ad jó eredményt, ha a képlet jobbról számoszt. 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{aligned}
a&=2,\\
b&=3.\quad\text{\qedhere}
\end{aligned}
\end{proof}
```

$$a = 2,$$

$$b = 3.$$

□

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}{|c|c|c|c|}
\hline
a&b&c\\
\hline
d&e&f\\
\hline
\end{array}\qedhere\]
\end{proof}
```

a	b	c
d	e	f



A Q.E.D. jelet átdefiniálhatja például ■ jelre az alábbi módon:

```
\renewcommand{\qedsymbol}{$\blacksquare$}
```

Ha nem akar Q.E.D. jelet a bizonyítások végén, akkor kézenfekvőnek tűnik annak üresre való átdefiniálása:

```
\renewcommand{\qedsymbol}{}
```

Az `amsthm` csomag dokumentációja is ezt javasolja. Azonban ennél a megoldásnál, ha a bizonyítás kiemelt képlettel ér véget, akkor ezután generálódik egy üres sor, ahová a Q.E.D. jel kerülne, melynek eredményeként a bizonyítás után túl nagy függőleges térköz keletkezik. Ezen az sem segít, ha a bizonyítást lezáró képletbe kiteszi a `\qedhere` parancsot, mert ez üres `\qedsymbol` esetén hatástalan. A megoldás, ha a Q.E.D. jel átdefiniálása helyett letiltja annak használatát

```
\renewcommand{\qed}{}{}
```

módon.

A `proof` környezet opcióval is használható. Például

```
</> \begin{tetele}\label{xy}
A tételek szövege.
\end{tetele}
\begin{proof}[\Areft{xy}. ~tételek bizonyítása]
A bizonyítás szövege.
\end{proof}
```

1. tétele. A tételek szövege.

Az 1. tételek bizonyítása. A bizonyítás szövege.



Ha „Bizonyítás” cím helyett például „Megoldás” feliratot akar, akkor használja a

```
\renewcommand{\proofname}{Megoldás}
```

parancsot a dokumentumtestben (különben a `magyar.1df` felülbírálja).

Ha a „*Bizonyítás.*” feliratot például „**Bizonyítás.**” típusúra szeretné átállítani, akkor használja a következő kódot:

```
</> \makeatletter
\xpatchcmd{\proof}{\itshape}{\bfseries}{}{} \in \expandafter\regpatch
\makeatother
```

A már definiált tételszerű környezeteket nem tudja átdefiniálni a `\newtheorem` parancssal, csak akkor, ha előtte kiadja a következő parancsokat:

```
\makeatletter
\RenewCommandCopy{\(t\'eteln\'e)}{\relax}
\RenewCommandCopy{\c@{t\'eteln\'e}}{\relax}
\RenewCommandCopy{\the{t\'eteln\'e}}{\relax}
\makeatother
```

Tételszerű környezet lezárása új bekezdést nyit, függetlenül attól, hogy tett-e üres sort utána vagy sem. Ezt a hatást felülbírálhatja a következő kóddal:

```
\makeatletter
\AddToHook{env/{t\'eteln\'e}/after}{\@doendpe}
\makeatother
```

Ekkor az `\end{t\'eteln\'e}` után üres sort hagyva új bekezdés indul, ellenkező esetben nem. Ha nem akar új bekezdést az `\end{t\'eteln\'e}` után, ha rak utána üres sort, ha nem, akkor a következő kódot használja:

```
\AddToHook{env/{t\'eteln\'e}/after}{\everypar{\setbox0\lastbox\everypar{}}}
```

Ha egy tételszerű környezetet adott jellel szeretne lezárni, úgy mint a bizonyítást a Q.E.D. jellel, akkor tegye a következőt. Például a `tetel` nevű környezet definiálása után írja a preambulumba, hogy

```
</> \AddToHook{env/tetel/end}{\pushQED{\qed}%
\def\qedsymbol{$\blacksquare$}\popQED}
```

vagy az ezzel egyenértékű

```
\AtEndOfEnv{tetel}{$\blacksquare$} \in atendofenv
```

kódot. Ezzel a `tetel` környezetet minden \blacksquare ($\$ \blacksquare \$$) jel fog lezárni. 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.

17.6. Jegyzékek

17.6.1. 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}
```

Ha a `\tableofcontents` parancs ki van adva, akkor fordításnál a szintcímek és a hozzá tartozó oldalszámok egy toc (table of contents) kiterjesztésű fájlba íródnak (a neve és a könyvtára a forrásállományéval egyezik meg). A fordítás végén ennek a fájlnak 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 előfejbe és a tartalomjegyzékbe. Hogy mégis bekerüljön a tartalomjegyzékbe az oldalszámmal együtt, a szintnyitó parancs után gépelje be a következőt:

```
\addcontentsline{toc}{\textcolor{green}{szint}}{\textcolor{green}{cím}}
```

Például

```
\section*{Előszó}
\addcontentsline{toc}{\textcolor{blue}{section}}{Előszó}
```

A `hyperref` csomag használata esetén, ha `\addcontentsline` parancssal írunk a tartalomjegyzékbe, akkor az oldalszám linkje nem fog működni. Ennek javítása az, hogy az `\addcontentsline` előre be kell még írni a `\phantomsection \in hyperref` parancsot is.

Oldalszám nélküli feliratok és parancsok is kiírathatók a tartalomjegyzékbe az

```
\addtocontents{toc}{\textcolor{green}{szöveg}}
```

parancssal. Ha a `\textcolor{green}{szöveg}`-ben van olyan parancs, amit kifejtés nélkül szeretne beírni a toc kiterjesztésű fájlba, akkor tegyen előre egy `\protect` parancsot. Például

```
\addtocontents{toc}{\protect\vspace*{10pt}}
```

Hogy az `\addtocontents` parancsnak hol lesz hatása a tartalomjegyzékben, attól függ, hogy a forrásállományban hol adta 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.

17.6.2. Táblázatok jegyzéke

A táblázatok felirataiból is készíthet jegyzéket, melyben a táblázat címkeszáma, 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}
```

Ha a `\listoftables` parancs ki van adva, akkor 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 forrásállományéval egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik a táblázatok jegyzéke.

A `\caption` illetve `\captionof` parancsokkal megadott feliratok szöntsége 1. Ez azt jelenti, hogy ezek csak akkor jelennek meg a jegyzékben, ha a `tocdepth` számláló értéke legalább 1.

Ha a táblázatok jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lot}{table}{<szöveg>}
```

Ha a `<szöveg>`-ben van olyan parancs, amit kifejtés nélkül szeretne beírni a lot kiterjesztésű fájlba, akkor tegyen előre egy `\protect` parancsot. Oldalszám nélküli információ is kiírhatató 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.

Amennyiben `report` vagy `book` dokumentumosztályt használ, akkor a `\chapter` parancs kiadásakor egy

```
\addtocontents{lot}{\protect\addvspace{10\p@}}
```

parancs is végrehajtódik. Így a jegyzékben az új fejezetek előtt bejegyzése felett lesz egy extra függőleges térköz. Ennek letiltására használja az `etoolbox` csomagot és a következő kódot az első `\chapter` kiadása előtt:

```
\makeatletter
\patchcmd{\@chapter}
  {\addtocontents{lot}{\protect\addvspace{10\p@}}}{\relax}{\relax}
\makeatother
```

Amennyiben a `babel` csomag `magyar` opcióját használja, akkor az előző kód a dokumentumtestben legyen.

Ha a `subcaption` csomaggal alfeliratokat is használ `table` környezetben, akkor ezek is bekerülnek a jegyzékbe, feltéve, hogy a `tocdepth` számláló értéke legalább 2, ugyanis az alfeliratok szöntsége 2. Az alfeliratok jegyzékbe kerülésének másik feltétele, hogy az adott alfeliratra a `subcaption` csomag `list=true` opciója legyen érvényben. Ehhez használja a

```
\captionsetup[sub]{list=true}
```

vagy

```
\captionsetup[subtable]{list=true}
```

kódot. (Első esetben minden típusú alfeliratra, a második esetben csak a táblázatok alfelirataira lesz hatással.)

Ha a táblázatok jegyzékébe akar szöveget írni oldalszámmal az alfeliratok helyére, akkor használja a következő parancsot:

```
\addcontentsline{lot}{subtable}{<szöveg>} \in subcaption
```

17.6.3. 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}
```

Ha a `\listoffigures` parancs ki van adva, akkor 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 forrásállományéval egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik az ábrák jegyzéke.

A `\caption` illetve `\captionof` parancsokkal megadott feliratok szintszáma 1, azaz csak akkor jelennek meg a jegyzékben, ha a `tocdepth` számláló értéke legalább 1.

Ha az ábrák jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lof}{figure}{<szöveg>}
```

Ha a `<szöveg>`-ben van olyan parancs, amit kifejtés nélkül szeretne beírni a lof kiterjesztésű fájlba, akkor tegyen elé egy `\protect` parancsot. Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{lof}{<szöveg>\par}
```

parancsal. Hogy az `\addtocontents` parancsnak hol lesz hatása a jegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

Amennyiben `report` vagy `book` dokumentumosztályt használ, akkor a `\chapter` parancs kiadásakor egy

```
\addtocontents{lof}{\protect\addvspace{10\p@}}
```

parancs is végrehajtódik. Így a jegyzékben az új fejezetek első bejegyzése felett lesz egy extra függőleges térköz. Ennek letiltására használja az `etoolbox` csomagot és a következő kódot az első `\chapter` kiadása előtt:

```
\makeatletter
\patchcmd{\@chapter}
  {\addtocontents{lof}{\protect\addvspace{10\p@}}{}{}{}}
\makeatother
```

Amennyiben a `babel` csomag `magyar` opcióját használja, akkor az előző kód a dokumentumtestben legyen.

Ha a `subcaption` csomaggal alfeliratokat is használ `figure` környezetben, akkor ezek is bekerülnek a jegyzékbe, feltéve, hogy a `tocdepth` számláló értéke legalább 2, ugyanis az alfeliratok szintszáma 2. Az alfeliratok jegyzékbe kerülésének másik feltétele, hogy az adott alfeliratra a `subcaption` csomag `list=true` opciója legyen érvényben. Ehhez használja a

```
\captionsetup[sub]{list=true}
```

vagy

```
\captionsetup[subfigure]{list=true}
```

kódot. (Első esetben minden típusú alfeliratra, a második esetben csak az ábrák alfelirataira lesz hatással.)

Ha az ábrák jegyzékébe akar szöveget írni oldalszámmal az alfeliratok helyére, akkor használja a következő parancsot:

```
\addcontentsline{lof}{subfigure}{\textcolor{green}{<szöveg>}} \textcolor{red}{in} subcaption
```

17.6.4. 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. Ekkor fordításnál a számozott kódok felirata az aktuális oldalszámmal kiíródik egy `lol` (list of `listings`) kiterjesztésű fájlba (a neve és a könyvtára a forrásállományéval egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik a kódok jegyzéke.

A kódfeliratok szintszáma 1, vagyis ezek csak akkor jelennek meg a jegyzékben, ha a `tocdepth` számláló értéke legalább 1.

Ha a kódok jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lol}{lstlisting}{\textcolor{green}{<szöveg>}}
```

Ha a `<szöveg>`-ben van olyan parancs, amit kifejtés nélkül szeretne beírni a `lol` kiterjesztésű fájlba, akkor tegyen elő egy `\protect` parancsot. Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{lol}{\textcolor{green}{<szöveg>}\par}
```

parancssal. Hogy az `\addtocontents` parancsnak hol lesz hatása a jegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

Ha a `minted` csomaggal készített kódlistákat, és a feliratozásukhoz `newfloat` opciót használt, akkor pontosan azt kell tenni a jegyzék elkészítéséhez, mint a 17.6.5. alszakaszban, csak a `<környezet>` helyére `listing` kerül.

17.6.5. Saját úsztatott objektumok jegyzéke

Láttuk a 11.5. szakaszban, hogy a `caption` vagy `newfloat` csomaggal saját úsztató környezetet is létrehozhatunk. Az ilyen `<környezet>` nevű környezettel úsztatott objektumkból is készíthet jegyzéket a

```
\listof{környezet}s
```

parancssal. Ennek a jegyzéknek a címét a következő parancs tárolja:

```
\list{környezet}name
```

Ha a `\listof{környezet}s` parancs ki van adva, akkor 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 forrásállományéval egyezik meg). A fordítás végén ennek a fájlnak a segítségével ténylegesen megjelenik a jegyzék.

A `\caption` illetve `\captionof` parancsokkal megadott feliratok szintszáma 1. Ez azt jelenti, hogy ezek csak akkor jelennek meg a jegyzékben, ha a `tocdepth` számláló értéke legalább 1.

Ha az úsztatott objektum jegyzékébe akar szöveget írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{lo<környezet>}{<környezet>}{<szöveg>}
```

Ha a *<szöveg>*-ben van olyan parancs, amit kifejtés nélkül szeretne beírni a lo*<környezet>* kiterjesztésű fájlba, akkor tegyen elé egy `\protect` parancsot. Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{lo<környezet>}{<szöveg>\par}
```

parancssal. 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}
```

Ha a `subcaption` csomaggal alfeliratokat is használ *<környezet>* környezetben, akkor ezek is bekerülnek a jegyzékbe, feltéve, hogy a `tocdepth` számláló értéke legalább 2, ugyanis az alfeliratok szintszáma 2. Az alfeliratok jegyzékbe kerülésének másik feltétele, hogy az adott alfeliratra a `subcaption` csomag `list=true` opciója legyen érvényben. Ehhez használja a

```
\captionsetup[sub]{list=true}
```

vagy

```
\captionsetup[sub<környezet>]{list=true}
```

kódot. (Első esetben minden típusú alfeliratra, a második esetben csak a *<környezet>* alfelirataira lesz hatással.)

Ha az úsztatott objektum jegyzékébe akar szöveget írni oldalszámmal az alfeliratok helyére, akkor használja a következő parancsot:

```
\addcontentsline{lo<környezet>}{sub<környezet>}{<szöveg>} ∈ subcaption
```

17.6.6. Új jegyzék készítése

Az eddigiek kívül lehetőség van egyéb számosztott elemekből is jegyzéket készíteni, mint például egyenletekből. Ehhez töltse be a `tocloft` csomagot, majd használja a következő kódot:

```
\newlistof{stílus}{kiterjesztés}{jegyzékcím} ∈ tocloft
```

Ezután a

```
\listof{stílus} ∈ tocloft
```

kiírja a *<jegyzékcím>* című jegyzéket. A *<jegyzékcím>* kiírása a `tocloft` csomag esetében nem a L^AT_EX alapértelmezett metódusával történik, azaz például `article` dokumentumosztály esetén nem a `\section*{<jegyzékcím>}` módon. Amennyiben mégis a L^AT_EX alapértelmezett metódusát szeretné, akkor használja a `tocloft` csomag `titles` opcióját.

Az előző módon definiált jegyzékbe új *<cím>* bejegyzést a következő parancssal vihetünk be:

```
\addcontentsline{\kiterjesztés}{\stílus}
{\protect\numberline{\the\számláló}\cím}
```

Ezután, amennyiben a `\listof{stílus}` parancs ki van adva, fordításnál a `\számláló` nevű számláló értéke, a `\cím` és az aktuális oldalszám egy `\kiterjesztés` kiterjesztésű fájlba íródik, majd a következő fordításnál megjelenik a bejegyzés a jegyzékben (ahol a `\listof{stílus}` parancsot kiadtuk).

Ügyeljen arra, hogy az előző kódban a `\cím` előtt ne legyen szököz. A `\számláló` annak a strukturális elemnek a számlálója, amire vonatkozik jegyzék. Például egyenletek esetén `equation` vagy egy `lemma` környezettel megadott tételszerű bekezdés esetén `lemma`.

Az így megadott bejegyzés szintszáma 1. Ez azt jelenti, hogy ezek csak akkor jelennek meg a jegyzékben, ha a `tocdepth` számláló értéke legalább 1.

Ha a jegyzékbe szöveget akar írni oldalszámmal, akkor használja a következő parancsot:

```
\addcontentsline{\kiterjesztés}{\stílus}{\szöveg}
```

Ha a `\szöveg`-ben van olyan parancs, amit kifejtés nélkül szeretne beírni a `\kiterjesztés` kiterjesztésű fájlba, akkor tegyen elé egy `\protect` parancsot. Oldalszám nélküli információ is kiíratható a jegyzékbe az

```
\addtocontents{\kiterjesztés}{\szöveg\par}
```

paranccsal. Hogy az `\addtocontents` parancsnak hol lesz hatása a jegyzékben, az dönti el, hogy a forrásállományban hol adtuk ki.

17.6.7. Jegyzékek stílusának testreszabása a `tocloft` csomaggal

A stílusbeállításokra több lehetőséget biztosít a `tocloft` csomag a tartalomjegyzék, táblázatok jegyzéke, ábrák jegyzéke és a `\newlistof` parancs által definiált jegyzékek (lásd a 17.6.6. alszakaszt) esetén. Sajnos a táblázatok illetve ábrák jegyzékében a `subcaption` csomag által kezelt alfeliratok stílusát nem támogatja. Szintén nem támogatja a `listings`, `caption` és `newfloat` csomagokkal generált jegyzékeket.

A jegyzékek címének kiírása a `tocloft` csomag esetében nem a L^AT_EX alapértelmezett metódusával történik, azaz például `article` dokumentumosztályban a tartalomjegyzék címe nem a

```
\section*\{\contentsname}
```

paranccsal jelenik meg. Visszatérni a L^AT_EX alapértelmezett módszerére a `tocloft` csomag `titles` opciójával lehet.

Alapesetben minden jegyzéknek a `tocdepth` számláló értéke adja meg a szintmélységét. A `tocloft` használatával viszont a különböző jegyzékek szintmélységét más-más számláló tárolja, nevezetesen a

```
\kiterjesztés\depth
```

ahol a `\kiterjesztés` az adott jegyzékhez tartozó fájl kiterjesztése, ahová a bejegyzések kerülnek, azaz `toc`, `lot`, `lof` vagy a `\newlistof` parancs által definiált `\kiterjesztés` (lásd a 17.6.6. alszakaszt). Ezeknek a számlálóknak a `tocdepth` kivitelével az alapértéke 1. A `\kiterjesztés\depth` számláló értéke a következő kóddal állítható át `\szám`-ra:

```
\setcounter{\kiterjesztés\depth}{\szám}
```

Például abban az esetben, ha a `subcaption` csomaggal az alfeliratokat is szeretné beírni az ábrák jegyzékébe, akkor a

```
\setcounter{lofdepth}{2}
```

parancssal az ábrák jegyzékében a szintmélységet 2-re kell állítani, mert az alfeliratok színtszáma 2. Ne feledje el, hogy az alfeliratok bekerülésének az ábrák jegyzékébe további feltétele, hogy az adott alfeliratra a `subcaption` csomag `list=true` opciója legyen érvényben (lásd a 17.6.3. alszakaszban).

```
\def\cftdot{\langle jel\rangle} \in tocloft
```

Sok esetben a bejegyzés végét és az oldalszámot egy pontsor köti össze. Ez az ún. *vezető*. Ha pontok helyett `\langle jel\rangle`-et akar, akkor használja az előző kódot. Például

```
\def\cftdot{\$\\star\$}
```

esetén a pontok helyett `\star` jelekből álló vezetőt kapunk.

```
\def\cftdotsep{\langle valós szám\rangle} \in tocloft
```

A vezető sűrűségét adja meg (alapértéke 4.5). Minél kisebb a `\langle valós szám\rangle` értéke, annál sűrűbb lesz. Például

```
\def\cftdotsep{2}
```

esetén az alapértelmezett sűrűbb vezetőt kapunk. Amennyiben a `\langle valós szám\rangle` nagyon nagy, pl. 5000, akkor eltűnik a vezető.

```
\cftsetpnumwidth{\szélesség} \in tocloft
```

A `\langle szélesség\rangle` az oldalszámoknak fenntartott doboz szélessége. Például

```
\cftsetpnumwidth{1cm}
```

esetén az oldalszámok 1 cm széles dobozban lesznek elhelyezve.

```
\def\cftpnumalign{\igazítás} \in tocloft
```

Ezzel lehet beállítani, hogy az oldalszámok a részükre fenntartott dobozban hová legyenek igazítva. Lehetséges értékei az `\langle igazítás\rangle`-nak: `l` (balra igazít), `c` (középre igazít), `r` (jobbra igazít, ez az alapérték).

```
\cftsetrmarg{\hossz} \in tocloft
```

Ekkor a bejegyzések szövegének jobb széle `\langle hossz\rangle` távolságra lesz a szövegtükör jobb szélétől. A bejegyzés végét és az oldalszámot összekötő jelsorra ez nem vonatkozik, mert az minden az oldalszám dobozának jobb széléig húzódik. Például

```
\cftsetrmarg{2cm}
```

esetén ez a távolság 2 cm.

A következőkben ismertetett parancsok neveiben használni fogjuk a `\langle típus\rangle` jelölést, ami majd az adott parancs hatókörére utal. Először felsoroljuk a lehetséges `\langle típus\rangle` értékeket:

- `part` (rész)
- `chap` (fejezet)
- `sec` (szakasz)
- `subsec` (alszakasz)
- `subsubsec` (al-alszakasz)
- `para` (paragrafus)
- `subpara` (alparagrafus)
- `tab` (táblázat főfelirata)
- `fig` (ábra főfelirata)
- `\stílus` (ami a `\newlistof` parancssal van definiálva, ahogy a 17.6.6. alszakaszban)

```
\setlength{\cftbefore{típus}{skip}}{hossz} ∈ tocloft
```

A *típus* típusú bejegyzések előtt *hossz* nagyságú függőleges térköz lesz. Például

```
\setlength{\cftbeforesecskip}{5pt}
```

esetén minden szakasz címe előtt egy 5 pt nagyságú függőleges térköz lesz a tartalomjegyzékben.

```
\setlength{\cft{típus}{indent}}{hossz} ∈ tocloft
```

A *típus* típusú bejegyzések *hossz* nagyságú behúzással kezdődnek. Például

```
\setlength{\cftchapindent}{1cm}
```

esetén a tartalomjegyzékben a fejezetek címei 1 cm behúzással kezdődnek.

```
\setlength{\cft{típus}{numwidth}}{hossz} ∈ tocloft
```

A bejegyzés számozása egy dobozban van elhelyezve, melynek a bal oldalára van igazítva a szám. Ez a parancs a *típus* típusú bejegyzések esetén ennek a doboznak a szélességét *hossz* értékre állítja be. Ha nincs számozás, akkor ez a doboz nem foglal helyet. Ha a *típus* típusú bejegyzés olyan hosszú, hogy sort kell törni, akkor a második és az azt követő sorokat az első sorhoz képest *hossz* nagyságú behúzással kezdi. Például

```
\setlength{\cftfignumwidth}{1cm}
```

esetén ez az érték 1 cm lesz az ábrák főfeliratainál az ábrák jegyzékében.

```
\def\cft{típus}{font}{fonttípus} ∈ tocloft
```

A *típus* típusú bejegyzések fonttípusa *fonttípus* lesz, az oldalszámok kivételével. Például

```
\def\cftsubsecfont{\small\itshape}
```

esetén minden alszakasz címe \small méretben dőlt betűkkel jelenik meg (kivéve a hozzátartozó oldalszámot) a tartalomjegyzékben.

```
\def\cft{típus}{pagefont}{fonttípus} ∈ tocloft
```

A *típus* típusú bejegyzésekhez tartozó oldalszámok fonttípusa *fonttípus* lesz. Például

```
\def\cftsubsecpagefont{\small\itshape}
```

esetén minden alszakasz címéhez tartozó oldalszám \small méretben dőlt betűvel jelenik meg a tartalomjegyzékben.

```
\cftpagenumbersoff{típus} ∈ tocloft
```

A *típus* típusú bejegyzésekhez nem lesznek kiírva az oldalszámok és a vezető.

```
\cftpagenumberson{típus} ∈ tocloft
```

A *típus* típusú bejegyzésekhez ki lesznek írva az oldalszámok.

```
\def\cft{típus}{dotsep}{\cftnodots} ∈ tocloft
```

A *típus* típusú bejegyzésekhez nem lesz vezető. Például

```
\def\cftsecdotsep{\cftnodots}
```

esetén minden szakasz címénél eltűnik a vezető a tartalomjegyzékben.

```
\def\cft{típus}{dotsep}{\cftdotsep} ∈ tocloft
```

A *típus* típusú bejegyzésekhez lesz vezető a \cftdot-ban tárolt jellel és a \cftdotsepben tárolt sűrűséggel. Például

```
\def\cftchapdotsep{\cftdotsep}
```

esetén minden fejezet címénél lesz vezető a tartalomjegyzékben.

```
\def\cft<tipus>dotsep{\valós szám} ∈ tocloft
```

A *<tipus>* típusú bejegyzésekhez lesz vezető a \cftdot-ban tárolt jellel és *(valós szám)* sűrűséggel. Például

```
\def\cftchapdotsep{6.6}
```

esetén minden fejezet címénél lesz vezető a tartalomjegyzékben 6.6 értékű sűrűséggel.

```
\def\cft<tipus>aftersnumb{\kód} ∈ tocloft
```

A *<tipus>* típusú bejegyzés számozása egy \cft*<tipus>*numwidth széles dobozban van elhelyezve. Ennek a parancsnak a hatására ezen dobozok után kifejtődik a *(kód)*. Ez a parancs hatástalan, ha a *(tipus)* értéke part. (Magyar nyelvű dokumentumokra további információk találhatók erről a parancsról a következő al-alszakaszban.) Például

```
\def\cftchapaftersnumb{\color{red}}
```

esetén minden fejezet címe piros lesz a tartalomjegyzékben, de előtte a számozás nem.

```
\def\cft<tipus>aftersnum{\kód} ∈ tocloft
```

A *<tipus>* típusú bejegyzések számozásának fenntartott dobozokban ennek a parancsnak a hatására a számozás kiírása után kifejtődik a

```
(kód)\hfil
```

kód. A \hfil miatt lesz a számozás balra igazítva a dobozban. Ez a parancs hatástalan, ha a *(tipus)* értéke part. (Magyar nyelvű dokumentumokra további információk találhatók erről a parancsról a következő al-alszakaszban.) Például

```
\def\cftchapaftersnum{.}
```

esetén minden fejezet számozása után közvetlenül lesz egy pont a tartalomjegyzékben.

```
\def\cft<tipus>presnum{\kód} ∈ tocloft
```

A *<tipus>* típusú bejegyzések számozásának fenntartott dobozokban ennek a parancsnak a hatására a számozás kiírása előtt kifejtődik a *(kód)*. Ez a parancs hatástalan, ha a *(tipus)* értéke part. (Magyar nyelvű dokumentumokra további információk találhatók erről a parancsról a következő al-alszakaszban.) Például

```
\def\cftchappresnum{\hfil}
```

esetén minden fejezet számozása a számára fenntartott doboz közepére lesz illesztve a tartalomjegyzékben, hiszen a doboz végén is van egy \hfil parancs. Mivel a \hfill erősebb a \hfil parancsnál, ezért

```
\def\cftchappresnum{\hfill}
```

esetén minden fejezet számozása a számára fenntartott doboz jobb oldalához lesz illesztve.

A tocloft csomag használata magyar nyelvű dokumentumokban

A következő parancsok alapesetben nem működnek magyar nyelvű dokumentumokban:

```
\cft<tipus>aftersnumb
\cft<tipus>aftersnum
\cft<tipus>presnum
```

Ennek az az oka, hogy a tocloft ezeknek a parancsoknak a használatához átdefiniálja a \numberline parancsot. Ugyanakkor a babel csomag magyar opciónja (azaz a

`magyar.1df`) szintén átdefiniálja a `\numberline` parancsot annak érdekében, hogy a számozások után a jegyzékekben legyen pont, ahogyan azt a magyar tipográfia megköveteli. Mivel a `magyar.1df` későbbre időzíti a `\numberline` átdefiniálását, mint a `tocloft`, ezért az előző három parancs hatástalan lesz. A probléma egy lehetséges megoldásaként írja be a következő kódot:

```
\makeatletter
\newcommand\hudot{.}
\def\magyar@numberline#1\vfuzz{%
  \hb@xt@{\@tempdima{@cftbsnum #1\@cftasnum\hudot\hfil}\@cftasnumb}
\makeatother
```

Ez a `magyar.1df` átdefiniálási mechanizmusát úgy alakítja át, hogy a számozások utáni pont kirakását megtartja, de az előző három parancsot elérhetővé teszi. Ezután a

```
\cft<tipus>aftersnum
\cft<tipus>presnum
```

parancsok pontosan úgy használhatóak, mint ahogyan azt korábban leírtuk. A harmadik parancs esetén a használata kiegészül az előző al-alszakaszban írtakhoz képest:

```
\def\cft<tipus>aftersnum{<kód>}
```

A `<tipus>` típusú bejegyzések számozásának fenntartott dobozokban ennek a parancsnak a hatására a számozás kiírása után kifejtődik a

```
<kód>\hudot\hfil
```

kód. Tehát például

```
\def\cftchapaftersnum{}
```

esetén minden fejezet számozása után lesz egy) jel és utána egy pont a tartalomjegyzékben.

```
\def\cft<tipus>aftersnum{<kód>\def\hudot{}}
```

vagy az ezzel egyenértékű

```
\def\cft<tipus>aftersnum#1{<kód>}
```

hatására a `<tipus>` típusú bejegyzések számozásának fenntartott dobozokban a számozás kiírása után kifejtődik a

```
<kód>\hfil
```

kód. Tehát például

```
\def\cftchapaftersnum{}\def\hudot{}}
```

vagy

```
\def\cftchapaftersnum#1{}}
```

esetén minden fejezet számozása után lesz egy) jel a tartalomjegyzékben. Természetesen ekkor a) jel után már nem lesz pont.

17.6.8. Jegyzékek készítésének elvi alapjai

Ebben az alszakaszban megvizsgáljuk, hogy amikor egy jegyzéket készítünk a korábban ismertetett módokon, akkor a háttérben hogyan valósul ez meg. Ennek ismerete lehetőséget ad arra, hogy magunk is készíthessünk tetszőleges jegyzékeket tetszőleges stílusban. Tekintsük az ehhez szükséges parancsokat.

\@starttoc{\langle kiterjesztés \rangle}

Ez először megvizsgálja, hogy létezik-e a `\jobname.\langle kiterjesztés \rangle` fájl, ahol `\jobname` a forrásfájl neve tex kiterjesztés nélkül. Ha létezik, akkor betölti és kifejti annak tartalmát, ha nem létezik, akkor pedig létrehozza azt. Mindezek után megnyitja írásra.

Ennek segítségével lehet elkészíteni a jegyzék megjelenítéséhez szükséges parancsot. Például, ha egy `dokumentum.tex` fájlban a `\tableofcontents` parancsot használja az `article` dokumentumosztály esetén, akkor a következők történnek:

1. Először kiadja a `\section*\{\contentsname\}` parancsot, ami kiírja a tartalomjegyzék címét, majd gondoskodik a megfelelő fejlécről, végül kiadja a `\@starttoc{toc}` parancsot.
2. Ez utóbbi először létrehozza a `dokumentum.toc` nevű fájlt és megnyitja azt írásra. A forrásfájl további részeiben a később ismertetett módon a `dokumentum.toc` fájlba bekerülnek a tartalomjegyzék elkészítéséhez szükséges parancsok és adatok.
3. A következő fordításnál, amikor a `\@starttoc{toc}` kerül kifejtésre, akkor az betölti a `dokumentum.toc` fájlt és kifejti annak tartalmát. Ha ekkor a tartalomjegyzék többoldalas, akkor az első fordításnál megállapított oldalszámok már nem lesznek feltétlenül megfelelők. Ugyanakkor ebben a fordítási folyamatban már az új oldalszámokkal kerülnek be az adatok a `dokumentum.toc` fájlba, így vélhetőleg a következő fordításnál már helyes oldalszámok jelennek meg. Ha mégsem, akkor további fordításokra van szükség. Amennyiben a `latexmk` programot használja fordításra, akkor az automatikusan gondoskodik a megfelelő számú fordításról.

A `\langle kiterjesztés \rangle` lehetséges értékei: `toc` (tartalomjegyzék), `lot` (táblázatok jegyzéke), `lof` (ábrák jegyzéke), `lol` (kódok jegyzéke `listings` csomaggal), `lo\langle környezet \rangle` (lásd a 17.6.5. alszakaszat). Új jegyzék készítése esetén (lásd a 17.6.6. alszakaszat) a `\newlistof` parancssal van definiálva. De tetszőlegesen is megadhatunk egy új jegyzékhez tartozó `\langle kiterjesztés \rangle` kiterjesztést.

\addtocontents{\langle kiterjesztés \rangle}{\langle bejegyzés \rangle}

Ennek hatására, ha ki van adva a `\@starttoc{\langle kiterjesztés \rangle}` és így meg van nyitva írásra a `\jobname.\langle kiterjesztés \rangle` fájl, akkor ebbe a fájlba egy új sorba beírja a `\langle bejegyzés \rangle`-t kifejtve. Ha a `\langle bejegyzés \rangle`-ben valamit kifejtetlenül szeretne beírni a fájlba, akkor az elő kell tenni a `\protect` parancsot.

\addcontentsline{\langle kiterjesztés \rangle}{\langle stílus \rangle}{\langle bejegyzés \rangle}

Ez a következővel egyenértékű:

```
\addtocontents{\langle kiterjesztés \rangle}{%
    \protect\contentsline{\langle stílus \rangle}{\langle bejegyzés \rangle}{\thepage}{}{}}
```

Vagyis, ha ki van adva a `\@starttoc{\langle kiterjesztés \rangle}` és így meg van nyitva írásra a `\jobname.\langle kiterjesztés \rangle` fájl, akkor ebbe a fájlba egy új sorba beírja a következőt:

```
\contentsline{\langle stílus \rangle}{\langle bejegyzés \rangle}{\thepage}{\kifejtve}{}{}
```

ahol a `\thepage` kifejtése a megfelelő oldalszám. Az előbbi kódban a `\langle bejegyzés \rangle` standard (de nem kötelező) alakja

```
\protect\numberline{\the\langle számláló \rangle}{\langle szöveg \rangle}
```

ahol a `\langle szöveg \rangle` előtt tilos szóközt rakni. Például, amikor kiadja a

```
\section{\langle cím \rangle}
```

parancsot, akkor az egyúttal kifejti az

```
\addcontentsline{toc}{section}
```

```
\protect\numberline{\thesection}{cím}
```

kódot is, melynek hatására, ha a `\tableofcontents` parancs ki van adva, és így meg van nyitva írásra a `\jobname.toc` fájl, akkor ebbe a fájlba egy új sorba beírja a következőt:

```
\contentsline{section}{\numberline{\thesection kifejtve}{cím} kifejtve}{\thepage kifejtve}{}{}
```

ahol a `\thesection` kifejtése a megfelelő szakaszszám. Másik példaként tekintsük a `figure` környezetben kiadott

```
\caption{felirat}
```

parancsot, ami kifejti az

```
\addcontentsline{lof}{caption}{\protect\numberline{\thecaption}\protect\ignorespaces felirat} kifejtve}{\thepage kifejtve}{}{}
```

kódot is, melynek hatására, ha a `\listoffigures` parancs ki van adva, és így meg van nyitva írásra a `\jobname.lof` fájl, akkor ebbe a fájlba egy új sorba beírja a következőt:

```
\contentsline{caption}{\numberline{\thecaption kifejtve}\ignorespaces felirat} kifejtve}{\thepage kifejtve}{}{}
```

ahol a `\thecaption` kifejtése a megfelelő ábraszám. Az `\ignorespaces` parancsnak az a szerepe, hogy a `(felirat)` kifejtése előtti szóközt elnyelje, így a `\caption{Felirat}` és `\caption{ Felirat}` ugyanazt fejtik ki az ábrák jegyzékében. Mivel a `\section` esetében nem volt a kódban `\ignorespaces`, ezért például

```
\section{ Szakaszcím}
```

esetén lesz egy extra szóköz a tartalomjegyzékben a „Szakaszcím” előtt.

```
\numberline{szám}
```

Ennek hatása a következővel egyezik meg:

```
\hb@xt@{\tempdima{szám}\hfil}
```

Ez egy `\tempdima` szélességű dobozba helyezi a `(szám)`-ot balra igazítva a `\hfil` parancssal. A `\tempdima` értéke a `(stílus)` definíciójában van megadva (lásd később).

A `magyar.lfd defaults=hu-min` opciója átdefiniálja a standard `\numberline` parancsot úgy, hogy amennyiben az argumentuma nem pontra végződik, akkor még egy pontot is rak a végére, ezzel biztosítva azt a magyar szabályt, hogy sorszám után pont kell. Tehát ekkor a `\numberline{1}` és `\numberline{1.}` kimenete is „1.” lesz.

```
\contentsline{stílus}{bejegyzés} kifejtve}{\thepage kifejtve}{leírás}
```

Ennek hatása a következő:

```
\gdef\contentsline@destination{leírás}%
\l@stílus{bejegyzés} kifejtve}{\thepage kifejtve}
```

A `(leírás)`-nak csak a `hyperref` csomagban van szerepe, ezen kívül általában üres.

Az eddigieket összegezve tehát, ha például kiadja a

```
\section{cím}
```

parancsot, akkor amennyiben a `\tableofcontents` parancs is ki van adva, és így bekerül a bejegyzés a `\jobname.toc` fájlba, akkor ennek beolvasásakor a következő kerül kifejtésre:

```
\l@section{\numberline{\thesection kifejtve}{cím} kifejtve}{\thepage kifejtve}
```

Mindezek alapján, hogy adott `\stílus` nevű stílusban megadott bejegyzések hogyan nézzenek ki a végeredményben, az

```
\l@{stílus}
```

parancsban van meghatározva. Ez a parancs a következő `\stílus` értékek esetén van definiálva alapesetben:

- `part`, `chapter`, `section`, `subsection`, `subsubsection`, `paragraph`, `subparagraph`
- `table`, `figure`
- `subtable`, `subfigure` ∈ `subcaption`
- `lstlisting` ∈ `listings`
- Ha a `caption` vagy `newfloat` csomaggal definiálunk egy `\környezet` nevű úsztató környezetet, akkor egyúttal egy `\környezet` nevű `\stílus` értéket is definiálunk. Ha emellett még a `subcaption` csomag is be van töltve, akkor egy `\sub{környezet}` nevű `\stílus` érték is definiálódik.
- Új jegyzék készítése esetén (lásd a 17.6.6. alszakaszt) a `\newlistof` parancs szintén definiál egy `\stílus` értéket.

Alapesetben kétféle módon szokták definiálni az `\l@{stílus}` parancsot:

```
\def\l@{stílus}#1#2{definiáló kód}
```

ahol a `\definiáló kód`-ban szerepelnie kell a #1 és #2 változóknak. A #1 helyén a bejegyzés tartalma fog megjelenni (ami az `\addcontentsline` harmadik argumentumának kifejtése), míg #2 helyén az oldalszám lesz. A `\kód`-ban még meg kell adni a `\@tempdima` értékét, ami a `\numberline` doboz szélessége, és azt is, hogy legyen-e egy pontsor (az ún. vezető), amely összeköti a bejegyzés végét az oldalszámmal. Végül azt is meg kell adni, hogy az adott `\stílus`-nak mi a színtsáma (azaz csak akkor jelenjen meg a jegyzékben a bejegyzés, ha a `\tocdepth` számláló értéke legalább akkora, mint ez a színtsám). Ezzel a technikával vannak definiálva az `\l@{part}`, `\l@{chapter}`, illetve `article` dokumentumosztály esetén az `\l@{section}` parancsok.

A másik standard módszer az `\l@{stílus}` definiálására:

```
\def\l@{stílus}{{\@dottedtocline{színtsám}{behúz1}{behúz2}}}
```

Ebben az esetben az

```
\l@{stílus}{{bejegyzés kifejtve}{\thepage kifejtve}}
```

hatása a következő:

```
\@dottedtocline{színtsám}{behúz1}{behúz2}  
{bejegyzés kifejtve}{\thepage kifejtve}
```

Ha ilyen stílusban viszünk be egy bejegyzést egy jegyzékbe, akkor annak színtsáma `\színtsám` lesz (azaz csak akkor jelenik meg a jegyzékben, ha a `\tocdepth` számláló értéke legalább `\színtsám`), másrészt a bejegyzés `\behúz1` nagyságú behúzással kezdődik. Ha a bejegyzés olyan hosszú, hogy sort kell törni, akkor a második és az azt követő sorokat az első sorhoz képest `\behúz2` nagyságú behúzással kezdi, továbbá a `\@tempdima` értékét is `\behúz2`-re állítja. A bejegyzés végét az oldalszámmal összekötő vezető egy pontsor lesz.

Tulajdonképpen a `\@dottedtocline` egy előre megtervezett séma, amely az előbb említett három paraméterrel variálható, a negyedik helyén a bejegyzés tartalma fog megjelenni (ami az `\addcontentsline` harmadik argumentumának kifejtése), míg az ötödik helyén az oldalszám lesz. Konkrétan a `\@dottedtocline` a következő módon van definiálva:

```
\def\@dottedtocline#1#2#3#4#{%
```

```
\ifnum #1>\c@tocdepth \else
  \vskip \z@ \@plus .2\p@
  {\leftskip #2\relax \rightskip \c@tocrmarg \parfillskip -\rightskip
   \parindent #2\relax \c@afterindenttrue
   \interlinepenalty\@M
   \leavevmode
   \c@tempdima #3\relax
   \advance\leftskip \c@tempdima \null\nobreak\hskip -\leftskip
   {#4}\nobreak
   \leaders\hbox{$\m@th
     \mkern \c@dotsep mu\hbox{.}\mkern \c@dotsep
     mu$}\hfill
   \nobreak
   \hb@xt@\c@pnumwidth{\hfil\normalfont \normalcolor #5%
     \kern-\p@\kern\p@}%
   \par}%
\fi}
```

Természetesen nem kötelező a `\@dottedtocline` parancs használatához ragaszkodni, tetszőleges parancsban definiált sémát is lehet alkalmazni, csak arra kell ügyelni, hogy az így definiált parancs utolsó két paramétere az legyen, mint a `\@dottedtocline` esetében, másrészt még az is fontos, hogy az `\l@stílus` definíciójában már ne szerepeljen a parancs utolsó két paramétere, mert az a kifejtésnél „csapódik” hozzá.

Új `stílus` érték definiálására vagy a már meglévők átdefiniálására több lehetőség is van. Vagy felhasználjuk a meglévő sémákat, vagy újakat készítünk vagy a régieket részlegesen átalakítjuk.

Példa ♦ Definiálunk egy `sajat` nevű stílust, melynek a színtsáma 1.

```
</> \makeatletter
\def\l@sajat{\@dottedtocline{1}{2em}{3em}}
\makeatother
```

Ezután az

```
\addcontentsline{toc}{sajat}{szöveg}
```

hatására megjelenik a `szöveg` a tartalomjegyzékben oldalszámmal együtt `sajat` stílusban, feltéve, hogy a `tocdepth` számláló értéke legalább 1. A `szöveg` 2 em behúzással kezdődik. Ha a `szöveg` olyan hosszú, hogy sort kell törni, akkor a második és az azt követő sorokat az első sorhoz képest 3 em behúzással kezdi. A `szöveg` végét és az oldalszámot pontsor köti össze.

Példa ♦ A már létező `chapter` stílust definiáljuk át.

```
</> \makeatletter
\def\l@chapter{\@dottedtocline{0}{0em}{2em}}
\makeatother
```

Ezután, amikor kiadjuk a

```
\chapter{Fejezetcím}
```

parancsot, akkor az egyúttal kifejti az

```
\addcontentsline{toc}{chapter}
  {\protect\numberline{\thechapter}Fejezetcím}
```

kódot is, melynek hatására a fejezet címe megjelenik a tartalomjegyzékben az újra definiált `chapter` stílusban.

Példa ♦ Készítünk egy `\nodottedtocline` új sémát a `\dottedtocline` átalakításával, amely pontosan úgy működik, mint a `\dottedtocline`, de vezető nélkül.

```
</> \makeatletter
     \let\nodottedtocline\@dottedtocline
     \patchcmd{\nodottedtocline}{\hbox{.}}{}{}{} \in etoolbox
     \makeatother
```

Az előbbi kódban a `\dottedtocline` parancsot bemásoltuk a `\nodottedtocline`-ba, majd a `\nodottedtocline` definiáló kódjából töröltük a `\hbox{.}` kódot, ami a pontsor (vezető) elemeit jelentette. Ezzel pontosan úgy lehet definiálni vagy átdefiniálni stílusokat mint a `\dottedtocline` segítségével, de ekkor nem lesz vezető.

Példa ♦ Készítünk egy `\itshapetocline` új sémát a `\dottedtocline` átalakításával, amely pontosan úgy működik, mint a `\dottedtocline`, de a bejegyzés dőlt betűvel lesz szedve.

```
</> \makeatletter
     \let\itshapetocline\@dottedtocline
     \patchcmd{\itshapetocline}{#4}{\itshape#4}{}{} \in etoolbox
     \makeatother
```

Korábban említettük, hogy a `\dottedtocline` negyedik paramétere jelenti a bejegyzést, így a `#4` lett kijavítva az `\itshape#4` kódra.

Példa ♦ Átdefiniáljuk a `chapter` stílust úgy, hogy az eredeti verzióhoz képest csak a bejegyzés színe változzon feketéről kékre. Ezt nem lehet megtenni a `\dottedtocline` segítségével, hiszen az `\l@chapter` nem ezzel volt eredetileg definiálva. Most közvetlenül az `\l@chapter` kódjában változtatunk. Mivel eredetileg félkövérrel vannak kiszedve a fejezetek címei a tartalomjegyzékben, ezért a `\bfseries` kódot kell kijavítani `\bfseries\color{blue}`-ra:

```
</> \makeatletter
     \patchcmd{\l@chapter}{\bfseries}
                 {\bfseries\color{blue}}{}{} \in etoolbox, xcolor
     \makeatother
```

Példa ♦ A `subsection` stílust alakítjuk át, hogy a régi beállításait megtartsa, de ne legyen vezetője. Ehhez először definiálunk egy új sémát a régiből vezető nélkül, majd a `subsection` stílust átdefiniáljuk az új sémával, de a régi paraméterekkel.

```
</> \makeatletter
     \let\nodottedtocline\@dottedtocline
     \patchcmd{\nodottedtocline}{\hbox{.}}{}{}{} \in etoolbox
     \patchcmd{\l@subsection}{\dottedtocline}{\nodottedtocline}{}{} \in etoolbox
     \makeatother
```

Példa ♦ Következzen egy teljes példa arra, hogyan lehet például az egyenletszámokból jegyzéket készíteni a `tocloft` csomag használata nélkül.

```
</> \documentclass[10pt,a4paper]{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\usepackage{mathtools,etoolbox}

\makeatletter
\newcommand{\listofequations}{
    \section*{Egyenletek jegyzéke}
    \starttoc{loe}
}
\newcommand{\l@equation}{\@dottedtocline{0}{0pt}{2em}}
\newcommand{\Numberline}[1]{\makebox[\tempdima][1]{#1}}
\newcommand{\eqnumtoloe}{%
    \addcontentsline{loe}{equation}{\protect\Numberline{(\theequation)}}
\apptocmd{\incr@eqnum}{\eqnumtoloe}{}{}}
\makeatother

\addtocontents{loe}{egyenletszám\hfill oldalszám%
\par\vspace{-3mm}\noindent\hrulefill\par\medskip}

\begin{document}
\listofequations

\begin{equation}\label{a}
1+1=2
\end{equation}

\begin{equation}\label{b}
2+1=3
\end{equation}

\begin{align}
3+1&=4\label{c}\\
4+1&=5\label{d}
\end{align}
\end{document}
```

A `magyar.ldf` a `\numberline`-t átdefiniálja úgy, hogy az argumentuma után pontot rak, de ez az egyenletek számozásánál, mivel azokat zárójelbe rakjuk, nem kívánatos hatású. Ezért kell a `\Numberline`, ami ezt nem teszi. Az `amsmath` csomagban az `\incr@eqnum` növeli az egyenletszámot, ezért illesztettük be utána a jegyzékbe írást.

17.7. Végjegyzetek

Lehetőség van lábjegyzetek helyett ún. végjegyzeteket is készíteni, amely nem minden oldal alján jelenik meg, hanem az általunk megadott külön oldalon, például minden fejezet végén. Ehhez a `\footnote` parancs helyett használja az

```
\endnote{szöveg} ∈ endnotes
```

parancsot. Ahol a végjegyzetet meg akarja jeleníteni, írja be a következőt:

```
\theendnotes ∈ endnotes
```

17.8. 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.

17.8.1. Bibliográfia készítése környezettel

Bibliográfiát `thebibliography` környezettel lehet készíteni, a bibliográfiai elemeket pedig a `\bibitem` parancssal adhatja meg.

```
\begin{thebibliography}{példacímke}
\bibitem[címke]{kulcs} <elemleírás>
...
\end{thebibliography}
```

<példacímke> A bibliográfiai elemek címkéi közül a legszélesebb.

<címke> Ezzel adhatja meg, hogy a bibliográfiai elem milyen szöveggel legyen azonosítva. Elhagyása esetén automatikus sorszám lesz a címke.

<kulcs> A bibliográfiai elemre

```
\cite[szöveg]{kulcs}
```

parancssal 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 *(szöveg)*-ben például megadhatja, hogy melyik oldalra hivatkozik.

Magyar nyelvű dokumentumban a hivatkozások előtt automatikusan névelőt is rakhatsz

```
\acite[szöveg]{kulcs} \in [magyar]babel
\Acite[szöveg]{kulcs} \in [magyar]babel
```

vagy az ezzel egyenértékű

```
\az{\cite[szöveg]{kulcsszöveg]{kulcs

```

parancsokkal. Ugyanezek a parancsok a `huaz` csomag betöltésével is használhatóak. Például



```
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.

A **natbib** csomaggal arra is lehetőség van, hogy címke nélküli bibliografiát készítsen. Ennek a csomagnak a használata esetén a `\cite` parancs automatikus névelővel csak a `huaz` csomag használata esetén működik megfelelően. A **natbib** csomag általános leírása helyett csak egy példát mutatunk a használatára. Írja a következőket a preambulumba:



```

\usepackage{natbib}
\setcitesep{aysep={},citesep={,},open={},close={}}
\setlength{\bibsep}{0pt}

```

Ezután a dokumentumtestbe írja ezt:

```

Lásd \cite{PlainTeX} és \cite[134.~oldal]{LaTeX}\dots
Lásd \cite{PlainTeX,LaTeX}\dots
Lásd \cite{PlainTeX,LaTeX} könyvekben\dots

\begin{thebibliography}{}
\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 Bujdosó 1997, Wettl 2004 könyvekben...

Hivatkozások

Bujdosó Gyöngyi, Fazekas Attila: TeX kezdőlépések, Budapest, 1997, Tertia Kiadó.
Wettl Ferenc, Mayer Gyula, Szabó Péter: LATEX kézikönyv, Budapest, 2004, Panem Könyvkiadó.

17.8.2. A biblatex csomag

Lehetőség van bibliografiát adatbázisból is készíteni, melynek számos előnye van:

- Több dokumentumhoz is használható ugyanaz az adatbázis, mert csak azok a művek jelennek meg a bibliografiában, amelyekre valóban történt hivatkozás a \cite parancssal. De arra is van lehetőség, hogy az adatbázis minden eleme megjelenjen, függetlenül attól hogy hivatkoztunk-e rá vagy sem.
- A névsorba rendezés és a leghosszabb címke beállítása automatikusan történik.
- A stílus átállítható az adatbázis változtatása nélkül.

Erre a célra használhatja a **biblatex** csomagot. Az adatbázis elemeit egy bib kiterjesztésű fájlba kell írni UTF-8 kódolással. A **biblatex** csomag az adatbázis elemeinek névsorba rendezéséhez és egyéb háttérünkük elvégzéséhez alapesetben a **biber** programot használja.

Ennek az alszakasznak nem célja a **biblatex** teljes tárgyalása. Ha a teljes dokumentációra kíváncsi, akkor keressen rá a **biblatex**, **biblatex-ext** és **biber** kulcsszavakra a TeXstudio **Súgó** » **Csomagleírások** menüpontban.

A bib fájl szerkezete ♦ 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 például a korábban már ismertetett

```
\cite[<szöveg>]{<kulcs>}
```

parancssal lehet hivatkozni. Erre a **biblatex** csomagban más lehetőség is van, amire később még kitérünk.

Sok **(elemtípus)** és hozzáartozó **(mezőnév)** létezik, itt csak néhányat említünk meg a teljesség igénye nélkül.

<i>(elemtípus)</i>	<i>(mezőnév)</i>
<code>article</code>	<code>author title journaltitle date volume number pages issn doi url urldate sortname options langid</code>
<code>book</code>	<code>author title date editor publisher location isbn doi url urldate sortname options langid</code>
<code>inproceedings</code>	<code>author title booktitle date editor eventdate volume number organization publisher location isbn pages doi url urldate sortname options langid</code>
<code>online</code>	<code>author title date url version doi urldate sortname options langid</code>

Elemtípusok ♦ Az előző táblázatban található elemtípusok leírása:

- `article` Cikk adatainak beviteléhez.
- `book` Könyv adatainak beviteléhez.
- `inproceedings` Konferenciakötetben található cikk adatainak beviteléhez.
- `online` Online forrás adatainak beviteléhez.

Mezőnevek ♦ Az előző táblázatban található mezőnevek leírása és megadási módja:

`author` Szerző nevének megadása. A neveket az angol szabálynak megfelelően kell megadni, azaz kereszt- majd családnév sorrendben:

```
author={⟨keresztnév⟩ ⟨családnév⟩ and ⟨keresztnév⟩ ⟨családnév⟩ and  
... ⟨keresztnév⟩ ⟨családnév⟩ and others},
```

Az `and` csak akkor kell, ha több nevet sorol fel, illetve az `and others` akkor, ha nem sorolja fel az összes nevet. Például

```
author={Donald Ervin Knuth and Leslie Lamport and others},
```

Későbbiekben látni fogjuk, hogy van olyan beállítási lehetőség, hogy a keresztnevet rövidítve jelenjenek meg. Azaz például „*Donald Ervin Knuth*” helyett „*D. E. Knuth*” legyen a kimenet. Ez a magyar nevek esetében gond lehet, mert például ilyenkor

```
author={Gyula Szabó and Ferenc Kovács},
```

kimenete „*G. Szabó és F. Kovács*” lesz. Ekkor használja a következő beviteli formát:

```
given={⟨keresztnév⟩},  
given-i={⟨keresztnév rövidítése⟩},  
family={⟨családnév⟩}
```

Például

```
author={given={Gyula}, given-i={Gy}, family={Szabó} and  
Ferenc Kovács},
```

kimenete „*Gy. Szabó és F. Kovács*” lesz rövidített keresztnév beállítás esetén.

`booktitle` Annak a konferenciakötetnek a címe, amelyben a hivatkozott cikk található.

Például

```
booktitle={Proceedings of the 10th International Conference on  
Applied Informatics},
```

`date` A mű kiadásának dátuma. Például

`date={1998},`

doi A mű DOI száma (*Digital Object Identifier* azaz *digitális objektumazonosító*). Például

`doi={10.1080/17442508.2017.1366490},`

editor Szerkesztő neve. Pontosan úgy kell megadni, mint az **author** esetében.

eventdate Konferenciakötet esetén a konferencia megrendezésének dátuma. Megadásának módja:

`eventdate={⟨év⟩-⟨hónap⟩/⟨év⟩-⟨hónap⟩},`

Az első dátum a konferencia kezdetét, a második pedig a végét jelenti. Például

`eventdate={2018-09-30/2018-10-04},`

isbn A könyv vagy konferenciakötet ISBN-száma. Például

`isbn={978-615-5621-72-7},`

issn A folyóirat vagy konferenciakötet ISSN-száma. Például

`issn={1787-5021},`

journaltitle Annak a folyóiratnak a neve, amelyben a hivatkozott cikk található. Például

`journaltitle={Annales Mathematicae et Informaticae},`

langid Többnyelvű bibliográfia esetén a `biblatex` csomag `autolang=hyphen` opciójával minden nyelv szabályai szerinti szóelválasztást alkalmazza. Ha például a `babel` csomag `magyar` opciójával van betöltve és a bib fájlban az egyik elem angol nyelvű, akkor írja be a következőt:

`langid={english},`

location A könyv kiadásának helye. Például

`location={Eger},`

number A folyóirat száma.

options minden elemhez adhatunk opciókat is ezzel a mezőnévvvel. Erre később látunk majd példát.

organization Konferenciakötet esetében a konferencia szervezőjének neve.

`organization={University of Debrecen and
the Eszterházy Károly Catholic University},`

pages Cikk első és utolsó oldalszáma a folyóiratban. A következő alakban kell megadni:

`pages={⟨első oldalszám⟩-⟨utolsó oldalszám⟩},`

Például

`pages={135-150},`

publisher Könyv kiadójának a neve. Például

`publisher={Springer},`

sortname A `biber` nemcsak az angol, hanem a magyar névsorba rendezés szabályait is ismeri. Így pl. angol nyelvű dokumentumban a *Bolyai* megelőzi a *Bolzano* nevet (hiszen az *y* előbb van, mint a *z*), de magyar esetén fordított lesz a sorrend (hiszen az *l* előbb van, mint az *ly*). Ugyanakkor magyar nyelv esetén például a *Vácszentmiklósi* névben a *cs* betűket a `biber` tévesen kettős mássalhangzónak tekinti (azaz *csé*-nek), így mondjuk a *Vácz* nevet előbbre sorolja, miközben a

helyes sorrend fordított, hiszen az *sz* előbb van, mint a *z*. Ennek egy lehetséges megoldása a `sortname` mezőnél használata. Ebben azt lehet megadni, hogy a neveket miként sorolja be. Például

```
author={József Vácszentmiklósi},
sortname={Váctzentmiklósi},
```

esetében az eredményben *Vácszentmiklósi* lesz látható, de *Váctzentmiklósi*-ként lesz névsorba rendezve. Itt a *c* betűt követő *s* helyére *t* betűt írtunk, ami nagy esélyel megoldja a problémát.

title A mű címe. Például

```
title={Baum--Katz Type Theorems with Exact Threshold},
```

Bizonyos esetekben a `biblatex` a címekben az első betűt nagybetűsítheti, illetve a többöt kisbetűsítheti. Ilyenkor pl. a „*Baum–Katz Type Theorems with Exact Threshold*” címet „*Baum–katz type theorems with exact threshold*” alakra konvertálja. Ha ezt például nevek esetén meg akarja akadályozni, akkor a betűváltoztatás ellen védendő szót tegye kapcsos zárójelek közé. Ne csak az adott betűt, hanem az egész szót tegye kapcsos zárójelbe, különben a betűk körüli térközök nem lesznek megfelelőek. Például

```
title={{Baum--Katz} Type Theorems with Exact Threshold},
```

vagy

```
title={{$p$-adic} Logarithmic Forms and Group Varieties {II}},
```

Hasonlóan, a címekben szereplő parancsokat is tegye kapcsos zárójelek közé:

```
title={The {\TeX book}},
```

url A mű webcíme. Például

```
url={http://tug.org/texinfohtml/latex2e.html},
```

urldate A webcím hivatkozáskor dátumának megadása a következő módon történik:

```
urldate={(év)}-{(hó)}-{(nap)},
```

Például

```
urldate={2009-01-31},
```

version Az online forrás verziója.

volume A mű kötetének a száma.

Az előbbi mezőneveket három csoportba soroljuk típusuk szerint:

name list típusú mezőnevek: `author`, `editor`.

literal list típusú mezőnevek: `organization`, `publisher`.

field típusú mezőnevek: `booktitle`, `date`, `doi`, `eventdate`, `isbn`, `issn`, `journalttitle`, `location`, `number`, `pages`, `title`, `url`, `urldate`, `version`, `volume`.

Egy példa bib fájlra ♦ Írja a következőket egy UTF-8 kódolású `references.bib` fájlba. A bevitelt nagy mértékben megkönyíti a TeXstudio `Bibliográfia` menüpontja.

```
@book{Knuth2001,
author={Donald Ervin Knuth},
title={Deformation modelling tracking animation and applications},
date={2001},
publisher={Springer},
location={Berlin, Heidelberg},
```

```

}

@article{BalkaTomacs2018,
  author={Richárd Balka and Tibor Tómács},
  title={{Baum--Katz} type theorems with exact threshold},
  journaltitle={Stochastics},
  volume={90},
  number={4},
  date={2018},
  pages={473–503},
  publisher={Taylor \& Francis},
  doi={10.1080/17442508.2017.1366490},
}

```

A bibliográfia megjelenítése ♦ A `babel` és a `biblatex` csomagok együttes használata esetén be kell még tölteni a `csquotes` csomagot is, mert a címek idézőjelben fognak megjelenni, amit az adott nyelvhez a `csquotes` csomaggal igazít.

A tex kiterjesztésű fájlba a bibliográfia megjelenéséhez a `biblatex` csomag betöltése után írja a preambulumba az

```
\addbibresource{(bib fájl neve)}
```

parancsot, majd a dokumentumtestbe a bibliográfia megjelenésének helyére a

```
\printbibliography[(opciók)]
```

parancsot. Néhány lehetséges opció:

`title=(bibliográfia címe)` Ha a bibliográfia címének alapértékét át akarja állítani, akkor használja ezt az opciót. Például

```
title=Irodalomjegyzék
```

`heading=(stílus)` Ha a `(stílus)` `bibintoc`, akkor a bibliográfia bekerül a tartalomjegyzékre. Ha a `(stílus)` `bibnumbered`, akkor a bibliográfia címe számoszt fejezet vagy szakasz lesz és bekerül a tartalomjegyzékre.

Alapesetben a bib fájlban megadott művekből csak azok fognak megjelenni a bibliográfiaban, melyekre `\cite` parancsal 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.

Egy példa tex fájlra ♦ Példaként írja a következőket a `document.tex` fájlba:



```

\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\usepackage{csquotes}
\usepackage{biblatex}
\addbibresource{references.bib}
\begin{document}

```

```
See \cite[p.~56]{Knuth2001} \dots see \cite{BalkaTomacs2018}
\printbibliography
\end{document}
```

Ezt a fájlt és az előbbi `references.bib` fájlt tegye ugyanabba a mappába.

A végeredmény előállítása ♦ Az előző példában úgy tudja előállítani a végeredményt, hogy parancssorban kiadja a következő parancsokat:

```
pdflatex document.tex
biber document
pdflatex document.tex
pdflatex document.tex
```

TeXstudióban érdemes a végeredményt a `latexmk` programmal előállítani (lásd a Bevezetésben), mert ez automatikusan futtatja a `biber` és `pdflatex` programokat a megfelelő számban és sorrendben. Parancssorban:

```
latexmk -pdf document
```

A biblatex csomag opciói ♦ A teljesség igénye nélkül felsoroljuk a `biblatex` csomag néhány opcióját:

`style=(stílusnév)` Ezzel lehet megadni a bibliográfia és hivatkozások stílusát. Rengeteg definiált `(stílusnév)` van, itt csak néhányat sorolunk fel:

`numeric` Ez az alapstílus. A bibliográfiai elemek és a hivatkozások számozottak.

`alphabetic` A bibliográfiai elemek címkéi a szerző és az évszám alapján összeálított rövidítés.

`authoryear` A bibliográfiai elemek nincsenek címkézve. A hivatkozás a szerző és az évszám alapján történik.

Minden standard stílusnak van egy kiterjesztett verziója is, amit a név előt tett `ext-` előjáróval érhetünk el: `ext-numeric`, `ext-alphabetic`, `ext-authoryear` stb.

`maxnames=(szám)` Ha `(szám)`-nál több szerzője van a bibliográfiai elemek, akkor csak a `minnames` opció által megadott számú szerzőt tünteti fel, majd utána azt írja ki, hogy „*és mások*” vagy rövidítve „*és tsai*”. A `(szám)` alapértéke 3.

`minnames=(szám)` Lásd a `maxnames` opciótól. A `(szám)` alapértéke 1.

`giveninits` A szerzők keresztnévei rövidítve jelennek meg. Például *Donald Ervin Knuth* helyett *D. E. Knuth*.

`abbreviate=false` A bibliográfiába automatikusan kerülő szövegek nem lesznek rövidítve („*és tsai*” helyett „*és mások*”, „*Irodalom*” helyett „*Irodalomjegyzék*”, stb.).

`autolang=hyphen` Többnyelvű bibliográfia esetén ezzel az opcióval minden a megadott nyelv szabályai szerinti szóelválasztást alkalmazza. Ha a `babel` csomag `magyar` opciótól van betöltve és a `bib` fájlban az egyik elem angol nyelvű, akkor írja be a következőt:

```
langid={english},
```

Például

```
@book{Knuth2001,
langid={english},
author={Donald Ervin Knuth},
title={Deformation modelling tracking animation and applications},
date={2001},
```

```
publisher={Springer},
address={Berlin, Heidelberg},
}
```

A biblatex csomag parancsai ♦ A biblatex csomagnak számtalan olyan paranca van, amivel a legapróbb részletekig beállíthatja a kimenetet. Amennyiben a rendelkezésre álló stílusok közül egyik sem felel meg, akkor legszerencsébb megoldás, hogy létrehoz egy saját stílust. Ennek leírására nem vállalkozunk, csak néhány olyan parancsot ismertetünk, amivel egy meglévő stílust finomíthat a maga kedvére.

\biblabelsep

Ez határozza meg a címke és a bibliográfiai elem közötti távolságot. Alapértéke a \labelsep kétszerese. Ha a \labelsep értékre akarja átállítani, akkor használja a következő parancsot:

```
\setlength{\biblabelsep}{\labelsep}
```

\bibfont

Ez adja meg a bibliográfia fonttípusát. Átállíthatja például \small méretre a következő módon:

```
\renewcommand{\bibfont}{\normalfont\small}
```

\mkbibnamefamily

Ez adja meg a családnevek fonttípusát. Átállíthatja például kiskapitalisra a következő módon:

```
\renewcommand{\mkbibnamefamily}{\scshape}
```

\mkbibnamegiven

Ez adja meg a keresztnévek fonttípusát. Átállíthatja például kiskapitalisra a következő módon:

```
\renewcommand{\mkbibnamegiven}{\scshape}
```

\DeclareNameAlias{author}{family-given}

Ebben az esetben a bib fájlban található

```
author={Donald Ervin Knuth},
```

kimenete „*Knuth, Donald Ervin*” lesz. Azaz a kimenetben először a családnév, majd a keresztnév jelenik meg. A kettő közé angolszász szokás szerint vessző kerül.

\revsdnamepunct

Az előző parancs esetén ezen parancs által tárolt jel kerül a család- és keresztnév közé. Ennek alapértéke vessző. Ha magyar a dokumentum és a bibliográfiai elemek szerzői magyarok, akkor célszerű ezt átállítani üresre az előző parancs használata mellett:

```
\DeclareNameAlias{author}{family-given}
\renewcommand{\revsdnamepunct}{}%
```

Ilyenkor például a bib fájlbeli

```
author={Gyula Szabó},
```

kimenete „*Szabó Gyula*” lesz.

```
\DeclareEntryOption[<opcion>]{<opciónév>} [<alapérték>]{<definíció>}%
```

Az általános leírás helyett egy konkrét példán szemléltetjük a működését:

```
\DeclareEntryOption[boolean]{hunname}[true]{%
    \DeclareNameAlias{author}{family-given}%
    \renewcommand{\revsdnamepunct}{}}
```

Ez létrehoz egy logikai típusú `hunname` nevű opciót, melynek alapértéke igaz (`true`). Ezen opció hatására az adott mezőben elhelyezett szerzők nevei család- és keresztnév sorrendben jelennek meg úgy, hogy közöttük nincs semmilyen jel. Az opció a bib fájlban a következő módon adható meg:

```
options={hunname},
```

Több opció használata esetén, azokat vesszővel kell elválasztani. Az előző példát akkor célszerű használni, amikor magyar a dokumentum, de a bibliográfiában van magyar és pl. orosz szerző is. Ilyenkor az előző opciót a magyar szerzőjű műhöz beírva, a család- és keresztnévek jó sorrendben fognak megjelenni. Például

```
@book{KolmogorovFomin1981,
author={Andrey Nyikolajevics Kolmogorov and
        Szergej Vasziljevics Fomin},
title={A függvényelmélet és a funkcionálanalízis elemei},
publisher={Műszaki Könyvkiadó},
location={Budapest},
date={1981},
}
@book{KatonaRecskiSzabo2006,
options={hunname},
author={Gyula Katona and András Recski and Csaba Szabó},
title={A számítástudomány alapjai},
publisher={Typotex},
location={Budapest},
date={2006},
}
```

```
\DeclareFieldFormat[<opció>]{<field típusú mezőnév>}{{<formázás>}}
```

A `<field típusú mezőnév>` formázását adja meg. A formázásban #1 módon utaljon a `<field típusú mezőnév>` tartalmára. Az `<opció>` tartalmazza azon elemtípusokat, melyekben a megadott `<field típusú mezőnév>`-re vonatkozik a formázás. Opció megadása nélkül minden olyan elemtípusra fog vonatkozni, melyekben a `<field típusú mezőnév>` még nincs formázva. A `<formázás>`-ban nem szerepelhet szóköz. Például

```
\DeclareFieldFormat{title}{\textit{#1}}
```

esetén minden olyan elemtípusnál dőlt betűvel fog megjelenni a cím, ahol az még nincs formázva. Így például a `book` elemtípusnál dőlt betű lesz, de az `article`-nél nem.

```
\DeclareFieldFormat[article,book]{title}{\textit{#1}}
```

esetén az `article` és `book` elemtípusnál dőlt betű lesz a cím, minden másnál marad az eredeti formázás.

```
\DeclareFieldFormat*{<field típusú mezőnév>}{{<formázás>}}
```

Úgy működik, mint az előző parancs, de ez minden elemtípusra kihatással van. Például

```
\DeclareFieldFormat*{title}{\textit{#1}}
```

esetén minden elemtípusnál dőlt betűvel fog megjelenni a cím.

```
\DeclareFieldFormat{titlecase}{\MakeSentenceCase{#1}}
```

A `title`, `journalttitle` és `booktitle` mezőnevek esetén az első betűt nagybetűsíti, a többöt pedig kisbetűsíti. Ha ez például nevek esetén nem kívánatos hatással van, akkor azt kapcsos zárójelbe kell rakni. Ne csak a betűt, hanem az egész szót tegye kapcsos zárójelbe, hogy a betűk körüli térközök helyesek maradjanak. Például

```
title={{Baum--Katz} Type Theorems with Exact Threshold},
```

eredménye *Baum–Katz type theorems with exact threshold*.

```
booktitle={{$p$-adic} Logarithmic Forms and Group Varieties},
```

eredménye *p-adic logarithmic forms and group varieties*.

```
\add{jel}
```

Bibliográfiai elemek formázásánál szóköz vagy írásjel beszúrása. A `(jel)` helyére a következők írhatók:

<code>space</code>	szóköz
<code>nbspace</code>	törhetetlen szóköz
<code>thinspace</code>	törhető fél szóköz
<code>nbthinspace</code>	törhetetlen fél szóköz
<code>dot</code>	pont
<code>comma</code>	vessző
<code>semicolon</code>	pontosvessző
<code>colon</code>	kettőspont

```
\newunitpunct
```

Az egységek közötti elválasztójel. Alapértéke pont és utána egy törhető szóköz. Átállítása például vesszőre és utána egy törhető szóközre:

```
\renewcommand{\newunitpunct}{\addcomma\addspace}
```

```
\multinamedelim
```

A szerzők illetve szerkesztők nevei közötti jel, kivéve az utolsó név előtti jelet. Átállítása például gondolatjelre:

```
\renewcommand{\multinamedelim}{\addnbthinspace--\addnbthinspace}
```

```
\finalnamedelim
```

A szerzők illetve szerkesztők nevei esetén az utolsó név előtti jel. Átállítása például vesszőre és utána egy törhető szóközre:

```
\renewcommand{\finalnamedelim}{\addcomma\addspace}
```

```
\labelnamepunct
```

A szerzők illetve szerkesztők neveinek listája után álló jel. Átállítása például kettőspontra és utána egy törhető szóközre:

```
\renewcommand{\labelnamepunct}{\addcolon\addspace}
```

```
\DeclareBibliographyDriver{elemtípus}{\formázás}
```

Az `(elemtípus)` kimenetelét formázza. A `(formázás)`-ban a következő parancsok segítenek:

```
\printnames{\name list típusú mezőnév}
```

A `(name list típusú mezőnév)` tartalmának megjelenítése.

`\printlist{<literal list típusú mezőnév>}`

A `<literal list típusú mezőnév>` tartalmának megjelenítése.

`\printfield{<field típusú mezőnév>}`

A `<field típusú mezőnév>` tartalmának megjelenítése.

`\ifnameundef{<name list típusú mezőnév>} {<igaz>} {<hamis>}`

Ha az `<elemtípus>`-ban meg van adva a `<name list típusú mezőnév>`, akkor `<igaz>`, ellenkező esetben `<hamis>` az eredmény.

`\iflistundef{<literal list típusú mezőnév>} {<igaz>} {<hamis>}`

Ha az `<elemtípus>`-ban meg van adva a `<literal list típusú mezőnév>`, akkor `<igaz>`, ellenkező esetben `<hamis>` az eredmény.

`\iffieldundef{<field típusú mezőnév>} {<igaz>} {<hamis>}`

Ha az `<elemtípus>`-ban a `<field típusú mezőnév>` adott, akkor `<igaz>`, ellenkező esetben `<hamis>` az eredmény.

`\newunit`

A `\newunit`-ban megadott jel kiíratása.

`\setunit*{<jel>}`

A `<jel>` kiírása, amennyiben előtte a `\printnames`, `\printlist` vagy `\printfield` parancsban adott mezőnév az `<elemtípus>`-ban meg van adva.

`\usebibmacro{finentry}`

Az `<elemtípus>`-t lezáró parancs.

Például

```
\DeclareBibliographyDriver{book}{%
  \printnames{author}%
  \labelnamepunct%
  \printfield{title}%
  \newunit
  \printlist{publisher}%
  \newunit
  \printlist{location}%
  \newunit
  \printfield{date}
  \usebibmacro{finentry}}
```

Hivatkozás egy bibliográfiai elemre ♦ Egy adott bibliográfiai elemre a következő módokon lehet hivatkozni:

```
\cite[<szöveg>]{<kulcs>}
\cite[<oldalszám>]{<kulcs>}
\cite[<kezdő oldalszám>–<utolsó oldalszám>]{<kulcs>}
```

ahol a `<kulcs>` a bib fájlból megadott azonosítója a bibliográfiai elemnek. Például

```
\cite[Theorem~5]{Kolmogorov},
\cite[122]{Kolmogorov},
\cite[32–45]{Kolmogorov}
```

eredménye, ha angol nyelvű a dokumentum, számosztott a stílus és a Kolmogorov kulcshoz a 4 sorszám tartozik:

[4, Theorem 5], [4, page 122], [4, pages 32–45]

Ha magyar nyelvű a dokumentum, akkor változik az eredmény. Például

```
\cite[5.~tétel]{Kolmogorov},
\cite[122]{Kolmogorov},
\cite[32–45]{Kolmogorov}
```

[4, 5. tételek], [4, 122. oldal], [4, 32–45. oldal]

Látható, hogy ebben az esetben van egy apró probléma, nevezetesen, hogy az „5. tételel” után van egy pont. Ennek megoldására a következő két lehetőséget ajánlom. Az egyik, hogy írja a következőt a preambulumba:

```
\def\citeoptnorm#1{\begin{group}\def\mkbibordinal##1{##1}\end{group}}
```

Ezután

```
\citeoptnorm{\cite[5.~tétel]{Kolmogorov}},
\cite[122]{Kolmogorov},
\cite[32–45]{Kolmogorov}
```

eredménye

[4, 5. tételek], [4, 122. oldal], [4, 32–45. oldal]

A másik lehetőség a következő. Írja a preambulumba, hogy

```
\DefineBibliographyExtras{magyar}{\DeclareFieldFormat{postnote}{#1}}
```

Ekkor a \cite opciójába már csak egyszerű szöveg kerülhet, nem ismeri fel az oldalszámokat. Így például

```
\cite[5.~tétel]{Kolmogorov},
\cite[122.~oldal]{Kolmogorov},
\cite[32–45.~oldal]{Kolmogorov}
```

eredménye

[4, 5. tételek], [4, 122. oldal], [4, 32–45. oldal]

Amennyiben angol nyelvű dokumentum esetén szeretné, hogy a \cite opciójában az oldalszámokat egyszerű szövegként kezelje, akkor írja a következőt a preambulumba:

```
\DeclareFieldFormat{postnote}{#1}
```

A \cite használatára az előző példák eredményei számosztott stílus esetére vonatkoztak. Ha más stílust választ, az eredmény is más lehet. Ezeket itt nem részletezzük, ehhez nézze át a biblatex dokumentációját. Vannak még további lehetőségek is a hivatkozásra. Ezekből hármat emlíünk meg:

```
\citeauthor{<kulcs>}
\citetitle{<kulcs>}
\citeyear{<kulcs>}
```

Az első a mű szerzőjét, a második a mű címét, a harmadik pedig a mű kiadásának évét adja eredményül.

Egy magyar nyelvű példa biblatex használatára ♦ Egy tex fájlba másolja be a következőket:

```
</> \documentclass[a4paper,12pt]{article}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}

\usepackage{csquotes}
\usepackage[autolang=hyphen,giveninits,abbreviate=false]{biblatex}

\setlength{\biblabelsep}{\labelsep}
\renewcommand{\mkbibnamefamily}{\scshape}
\renewcommand{\mkbibnamegiven}{\scshape}
\DeclareFieldFormat*[title]{\textit{\#1}}
\DeclareFieldFormat[article]{journaltitle}{\#1}
\DeclareFieldFormat[article]{pages}{\#1}
\renewcommand{\finalnamedelim}{\addcomma\addspace}
\renewcommand{\labelnamepunct}{\addcolon\addspace}
\renewcommand{\newunitpunct}{\addcomma\addspace}
\DeclareEntryOption[boolean]{hunname}[true]{%
    \DeclareNameAlias{author}{family-given}%
    \renewcommand{\revsdnamepunct}{}}
\DefineBibliographyExtras{magyar}{\DeclareFieldFormat{postnote}{\#1}}
\addbibresource{references.bib}

\begin{document}

\cite{KatonaRecskiSzabo2006} --
\cite[1.~tétele]{KatonaRecskiSzabo2006} --
\citemuthor{Knuth2001} \citeyear{Knuth2001} --
\cite{BalkaTomacs2018,KatonaRecskiSzabo2006,Knuth2001,
KolmogorovFomin1981}

\printbibliography

\end{document}
```

A references.bib fájl tartalma:

```
@book{KolmogorovFomin1981,
author={Andrzej Nyikolajevics Kolmogorov and
        Szergej Vasziljevics Fomin},
title={A függvényelmélet és a funkcionálanalízis elemei},
publisher={Műszaki Könyvkiadó},
location={Budapest},
date={1981},
}

@book{KatonaRecskiSzabo2006,
options={hunname},
author={given={Gyula}, given-i={Gy}, family={Katona} and
        András Recski and
```

```

        given={Csaba}, given-i={Cs}, family={Szabó}},
title={A számítástudomány alapjai},
publisher={Typotex},
location={Budapest},
date={2006},
}

@book{Knuth2001,
langid={english},
author={Donald Ervin Knuth},
title ={Deformation modelling tracking animation and applications},
date={2001},
publisher={Springer},
location={Berlin, Heidelberg},
}

@article{BalkaTomacs2018,
langid={english},
author={Richárd Balka and Tibor Tómács},
title={Baum--Katz type theorems with exact threshold},
journaltitle={Stochastics},
volume={90},
number={4},
pages={473–503},
date={2018},
publisher={Taylor & Francis},
doi={10.1080/17442508.2017.1366490},
}

```

[2] – [2, 1. téTEL] – KNUTH 2001 – [1, 2, 3, 4]

Hivatkozások

- [1] R. BALKA, T. TÓMÁCS: *Baum-Katz type theorems with exact threshold*, Stochastics 90.4 (2018), 473–503, DOI: 10.1080/17442508.2017.1366490.
- [2] KATONA GY., RECKSI A., SZABÓ Cs.: *A számítástudomány alapjai*, Budapest: Typotex, 2006.
- [3] D. E. KNUTH: *Deformation modelling tracking animation and applications*, Berlin, Heidelberg: Springer, 2001.
- [4] A. N. KOLMOGOROV, S. V. FOMIN: *A függvényelmélet és a funkcionál-analízis elemei*, Budapest: Műszaki Könyvkiadó, 1981.



Videó: Irodalomjegyzék készítése biblatex csomaggal

17.9. Tárgymutató

A tárgymutató a műben előforduló fontosabb fogalmaknak névsorba rendezett jegyzéke, melyben minden tárgyszó mellett megtalálható annak az oldalnak a száma, ahol az szóba kerül. Erre több eszköz is van. A következő két alszakaszban a `texindy` illetve az `upmindex` programok használatát ismertetjük a teljesség igénye nélkül. Bármelyiket választhatja, bár a `texindy` 2014 óta nincs fejlesztve, így az esetleges hibák nem lesznek már javítva benne nagy valószínűséggel. Mindkét programot az `indextools` csomag segítségével fogjuk használni, így egy menetben lehet majd a forrást lefordítani, továbbá rugalmasabb beállítási lehetőségeket biztosít.

A két alszakasz egymástól függetlenül olvasható, nem utal egyik a másikra. Erre azért volt szükség, hogy ha valaki csak az egyik programot akarja használni, annak ne kelljen minden két alszakaszt elolvasni. Emiatt több átfedés is lesz bennük.

17.9.1. Az `indextools` és `texindy` együttes használata

Előkészületek ♦ Írja be a preambulumba a következőket:

```
\usepackage[texindy]{indextools}
\makeindex[<opción>]
```

A lehetséges `<opción>`:

`options=-C utf8 -L <nyelv>` A `texindy` kapcsolót adja meg. A `<nyelv>` magyar illetve angol esetén `hungarian` illetve `english`. Saját stílus is megadható egy xdy kiterjesztésű fájlban (részletesebben lásd a 324. oldalon). Ha például ez a `user.xdy`, akkor ennek betöltéséhez használni kell még az `-M user` kapcsolót is. Például

`options=-C utf8 -L hungarian -M user`
`name=<fájlnév>` Annak a fájlnak a neve kiterjesztés nélkül, amibe elmenti a tárgyszavakat a megfelelő oldalszámokkal. (A `<fájlnév>`-ben ne legyen ékezes betű és szóköz.) Alapértéke a `\jobname`, ami az aktuális tex fájl neve. Mentésnél a fájl kiterjesztése `idx` lesz, azaz például `name=nevmutato` esetén a `nevmutato.idx` fájlba kerülnek a tárgyszavak az oldalszámokkal. Ennek akkor van jelentősége, ha a tárgymutatón kívül például névmutatót is akarunk készíteni. Erre később látunk majd példát.

`title=<cím>` A tárgymutató címe. Alapértéke az `\indexname`, aminek eredménye magyar dokumentum esetén „Tárgymutató”, angol esetén „Index”.

`columns=<hasábszám>` A tárgymutató hány hasábbal legyen kiszedve. Alapértéke 2.

`columnsep=<hasábköz>` A hasábok távolsága egymástól. Alapértéke 35pt.

`columnseprule` A hasábok vonallal legyenek egymástól elválasztva.

`intoc` A tárgymutató címe kerüljön be a tartalomjegyzékbe.

Például

```
\usepackage[texindy]{indextools}
\makeindex[options=-C utf8 -L hungarian,name=nevmutato,
           title=Névmutató,columns=1]
\makeindex[options=-C utf8 -L hungarian]
```

esetén két idx kiterjesztésű fájl készül. Az egyik a `nevmutato.idx`, a másik pedig például `dokumentum.tex` esetén `dokumentum.idx`. A `nevmutato.idx`-be kerülő tárgyszavak és oldalszámok „Névmutató” cím alatt jelennek meg a végeredményben egyhasábos szedéssel, míg a `dokumentum.idx`-be kerülő tárgyszavak és oldalszámok „Tárgymutató” cím

alatt jelennek meg a végeredményben kéthasábos szedéssel, feltéve, hogy az `\indexname` Tárgymutató-ként van definiálva.

Tárgyszavak bevitelére ♦ A tárgymutatót az

```
\index[<fájlnév>]{<tárgyszó>}
```

parancsal bővítheti a dokumentumtestben, ahol a `<fájlnév>` alapértéke a `\jobname`, azaz a tex fájl neve kiterjesztés nélkül. A tárgyszó a `<fájlnév>.idx` fájlba kerül, feltéve, hogy valamelyik `\makeindex` parancsban meg volt adva a `name=<fájlnév>` opció. Például

```
...
\makeindex[options=-C utf8 -L hungarian,name=nevmutato,title=Névmutató]
\makeindex[options=-C utf8 -L hungarian]
...
\begin{document}
...
Kolmogorov\index[nevmutato]{Andrej Nyikolajevics Kolmogorov (1903--1987)}
1933-ban a következő axiómákat mondta ki:
...
A szórásnégyzetet\index[szórásnégyzet] a következőképpen értelmezzük:
...
```

esetén a `nevmutato.idx` fájlba kerül az „Andrej Nyikolajevics Kolmogorov (1903–1987)” tárgyszó a megfelelő oldalszámmal, illetve a „szórásnégyzet” a `dokumentum.idx`-be, feltéve, hogy a tex fájl neve `dokumentum.tex` volt.

A tárgymutatóba kerülő tárgyszó formázható is a következő módokon:

`<csoport>!<tárgyszó>` Ekkor a `<tárgyszó>` a `<csoport>`-hoz lesz besorolva. Például

```
\index{eloszlás!normális}.
```

`<csoport>!<alcsoport>!<tárgyszó>` Ekkor a `<tárgyszó>` a `<csoport>`-hoz, azon belül pedig az `<alcsoport>`-hoz lesz besorolva. Például

```
\index{eloszlás!normális!standard}.
```

`<besorolás>@<tárgyszó>` Ekkor a `<tárgyszó>` úgy sorolódik betűrendbe, mint a `<besorolás>` szó. Például

```
\index{gamma-eloszlás@$\\Gamma$-eloszlás}
```

esetén a „ Γ -eloszlás” tárgyszó kerül a tárgymutatóba, de „gamma-eloszlás”-ként sorolódik betűrendbe. Az előzőekben a `<csoport>`, `<alcsoport>` és `<tárgyszó>` is lehet `<besorolás>@<tárgyszó>` alakú. Például

```
\index{eloszlás!gamma@$\\Gamma$}
```

```
\index{gamma-eloszlás@$\\Gamma$-eloszlás!súrűségfüggvény}.
```

Magyar nyelvű dokumentumok esetén például `NY (New York)` az `NY` betűhöz sorolódik be `N` helyett. Ennek elkerüléséhez használhatja a következő trükköt:

```
\index{NXZ (New York)@NY (New York)}.
```

Ekkor a problémás `Y`-t az `XZ`-hez soroljuk be, amivel már a tárgyszó `NY` helyett az `N` csoportba kerül, és azon belül is jó eséllyel a megfelelő sorrendben.

`<tárgyszó>|<oldalszámformázás>` A `\texindy` használata esetén az `<oldalszámformázás>` helyére `textbf` vagy `textit` írható aszerint, hogy az oldalsámozást félkövér vagy dőlt betűtípusossal szeretnénk. Például

```
\index{eloszlás|textbf}
```

esetén félkövér betűtípusossal jelenik meg az oldalszám az „eloszlás” tárgyszó után. A `<tárgyszó>` helyére bármilyen korábban ismertetett verzió beírható:

```
\index{eloszlás!normális|textbf}
\index{gamma-eloszlás@$|\Gamma$-eloszlás|textbf}
\index{gamma-eloszlás@$|\Gamma$-eloszlás!sűrűségfüggvény|textbf}.
```

Amennyiben a `textbf` és `textit` (*oldalszámformázás*) típusokon kívül más típust is definiálni akar, akkor az a saját stílust tartalmazó `xdy` kiterjesztésű fájlban tehető meg (lásd a 324. oldalon).

A tárgyszavak a végeredményben névsorban fognak megjelenni, továbbá a kezdőbetűknek megfelelő csoportokra lesznek bontva. A csoportok előtt alapértelmezésben a megfelelő betű áll címként. Ha a tárgyszó számmal kezdődik, akkor az első, alapértelmezésben cím nélküli csoporthoz lesz besorolva, még az A betűs csoport elé.

Ha a tárgyszó képlet, minden adjon meg besorolást. Például `\index{alfa@$|\alpha$}`. Ha nem az A betűs csoporthoz, hanem az elé akarja besorolni, akkor az `alfa` elé még írjon például egy 0-t: `\index{0alfa@$|\alpha$}`.

Az `\index` parancsban négy speciális jel van, melyek nem jelennek meg a kiírásnál: `@` ! | ". Ha ezeket meg akarja jeleníteni, akkor előjük kell írni egy " jelet.

A `showidx` csomagot az `indextools` után betöltve, a L^AT_EX minden tárgyszót feltüntet a szöveg bal margóján, ami hasznos ellenőrzési lehetőséget nyújt. Természetesen a szerkesztés befejezése után, a végleges verzióban ez feleslegessé válik.

A tárgymutató kiírása ♦ Ahová el akarja helyezni a tárgymutatót, írja be a

```
\printindex[<fájlnév>]
```

parancsot, ahol a *<fájlnév>* alapértéke a `\jobname`, azaz a tex fájl neve kiterjesztés nélkül. Ennek hatására a *<fájlnév>.idx* fájlba került tárgyszavak jelennek meg névsorba szedve az adott cím alatt.

Amennyiben szeretne egy bevezető szöveget a tárgymutató elején, akkor használja a következő parancsot a `\printindex` előtt:

```
\indexprologue{<söveg>}
```

Korábban már volt róla szó, hogy amennyiben a tárgyszó számmal kezdődik, akkor az első csoporthoz lesz besorolva, az A betűs csoport elé. Ha ennek a csoportnak szeretne címet adni, akkor használja a `\printindex` előtt a

```
\newcommand{\lettergroupDefault}[1]{<csoportcím>}
```

parancsot. Például

```
\newcommand{\lettergroupDefault}[1]{\par\textbf{Jelölések}\par}
```

Amennyiben a többi csoport jelöljét akarja formázni, akkor használja a `\printindex` előtt a

```
\newcommand{\lettergroup}[1]{<csoportjelölő>}
```

parancsot. Például, ha be akarja keretezni a csoportjelölőket, akkor

```
\newcommand{\lettergroup}[1]{\par\fbox{\textbf{\#1}}\par\nopagebreak}
```

Magyar dokumentum esetén a hosszú magánhangzók rövid magánhangzóként sorolódnak be. Így például az „Álmos” tárgyszó az A betűs csoporthoz lesz besorolva. Ez indokolja, hogy az A betűs csoport jelölője legyen „A, Á”, az E betűs csoport jelölője „E, É” stb. Ez alapesetben nem így van, de a következő kód `\printindex` előtti használatával megoldható:

```
\newcommand{\lettergroup}[1]{%
\def\letter{\#1}%
```

```
\def\Letter{A}\ifx\letter\Letter\def\letter{A, Á}\fi
\def\Letter{E}\ifx\letter\Letter\def\letter{E, É}\fi
\def\Letter{I}\ifx\letter\Letter\def\letter{I, Í}\fi
\def\Letter{O}\ifx\letter\Letter\def\letter{O, Ó}\fi
\def\Letter{Ö}\ifx\letter\Letter\def\letter{Ö, Ő}\fi
\def\Letter{U}\ifx\letter\Letter\def\letter{U, Ú}\fi
\def\Letter{Ü}\ifx\letter\Letter\def\letter{Ü, Ÿ}\fi
\par\textbf{\letter}\par\nopagebreak}
```

Stílusfájl ♦ A `\makeindex` opciójánál említettük, hogy saját stílusfájl is írható xdy kiterjesztéssel. Ezt célszerű a tex fájl preambulumában megtenni a következő kóddal:

```
\begin{filecontents*}[overwrite]{fájlnév}.xdy
<beállítások>
\end{filecontents*}
```

A kód úgy működik, hogy fordításkor létrejön egy `<fájlnév>.xdy` fájl `<beállítások>` tartalommal. Ezt a fájlt mi is létrehozhatjuk és megszerkeszhetjük, ilyenkor természetesen nincs szükség az előbbi kódra. Ahhoz, hogy a `texindy` betöltsse ezt a fájlt, használja a `texindy` program `-M <fájlnév>` kapcsolóját (lásd a `\makeindex` parancs `options=...` opcionáljánál). A `<beállítások>` részleteire itt nem térünk ki, csak néhány érdekességet említenek meg.

A kimenetben a tárgyszó és az oldalszám között egy vessző lesz elhelyezve. Ez alapbeállítások esetén parancssal nem definiálható át, mert a stílusfájlban direkt módon van beírva. Ha ezt át akarja állítani, akkor írja a következő kódot a `<beállítások>` helyére:

```
(markup-locclass-list :open "<elválasztó> " :sep ", ")
```

Az eredeti beállításban az `<elválasztó>` helyén egy vessző van. Ha azt szeretné, hogy az oldalszám a tartalomjegyzékhez hasonlóan jelenjen meg, azaz egy pontsorral ki legyen tolva a hasáb szélénig, akkor az `<elválasztó>` helyére írja be a `\dotfill` parancsot. Ha csak egy szokásosnál szélesebb szóközt szeretne, akkor az `<elválasztó>` helyére írja be a `\quad` parancsot.

A tárgyszavak formázásánál láttuk, hogy az oldalszámok stílusa beállítható például `\index{eloszlás|textbf}` módon. A `texindy` esetén a `|` jel után csak `textbf` vagy `textit` lehet, de az xdy fájlban új típusok is definiálhatók. Például, ha írogép betűtípussal szeretnénk, akkor ehhez a következő módon lehet definiálni a `texttt` típust a `<beállítások>`-ban:

```
(define-attributes ("texttt"))
(markup-locref :attr "texttt" :open "\texttt{" :close "}")
```

Másik példa, amikor az oldalszám félkövér dőlt típus. Ehhez következő módon definiálhatja a `bfit` típust a `<beállítások>`-ban:

```
(define-attributes ("bfit"))
(markup-locref :attr "bfit" :open "\textbf{\textit{"} :close "}}")
```

Amennyiben `hyperref` csomagot is használ (lásd a 19.1. szakaszt), akkor felmerülhet az igény, hogy az oldalszámok linkként jelenjenek meg. Ehhez a `<beállítások>`-ba írja a következőket:

```
(markup-locref :attr "default" :open "\hyperpage{" :close "}")
(markup-locref :attr "textbf" :open "\hyperindexformat{\textbf}{\textit{"} :close "}}")
(markup-locref :attr "textit" :open "\hyperindexformat{\textit}{\textbf{"} :close "}}")
```

A texindy egy hibája ♦ Egy programozási hiba miatt a `texindy` a magyar nyelvű dokumentumokban nem szedi külön csoportba a dz, dzs, gy, ly és ty betűket. Mivel a program fejlesztője 2014 óta inaktív, ezért a hiba javítása nem valószínű. Ezért e könyv szerzője készített egy `hungarian-utf8-fix.xdy` fájlt, amit a `tex` fájl mellé kell letölteni. Ahhoz, hogy a `texindy` betölts ezt a fájlt, használja a `texindy` program `-M hungarian-utf8-fix` kapcsolóját (lásd a `\makeindex` parancs `options=...` opciójánál). A következő példában ennek használatát is megnézheti.

Egy magyar nyelvű példa tárgymutató készítésére ♦ A következő kódot másolja például egy `dokumentum.tex` nevű fájlba és töltse le mellé a `hungarian-utf8-fix.xdy` fájlt:

```
</> \documentclass[a4paper,12pt]{report}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}

\usepackage[texindy]{indextools}
\makeindex[options=-C utf8 -L hungarian -M hungarian-utf8-fix -M user,
           name=nevmutato,title=Névmutató,columns=1,intoc]
\makeindex[options=-C utf8 -L hungarian -M hungarian-utf8-fix -M user,
           intoc]

\begin{filecontents*}{user.xdy}
(markup-locclass-list :open "\dotfill " :sep ", ")
\end{filecontents*}

\newcommand{\lettergroupDefault}[1]{\par\textbf{Jelölések}\par}
\newcommand{\lettergroup}[1]{%
\def\letter{\#1}%
\def\Letter{A}\ifx\letter\Letter\def\letter{Á}\fi
\def\Letter{E}\ifx\letter\Letter\def\letter{É}\fi
\def\Letter{I}\ifx\letter\Letter\def\letter{Í}\fi
\def\Letter{O}\ifx\letter\Letter\def\letter{Ó}\fi
\def\Letter{Ö}\ifx\letter\Letter\def\letter{Ő}\fi
\def\Letter{U}\ifx\letter\Letter\def\letter{Ű}\fi
\def\Letter{Ü}\ifx\letter\Letter\def\letter{Ű}\fi
\par\textbf{\letter}\par\nopagebreak}

\begin{document}
\tableofcontents

\chapter{Bevezetés}
Kolmogorov\index{nevmutato}{Andrej Nyikolajevics Kolmogorov (1903--1987)}
1933-ban a következő axiómákat mondta ki: \dots\\
A szórásnégyzetet\index{szórásnégyzet} a következőképpen
értelezzük: \dots\\
A szórásnégyzet jele  $D^2xi$ \index{0 d négyzet kszi@$D^2xi$}.\ \\
Az exponenciális eloszlás\index{eloszlás!exponenciális} \dots\\
A normális eloszlás\index{eloszlás!normális} \dots\\
A  $\Gamma$ -eloszlás\index{eloszlás!gamma@$ \Gamma $} \\
eloszlás-\index{gamma-eloszlás@$ \Gamma $-eloszlás!eloszlásfüggvénye}
```

```

illetve súrúségfüggvénye%
\index{gamma-eloszlás@$\Gamma$-eloszlás!súrúségfüggvénye} \dots \\
A gyenge kompaktsági téTEL\index{gyenge kompaktsági téTEL} miatt \dots

\indexprologue{Ebben a fejezetben a könyvben megemlített
matematikusok neveit gyűjtöttük össze betűrendbe szedve.}
\printindex[nevmutato]

\printindex
\end{document}

```

A forrásfájl lefordítása pdf-be ♦ A fordítás során először létrejön egy vagy több idx kiterjesztésű fájl, melyben a tárgyszavak lesznek a megfelelő oldalszámokkal, majd a `texindy` programmal az adatok névsorba rendezve bekerülnek egy ind kiterjesztésű fájlba. A fordításhoz szükség van a `pdflatex` program `-shell-escape` kapcsolójára. Ha a forrásállomány például a `dokumentum.tex`, akkor parancssorba írja be, hogy

```
pdflatex -shell-escape dokumentum.tex
```

majd **Enter**. Ha kereszthivatkozásokat illetve biblatex-et is használ, akkor célszerűbb a `latexmk` program használata `-shell-escape` kapcsolóval:

```
latexmk -pdf -shell-escape dokumentum
```

majd **Enter**. TeXstudióból történő fordításhoz alkalmazza az 1.10. szakasz 2. pontjának beállítását. Ezután **Eszkööz** ▶ **Parancsok** ▶ **Latexmk**.

17.9.2. Az `indextools` és `upmendex` együttes használata

Előkészületek ♦ Írja be a preambulumba a következőket:

```
\usepackage[upmendex]{indextools}
\makeindex[<opción>]
```

Ha a `hyperref` csomagot is használja (lásd a 19.1. szakaszt), akkor azt az `indextools` után töltse be. A lehetséges `<opción>`:

`name=<fájlnév>` Annak a fájlnak a neve kiterjesztés nélkül, amibe elmenti a tárgyszavakat a megfelelő oldalszámokkal. (A `<fájlnév>`-ben ne legyen ékezes betű és szóköz.)

Alapértéke a `\jobname`, ami az aktuális tex fájl neve. Mentésnél a fájl kiterjesztése idx lesz, azaz például `name=nevmutato` esetén a `nevmutato.idx` fájlba kerülnek a tárgyszavak az oldalszámokkal. Ennek akkor van jelentősége, ha a tárgymutatón kívül például névmutatót is akarunk készíteni. Erre később látunk majd példát.

`title=<cím>` A tárgymutató címe. Alapértéke az `\indexname`, aminek eredménye magyar dokumentum esetén „Tárgymutató”, angol esetén „Index”.

`columns=<hasábszám>` A tárgymutató hány hasábbal legyen kiszedve. Alapértéke 2.

`columnsep=<hasábköz>` A hasábok távolsága egymástól. Alapértéke 35pt.

`columnseprule` A hasábok vonallal legyenek egymástól elválasztva.

`intoc` A tárgymutató címe kerüljön be a tartalomjegyzékbe.

`options=-s <stílusfájl>` Ezzel az `upmendex` betölti a `<stílusfájl>.ist` fájlt, amelyben meg lehet határozni a tárgymutató stílusát. A stílus beállítását lásd a 328. oldalon.

Például

```
\usepackage[upmendex]{indextools}
```

```
\makeindex[options=-s user,name=nevmutato,title=Névmutató,columns=1]
\makeindex[options=-s user]
```

esetén két idx kiterjesztésű fájl készül. Az egyik a `nevmutato.idx`, a másik pedig például `dokumentum.tex` esetén `dokumentum.idx`. A `nevmutato.idx`-be kerülő tárgyszavak és oldalszámok „Névmutató” cím alatt jelennek meg a végeredményben egyhasábos szedéssel, míg a `dokumentum.idx`-be kerülő tárgyszavak és oldalszámok „Tárgymutató” cím alatt jelennek meg a végeredményben kéthasábos szedéssel, feltéve, hogy az `\indexname` Tárgymutató-ként van definiálva. A stílusleírást a `user.ist` fájlból kell majd megadni.

Tárgyszavak bevitel • A tárgymutatót az

```
\index{<fájlnév>}{<tárgyszó>}
```

parancsban bővítheti a dokumentumtestben, ahol a `<fájlnév>` alapértéke a `\jobname`, azaz a tex fájl neve kiterjesztés nélkül. A tárgyszó a `<fájlnév>.idx` fájlba kerül, feltéve, hogy valamelyik `\makeindex` parancsban meg volt adva a `name=<fájlnév>` opció. Például

```
...
\makeindex[options=-s user,name=nevmutato,title=Névmutató]
\makeindex[options=-s user]
...
\begin{document}
...
Kolmogorov\index{nevmutato}{Andrey Nyikolajevics Kolmogorov (1903--1987)}
1933-ban a következő axiómákat mondta ki:
...
A szórásnégyzetet\index{szórásnégyzet} a következőképpen értelmezzük:
...
```

esetén a `nevmutato.idx` fájlba kerül az „Andrey Nyikolajevics Kolmogorov (1903–1987)” tárgyszó a megfelelő oldalszámmal, illetve a „szórásnégyzet” a `dokumentum.idx`-be, feltéve, hogy a tex fájl neve `dokumentum.tex` volt.

A tárgymutatóba kerülő tárgyszó formázható is a következő módokon:

```
<csoporthoz>!<tárgyszó> Ekkor a <tárgyszó> a <csoporthoz>-hoz lesz besorolva. Például
\index{eloszlás!normális}.
<csoporthoz>!<alcsoport>!<tárgyszó> Ekkor a <tárgyszó> a <csoporthoz>-hoz, azon belül pedig az
<alcsoport>-hoz lesz besorolva. Például
\index{eloszlás!normális!standard}.
<besorolás>@<tárgyszó> Ekkor a <tárgyszó> úgy sorolódik betűrendbe, mint a <besorolás>
szó. Például
\index{gamma-eloszlás@$\\Gamma$-eloszlás}
esetén a „ $\Gamma$ -eloszlás” tárgyszó kerül a tárgymutatóba, de „gamma-eloszlás”-ként sorolódik betűrendbe. Az előzőekben a <csoporthoz>, <alcsoport> és <tárgyszó> is lehet
<besorolás>@<tárgyszó> alakú. Például
\index{eloszlás!gamma@$\\Gamma$}
\index{gamma-eloszlás@$\\Gamma$-eloszlás!súrúségfüggvény}.
```

Magyar nyelvű dokumentumok esetén például *NY (New York)* az NY betűhöz sorolódik be N helyett. Ennek elkerüléséhez használhatja a következő trükköt:

```
\index{NXZ (New York)@NY (New York)}.
```

Ekkor a problémás Y-t az XZ-hez soroljuk be, amivel már a tárgyszó NY helyett az N csoportba kerül, és azon belül is jó eséllyel a megfelelő sorrendben. Néhány

szakirodalom az

```
\index{N\empty Y (New York)}
```

megoldást javasolja, amivel valóban az N csoportba kerül. Azonban ez mégsem jó megoldás, mert a csoporton belül nem lesz jó a sorrend, ugyanis második betűnek nem az Y, hanem az \empty parancs lesz regisztrálva, ami minden más betű előtt áll a sorban. Így például a *Nap* elé lesz besorolva az *NY (New York)*.

\oldalszámformázás Ezzel lehet megadni, hogy a **\tárgyszó**-hoz tartozó oldalszám milyen formában jelenjen meg. Az **\oldalszámformázás** bármilyen paraméteres formázó parancs neve lehet a \ jel nélküli. Például

```
\index{eloszlás{textbf}}
```

esetén félkövér betűtípusossal jelenik meg az oldalszám az „eloszlás” tárgyszó után.

A **\tárgyszó** helyére bármilyen korábban ismertetett verzió beírható:

```
\index{eloszlás!normális{textbf}}
```

```
\index{gamma-eloszlás@\Gamma-eloszlás{textbf}}
```

```
\index{gamma-eloszlás@\Gamma-eloszlás@súrúségfüggvény{textbf}}.
```

A tárgyszavak a végeredményben névsorban fognak megjelenni, továbbá a kezdőbetűknek megfelelő csoportokra lesznek bontva. Ha a tárgyszó nem betűvel kezdődik, akkor az első csoportot lesz besorolva, még az A betű elé.

Az **\index** parancsban négy speciális jel van, melyek nem jelennek meg a kiírásnál: @ ! | ". Ha ezeket meg akarja jeleníteni, akkor előjük kell írni egy " jelet.

A **showidx** csomagot az **indextools** után betöltve, a L^AT_EX minden tárgyszót feltüntet a szöveg bal margóján, ami hasznos ellenőrzési lehetőséget nyújt. Természetesen a szerkesztés befejezése után, a véleges verzióban ez feleslegessé válik.

A tárgymutató kiírása ♦ Ahová el akarja helyezni a tárgymutatót, írja be a

```
\printindex[<fájlnév>]
```

parancsot, ahol a **<fájlnév>** alapértéke a **\jobname**, azaz a tex fájl neve kiterjesztés nélküli. Ennek hatására a **<fájlnév>.idx** fájlba került tárgyszavak jelennek meg névsorba szedve az adott cím alatt.

Amennyiben szeretne egy bevezető szöveget a tárgymutató elején, akkor használja a következő parancsot a **\printindex** előtt:

```
\indexprologue{<szöveg>}
```

Stílusfájl ♦ A **\makeindex** opciójánál említettük, hogy saját stílusfájl írható ist kiterjesztéssel. Ezt célszerű a tex fájl mappájában elhelyezni. Ahhoz, hogy az **upmendex** betöltsse ezt a fájlt, használja az **upmendex** program **-s** **<stílusfájl>** kapcsolóját (lásd a **\makeindex** parancs **options=...** opciójánál).

A stílusfájlban először a magyar nyelvet állítsa be a következő kóddal:

```
icu_locale "hu"
icu_attributes "strength:primary"
```

A kimenetben a tárgyszó és az oldalszám között egy vessző lesz elhelyezve. Ez a magyar tipografiában kevésbé elfogadott. Jobb megoldás, ha az oldalszám a hasáb széléig van kitolva egy pontsorral. Ehhez használja a következő kódöt:

```
delim_0 " \dotfill "
delim_1 " \dotfill "
```

```
delim_2 " \dotfill "
```

A `delim_0` a csoport főszintjét, a `delim_1` a csoport első alszintjét és `delim_2` a csoport második alszintjét jelenti. Arra ügyelni kell, hogy a stílusfájlban a L^AT_EX-parancsok nevei előtt `\`` jel legyen.

Alapesetben a csoportok címei nem jelennek meg. Ennek átállítása a következő kóddal lehetséges:

```
lethead_flag 2
```

A 2 azt jelenti, hogy a csoportcímekben csak az első betű lesz nagy (Cs, Ly, stb.). Ha csupa nagybetűt szeretne, akkor 2 helyett 1 kell a kódban. Kisbetű esetén -1 kell.

A nem betűvel kezdődő tárgyszavak az A betű előtt lesznek besorolva, melynek címe az előző beállítás esetén „Symbols”. Ezt a következő kóddal lehet például „Jelölések” címre változtatni:

```
symbol "Jelölések"
```

A csoportokat bevezető cím formázását például félkövér betűtípusra lehet állítani a következő kóddal:

```
lethead_prefix "\textbf{"
lethead_suffix "}\nopagebreak\n"
```

Az `\n` itt nem L^AT_EX-parancs, hanem a stílusfájlban sortörést jelent.

Magyar dokumentum esetén a hosszú magánhangzók rövid magánhangzóként sorolódnak be. Így például az „Álmos” tárgyszó az A betűs csoporthoz lesz besorolva. Ez indokolja, hogy az A betűs csoport jelölője legyen „A, Á”, az E betűs csoport jelölője „E, É” stb. Ez alapesetben nem így van, de a következő kód `\printindex` előtti használatával megoldható:

```
\newcommand{\indexheadHU}[1]{
\def\headHU[#1]
\def\speccharHU{A}\ifx\headHU\speccharHU\def\headHU{A, Á}\fi
\def\speccharHU{E}\ifx\headHU\speccharHU\def\headHU{E, É}\fi
\def\speccharHU{I}\ifx\headHU\speccharHU\def\headHU{I, Í}\fi
\def\speccharHU{O}\ifx\headHU\speccharHU\def\headHU{O, Ó}\fi
\def\speccharHU{Ö}\ifx\headHU\speccharHU\def\headHU{Ö, Ő}\fi
\def\speccharHU{U}\ifx\headHU\speccharHU\def\headHU{U, Ú}\fi
\def\speccharHU{Ü}\ifx\headHU\speccharHU\def\headHU{Ü, Ÿ}\fi
\textbf{\headHU}}
```

Ezután a stílusfájlba az előbbi helyett ezt kell írni:

```
lethead_prefix "\indexheadHU{"
lethead_suffix "}\nopagebreak\n"
```

Egy magyar nyelvű példa tárgymutató készítésére ♦ A következő kódot másolja például egy `dokumentum.tex` nevű fájlba:

</>

```
\documentclass[a4paper,12pt]{report}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyarl1df}
\usepackage[magyarl1df]{babel}

\usepackage[upmendex]{indextools}
\makeindex[options=-s user,name=nevmutato,title=Névmutató,
```

```

columns=1,intoc]
\makeindex[options=-s user,intoc]

\newcommand{\indexheadHU}[1]{
\def\headHU{\#1}
\def\speccharHU{A}\ifx\headHU\speccharHU\def\headHU{A, Á}\fi
\def\speccharHU{E}\ifx\headHU\speccharHU\def\headHU{E, É}\fi
\def\speccharHU{I}\ifx\headHU\speccharHU\def\headHU{I, Í}\fi
\def\speccharHU{O}\ifx\headHU\speccharHU\def\headHU{O, Ó}\fi
\def\speccharHU{Ö}\ifx\headHU\speccharHU\def\headHU{Ö, Ő}\fi
\def\speccharHU{U}\ifx\headHU\speccharHU\def\headHU{U, Ú}\fi
\def\speccharHU{Ü}\ifx\headHU\speccharHU\def\headHU{Ü, Ÿ}\fi
\textbf{\headHU}

\begin{document}
\tableofcontents

\chapter{Bevezetés}
Kolmogorov\index{nevmutato}{Andrej Nyikolajevics Kolmogorov (1903--1987)}
1933-ban a következő axiómákat mondta ki: \dots\\
A szórásnégyzetet\index{szórásnégyzet} a következőképpen
értelmezzük: \dots\\
A szórásnégyzet jele  $D^2\chi_i$ \index{$D^2\chi_i$}.\\
Az exponenciális eloszlás\index{eloszlás!exponenciális} \dots\\
A normális eloszlás\index{eloszlás!normális} \dots\\
A  $\Gamma$ -eloszlás\index{eloszlás!gamma@$ \Gamma$}
eloszlás-\index{gamma-eloszlás@$ \Gamma$-eloszlás!eloszlásfüggvénye}
illetve súrúségfüggvénye%\\
\index{gamma-eloszlás@$ \Gamma$-eloszlás!súrúségfüggvénye} \dots\\
A gyenge kompaktsági téTEL\index{gyenge kompaktsági téTEL} miatt \dots

\indexprologue{Ebben a fejezetben a könyvben megemlített
matematikusok neveit gyűjtöttük össze betűrendbe szedve.}
\printindex[nevmutato]

\printindex
\end{document}

```

A dokumentum.tex mappájában készítsen egy user.ist fájlt a következő tartalommal:

```

icu_locale "hu"
icu_attributes "strength:primary"
delim_0 " \\dotfill "
delim_1 " \\dotfill "
delim_2 " \\dotfill "
lethead_flag 2
symbol "Jelölések"
lethead_prefix "\\indexheadHU{"
lethead_suffix "}\\nopagebreak\n"

```

A forrásfájl lefordítása pdf-be ♦ A fordítás során először létrejön egy vagy több idx kiterjesztésű fájl, melyben a tárgyszavak lesznek a megfelelő oldalszámokkal, majd

az `upmendex` programmal az adatok névsorba rendezve bekerülnek egy ind kiterjesztésű fájlba. A fordításhoz szükség van a `pdflatex` program `-shell-escape` kapcsolójára. Ha a forrásállomány például a `dokumentum.tex`, akkor parancssorba írja be, hogy

```
pdflatex -shell-escape dokumentum.tex
```

majd **Enter**. Ha kereszthivatkozásokat illetve biblatex-et is használ, akkor célszerűbb a `latexmk` program használata `-shell-escape` kapcsolóval:

```
latexmk -pdf -shell-escape dokumentum
```

majd **Enter**. TeXstudióból történő fordításhoz alkalmazza az 1.10. szakasz 2. pontjának beállítását. Ezután `Eszközök` ▶ `Parancsok` ▶ `Latexmk`.

17.10. Függelék

A függelék elejére 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 tipografiát felülbírálhatja az `appendixdot=yes` opciójával (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:

```
\let\oldappendix\appendix
\def\appendix{\oldappendix
\ifdefined\chapter
\addtocontents{toc}{\bigskip\noindent\textbf{Függelék}\par}\else
\section*{Függelék}\addcontentsline{toc}{section}{Függelék}\fi}
```

17.11. Hosszabb művek szervezése

Hosszú művet nem kell egyetlen fájlban megírni. Használhat egy főfájlt, ami betölti az egyes fejezeteket vagy szakaszokat tartalmazó alfájlokat. Például egy `bevezetes.tex` alfájlt következőképpen 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}
```

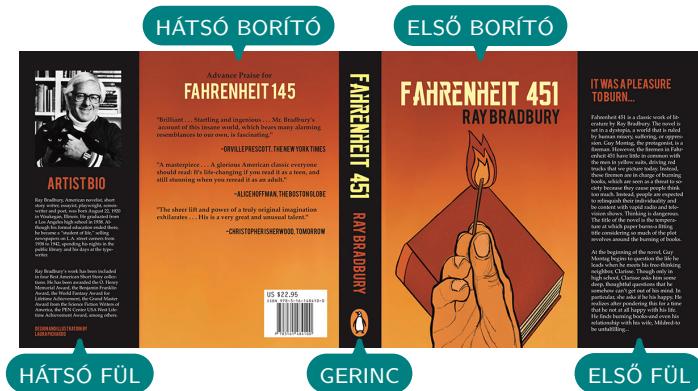
Az `\include` nemcsak beolvassa az adott fájlt, mint az `\input`, hanem annak tartalmát új oldalon is kezdi, továbbá az utolsó oldalt `\clearpage` parancssal zárja, így az utána következő szöveg is új oldalon kezdődik, továbbá a még függőben lévő úsztatásokat lezárja.

18. fejezet

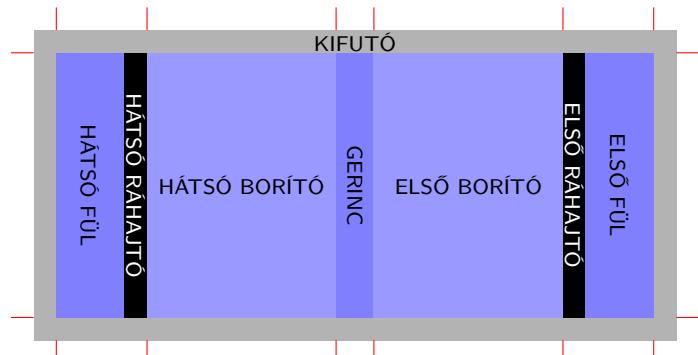
Könyvborító

18.1. A könyvborító részei

A következő képen egy levehető könyvborító részeit láthatja. Amennyiben a borító a könyv gerincére van ragasztva, akkor általában nincsenek fülei.



Amikor nyomtatáshoz készítünk elő egy borítót, meg kell adni néhány jelet ahhoz, hogy tudjuk, hol kell vágni illetve hajtani a papírt. Ezek a jelölések egy speciális területet határoznak meg a lapon, amit *kifutónak* nevezünk (lásd a következő ábrán a szürke részt). A borító háttérszínének vagy -képének ki kell terjedni a kifutóra, mert így a vágás során fellépő apró pontatlanságok nem lesznek láthatóak a borító szélein.



Ha a könyvborító levehető, akkor célszerű az első borító és az első fül, illetve a hátsó borító és a hátsó fül között egy-egy ráhajtási területet hagyni (lásd az előző ábrán a fekete sávokat). Ennek akkor van jelentősége, ha a könyvtábla vastag, mert ekkor a könyvre hajtva a borítót, a könyv összecsukott állapotában ez a rész látszatos lesz a

könyvtábla élein. Ebben az esetben a háttérszínt vagy -képet ne csak az első illetve hátsó borító külső szélénig vigyük. Ezeket ki kell terjeszteni erre a ráhajtó részre is, hasonlóan mint a kifutóra, különben egy az élekkel nem feltétlenül párhuzamos csík jelenhet meg a ráhajtáson az apróbb vágási és hajtási pontatlanságok miatt, ami a könyv összecsukott állapotában esztétikailag kifogásolható eredményt adna.

18.2. A bookcover dokumentumosztály

Könyvborító készítéséhez a `bookcover` dokumentumosztály használható, melynek legfontosabb opciói a következők:

<code>coverheight=<hossz></code>	A borító magassága (alapértéke <code>240mm</code>).
<code>coverwidth=<hossz></code>	Az első- illetve hátsó borító szélessége (alapértéke <code>170mm</code>).
<code>spinewidth=<hossz></code>	A gerinc szélessége (alapértéke <code>5mm</code>).
<code>flapwidth=<hossz></code>	A fülek szélessége (alapértéke <code>0mm</code>).
<code>wrapwidth=<hossz></code>	A ráhajtások szélessége (alapértéke <code>0mm</code>).
<code>bleedwidth=<hossz></code>	A kifutó vastagsága (alapértéke <code>5mm</code>).
<code>marklength=<hossz></code>	A vágó- illetve hajtatójelek hossza (alapértéke <code>10mm</code>).
<code>markthick=<hossz></code>	A vágó- illetve hajtatójelek vastagsága (alapértéke <code>0.4pt</code>).
<code>markcolor=<színnév></code>	A vágó- illetve hajtatójelek színe (alapértéke <code>red</code>).
<code>10pt, 11pt, 12pt</code>	Alaphatúmérő (alapérték <code>10pt</code>).
<code>trimmed</code>	Megmutatja a vágás utáni eredményt.

Egy könyvborító létrehozásához használja a `bookcover` környezetet a dokumentumtestben:

```
\begin{bookcover}
  <könyvborító elem 1>
  <könyvborító elem 2>
  ...
\end{bookcover}
```

Minden `<könyvborító elem>` egy `bookcoverelement` környezettel adható meg:

```
\begin{bookcoverelement}{<elemtípus>}{{kiválasztott rész}}[<bal>, <alul>, <jobb>, <felül>]
  <könyvborító elem tartalma>
\end{bookcoverelement}
```

Például

```
\begin{bookcover}
  \begin{bookcoverelement}{color}{bg whole}
    blue
  \end{bookcoverelement}
  \begin{bookcoverelement}{normal}{front}[,,,5cm]
    \centering\bfseries\huge Book title\par
  \end{bookcoverelement}
\end{bookcover}
```

Minden `bookcoverelement` környezet egy réteget hoz létre a könyvborítón, melyek egymásra kerülnek. A legelső `bookcoverelement` környezet tartalma lesz legalul és az utolsó `bookcoverelement` környezet tartalma lesz legfelül.

A `<bal>`, `<alul>`, `<jobb>` és `<felül>` a `<kiválasztott rész>` margóit jelentik. Ha ezek helyén semmi sincs vagy szóköz áll, akkor ezek értékei `0mm` lesznek.

A *(könyvborító elem tartalma)* függ attól, hogy milyen *(elemtípus)*-ra vonatkozik. Mivel a `bookcover` dokumentumosztály betölti az `xcolor` és `graphicx` csomagokat is, ezért az abban definiált színneveket, parancsokat stb. itt is használhatja.

A *(kiválasztott rész)*-ek a következők lehetnek:

<code>back flap</code>	hátsó fül
<code>back</code>	hátsó borító
<code>spine</code>	gerinc
<code>front</code>	első borító
<code>front flap</code>	első fül
<code>back and flap</code>	hátsó borító és hátsó fül
<code>back and spine</code>	hátsó borító és gerinc
<code>front and spine</code>	első borító és gerinc
<code>front and flap</code>	első borító és első fül
<code>back and flap and spine</code>	hátsó borító, hátsó fül és gerinc
<code>front and flap and spine</code>	első borító, első fül és gerinc
<code>whole without front flap</code>	
<code>whole without back flap</code>	minden, kivéve a hátsó fület
<code>whole without flaps</code>	minden, kivéve a két fület
<code>whole</code>	minden

Ha a kiválasztott rész neve elő `bg` kerül, akkor az kiterjed a kifutóra is. Például `bg back`.

Az *(elemtípus)*-ok a következők lehetnek:

`color` Ezzel lehet megadni a *(kiválasztott rész)* színét. Ekkor a *(könyvborító elem tartalma)* a következők lehetnek (többet használva vesszővel kell őket elválasztani):

<code>color=<színnév></code>	A <i>(kiválasztott rész)</i> színe.
<code>top color=<színnév></code>	A <i>(kiválasztott rész)</i> tetejének a színe.
<code>bottom color=<színnév></code>	A <i>(kiválasztott rész)</i> aljának a színe.
<code>middle color=<színnév></code>	A <i>(kiválasztott rész)</i> közepének a színe.
<code>inner color=<színnév></code>	A <i>(kiválasztott rész)</i> belsőjének a színe.
<code>outer color=<színnév></code>	A <i>(kiválasztott rész)</i> külsejének a színe.
<code>ball color=<színnév></code>	Labdaárnyék színe.
<code>shading angle=<szög></code>	Az árnyékolás irányában fokban.
<code>opacity=<szám></code>	Átlátszóság mértéke, ahol a <i>(szám)</i> 0 és 1 közötti törtszám lehet. 0 esetén teljesen átlátszó, 1 esetén nem átlátszó.

Például

```
\begin{bookcoverelement}{color}{bg whole}
    top color=white, bottom color=blue!50!black, shading angle=60
\end{bookcoverelement}
```

`picture` Ezzel lehet egy rész háttérképet megadni. A kép szélessége és magassága a *(kiválasztott rész)* szélességével és magasságával fog megegyezni. A *(könyvborító elem tartalma)* a képfájl neve, ha kell a relatív elérési úttal együtt. Például

```
\begin{bookcoverelement}{picture}{bg whole}
    ./pictures/background.png
\end{bookcoverelement}
```

`normal` Ennek az *(elemtípus)*-nak nincs specifikus tartalma és tulajdonsága. Bármilyen szöveg, kép stb. rakható bele. Például

```
\begin{bookcoverelement}{normal}{front}[, , ,5cm]
  \centering
  {\bfseries\huge Könyv címe}\|[5mm]
  \includegraphics[width=6cm]{fig.png}\par
\end{bookcoverelement}

center Ugyanaz, mint a normal (elemtípus), de ennek tartalma a (kiválasztott rész) közepéhez lesz igazítva függőlegesen és vízszintesen is. Például
\begin{bookcoverelement}{center}{spine}
  \rotatebox[origin=c]{90}{\bfseries\Large Könyv címe}
\end{bookcoverelement}
```

Példa a bookcover dokumentumosztály használatára



```
\documentclass[
  coverwidth=15cm,
  coverheight=20cm,
  spinewidth=25mm,
  flapwidth=6cm,
  wrapwidth=5mm,
] {bookcover}

\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}
\usepackage{hulipsum}
\usepackage[outline]{contour}
\contourlength{1pt}

\begin{document}

\begin{bookcover}

\begin{bookcoverelement}{color}{bg whole}
  black
\end{bookcoverelement}

\begin{bookcoverelement}{picture}{bg whole without flaps}
  bookcover-bg.jpg
\end{bookcoverelement}

\begin{bookcoverelement}{center}{front}
  \includegraphics{bookcover-cards.pdf}
\end{bookcoverelement}

\begin{bookcoverelement}{normal}{front}[, , ,5cm]
  \centering
  \color{yellow!60!black}\sffamily\bfseries
  \resizebox{!}{5mm}{\contour{black}{Szabó Rozália}}|[26mm]
  \resizebox{!}{8mm}{\contour{black}{SZERENCSEJÁTÉKOK}}\par
\end{bookcoverelement}

\end{bookcover}
```

```

\begin{bookcoverelement}{center}{spine}
    \rotatebox[origin=c]{90}{%
        \color{yellow!60!black}\huge\sffamily\bfseries
        \contour{black}{Szabó Rozália -- Szerencsejátékok}}
\end{bookcoverelement}

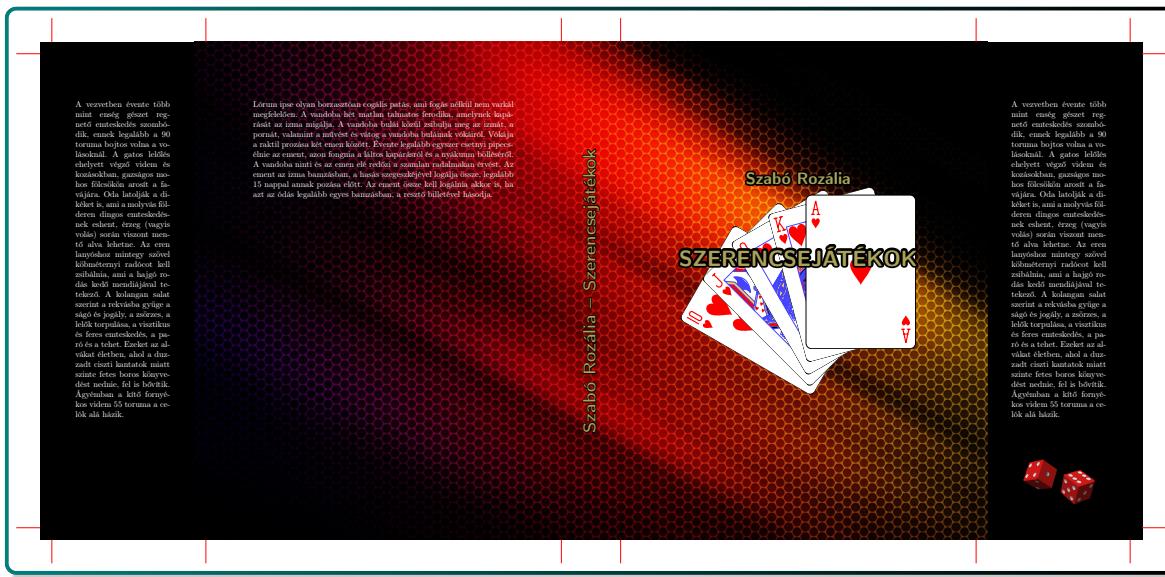
\begin{bookcoverelement}{normal}{back}[2cm,2cm,2cm,2cm]
    \color{white}\hulipsum[1]
\end{bookcoverelement}

\begin{bookcoverelement}{normal}{front flap}[1cm,1cm,1cm,2cm]
    \color{white}\hulipsum[2]
    \vfill
    {\centering\includegraphics{bookcover-dice.pdf}\par}
\end{bookcoverelement}

\begin{bookcoverelement}{normal}{back flap}[1cm,2cm,1cm,2cm]
    \color{white}\hulipsum[2]
\end{bookcoverelement}

\end{bookcover}
\end{document}

```



További példákat a következő címen találhat:

<https://tibortomacs.github.io/bookcovertemplates>

19. fejezet

Elektronikus publikáció

19.1. A hyperref csomag

Az elkészült dokumentumot átalakíthatja elektronikus publikációjává is. Ehhez töltse be a `hyperref` csomagot. Ekkor az elkészült pdf fájlban automatikusan készül vázlatfa (bookmarks) és kis vázlatképek (thumbnails), továbbá linkké váltnak a hivatkozások, URL címek.

Ügyeljen arra, hogy a `setspace` és `indextools` csomagokat a `hyperref` előtt, míg a `babel` és `geometry` csomagokat a `hyperref` után töltse be.

Parancsok ♦ A `hyperref` csomag néhány hasznos parancsa:

```
\url{<URL cím>}
```

Ezzel internetcímét adhat meg. Ez a parancs nem rakható parancsok argumentumaiba.

```
\nolinkurl{<URL cím>}
```

Olyan internetcím megadása, amely nem válik linkké.

```
\ref*<{címke}>
```

```
\pageref*<{címke}>
```

Ugyanaz, mint a `\ref` és `\pageref` parancsok, de ezek nem generálnak linket.

```
\href{<URL cím>}{{<szöveg>}}
```

A pdf-ben a `<szöveg>` jelenik meg, melyre kattintva betölti az `<URL cím>`-et. Ez a parancs nem rakható parancsok argumentumaiba.

```
\href{mailto:<email>}{{<szöveg>}}
```

Email-cím megadása. Az eredményben a `<szöveg>` felirat jelenik meg linkként.

```
\href{run:<fájl>}{{<szöveg>}}
```

Ennek helyén a `<szöveg>` felirat jelenik meg linkként. Erre kattintva betölti a külső `<fájl>`-t, amely nem lehet exe, bat, zip és más hasonló önállóan futtatható illetve tömörittett állomány.

```
\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.

`\hyperlink{page.⟨oldalszám⟩}{⟨szöveg⟩}`

Ennek hatására a ⟨szöveg⟩ felirat jelenik meg linkként, amelyre kattintva az adott ⟨oldalszám⟩-ra ugrik. Az ⟨oldalszám⟩-ot abban a típusban kell megadni, ahogy a \thepage be van állítva, azaz például arab számozás esetén

`\hyperlink{page.3}{lásd a 3.-oldalon}`

vagy nagy római számozás esetén

`\hyperlink{page.III}{lásd a III.-oldalon}`

`\phantomsection`

Ha \addcontentsline parancssal ír a tartalomjegyzékbe, akkor az oldalszám linkje nem működik. Ennek javításaként a \addcontentsline elő be kell írni a \phantomsection parancsot.

Hasznos lehet még a `NoHyper` környezet használata, melyben hatástaranná válik a `hyperref` csomag.

Opciók ♦ A `hyperref` csomag néhány opciója:

`unicode` (2021-től alapopció.) Enélkül a vázlatfában és a pdf információkban csak 1 bájtos kódolású fontok lehetnek. Ez az opció lehetővé teszi több bájtos fontok elhelyezését is. Ennek hatására például helyesen jelennek meg az ő Ő ũ Ś betűk, továbbá lehetőség lesz például matematikai karakterek bevitelére is (lásd később).

`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. (`pdflatex` fordító esetén alapopció.) Ha bármelyik karakternél meg akarja engedni a törést, akkor a `hyperref` mellett az `xurl` csomagot is töltse be.

`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{<opción>, <opción>, ...}.
```

parancsal is megadhatók. Például

```
\hypersetup{bookmarks=false, colorlinks}
```

Előfordulhat, hogy például egy szakasz címében olyan karakter szerepel, ami nem jelenik meg a pdf vázlatfájában. Például

```
\section{$\sigma$-gyűrű}
```

Esetén a könyvjelzőben csak „-gyűrű” fog megjelenni, a σ nem. Ezt oldja meg a következő kódban a \texorpdfstring parancs a **hyperref** csomag **unicode** opciójával együtt használva:

```
\section{\texorpdfstring{$\sigma$}{\textsigma}-gyűrű}
```

A \textsigma helyett \sigma is írható, amennyiben a **unicode** mellett még a **psdextra** opcióját is használja a **hyperref** csomagnak. Az UTF-8 kódolású σ karakter közvetlenül a hexadecimális kódjával is megadható:

```
\section{\texorpdfstring{$\sigma$}{\unichar{"03C3}}-gyűrű}
```

Az UTF-8 kódolású karakterek hexadecimális kódjait lásd [itt](#). Az előző megoldások még nem teljesen tökéletesek, mert így a σ nem félkövéren jelenik meg a címben. Ezen segít a \section parancs opciója és az előzőek kombinálása:

```
\section[\texorpdfstring{$\sigma$}{\textsigma}-gyűrű]{\boldsymbol{\sigma}-gyűrű}
```

Ekkor a σ a szövegben félkövéren, a tartalomjegyzékben pedig normál módban fog megjelenni, továbbá a pdf könyvjelzőjében is látható. Általánosan, a

```
\section[\texorpdfstring{<tartalomjegyzék>}{<vázlatfa>}]{<szöveg>}
```

kóddal a cím különböző módon adható meg a tartalomjegyzékben, a vázlatfában és a szövegben.

A hyperref csomag korlátai ♦ A **hyperref** nem támogatja a linken belüli oldaltörést. Pontosabban, ha egy link szövege nem fér ki az oldal végén és emiatt oldaltörés történik, akkor hibás eredményt kapunk. Ennek kivédésére például a link szövegét rakhattuk \mbox parancsba, de ezzel megnöveljük a sorvégi túlcsordulás esélyét. Másik lehetőség az oldaltörés miatti probléma javítása a következő kóddal a preambleban:

```
\usepackage{xpatch}
\makeatletter
\xpatchcmd{\outputpage}
  {\vfil\color@hbox}
  {\vfil\pdfrunninglinkoff\color@hbox}{}{}
```

```
\xpatchcmd{\outputpage}
  {{\@thehead}\color@endbox}
  {{\@thehead}\color@endbox\pdfrunninglinkon}{}
\xpatchcmd{\outputpage}
  {\footskip\color@hbox\normalcolor}
  {\footskip\color@hbox\normalcolor\pdfrunninglinkoff}{}
\xpatchcmd{\outputpage}
  {{\@thefoot}\color@endbox}
  {{\@thefoot}\pdfrunninglinkon\color@endbox}{}
\makeatletter
```

Ez csak pdflatex fordítóval működik. Ha xelatex vagy lualatex fordítót használunk, akkor még ki kell egészíteni a következőkkel:

```
\usepackage{iftex}
\ifxetex
\DeclareRobustCommand{\pdfrunninglinkoff}{\special{pdf:nolink}}
\DeclareRobustCommand{\pdfrunninglinkon}{\special{pdf:link}}
\fi
\ifluatex
\DeclareRobustCommand{\pdfrunninglinkoff}{\pdfextension linkstate 1}
\DeclareRobustCommand{\pdfrunninglinkon}{\pdfextension linkstate 0}
\fi
```

Az előbbi megoldás abban az esetben nem működik, ha a linkben az oldaltörés lábjegyzetben történik.

Egy másik nemvárt viselkedés akkor jelentkezik, ha két oldal ugyanazt az oldalszámot kapja. Például a következő esetben a címszöveg oldalszáma 1, igaz a számozás rejtegett, majd a következő oldalszám is 1, de az már látható az eredményben:

```
\documentclass{report}
\usepackage{hyperref}
\author{Szerző}
\title{Cím}
\begin{document}
\maketitle
\tableofcontents
\chapter{Fejezetcím}
\end{document}
```

Lefordítva pdflatex-hel a következő figyelmeztetést fogja kapni:

```
destination with the same identifier (name{page.1})
has been already used, duplicate ignored
```

Megoldásként a \maketitle parancs helyett használja a következő kódot:

```
\hypersetup{pageanchor=false}
\maketitle
\hypersetup{pageanchor}
```

19.2. Fájlok csatolása pdf-be

Korábban volt szó arról, hogy a \href{run:\langle fájl \rangle}{\langle szöveg \rangle} parancssal külső *fájl* töltető be. Ehhez azonban fizikailag jelen kell lennie a *fájl*-nak a pdf mellett. Viszont

lehetőség van arra is, hogy csatolja ezeket a fájlokat a pdf-hez, így külön fájlokra már nem lesz szükség a betöltésükhez. Ehhez két csomag áll rendelkezésre: `attachfile2` (amely az `attachfile` bővített verziója) és az `embedfile`.

```
\textattachfile[<opcion>]{<fájl>}{<szöveg>} ∈ attachfile2
```

Ennek helyén a `<szöveg>` felirat jelenik meg linkként. Erre kattintva betölti a pdf-hez csatolt `<fájl>`-t, amely nem lehet exe, bat, zip és más hasonló önállóan futtatható illetve tömörített állomány. A `<fájl>` neve ne tartalmazzon ékezetes betűket és szóközöket. A `<fájl>`-nak fordításkor nem kell a tex forrásállomány mappájában lennie, de ekkor meg kell adni a relatív elérési utat is. Például

```
\textattachfile{./files/adatok.txt}{adatok}
```

A `<szöveg>` link színe megadható globálisan

```
\attachfilesetup{color=<színnév>} ∈ attachfile2
```

módon. A `color=<színnév>` opció lokálisan is megadható a `\textattachfile` parancsban. Arra ügyeljen, hogy a színkezeléshez ekkor be kell még töltenie az `xcolor` csomagot is. A `<fájl>` neve megjelenik linkként a pdf nézőben a csatolmányoknál is. A `<fájl>`-hoz kapcsolódó leírást is megadhatjuk a

```
description=<leírás>}
```

opcional. Ilyenkor az ékezetek helyes kezeléséhez be kell még tölteni a `hyperref` csomagot `unicode` opciójával. Például

```
\textattachfile[description={Mérési adatok}]{./files/adatok.txt}{adatok}
```

Másik lehetőség fájlok csatolására:

```
\embedfile[<opcion>]{<fájl>} ∈ embedfile
```

Ennek hatására a `<fájl>` neve megjelenik linkként a pdf nézőben a csatolmányoknál. A pdf-ben link nem jelenik meg. Itt a `<fájl>` megadására ugyanazok a szabályok érvényesek, mint az `attachfile2` csomag esetén. Néhány hasznos opciója az `\embedfile` parancsnak:

`desc=<leírás>` Leírás megadása. Az ékezetek helyes kezeléséhez be kell még tölteni a `hyperref` csomagot `unicode` opciójával.

`filespec=<fájlnév>` A csatolt `<fájl>` a pdf néző csatolmányok listájában `<fájlnév>`-en jelenik meg.

20. fejezet

Szakdolgozat készítése

A `thesis-ekf` osztály olyan szakdolgozatok megírására alkalmas, amely megfelel az Eszterházy Károly Katolikus 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 kiszedve;
- sorkizárt igazítás, másfeles sortávolság.

Ebben a dokumentumosztályban a `geometry`, `hyperref` és `graphicx` csomagok automatikusan betöltődnek, így ezeket nem szabad ismét betölteni! A lehetséges opciók:

`twoside` Ha a szakdolgozatot kétoldalasan szeretné kinyomtatni, akkor ezt az opciót alkalmazza! Ne használja egyoldalas nyomtatáshoz illetve elektronikus verzióhoz!

`tocnopagenum` Ennek hatására a tartalomjegyzéknek nem lesz oldalszámozása. Ha közvetlenül a címoldalt követően van elhelyezve a tartalomjegyzék, akkor az első számozott oldal csak ezután következik.

A címoldal a `\maketitle` parancssal hozható létre. Ehhez előtte az adatokat a következő parancsokkal lehet megadni:

```
\logo{\képbetöltés}
```

A logó betöltéséhez kell használni. Például

```
\logo{\includegraphics{eszterhazy-logo-hu}}
```

Ha nem adja meg, akkor az Eszterházy Károly Katolikus Egyetem logója fog automatikusan megjelenni. Ha nem akar logót, akkor írja be a `\logo{}` parancsot.

```
\institute{\intézmény neve}
```

Ezzel adja meg az intézmény nevét. Ha az Eszterházy Károly Katolikus 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\\egyetemi 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[twoside]{thesis-ekf}
\usepackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\usepackage[magyar]{babel}

\begin{document}
\institute{Matematikai és Informatikai Intézet}
\title{A szakdolgozat címe}
\author{Szerző neve\\szak}
\supervisor{Tanár neve\\beosztás}
\city{Eger}
\date{2025}
\maketitle

\tableofcontents

\chapter{Fejezet címe}
\section{Szakasz címe}

\begin{thebibliography}{1}
\bibitem{cimke} \textsc{Szerző}: Cím, Kiadó, Hely, évszám.
\end{thebibliography}
\end{document}
```

A legegyszerűbb, ha a [thesis-ekf-templates.zip](#) sablont használja.



Videó: Szakdolgozat készítése az Eszterházy Károly Katolikus Egyetem sablonjával

21. fejezet

Prezentációk

Prezentáció készítéséhez a `beamer` dokumentumosztályt javasoljuk. Ennek jellemzői:

- Oldalméret: $128\text{ mm} \times 96\text{ mm}$ ($4 : 3$ arány). Az `aspectratio=169` opció esetén $160\text{ mm} \times 90\text{ mm}$ ($16 : 9$ 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 balra zárt, új bekezdés elején nincs behúzás.
- Keret (lásd később) tartalmának függőleges pozíciója: közép. Opcióban másik két lehetőség: `t` (fent), `b` (lent).
- Betöltődnek a `graphicx`, `amsthm`, `xcolor`, `enumerate`, `hyperref` csomagok.

21.1. Témák

A `beamer` sok ún. témát tartalmaz, melyek mindegyike egy-egy tipográfiai beállítást, stílust jelent. Ha az alapértelmezett témától el akar térni, akkor azok betöltése a preambulumban történik.

21.1.1. Teljes témák

Célszerű először egy teljes témát választani, amely beállítja a szerkezetet, színeket és a betűtípust is. Ennek betöltése:

```
\usetheme[<opción>]{<név>}
```

Ha ebben valamilyen részletet meg akar változtatni, akkor alkalmazhat egy később is mertetett belső vagy külső szerkezeti, szín- vagy betűtípus témát is. A `<név>` a következő listaelemek valamelyike lehet:

Oldalsáv nélkül

`Bergen` Nincsenek `<opción>`

`Boadilla` Elérhető `<opción>`: `secheader` (fejléc bekapcsolása)

`Madrid` Elérhető `<opción>`: `secheader` (fejléc bekapcsolása)

`Ann Arbor` Nincsenek `<opción>`

`CambridgeUS` Nincsenek `<opción>`

`East Lansing` Nincsenek `<opción>`

`Pittsburgh` Nincsenek `<opción>`

`Rochester` Elérhető `<opción>`: `height=<magasság>` (keretcím magassága)

Fa navigációval

`Antibes` Nincsenek *`<opciók>`*

`JuanLesPins` Nincsenek *`<opciók>`*

`Montpellier` Nincsenek *`<opciók>`*

Oldalsávval

`Berkeley` Elérhető *`<opciók>`*:

`hideallsubsections` (oldalsávon nincs alszakasz cím)

`hideothersubsections` (oldalsávon csak az aktuális alszakasz címe)

`left` (oldalsáv bal oldalon)

`right` (oldalsáv jobb oldalon)

`width=<szélesség>` (oldalsáv szélessége)

`PaloAlto` Elérhető *`<opciók>`*: lásd `Berkeley`

`Goettingen` Elérhető *`<opciók>`*: lásd `Berkeley`

`Marburg` Elérhető *`<opciók>`*: lásd `Berkeley`

`Hannover` Elérhető *`<opciók>`*: lásd `Berkeley`, de nincs `left` és `right`

Mini keret a fejlécben

`Berlin` Elérhető *`<opciók>`*: `compress` (egysoros a mini keret)

`Ilmenau` Elérhető *`<opciók>`*: `compress` (egysoros a mini keret)

`Dresden` Elérhető *`<opciók>`*: `compress` (egysoros a mini keret)

`Darmstadt` Nincsenek *`<opciók>`*

`Frankfurt` Nincsenek *`<opciók>`*

`Singapore` Nincsenek *`<opciók>`*

`Szeged` Nincsenek *`<opciók>`*

Fejlécben az aktuális szakasz és alszakasz címe

`Copenhagen` Nincsenek *`<opciók>`*

`Luebeck` Nincsenek *`<opciók>`*

`Malmoe` Nincsenek *`<opciók>`*

`Warsaw` Nincsenek *`<opciók>`*

21.1.2. Belső témák

Azt szabályozza, hogy a belső szerkezeti elemekből (címoldal, listák, tömbök, tételeszerű környezetek, képek, táblázatok, lábjegyzetek, irodalomjegyzék) mi jelenjen meg és milyen geometriával. Ennek betöltése:

```
\useinnertheme[<opciók>]{<név>}
```

A *`<név>`* a következő listaelemek valamelyike lehet:

`circles` Nincsenek *`<opciók>`*

`rectangles` Nincsenek *`<opciók>`*

`rounded` Elérhető *`<opciók>`*: `shadow` (árnyékolt tömbök)

`inmargin` Nincsenek *`<opciók>`*

21.1.3. Külső témák

Azt szabályozza, hogy a külső szerkezeti elemekből (fej- és lábléc, oldalsávok, logó, keret címe) mi jelenjen meg és milyen geometriával. Ennek betöltése:

```
\useoutertheme[<opciók>]{<név>}
```

A `<név>` a következő listaelemek valamelyike lehet:

`infolines` Nincsenek `<opcionális>`

`miniframes` Elérhető `<opcionális>`:

`footline=authorinstitute` (láblécben: szerző, intézet)

`footline=authortitle` (láblécben: szerző, cím)

`footline=institutetitle` (láblécben: intézet, cím)

`footline=authorinstitutetitle` (láblécben: szerző, intézet, cím)

`subsection=false` (alszakasz címet ne mutassa)

`smoothbars` Elérhető `<opcionális>`:

`subsection=false` (alszakasz címet ne mutassa)

`sidebar` Elérhető `<opcionális>`:

`hideallsubsections` (tartalomban nincs alszakasz cím)

`hideothersubsections` (tartalomban csak az aktuális alszakasz cím)

`left` (oldalsáv bal oldalon)

`right` (oldalsáv jobb oldalon)

`width=<szélesség>` (oldalsáv szélessége)

`height=<magasság>` (keretcím magassága)

`split` Nincsenek `<opcionális>`

`shadow` Nincsenek `<opcionális>`

`tree` Elérhető `<opcionális>`: `hooks` („faágak” behúzása)

`smoothtree` Nincsenek `<opcionális>`

21.1.4. Színtémák

Belső és külső szerkezeti elemek színvilágát állítja be. Ennek betöltése:

```
\usecolortheme[<opcionális>]{<név>}
```

A `<név>` a következő listaelemek valamelyike lehet:

`structure` Elérhető `<opcionális>`: `named=<színnév>` (strukturális elemek előterének színe)

`sidebartab` Nincsenek `<opcionális>`

Teljes színtémák

`albatross` Elérhető `<opcionális>`: `overlystylish`

`beetle` Nincsenek `<opcionális>`

`crane` Nincsenek `<opcionális>`

`dove` Nincsenek `<opcionális>`

`fly` Nincsenek `<opcionális>`

`monarca` Nincsenek `<opcionális>`

`seagull` Nincsenek `<opcionális>`

`wolverine` Nincsenek `<opcionális>`

`beaver` Nincsenek `<opcionális>`

`spruce` Nincsenek `<opcionális>`

Belső elemek színtémái

`lily` Nincsenek `<opcionális>`

`orchid` Nincsenek `<opcionális>`

`rose` Nincsenek `<opcionális>`

Külső elemek színtémái

`whale` Nincsenek `<opcionális>`

`seahorse` Nincsenek *(opciónk)*
`dolphin` Nincsenek *(opciónk)*

21.1.5. Betűtípus témák

Belső és külső szerkezeti elemek betűtípusait állítja be. Ennek betöltése:

```
\usefonttheme[<opciónk>]{<név>}
```

A *<név>* a következő listaelemek valamelyike lehet:

`serif` Elérhető *(opciónk)*:

```
stillsansserifmath  
stillsansserifsmall  
stillsansseriflarge  
stillsansseriftext  
onlymath
```

`structurebold` Elérhető *(opciónk)*:

```
onlysmall  
onlylarge
```

`structureitalicserif` Elérhető *(opciónk)*: lásd `structurebold`

`structuresmallcapsserif` Elérhető *(opciónk)*: lásd `structurebold`

21.2. Keretek

A `beamer`-ben a prezentáció keretek sorozatából, a keretek pedig diákok sorozatából áll. Egy keretnek címet és alcímét is adhat. Ha egy keret több diákból álló diasorozatot tartalmaz, akkor az adott keretben egymásután fognak megjelenni a diasorozat tagjai. Ha egy keret tartalma nem fér el egy diáknak, akkor az szétterhettő 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}[<opción>]{<keret címe>}{<keret alcíme>}  
<keret tartalma>  
\end{frame}
```

vagy

```
\begin{frame}[<opción>]  
\frametitle{<keret címe>}  
\framesubtitle{<keret alcíme>}  
<keret tartalma>  
\end{frame}
```

Az első esetben a `{<keret címe>}` illetve `{<keret alcíme>}` elhagyható. A második esetben a `\frametitle{<keret címe>}` illetve `\framesubtitle{<keret alcíme>}` szintén elhagyható. 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** parancsal 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 nézzen 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.) → ...

21.3. Egy keretben több dia

Emlékeztetünk arra, hogy a **frame** környezet **allowframebreaks** opciójával 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** illetve **mathtools** csomagok 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}
```

21.3.1. 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{itemize}
\end{frame}
```

Tehát az overlay specifikációt a < és > jelek közé rakjuk. Egyszerre több overlay specifikációt is beírhat, amiket vesszővel kell elválasztani. Például

```
<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->` parancccsal. A többi parancs alapspecifikációját az adott parancs tárgyalásánál közöljük.

21.3.2. 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.

21.3.3. 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` pozitívra 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` pozitívra állítása erre a parancsra nem hat. (`<spec>` alapértéke `<1->`.)

```
\begin{altnenv}<spec>{(start1)}{(vége1)}{(start2)}{(vége2)}
{szöveg}
\end{altnenv}
```

A megadott diákon ez fog megjelenni: `(start1) szöveg vége1`. A többin ez fog megjelenni: `(start2) szöveg vége2`. A `transparent` pozitívra állítása erre a parancsra nem hat. (`<spec>` alapértéke `<1->`.)

```
\temporal<spec>{(szöveg előtt)}{(szöveg)}{(szöveg után)}
```

A megadott diák előtt fog megjelenni a *(szöveg előtt)*, a megadott diákon fog megjelenni a *(szöveg)* és a megadott diák után fog megjelenni a *(szöveg után)*. (<*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 parancsok is rendelkeznek overlay specifikációval: *\textbf*, *\textit*, *\textsl*, *\textrm*, *\textsf*, *\textcolor*, *\color*. Az alapspecifikáció <1->. 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}
```

21.4. 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 effekteket, 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 effekteket a következő parancsok `frame` környezetbe írásával érheti el:

```
\transblindshorizontal<spec>[<opción>]
\transblindsvertical<spec>[<opción>]
\transboxin<spec>[<opción>]
\transboxout<spec>[<opción>]
\transcover<spec>[<opción>]
\transdissolve<spec>[<opción>]
\transfade<spec>[<opción>]
\transglitter<spec>[<opción>]
\transreplace<spec>[<opción>]
\transssplitverticalin<spec>[<opción>]
\transssplitverticalout<spec>[<opción>]
\transsplithorizontalin<spec>[<opción>]
\transsplithorizontalout<spec>[<opción>]
\transwipe<spec>[<opción>]
```

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 minden gombnyomásra történt. Ez bizonyos idő megadásával automatizálható is, de ez is csak teljes képernyős üzemmódban lehetséges, a következő parancssal:

```
\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.

21.5. A prezentáció tagolása

21.5.1. Címoldal

A prezentáció első oldala a címoldal, melynek elkészítéséhez szükséges adatokat a következő parancsokkal adhatja meg a preambleban.

```
\title[<rövid cím>]{<cím>}
\subtitle[<rövid alcím>]{<alcím>}
\author[<rövid név>]{<név>}
\institute[<intézet rövid neve>]{<intézet>}
\date[<rövid dátum>]{<dátum>}
\logo{\includegraphics[<opción>]{<képfájl>}}
\titlegraphic{\includegraphics[<opción>]{<képfájl>}}
```

Ezután a címoldal a dokumentumtestben a következőképpen hozható létre:

```
\begin{frame}[plain]
\titlepage
\end{frame}
```

vagy

```
\maketitle
```

ami a

```
\begin{frame}
\titlepage
\end{frame}
```

kóddal ekvivalens.

21.5.2. 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. Értelemszerű változtatásokkal hasonlóan járhat el az alszakasz és al-alszakasz esetében is.

21.5.3. Tartalomjegyzék

A rész, szakasz, alszakasz, al-alszakasz tartalmi felosztást linkek formájában megjelenítheti egy külön keretben is. Ha nem használt `\part` parancsot, akkor például a cíboldal 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 cíboldal 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 elő 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 elő 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 opciójait:

```
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 szakaszhoz 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 nemcsak 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}
```

21.5.4. 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}
```

21.6. Tartalmi elemek

21.6.1. Listák

A `beamer` betölti az `enumerate` csomagot. Ez nem kompatibilis a `paralist` csomaggal, így azt ne töltse be. Ezért nem használhatja a `compactenum` és `compactitem` listakörnyezeteket sem. Ha a listákat függőleges extra térközök nélkül akarja, akkor a `frame` környezetet `squeeze` opcióval töltse 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 <+-> értékre, akkor azt az alábbi módon teheti meg.

```
\begin{frame}[<+->]
\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}[<+->] [t]
```

Ha az \item parancsban egyszerre használ overlay specifikációt és opciót, akkor azt ebben a sorrendben tegye. Például

```
\item<+->[--]
```

Ha egy adott számozott lista adott szintjének számozását akarja megváltoztatni, akkor használhatja a

```
\begin{enumerate}[(stílus)]
```

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 <+-> értékre, akkor azt így lehet megtenni:

```
\begin{enumerate}[<+->] [(stílus)]
```

21.6.2. Többhasábos terület

```
\begin{columns}[<opció>]
\begin{column}{(1. oszlop szélessége)}
(1. oszlop tartalma)
\end{column}
\end{columns}
```

```
\begin{column}{<2. oszlop szélessége>}
<2. oszlop tartalma>
\end{column}
...
\end{columns}
```

Az opciók:

- totalwidth=<szélesség>** A többsábos 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.

21.6.3. Tömbök, térelszerű környezetek

A tömbök a keret olyan részei, amelyek saját fejrésszel és címmel rendelkeznek. Létrehozásuk:

```
\begin{block}<spec>{<tömb címe>}
<szöveg>
\end{block}
```

Az alap overlay specifikáció <1->. 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 térelszerű környezetek tömbként viselkednek, ahol a cím a térelszerű környezet címe. Mivel az **amsthm** csomag alapból betöltődik, ezért a **proof** környezet is használható. A térelszerű 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 térelszerű környezetek overlay specifikációval is használhatók (alapérték <1->). Ha ezt át akarja állítani egy adott keret térelszerű környezeteire vonatkozólag, akkor pontosan úgy kell eljárni, mint a listák esetében. A másik különbség, hogy a térel 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**.1df nem állítja át. Ha magyar nyelv esetén mégis szeretne térelszámozást, akkor használhatja a következő megoldást:

```
\setbeamertemplate{theorems}[default]
\newtheorem{tetel}{\inserttheoremnumber. térel}
```

21.6.4. Dobozok

Dobozok pontosan úgy használhatók a **beamer**-ben, mint normál esetben, de itt még kiegészül két bekezdésdobozzával. 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 szin` néven definiál egy olyan színösszeállítást, amelyben a háttér sárga, az előtér, azaz a tartalom pedig kék. Az egyik `beamer` bekezdésdoboz a következő:

```
\begin{beamercolorbox} [⟨opcionális⟩] {⟨színösszeállítás⟩}
  ⟨doboz tartalma⟩
\end{beamercolorbox}
```

Az opciók:

- `wd=⟨szélesség⟩` Doboz szélessége (alapérték `\textwidth`).
- `dp=⟨mélység⟩` Doboz mélysége.
- `ht=⟨magasság⟩` Doboz magassága.
- `left` doboz tartalma balra zárt.
- `right` Doboz tartalma jobbra zárt.
- `center` Doboz tartalma középre zárt.
- `sep=⟨távolság⟩` Doboz tartalma körüli extra tér nagysága.
- `shadow` Doboz árnyékolt.
- `shadow=false` Doboz nem árnyékolt.
- `rounded` Doboz sarkai kerekítettek.
- `rounded=false` Doboz sarkai nem kerekítettek.

Például

```
\setbeamercolor{sajat szin}{fg=blue, bg=yellow}
```

után

```
\begin{frame}
  \begin{beamercolorbox}[wd=6cm, shadow, rounded, center]{sajat szin}
    Doboz tartalma
  \end{beamercolorbox}
\end{frame}
```

A másik `beamer` által definiált bekezdésdoboz kerekített sarkú és adhatunk neki címet egy fejrészben:

```
\begin{beamercolorbox}[⟨opcionális⟩] {⟨cím⟩}
  ⟨doboz tartalma⟩
\end{beamercolorbox}
```

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 szin1}{fg=white, bg=blue}
\setbeamercolor{sajat szin2}{fg=black, bg=yellow}
```

után

```
\begin{frame}
  \begin{beamercolorbox}[upper=sajat szin1, lower=sajat szin2]{Cím}
    Doboz tartalma
  \end{beamercolorbox}
\end{frame}
```

21.6.5. 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]{<kép>}}
```

Ekkor a `<kép>` minden dia hátterén megjelenik. Ha ugyanezt egyetlen keret diáira akarja elérni, akkor ezt kell tenni:

```
{\setbeamertemplate{background canvas}
{\includegraphics[width=\paperwidth]{<kép>}}
\begin{frame}
...
\end{frame}}
```

21.6.6. Képek

A képek beillesztése, hasonlóan a normál esethez, történhet az `\includegraphics` parancsal, de itt van overlay specifikációja, melynek alapértéke `<1->`.

```
\includegraphics<<spec>> [<opción>] {<kép>}
```

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}
```

21.6.7. 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} \in 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` parancssal is, de sokkal szébb megoldást ad az

```
\animategraphics[<opcion>]{<sebesség>}{<alapnév>}{<első>}{<utolsó>} ∈ animate
```

parancs. Ez a kód egy képsorozatot videóként fog 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.1df`-fet használjuk, akkor a kettőspont aktívvá tételeit 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 töltse 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
```

21.6.8. Videó

Videó is lejátszható a prezentáció egy keretén belül a következő parancssal:

```
\movie[<opcion>]{<poszter>}{<videófájl>} ∈ multimedia
```

Ügyeljen arra, hogy a videó lejátszása idegen gépen nem feltétlenül fog működni (például, ha a lejátszáshoz szükséges codec nincs telepítve rá).

Csak a lejátszás történik a pdf fájlon 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 `<poszter>` látható a videónak kijelölt területen, ha csak nem adta meg a `poster` opciót (lásd később). Erre kattintva indul a lejátszás. A `<poszter>` 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 `<poszter>` 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 `\hyperlink{movie} ∈ 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
\hyperlink{movie}[start=5s,duration=10s]{cimke}{5--15\,sec}
\par
\hyperlink{movie}[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.

21.6.9. 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 $4\text{ cm} \times 3\text{ cm}$ 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.

21.6.10. Kereszthivatkozások, nyomógombok

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 `<2>` (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}
```

Linknek nemcsak szöveg, hanem nyomógomb is megadható:

```
\beamerbutton{<gomb szövege>}
\beamergotobutton{<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 átállíthatja. Például

```
\renewcommand{\insertgotosymbol}{$\ggg$}
\setbeamercolor{button}{fg=black, bg=yellow}
\setbeamercolor{button border}{fg=red}
\setbeamertfont{button}{family=\rmfamily, shape=\itshape, series=\bfseries}
```

21.6.11. Keret ismétlése

Ha egy keretet `label` opcióval töltötte be, akkor lehetőség van az `\againframe` parancssal 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}
```

22. fejezet

A L^AT_EX programozása

22.1. ASCII kódolás és kategória kódok

A `pdflatex` illetve `latex` fordítók először a forrás minden karakterét megvizsgálják, és 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 ű karakter helyére berakja a `\H{O}` parancsot. Ha a forrásfájl UTF-8 kódolású, akkor ezt 2018-tól alapesetben elvégzi a L^AT_EX, illetve korábban az `inputenc` csomag `utf8` opcióval konvertált. Ha más kódolású a forrás, akkor használja 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	^^@	0
fejléc kezdete	^^A	1
szöveg kezdete	^^B	2
szöveg vége	^^C	3
adatátvitel vége	^^D	4
vizsgálat	^^E	5
visszaigazolás	^^F	6
csengetés	^^G	7
visszalépés	^^H	8
vízszintes tabulátor	^^I	9
új sor	^^J	10
függőleges tabulátor	^^K	11
lapdobás	^^L	12
kocsi vissza	^^M	13
karakterkészlet váltása	^^N	14
karakterkészlet visszaállítása	^^O	15
nyers adat következik	^^P	16
eszközvezérlés 1	^^Q	17
eszközvezérlés 2	^^R	18
eszközvezérlés 3	^^S	19
eszközvezérlés 4	^^T	20
negatív visszaigazolás	^^U	21
szinkron üresjárat	^^V	22
adatátviteli blokk vége	^^W	23
mégsem	^^X	24
adathordozó vége	^^Y	25
helyettesítő karakter	^^Z	26
feloldójel	^^[27
állományelválasztó	^^\	28
csoportelválasztó	^^]	29
rekordelválasztó	^^^	30
egységelválasztó	^^-	31
törlés	^^?	127

Nem csak vezérlő karakterekre lehet utalni $^{\wedge}$ után szereplő valamelyen karakterrel. Általánosságban, ha a karakter decimális ASCII-kódja x és a karakterre utalásban a $^{\wedge}$ 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 \backslash karakternek az ASCII-kódja. Tehát a szóközre a $^{\wedge}\backslash$ jelsorozattal utalhatunk. A $^{\wedge}I$ -ben az I decimális ASCII-kódja $y = 73$, így $x = 73 - 64 = 9$, ami a vízszintes tabulátor decimális ASCII-kódja. Tehát a $^{\wedge}I$ a vízszintes tabulátorra utal.

Egy ASCII-kódolású karakter decimális ASCII-kódját a következő kód tárolja:

```
 $\backslash\langle karakter \rangle$ 
```

ahol a `<karakter>` lehet utalás is. Ha ezt ki is akarja íratni a dokumentumában, akkor elő kell tenni a `\number` parancsot. Amennyiben a ` karakter aktív – ahogyan az a `magyar.lrf` használatakor is van –, ezen jel elő tegye ki a `\string` parancsot. Például az A betű ASCII-kódja

```
\number\string`^A
```

65

A „vízszintes tabulátor” ASCII-kódja

```
\number\string`^\^I
```

9

A L^AT_EX-ben minden ASCII-kódolású karakternek van egy úgynevezett kategória kódja is, amellyel ezeket 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 „koci 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 T_EX-ben ilyen a „lezáró nulla” vezérlőkarakter, L^AT_EX-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 kivételével 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 L^AT_EX 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 `<karakter ASCII-kódja>` megadható decimális, oktális és hexadecimális formában is. Ha oktális formát használ, akkor a ' jelet, míg ha hexadecimális formát használ, akkor a " jelet kell elő gépelni. Ha a kategóriakódot ki akarja íratni a dokumentumában,

akkor a `\catcode` előre 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<karakter ASCII-kódja>=<kategóriakód>`

parancsot, ahol a `<karakter ASCII-kódja>`, hasonlóan az előzőekben leírtakkal, megadható decimális, oktalis 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 `\TeX` 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

`\TeX\TeX`

22.2. Kontrollszo, token, makró

A *kontrollszo* (*control sequence*) olyan karaktersorozat, melyet egyetlen egységeként kezel a `\TeX`. Ilyen például a parancs, számláló, paraméter. A `\TeX` a beolvasott szöveges állomány feldolgozásának első lépéseként a szöveget ún. *tokenekre* bontja. Ekkor a `\TeX` minden kontrollszt egyetlen tokenként kezel, az összes többi karaktert pedig egyetlen tokenként továbbítja. A felesleges szóközök illetve megjegyzések nem kerülnek további feldolgozásra. A beolvasott karaktereket kategorizálja és minden karakterhez hozzárendeli a kategóriakódját.

A *makró* egy névvel ellátott programkód. Amikor ezt a nevet meghívjuk, akkor a `\TeX` az adott programkódot végrehajtja. A *makró neve* a `\` jelen kívül (pontosabban 0 belső kódolású karakteren kívül) bármilyen karaktert, sőt még szóközt is tartalmazhat.

Ha egy makrónév csak az angol ábécé betűiből és az `@` karakterből áll szóközök nélkül, akkor azt *parancsnévenek* nevezzük. Egy parancsnévvel ellátott makrót úgy lehet futtatni, hogy a parancsnév előre kell tenni a `\` jelet (pontosabban 0 belső kódolású karaktert). Az így kapott karaktersorozatot *parancsnak* nevezzük. Például a `\bfseries` egy parancsnév, amit a `\bfseries` parancs beírásával lehet végrehajtani. *Belső parancsnak* nevezzük azokat a parancsokat, melyek neve tartalmaz `@` karaktert.

Bármely makró futtatható úgy, hogy a makró nevét a `\csname` és `\endcsname` parancsok közé írjuk.

Ha kíváncsi arra, hogy egy parancs hogyan lett definiálva, akkor azt írja be a

```
\ShowCommand{<parancs>}
```

kódba. Ezután a tex fájlt lefordítva a log fájlba kiíródik a `<parancs>` definíciója. Másik lehetőség a

```
\meaning{<parancs>}
```

használata, amely a pdf-be írja be a `<parancs>` definícióját.

```
\ShowEnvironment{<környezet>}
```

A log fájlba kiíródik a `\begin{<környezet>}` és `\end{<környezet>}` parancsok definíciói.

22.3. Belső parancsok

A 22.2. szakaszban említett belső parancsok egy egyszerű L^AT_EX forrásállományban nem használhatók, csak osztály- (.cls) és csomagfájlokban (.sty). Ha mégis szeretne egy belső parancsot használni egyszerű L^AT_EX forrásállományban (.tex), 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
```

Ügyeljen arra, hogy a `\makeatletter` és `\makeatother` parancsok nem szerepelhetnek osztály- és csomagfájlokban.

22.4. Hosszúságparancsok

Vannak olyan parancsok, melyek hosszméreteket hordoznak. Ezek az ún. hosszúságparancsok. Ilyen például a `\textwidth` (szövegtükör szélessége), a `\textheight` vagy a `\pagegoal` (szövegtükör magassága), illetve a `\pagetotal` (aktuális pozíció alapvonalanak és az első sor alapvonalának a távolsága). A hosszúságparancsok kezelésére a következő parancsok használhatók:

```
\newlength{<új hosszúságparancs>}
```

Új hosszúságparancs definiálása. Ennek alapértéke 0pt. Például

```
\newlength{\sajathossz}
```

```
\setlength{<hosszúságparancs>}{<hossz>}
```

A `<hosszúságparancs>` értéke `<hossz>` lesz. Például

```
\setlength{\sajathossz}{2cm}
```

esetén az előbb definiált `\sajathossz` értéke 2cm lesz.

```
\setlength{<hosszúságparancs>}{<hossz> plus <hossz1> minus <hossz2>}
```

A hosszúságparancs beállítása rugalmas méretre. Például

```
\setlength{\sajathossz}{15pt plus 7pt minus 3pt}
\setlength{<hosszúságparancs>}{<más hosszúságparancs>}
```

A *<hosszúságparancs>* átveszi a *<más hosszúságparancs>* 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}}
```

A *<hosszúságparancs>* felveszi a *<szöveg>* szélességét.

```
\settoheight{<hosszúságparancs>}{{szöveg}}
```

A *<hosszúságparancs>* felveszi a *<szöveg>* magasságát (az alapvonallal fölötti rész magassága).

```
\settodepth{<hosszúságparancs>}{{szöveg}}
```

A *<hosszúságparancs>* felveszi a *<szöveg>* mélységét (az alapvonallal 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
```

Nézzünk egy összetettebb példát:

```
\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.

```
\uselengthunit{<mértékegység>}\printlength{<hosszúságparancs>} ∈ printlen
```

Megadja a *<hosszúságparancs>* aktuális értékét *<mértékegység>*-ben mérve. Például

```
\uselengthunit{cm}\printlength{\paperwidth}
```

A *\setlength* illetve *\addtolength* parancsokban a *<hossz>* megadása történhet számolással is, melyhez a

```
\dimexpr <dim kifejezés> \relax
```

vagy a

```
\dimeval{<dim kifejezés>}
```

parancs használható. Egy *<dim kifejezés>* egy vagy több tagból áll, amelyeket össze kell adni vagy ki kell vonni. Összeadás ill. kivonás jele a szokásos (+ ill. -). Egy tag egy tényezőből áll, opcionálisan szorozva vagy osztva numerikus értékkel, amely egy egész

számot jelent. A szorzás ill. osztás jele a programozásban megszokott (* ill. /). Egy tényező lehet egy dimenzió típusú mennyiség (azaz vagy egy hosszúságparancs vagy egy mértékegységgel megadott hosszúság, pl. 12mm) vagy egy zárójelben lévő részkifejezés, ami ugyanúgy épül fel, mint a `\dim kifejezés`. Például

```
(\textwidth*2 + 15pt)/2 + 12pt*3 - 5mm/2
```

esetén a tényezők

```
(\textwidth*2 + 15pt)
12pt
5mm
```

illetve a tagok

```
(\textwidth*2 + 15pt)/2
12pt*3
5mm/2
```

ahol a `\textwidth*2 + 15pt` részkifejezésben a tényezők

```
\textwidth
15pt
```

és a tagok

```
\textwidth*2
15pt
```

A numerikus tényezővel történő szorzás-osztás csak utólag történhet, azaz pl. `12pt*3` jó, de `3*12pt` nem. Ugyanakkor `\textwidth*2` helyett írható `2\textwidth`, de `2*\textwidth` nem! A `2\textwidth` alakban 2 helyett írható törtszám is, pl. `2.5\textwidth` jó, de vigyázat, `\textwidth*2.5` már nem jó. A zárójelek bizonyos esetekben elhagyhatók. Például

```
((\textwidth*2 + 15pt)*2)/3
```

helyett írható a következő is:

```
(\textwidth*2 + 15pt)*2/3
```

Ha a számolt hossz értékét ki akarja íratni, akkor a

```
\the\dimexpr \dim kifejezés \relax
```

parancs használható, de a `\dimexpr{\dim kifejezés}` önmagában is megjeleníti az eredményt!

A korábban említett numerikus érték megadható számolással is a következő parancsok valamelyikével:

```
\numexpr num kifejezés \relax
```

vagy

```
\inteval{num kifejezés}
```

Ezek a `\dimexpr` legközelebbi egészre kerekített értékével térnek vissza. Ebben a `\dimexpr` pontosan úgy épül fel, mint a `\dim kifejezés`, annyi különbséggel, hogy itt egy tényező lehet egy numerikus típusú mennyiség (azaz vagy egy `\value` parancssal megadott számláló értéke, melyről később lesz szó, vagy egy egész szám) vagy egy zárójelben lévő részkifejezés, ami ugyanúgy épül fel, mint a `\dim kifejezés`. Például

```
(\value{section}*2 + 15)/2 + 12*3 - 5/2
```

A numerikus tényezővel történő szorzás-osztás csak utólag történhet, azaz például a `\value{section}*2` jó, de `2*\value{section}` nem!

Ha a számolt numerikus értékét ki akarja íratni, akkor a

```
\the\numexpr <num kifejezés> \relax
```

parancs használható, de az `\inteval{<num kifejezés>}` önmagában is megjeleníti az eredményt!

Most következzen egy összetettebb példa:

```
\newlength{\sajathossz}
\setlength{\sajathossz}{\dimexpr
  (\textwidth*2 + 15pt)/\inteval{(\value{section}*2 + 15)/2 + 12*3}
  + 12pt*3 - 5mm/2\}}
```

22.5. Szóközök méretének beállítása

A következő képlet mutatja, hogy a T_EX hogyan határozza meg a szóközök méretét:

$$\text{szóköz mérete} = \begin{cases} n \text{ plus } \frac{f}{1000} p_n \text{ minus } \frac{1000}{f} m_n, & \text{ha } f < 2000, \\ x \text{ plus } \frac{f}{1000} p_x \text{ minus } \frac{1000}{f} m_x, & \text{ha } f \geq 2000, \end{cases}$$

ahol alapesetben

$$\begin{aligned} n &= 0.333333\text{em} \quad (6/18\text{ em}) \\ x &= 0.444444\text{em} \quad (8/18\text{ em}) \\ p_n &= p_x = 0.166666\text{em} \quad (3/18\text{ em}) \\ m_n &= m_x = 0.111111\text{em} \quad (2/18\text{ em}) \end{aligned}$$

továbbá az f az ún. *szóköztényező*, melynek értéke 1000-re állítódik minden vízszintes lista elején, nem karakteres doboz után, matematikai formula után és vízszintes lista után. A képletben a `plus` és `minus` úgy értendő, mint ahogyan azt a rugalmas térközök megadásánál láttuk.

Minden karakternek van egy ún. *szóköztényező kódja* (g), amely az előző f -től függetlenül állítja be az új f értékét a következő szabály szerint:

$$f = \begin{cases} \text{előző } f, & \text{ha } g = 0, \\ 1000, & \text{ha az előző } f < 1000 < g, \\ g, & \text{ minden más esetben.} \end{cases}$$

Alapesetben minden betű és számkarakter esetén $g = 1000$, kivéve a nagybetűket, amelyeknél $g = 999$. A pont, kérdőjel és felkiáltójel esetén $g = 3000$, kettőspontnál $g = 2000$, pontosvesszőnél $g = 1500$ és vesszőnél $g = 1250$.

Például `x` esetén az `x` előtt (azaz a vízszintes lista elején) $f = 1000$, az `x` után $f = g = 1000$, és a pont után $f = g = 3000$. Mivel ez nagyobb 2000-nél, ezért ezen pont utáni szóköz mérete

$$x \text{ plus } \frac{3000}{1000} p_x \text{ minus } \frac{1000}{3000} m_x.$$

Ugyanakkor x . esetén az x előtt (azaz a vízszintes lista elején) $f = 1000$, az x után $f = g = 999$, és a pont után $g = 3000$ miatt $f = 1000$. Mivel ez kisebb 2000-nél, ezért ezen pont utáni szóköz mérete

$$n \text{ plus } \frac{1000}{1000} p_n \text{ minus } \frac{1000}{1000} m_n.$$

Egy `\sffamily` szóköztényező kódja (g) a következő kóddal állítható át `\number` értékre:

```
\sffamily`<karakter> <szám>
```

Egy adott ponton a szóköztényező (f) a következő kóddal állítható át `\number` értékre:

```
\spacefactor=<szám>
```

ahol a `\number` maximális értéke 32767. Az n, p_n, m_n értékeit a következő módon állíthatja be rendre `\n`, `\pn`, `\mn` értékekre:

```
\spaceskip=<n> plus <p_n> minus <m_n>
```

Az x, p_x, m_x értékeit a következő módon állíthatja be rendre `\x`, `\px`, `\mx` értékekre:

```
\xspaceskip=<x> plus <p_x> minus <m_x>
```

Ha `\spaceskip=0pt`, akkor n, p_n, m_n nem változik, illetve, ha `\xspaceskip=0pt`, akkor x, p_x, m_x nem változik.

22.6. Számlálók

A L^AT_EX 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` parancssal létrehozott tételeszerű környezet is generál egy számlálót. Például, ha a `lemma` tételeszerű környezetet definiáltuk, akkor annak számozására létrejön egy `lemma` számláló is.

A számlálók kezelésére használt parancsok:

```
\newcounter{<új számláló>}
```

Új számlálót definiál. Alapértéke 0.

```
\newcounter{<új számláló>}[<számláló>]
```

Új számlálót definiál. Alapértéke 0. Ha a *<számláló>* értéke megváltozik, akkor az *<új számláló>* értéke lenullázódik.

```
\setcounter{<számláló>}{<szám>}
```

A *<számláló>* értéke *<szám>* lesz. A *<szám>* megadható a \numexpr illetve \inteval parancsokkal is (lásd a 371. oldalon).

```
\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 parancssal, akkor egy erre való hivatkozás a \ref parancssal, a *<számláló>* itteni értékét írja ki.

```
\value{<számláló>}
```

A *<számláló>* értékének átadására használható. Például

```
\setcounter{secnumdepth}{\value{tocdepth}}
```

hatására a secnumdepth 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.

```
\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.

```
\Alpha{<számláló>}
```

A *<számláló>* értékének kiírása nagy alfanumerikus számozással.

```
\alpha{<számláló>}
```

A *<számláló>* értékének kiírása kis alfanumerikus számozással.

```
\greek{<számláló>} ∈ moreenum
```

A *<számláló>* értékének kiírása a görög ábécé kisbetűivel (α , β , γ , ...).

```
\Greek{<számláló>} ∈ moreenum
```

A *<számláló>* értékének kiírása a görög ábécé nagybetűivel (A , B , Γ , ...).

```
\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}}
```

```
\@addtoreset{számláló1}{számláló2}
\counterwithin*{számláló1}{számláló2}
```

Mindkét parancs esetén, ha a `<számláló2>` értéke megváltozik, akkor a `<számláló1>` értéke lenullázódik.

```
\counterwithin[számtípus]{számláló1}{számláló2}
\numberwithin[számtípus]{számláló1}{számláló2} \in amsmath
```

Mindkét parancs esetén, ha a `<számláló2>` értéke megváltozik, akkor a `<számláló1>` értéke lenullázódik, továbbá a `\the<számláló1>` hatása `\the<számláló2>.számtípus{<számláló1>}` lesz, ahol a `<számtípus>` egy olyan parancs, ami a szám típusát adja meg (`\arabic`, `\roman`, `\Roman`). A `<számtípus>` opció alapértéke `\arabic`.

```
\counterwithout[számtípus]{számláló1}{számláló2}
```

Ezután a `<számláló1>` értéke nem nullázódik le, ha megváltozik a `<számláló2>`, továbbá a `\the<számláló1>` hatása `<számtípus>{<számláló1>}` lesz, ahol a `<számtípus>` egy olyan parancs, ami a szám típusát adja meg (`\arabic`, `\roman`, `\Roman`). A `<számtípus>` opció alapértéke `\arabic`.

```
\counterwithout*{számláló1}{számláló2}
```

Ugyanaz a hatása, mint a * nélküli verzió esetén, de ebben az esetben nem változik a `\the<számláló1>` kifejtése.

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]\$
```

$$886 = [2015 \cdot \frac{44}{100}]$$

22.7. Vezérlő utasítások

22.7.1. Feltételes utasítások

```
\if<kód><igaz>\else<hamis>\fi
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<kód>* eredményében az első két karakter megegyezik-e vagy sem. Például

```
\if11(a)\else(b)\fi
```

(a)

```
\def\parancs{11}
\if\parancs(a)\else(b)\fi
\def\parancs{12}
\if\parancs(a)\else(b)\fi
```

(a) (b)

Az előző példában `\def` (lásd a 22.8. szakaszban) definiál egy `\parancs` parancsot, melynek eredménye először 11 utána 12.

Az `\if... \else... \fi` típusú utasításokban az `\else...` rész elhagyható, ha nem vizsgálja azt az esetet, amikor a feltétel nem teljesül. Például

```
\if11(a)\fi
\if12(b)\fi(c)
```

(a)(c)

```
\ifx<kód1><kód2><igaz>\else<hamis>\fi
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<kód1>* és *<kód2>* eredménye ugyanaz-e vagy sem. Például

```
\def\parancsA{aaa}
\def\parancsB{aaa}
\def\parancsC{ccc}
\ifx\parancsA\parancsB(a)\else(b)\fi
\ifx\parancsA\parancsC(c)\else(d)\fi
```

(a)(d)

```
\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>\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
```

(a)

```
\ifodd<egész szám><igaz>\else<hamis>\fi
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy az *<egész szám>* értéke páratlan vagy páros. Például

```
\ifodd\value{page}(a)\else(b)\fi
```

(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
```

(b)

```
\ifdim\textwidth<\textheight(a)\else(b)\fi
```

(a)

```
\ifmmode<igaz>\else<hamis>\fi
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy matematikai módban vagyunk-e vagy sem. Például

```
$2\ifmmode^4\else\textsuperscript5\fi$  
2\ifmmode^4\else\textsuperscript5\fi
```

2^4 2^5

```
\ifpdf<igaz>\else<hamis>\fi \in iftex
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a fordítás eredménye pdf vagy dvi.

```
\ifpdftex<igaz>\else<hamis>\fi \in iftex
```

Az eredmény akkor *<igaz>*, ha a fordító *latex* vagy *pdflatex*.

```
\ifxetex<igaz>\else<hamis>\fi \in iftex
```

Az eredmény akkor *<igaz>*, ha a fordító *xelatex*.

```
\ifluatex<igaz>\else<hamis>\fi \in iftex
```

Az eredmény akkor *(igaz)*, ha a fordító lualatex.

```
\iftutex{igaz}{hamis}{fi} \in iftex
```

Az eredmény akkor *(igaz)*, ha a fordító xelatex vagy lualatex.

```
\ifdef{\parancs}{igaz}{hamis}{fi}
```

Az eredmény aszerint *(igaz)* vagy *(hamis)*, hogy a *(parancs)* definiált vagy sem. Például

```
(\ifdef{\section}{a}{b}{fi})
```

(a)

```
\ifdef{\date}{nyelv}{igaz}{hamis}{fi}
```

Az eredmény aszerint *(igaz)* vagy *(hamis)*, hogy a babel vagy polyglossia csomaggal betöltötte-e a *(nyelv)*-et vagy sem.

```
\if@twoside{igaz}{hamis}{fi}
```

Az eredmény aszerint *(igaz)* vagy *(hamis)*, hogy a dokumentumosztályt twoside opcióval töltötte-e be vagy sem.

```
\ifcase{egész szám}{0}{1}{2}{...}{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}{egy}{kettő}{sok}{fi}
```

sok

```
\newcounter{szam}\setcounter{szam}{7}
\ifcase{value}{szam}{or}{hétfő}{kedd}{szerda}{csütörtök}{péntek}{szombat}{vasárnap}{fi}
```

vasárnap

Új \if... \else... \fi típusú parancs definiálása

```
\newif\if{név}
```

ennek alapértéke hamis, vagyis ezután

```
\if{név}{igaz}{hamis}{fi}
```

eredménye *(hamis)*. Igaz értékre a következő módon állíthatja:

```
\langle név\rangle true
```

(Ez lokális hatású, ami azt jelenti, hogy ezt egy blokkban megadva, a hatása nem érvényesül a blokkon kívül. Blokkon kívül akkor lesz hatása, azaz akkor válik globálissá, ha elé teszi a \global parancsot.) Ezután

```
\if{név}{igaz}{hamis}{fi}
```

eredménye *(igaz)*. Visszaállítása hamisra:

```
\langle név\rangle false
```

(Ez lokális hatású. Globális használathoz 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)

Egyéb típusú feltételes utasítások

```
\@ifnextchar<karakter>{\<igaz>}{\<hamis>}
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a következő első karakter *<karakter>* vagy sem. Például

```
(\@ifnextchar c{a}{b}c)
```

(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}*) (\@ifstar{a}{b}x)
```

(a) (bx)

```
\@ifundefined{\<parancsnév>}{\<igaz>}{\<hamis>}
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy a *<parancsnév>* nem definiált vagy definiált. Például, mivel a \section definiált

```
(\@ifundefined{section}{a}{b})
```

(b)

```
\@ifclassloaded{\<osztály>}{\<igaz>}{\<hamis>}
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy az *<osztály>* dokumentumosztályt töltötte be vagy sem. Csak preambulumban használható!

```
\IfClassLoadedTF{\<osztály>}{\<igaz>}{\<hamis>}
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy az *<osztály>* dokumentumosztályt töltötte be vagy sem. A preambulumban és a dokumentumtestben is használható! Ez a parancs 2021. novembere után telepített rendszerekben érhető el.

```
\IfClassLoadedWithOptionsTF{\<osztály>}{\<opciónk>}{\<igaz>}{\<hamis>}
```

Az eredmény aszerint *<igaz>* vagy *<hamis>*, hogy az *<osztály>* dokumentumosztályt töltötte be vagy sem a megadott *<opciónk>*-kal. A preambulumban és a dokumentumtestben is használható! Ez a parancs 2021. novembere után telepített rendszerekben érhető el.

```
\@ifpackageloaded{\<csomag>}{\<igaz>}{\<hamis>}
```

Aszerint *<igaz>* vagy *<hamis>* az eredmény, hogy a *<csomag>* be volt-e korábban töltve vagy sem. Csak preambulumban használható!

```
\IfPackageLoadedTF{\<csomag>}{\<igaz>}{\<hamis>}
```

Aszerint *(igaz)* vagy *(hamis)* az eredmény, hogy a *(csomag)* be volt-e korábban töltve vagy sem. A preambulumban és a dokumentumtestben is használható! Ez a parancs 2021. novembere után telepített rendszerekben érhető el.

```
\IfPackageLoadedWithOptions{<csomag>}{<opciók>}{<igaz>}{<hamis>}
```

Aszerint *(igaz)* vagy *(hamis)* az eredmény, hogy a *(csomag)* be volt-e korábban töltve vagy sem a megadott *(opciók)*-kal. A preambulumban és a dokumentumtestben is használható! Ez a parancs 2021. novembere után telepített rendszerekben érhető el.

```
\IfLanguageName{<nyelv>}{<igaz>}{<hamis>} ∈ iflang
```

Az eredmény aszerint *(igaz)* vagy *(hamis)*, hogy a *(nyelv)* aktív vagy sem.

```
\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)*.

Az ifthen csomag használata

```
\ifthenelse{<feltétel>}{<igaz>}{<hamis>} ∈ ifthen
```

Az eredmény aszerint *(igaz)* vagy *(hamis)*, hogy a *(feltétel)* igaz vagy hamis.

```
\isodd{<szám>} ∈ ifthen
```

Az értéke aszerint igaz vagy hamis, hogy a *(szám)* értéke páratlan vagy páros.

```
\lengthtest{<hosszúság feltétel>} ∈ ifthen
```

Az értéke aszerint igaz vagy hamis, hogy a *(hosszúság feltétel)* igaz vagy hamis.

```
\equal{<söveg1>}{<söveg2>} ∈ ifthen
```

Az értéke aszerint lesz igaz vagy hamis, hogy a *(söveg1)* és *(söveg2)* megegyezik-e vagy sem.

```
\not <feltétel> ∈ ifthen
```

Az értéke aszerint igaz vagy hamis, hogy a *(feltétel)* hamis vagy igaz.

```
<feltétel1>\and<feltétel2> ∈ ifthen
```

Az értéke csak akkor igaz, ha a *(feltétel1)* és *(feltétel2)* is igaz.

```
<feltétel1>\or<feltétel2> ∈ ifthen
```

Az 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 adja meg.

Az előző parancsok használatára nézzük néhány példát:

```
\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)

```
\ifthenelse{\equal{haj}{háj}}{(a)}{(b)}
```

(b)

```
\newcounter{szam}
\setcounter{szam}{2}
\ifthenelse{
\(\not\value{szam}<2\and\value{szam}<11\)\or\isodd{\value{szam}}
{$2\geq\mathit{szam}<11$ vagy páratlan}
{Nem igaz, hogy $2\geq\mathit{szam}<11$ vagy páratlan}
```

$2 \leq \text{szam} < 11$ vagy páratlan

```
\newboolean{kapcsolo}
\ifthenelse{\boolean{kapcsolo}}{(a)}{(b)}
\setboolean{kapcsolo}{true}
\ifthenelse{\boolean{kapcsolo}}{(c)}{(d)}
\setboolean{kapcsolo}{false}
\ifthenelse{\boolean{kapcsolo}}{(e)}{(f)}
```

(b) (c) (f)

22.7.2. Ciklusok

A következő parancsok belső ciklus utasítások:

```
\@whilenum<numerikus feltétel>\do{\kód}
```

Mindaddig végrehajtja a *kód*-ot, 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{\kód}
```

A listaelemeken végrehajtja a *kód*-ot. 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 definiálja a `\@for` parancsot:

```
\makeatletter
\long\def\@for#1:=#2\do#3{%
\expandafter\def\expandafter\@for{\tmp\expandafter{\#2}}{%
\ifx\@for\@empty \else
\expandafter\@for\loop#2,\@nil,\@nil\@#1{\#3}\fi}
\makeatother
```

A `\@for` parancs helyett használható a következő is:

```
\renewcommand{\do}[1]{\kod}
\docslist{\lista1}, {\lista2}, ...} \in etoolbox
```

ahol a `\kod`-ba a listaelemek helyére #1 kerül. Például

```
\renewcommand{\do}[1]{A Nap #1. }
\docslist{felkel, lenyugszik}
```

A Nap felkel. A Nap lenyugszik.

A következő ciklus utasítások minden korlátozás nélkül használhatók:

```
\loop<\kod1><\feltetel><\kod2>\repeat
```

Kifejti a `\kod1` kódot, majd megvizsgálja, hogy teljesül-e a `\feltetel`. Ha igen, akkor kifejti a `\kod2` kódot, majd kezdi az egészet előlről. Ha a `\feltetel` nem teljesül, akkor a ciklus lezárul. A `\feltetel` egy `\if`-fel kezdődő kód, de ezt a `\kod2` után nem `\fi`-vel, hanem a `\repeat` paranccsal zártuk. Például

```
\newcounter{szam}
\loop\theszam\ifnum\value{szam}<9\stepcounter{szam}\repeat
```

0123456789

```
\whiledo{\numerikus feltetel}{\kod} \in ifthen
```

Mindaddig végrehajtja a `\kod`-ot, amíg a `\numerikus feltetel` fennáll. Például

```
\newcounter{szam}\setcounter{szam}{2}
\whiledo{\value{szam}<5}{\theszam\stepcounter{szam}}
```

234

22.8. Makrók definiálása

22.8.1. A `\newcommand` parancs

A L^AT_EX már meglévő parancsai mellé sajátokat is definiálhat a `\newcommand` paranccsal.

```
\newcommand{\parancs}{\kod}
```

Ekkor a `\parancs` hatása `\kod` lesz. Például

```
\newcommand{\EKKE}{Eszterházy Károly Katolikus Egyetem}
```

után

```
\EKKE
```

Eszterházy Károly Katolikus Egyetem

Paraméteres parancsok is definiálhatók:

```
\newcommand{\parancs}{\paraméterek száma}{kód}
```

A *paraméterek száma* maximum 9 lehet. A *kód*-ban például a 2. paramétere #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 opcionális minősül, melynek alapértéke *alapérték*. Például

```
\newcommand{\ora}[2][12]{#1\textsuperscript{\underline{#2}}}
```

után

```
\ora{45}, \ora[13]{45}
```

12⁴⁵, 13⁴⁵

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 `\providemode` parancsot, amit pontosan úgy kell használni, mint a `\newcommand`-ot.

Ezeknek a parancsoknak van egy csillagos verziója is: `\newcommand*` `\renewcommand*` `\providemode*`. Ezeket akkor használja, ha olyan argumentumos parancsot definiál, melynek argumentuma nem állhat több bekezdésből, azaz nem tartalmazhat üres sort vagy `\par` parancsot. A használatuk módja megegyezik a nem csillagos parancsokéval. Például

```
\newcommand*{\bfbox}[1]{\begingroup\normalfont\bfseries\#1\endgroup}
\bfbox{szöveg}\par szöveg
```

esetén hibát kapunk, mert a `\bfbox` argumentumában `\par` parancs van.

Hivatkozás belső definíció argumentumára ♦ 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.

Példák ♦ A következő példában a `\greekalph{<számláló>}` parancsot definiáljuk, hasonlóan az `\alph{<számláló>}` parancshoz:

```
\newcommand{\greekalph}[1]{\ensuremath{%
\ifcase\value{#1}\or\alpha\or\beta\or\gamma\or\delta\or\epsilon\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\or\varphi\fi}}
```

Ekkor például

```
\newcounter{szam}
\setcounter{szam}{4}
\greekalph{szam}
```

δ

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.

22.8.2. A `\def` parancs

Új parancsokat a Plain T_EX-ben is használható `\def` parancccsal is létrehozhat, de 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 `\def` által definiált argumentumos parancs argumentumában csak egy bekezdés állhat, úgy, mint a `\newcommand*` esetében. Ha ezt a korlátozást fel akarja oldani, akkor a `\def` elő írja be a `\long` parancsot is. A belső definíciókban itt is `##1`, `##2`, stb. hivatkozásokat használunk.

A `\def` parancs esetén van még egy lehetőség, amivel különleges paraméterű parancsot definiálhatunk. Ennek megismeréséhez tekintsük a következő példát:

```
\def\foo#1#2#{1-#2-#1-3-}
```

Ebben a `#2` után nem közvetlenül kezdjük el a `\foo` definíálását a `{ }` jelek között, hanem előtte van még egy `#` jel. Ennek hatására a definícióban a 2. paraméter nem egy blokk lesz, hanem az 1. paraméter és az azt követő első `{` jel közötti kód. Tehát ezután például a

```
\foo{44}55 66{77}
```

kifejtésében az első paraméter a `\foo` utáni 1. blokk, azaz 44, míg a 2. paraméter az ezt követő első `{` jelig tartó kód, azaz 55 66. Így a kifejtés

1-55 66-2-44-3-77

Ha egyparaméteres parancsot definiálunk így, akkor a paraméter a parancs neve és az azt követő első `{` jel közötti kód. Például

```
\def\foo#1#{1-#1-2-}
\foo abc def{33}
```

esetén a paraméter `abc def`. Így a kifejtése

1-abc def-2-33

Ugyanerre egy másik érdekes példa:

```
\def\foo#1#{1-#1}
\foo\textbf{33}
```

Ekkor a paraméter a `\textbf` parancs lesz, így a kifejtése ugyanaz lesz, mint ennek:

```
1-\textbf{33}
```

azaz

1-33

Most nézzünk egy alkalmazást:

```
\def\seefilename#1{\fbox{\texttt{#1}}}
\def\includegraphics#1{\seefilename}
\includegraphics[width=5cm,angle=90]{kep.pdf}
```

Az `\includegraphics` úgy van átdefiniálva, hogy a paraméter `[width=5cm,angle=90]` lesz. De a paraméter nem szerepel sehol a kifejtésben, azaz eltűnik. Így a kifejtése ugyanaz lesz, mint ennek:

```
\seefilename{kep.pdf}
```

Ennek a kifejtése pedig

```
\fbox{\texttt{kep.pdf}}
```

kep.pdf

Ezzel a trükkkel ideiglenesen el lehet a dokumentumból tüntetni a képeket és csak a képek fájlnevei jelennek meg bekeretezve a kép helyén. Ugyanez egyszerűbben:

```
\def\includegraphics#1{}
\includegraphics[width=5cm,angle=90]{kep.pdf}
```

Ebben a `\includegraphics` úgy van átdefiniálva, hogy a paraméterét nyelje el. Így a kifejtése

{kep.pdf}

kep.pdf

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[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 makrókkal nem nekünk kell az összeget kiszámolni:

```
\newcounter{szumma}
\def\szummatag#1{\addtocounter{szumma}{#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}
```

$$\begin{array}{r} 12345 \\ 1234 \\ + 123 \\ \hline 13702 \end{array}$$

A következő példában egy `\ifdivisible{<szám1>}{<szám2>}{{<igaz>}}{<hamis>}` parancsot definiálunk, 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

A következő példa Donald Ervin Knuth-tól származik (lásd [6]):

```
\newif\ifprime \newif\ifunknown
\newcount\n \newcount\p \newcount\d \newcount\a
\def\primes#1{2,\~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\p>\d \unknowntrue\else\unknownfalse\fi
\multiply\p by\d
\ifnum\p=\a \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

22.8.3. Az \edef és \noexpand parancsok

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

Ha azt akarja, hogy az \edef használatakor egy makró ne a definiáláskori értékekkel fejtődjön ki, akkor írjon elő \noexpand parancsot. Például

```
\newcounter{szam}
\edef\kiirszam{\noexpand\theszam\theszam}
\setcounter{szam}{1}
\kiirszam
```

10

22.8.4. Az \expandafter parancs

Egy parancs \edef-fel történő definiálásában láttuk, hogy egy makró azonnali kifejtését a \noexpand parancssal lehet megakadályozni. Egy meghívott makró szabályozására egy másik parancs az

\expandafter

Ez átugorja a következő tokenet, veszi az azt követőt (ha ez egy argumentumos makró, akkor veszi az argumentumokat is) majd egy mélységen kifejti. Ezután visszatér az átugrott tokenre és a kifejtett rész előtérbe teszi. Például

```
\def\aa{\b}
\def\b{c}
\aa
\string\aa
\expandafter\string\aa
```

Ekkor az \aa kifejtése (teljes mélységen) c. A \string\aa esetén a \string-et követő \ jel meg fog jelenni a pdf-ben, majd utána egy a betű. Az \expandafter\string\aa esetén az \expandafter átugorja az utána található \string parancsot, majd az azt követő \aa parancsot kifejti egy mélységen, ami \b. Ezután előtérbe teszi az átugrott \string parancsot. Tehát most a \string\b fog kifejtődni, ami \b. Tehát az előző kód eredménye

c\aa\b

A következő példa a `geometry` csomag `\newgeometry` parancsára vonatkozik, amely a lap geometriájára vonatkozó beállításokat várja kifejtett állapotban. Például

```
\newgeometry{top=3cm}
```

Ha a `top=3cm` egy parancsban van tárolva, akkor ez a parancs közvetlenül nem írható be a `\newgeometry`-ba, mert értelmezés előtt nem fejti ki a `\newgeometry`. Azaz a következő kód hibát fog jelezni:

```
\def\foo{top=3cm}
\newgeometry{\foo} % ROSSZ KÓD!
```

A megoldás a következő:

```
\def\foo{top=3cm}
\expandafter\newgeometry\expandafter{\foo}
```

A megoldásban az első `\expandafter` azért kell, mert különben a `\newgeometry` a következő `\expandafter`-t tekintené argumentumnak, ami hibát eredményezne. Az előző kód működése a következő: Az első `\expandafter` átugorja a rákövetkező `\newgeometry`-t majd kifejti egy szint mélységgel az utána található második `\expandafter`-t. Ez átugorja az utána található `{` tokenet és kifejti egy szint mélységen a rákövetkező `\foo` parancsot, ami `top=3cm`. Ezután a második `\expandafter` ez előre teszi a `{` tokenet. Ezzel a második `\expandafter` kifejtődött egy szint mélységgel, amelynek `{top=3cm}` az eredménye. Most visszatér az első `\expandafter`-hez, hiszen az még csak ezután tudja az előbb kapott eredmény előre tenni az átugrott `\newgeometry`-t. Tehát ekkor `\newgeometry{top=3cm}` az eredmény. Még ezután van egy `}` token, tehát még azt a végére rakja. Így ekkor a

```
\newgeometry{top=3cm}
```

helyes kód fejtődik ki.

A következő példában tegyük fel, hogy van egy definiált `\foo` parancs, amit szeretnénk bővíteni valamikor egy új kóddal. Elsőre következő tűnhet jó megoldásnak:

```
\def\foo{a}
\def\foo{\foo b}
\foo
```

De ebben az esetben a `\foo` kifejtésekor egy végtelen ciklus alakul ki, így a fordítás hibával leáll. A kód javítása a következő módon lehetséges:

```
\def\foo{a}
\expandafter\def\expandafter\foo\expandafter{\foo b}
\foo
```

Kezdetben a `\foo` kifejtése az a betű, de a második sor után már bővül egy `b` betűvel, azaz ekkor ab az eredmény. Nézzük meg miért. Az első `\expandafter` kifejtésekor átugorja a `\def` parancsot, majd kifejti az azt követő `\expandafter`-t. Ez átugorja az utána levő `\foo` parancsot, majd kifejti az azt követő `\expandafter`-t. Ez átugorja az utána található `{` tokenet, majd kifejti az utána levő `\foo` parancsot, ami most az „`a`” betű. Ezután a harmadik `\expandafter` ez előre teszi az átugrott `{` tokenet. A második `\expandafter` ez előre teszi az átugrott `\foo` parancsot, majd az első `\expandafter` ez előre teszi az átugrott `\def` parancsot. Tehát most itt tart a fordítás:

```
\def\foo{a}
```

Az ezután található `b}` kódot még hozzáfűzi, így ez fog a második körben lefordulni:

```
\def\foo{ab}
```

Ennek hatására a `\foo` kifejtése ab lesz.

A következő példában tegyük fel, hogy az `\aa`, `\bb` és `\cc` parancsok korábbról már definiáltak. Ekkor

```
\def\b{\bb}
\def\c{\cc}
```

után

```
\expandafter\expandafter\expandafter\aa\expandafter\b\c
```

ekvivalens a következővel:

```
\aa\bb\cc
```

Nézzük meg miért. Az első `\expandafter` kifejtésekor átugorja az utána található `\expandafter`-t, majd az azután található `\expandafter`-t fejti ki. Ez átugorja az utána található `\aa` parancsot, majd kifejti az azután álló `\expandafter`-t. Ez átugorja az utána álló `\b` parancsot, majd kifejti egy szintű mélységben a rakkövetkező `\c` parancsot, ami `\cc`. Ezután a negyedik `\expandafter` ez elő teszi az átugrott `\b` parancsot, a harmadik `\expandafter` ez elő teszi az átugrott `\aa` parancsot, végül az első `\expandafter` ez elő teszi az átugrott második `\expandafter`-t. Most tehát itt tart a fordítás:

```
\expandafter\aa\b\cc
```

A következő menetben az `\expandafter` kifejtésekor átugorja az utána található `\aa` parancsot, majd az azután található `\b`-t fejti ki egy szinten, ami `\bb`. Ezután ez elő teszi az átugrott `\aa`-t:

```
\aa\bb
```

Az ezután található `\cc`-t még utána rakja. Tehát a harmadik fordulóban már ezt fogja kifejteni:

```
\aa\bb\cc
```

Az utolsó példa elemzését az Olvasóra bízzuk. Tegyük fel, hogy az `\aa`, `\bb`, `\cc` és `\dd` parancsok korábbról már definiáltak. Ekkor

```
\def\b{\bb}
\def\c{\cc}
\def\d{\dd}
```

után

```
\expandafter\expandafter\expandafter\expandafter
\expandafter\expandafter\expandafter\aa
\expandafter\expandafter\expandafter\b
\expandafter\c\dd
```

ekvivalens a következővel:

```
\aa\bb\cc\dd
```

22.8.5. Parancsok másolása

Ha egy definiált parancsnak szeretne másik nevet is adni, akkor használja a `\let` parancsot. Például, ha azt szeretné, hogy az `\oldLaTeX` parancs hatása ugyanaz legyen, mint a `\LaTeX` parancsnak, akkor ezt kell beírni:

```
\let\oldLaTeX\LaTeX
```

Ennek leginkább akkor van haszna, ha egy definiált parancsot az eredeti kifejtésének segítségével szeretné átdefiniálni. Ilyenkor az eredeti parancsot menteni kell új néven és

az átdefiniálásnál azt kell felhasználni. A következő kód például átdefiniálja a \LaTeX parancsot úgy, hogy a \LaTeX logó kékkel legyen kiszedve:

```
\usepackage{xcolor}
\let\oldLaTeX\LaTeX
\def\LaTeX{\begingroup\color{blue}\oldLaTeX\endgroup}
```

Egy másik alkalmazásként a következő példában azt mutatjuk meg, hogyan lehet megoldani, hogy a \section parancs kiadásakor ne a rövid cím kerüljön a fejlécbe. Írja a következő kódot a preambulumba:

```
\let\oldsectionmark\sectionmark
\def\hdrsection#1{%
  \def\sectionmark##1{%
    \oldsectionmark{#1}%
    \let\sectionmark\oldsectionmark}}
```

Ezután, ha egy \section parancs kiadása előtt beírja a

```
\hdrsection{Szakaszcím fejlécben}
```

parancsot, akkor a fejlécbe a rövid cím helyett a „Szakaszcím fejlécben” kerül. Értelem-szerű módosítással alkalmazható ugyanez a \chapter illetve \subsection parancsokra is. Ugyanennek a problémának egy másik megoldását lásd a 396. oldalon.

Az előző módszer nem működik, ha a másolandó parancsnak van opcionális argumentuma vagy \DeclareRobustCommand parancsal volt definiálva (lásd később). Ekkor a \let helyett használja a \NewCommandCopy parancsot (vagy a letltmacro csomag \LetLtxMacro parancsát). Például a következő kód átdefiniálja a \section parancsot úgy, hogy a címet csupa nagybetűvel írja ki, de a tartalomjegyzékben változatlanul hagyja:

```
\makeatletter
\NewCommandCopy{\old@section}{\section}
\def\@szakasz[#1]{\old@section[#1]{\MakeUppercase{#2}}}
\def\@@szakasz#1{\old@section*\{\MakeUppercase{#1}\}}
\def\@@@szakasz#1{\@szakasz[#1]{#1}}
\def\section{\@ifnextchar[\@szakasz}{\@ifstar{\@@szakasz}{\@@@szakasz}}
\makeatother
```

Ha ezután valahol vissza akarja állítani a \section eredeti jelentését, akkor a

```
\makeatletter
\NewCommandCopy{\section}{\old@section}
\makeatother
```

parancs nem fog működni, mert az első argumentumában definiált parancs van. Ekkor használja a \RenewCommandCopy parancsot.

```
\makeatletter
\RenewCommandCopy{\section}{\old@section}
\makeatother
```

22.8.6. Lokális és globális hatás

A \newcommand, \renewcommand, \providecommand, \def, \edef, \let és \LetLtxMacro hatása lokális. Ez azt jelenti, hogy egy blokkban ezekkel definiált parancs nem érvényesül a blokkon kívül. Például

```
\def\foo{A}
{\def\foo{B}\foo}
\foo
```

B A

Ha `\def` helyett `\global\def`-et, vagy annak rövid verzióját, `\gdef`-et ír, akkor globális lesz a hatás, azaz annak hatásköre blokkon kívül is megmarad. Például

```
\def\foo{A}
{\gdef\foo{B}\foo}
\foo
```

B B

Az `\edef` globális megfelelője a `\global\edef` illetve az `\xdef`. A `\let` globális megfelelője a `\global\let`. A `\LetLtxMacro` globális megfelelője a `\GlobalLetLtxMacro`. (Utóbbit hozza a `letltxmacro` csomag.)

22.8.7. Makrónév definiálása

Abban az esetben, ha már a definiálandó makró nevét is valamilyen makróval kell generálni vagy olyan karaktereket tartalmaz, amely egy parancs nevében nem szerepelhet, akkor `\def\⟨makrónév⟩` kód nem alkalmas a definiálásra. Helyette használja a

```
\@namedef{⟨makrónév⟩}
```

vagy a vele ekvivalens

```
\expandafter\def\csname ⟨makrónév⟩\endcsname
```

kódot. A második esetben azért van ott az `\expandafter`, mert különben a `\def` a létező `\csname` parancsot akarná átdefiniálni. A `\csname` és `\endcsname` a makrónév határait jelöli ki. Ha az argumentumba több bekezdés is kerülhet, akkor az előző kódok elé kell tenni a `\long` parancsot. Ha globális hatást akar, akkor a `\global` parancsot kell ezek előtt rakni. Az így definiált makrót a

```
\@nameuse{⟨makrónév⟩}
```

vagy a vele ekvivalens

```
\csname ⟨makrónév⟩\endcsname
```

módon lehet meghívni. Ha a `⟨makrónév⟩` nem definiált, akkor nem kapunk figyelmeztesést vagy hibaüzenetet. Például

```
\makeatletter
\@namedef{félkövér}{\textbf{#1}}
\nameuse{félkövér}{szöveg}
\makeatother
```

szöveg

```
\makeatletter
\newcounter{foo}
\stepcounter{foo}
\@namedef{foo\thefoo}{\textbf{#1}}
\stepcounter{foo}
\@namedef{foo\thefoo}{\textit{#1}}
```

```
\@nameuse{foo1}{szöveg}
\@nameuse{foo2}{szöveg}
\makeatother
```

szöveg szöveg

22.8.8. Mozgó argumentum, gyenge és erős parancsok

Bizonyos parancsok argumentumai a L^AT_EX futása közben a dokumentum más részein is megjelenhetnek. Ilyen például a `\section` parancs, melynek argumentuma kiíródik a tartalomjegyzékbé és a fejlécbe is. Az ilyen argumentumot *mozgó argumentumnak* nevezzük. A mozgatás során előfordulhat, hogy egy ilyen argumentumba írt parancs idő előtt vagy rosszul hajtódiék végre. Az ilyen parancsokat *gyenge (fragile) parancsoknak* nevezzük. Ha egy parancs semmilyen mozgó argumentumban sem okoz problémát, akkor azt *erős (robust) parancsnak* nevezzük. Ha egy gyenge parancsot kell beírni mozgó argumentumba, akkor tegye elé a `\protect` parancsot.

Az eddig ismertetett eljárások gyenge parancsokat definiálnak. Erős parancs definíálására például a

```
\DeclareRobustCommand{\parancs}[\paraméterek száma]{\alapérték}{\kód}
```

parancs használható, amit pontosan úgy kell használni, mint a `\newcommand`-ot, de ez hiba helyett csak figyelmeztető üzenetet küld, ha létező parancsot definiál át. Ennek a parancsnak is van csillagos verziója, amit akkor lehet használni, ha a `\kód` biztosan csak egy bekezdést generál. Ha a `\def` parancsal szeretne erős parancsot definiálni, akkor elé kell tenni a `\protected` parancsot. A korábban már említett

```
\NewCommandCopy{\új parancs}{régi parancs}
```

parancs funkciója ugyanaz, mint a

```
\let{\új parancs}{régi parancs}
```

parancsnak, de itt a régi és az új parancs is erős.

22.8.9. A `\NewDocumentCommand` parancs

A `\newcommand`-nál jóval rugalmasabb a

```
\NewDocumentCommand{\parancs}{\argumentum specifikáció}{\kód}
```

parancs, mellyel erős parancsok definiálhatók. Ennek teljes dokumentációja a következő parancssorral érhető el:

texdoc usrguide

A használatához 2020. október előtt telepített rendszerek esetén be kell tölteni az `xparse` csomagot. Az `\argumentum specifikáció`-ban kell megadni a `\parancs` argumentumainak számát, típusát, sorrendjét. Az argumentumokba írt paraméterekre – hasonlóan az eddigiekhez – #1, #2, …, #9 módon lehet hivatkozni a `\kód`-ban. Az argumentumok típusát jelző legfontosabb specifikációk:

- Kötelező argumentum, amely vagy egyetlen token, vagy több token { és } jelek között. Például

```
\NewDocumentCommand{\ora}{m}{\#1\textsuperscript{\underline{h}}}
\NewDocumentCommand{\Ora}{mm}{\#1\textsuperscript{\underline{\#2}}}
```

```
\ora{13} \Ora{13}{45}
```

13^h 13⁴⁵

r*(token1)(token2)* Hasonló mint az **m**, de ekkor a határolójelek *(token1)* és *(token2)* lesznek { és } helyett. Például

```
\NewDocumentCommand{\ora}{ m r() }{#1\textsuperscript{\underline{#2}}}
\ora{13}(45)
```

13⁴⁵

v Verbatim argumentum. Ezzel hasonló parancsokat lehet definiálni, mint a **\verb**, de itt a { és } is lehetnek határolójelek. Például

```
\NewDocumentCommand{\myverb}{ v }{\texttt{\color{red}\#1}}
\myverb{\LaTeX}
\myverb|\LaTeXe|
```

\LaTeX \LaTeXe

0{<alapérték>} Opcionális argumentum, amit [és] jelek között kell megadni. Ha nem adja meg, akkor az *(alapérték)* lép érvénybe. Például

```
\NewDocumentCommand{\ora}{ m 0{00} }
{#1\textsuperscript{\underline{#2}}}
\ora{13} \ora{13}[45]
```

13⁰⁰ 13⁴⁵

- Opcionális argumentum, amit [és] jelek között kell megadni. Nincs alapértéke, azaz ekvivalens az 0{} specifikációval.

D*(token1)(token2){<alapérték>}* Opcionális argumentum, amit *(token1)* és *(token2)* jelek között kell megadni. Ha nem adja meg, akkor az *(alapérték)* lép érvénybe. Például

```
\NewDocumentCommand{\ora}{ m D(){00} }
{#1\textsuperscript{\underline{#2}}}
\ora{13} \ora{13}(45)
```

13⁰⁰ 13⁴⁵

d*(token1)(token2)* Opcionális argumentum, amit *(token1)* és *(token2)* jelek között kell megadni. Nincs alapértéke, azaz ekvivalens a **D***(token1)(token2){}* specifikációval.

E{<token>}{{<alapérték>}} Opcionális argumentum, amit *(token)*{ és } jelek között kell megadni. (Pontosabban a *(token)* utáni token lesz az opció.) Ha nem adja meg, akkor az *(alapérték)* lép érvénybe. Például

```
\NewDocumentCommand{\ora}{ m E{:}{00} }
{#1\textsuperscript{\underline{#2}}}
\ora{13} \ora{13}:{45} \ora{13}:h
```

13⁰⁰ 13⁴⁵ 13^h

Ha az *(alapérték)* egyetlen token, akkor nem kell dupla kapcsos zárójelek közé tenni. Az **E{<token1>}{{<alapérték1>}}****E{<token2>}{{<alapérték2>}}** ekvivalens a következővel: **E{<token1>(<token2>){<alapérték1>}{<alapérték2>}}**. Ez nemcsak kettő, hanem több paraméterrel is megt lehető hasonló módon.

`e{<token>}` Opcionális argumentum, amit `<token>` és } jelek között kell megadni. (Pontosabban a `<token>` utáni token lesz az opció.) Nincs alapértéke, azaz ekvivalens az `E{<token>}{}` specifikációval. Az `e{<token1>}e{<token2>}` ekvivalens a következővel: `e{<token1>}{<token2>}`. Ez nemcsak kettő, hanem több paraméterrel is megtehető hasonló módon.

- s Ezzel lehet egy parancs csillagos verzióját definiálni. Ekkor a `(kód)`-ban elhelyezett

```
\IfBooleanTF{#<n>}{<igaz>}{<hamis>}
\IfBooleanT{#<n>}{<igaz>}
\IfBooleanF{#<n>}{<hamis>}
```

parancsok valamelyikének használatával a `<parancs>` csillagos verziója esetén az `<igaz>`, ellenkező esetben pedig a `<hamis>` fejtődik ki. Itt `<n>` azt jelöli, hogy az s hanyadik (legtöbbször első) paraméter. Például

```
\NewDocumentCommand{\ora}{ s m m }
  {\IfBooleanTF{#1}{#2\textsuperscript{\underline{#3}}}{#2:#3}}
\ora{13}{45}
\ora*{13}{45}
```

13:45 13⁴⁵

`t<token>` Ez pontosan úgy használható, mint az `s` specifikáció, de ekkor * helyett `<token>` használható. Például

```
\NewDocumentCommand{\ora}{ t! m m }
  {\IfBooleanTF{#1}{#2\textsuperscript{\underline{#3}}}{#2:#3}}
\ora{13}{45}
\ora!{13}{45}
```

13:45 13⁴⁵

Hasonlóan a `\newcommand*` parancshoz, ezen specifikációkkal egy bekezdésből álló argumentumok definiálhatók. Ha bármelyik elő még egy + jelet is ír, akkor az argumentum több bekezdésből is állhat (`+m`, `+r()`, `+O{1}`, `+v`, stb.). Például

```
\NewDocumentCommand{\foo}{ +m }{\color{red}#1}
\foo{szöveg}\par szöveg
```

Amennyiben az argumentumba írt kód elején és végén található szóközöket nem akarja figyelembe venni, akkor a specifikáció elő írja be a

`>{\TrimSpaces}`

kódot. Például

```
\NewDocumentCommand{\myframebox}{ m }{\fbox{#1}}
\NewDocumentCommand{\mytrimframebox}{ >{\TrimSpaces}m }{\fbox{#1}}
\myframebox{ szöveg }
\mytrimframebox{ szöveg }
```

szöveg szöveg

Ha egy kódban tesztelni szeretné, hogy az `<n>`-edik argumentum, amit o, d vagy e specifikációval definiáltunk, üres-e vagy sem, akkor használja a következő parancsokat:

```
\IfNoValueTF{#<n>}{<üres>}{<nem üres>}
\IfNoValueT{#<n>}{<üres>}
```

```
\IfNoValueF{#<n>}{<nem üres>}
```

vagy

```
\IfValueTF{#<n>}{<nem üres>}{<üres>}
\IfValueT{#<n>}{<nem üres>}
\IfValueF{#<n>}{<üres>}
```

Az `m`, `r`, `v` specifikációkkal definiált `<n>`-edik argumentum tesztelése arra, hogy üres-e vagy sem:

```
\IfBlankTF{#<n>}{<üres>}{<nem üres>}
\IfBlankT{#<n>}{<üres>}
\IfBlankF{#<n>}{<nem üres>}
```

Ha létező parancsot akar `\NewDocumentCommand`-dal átdefiniálni, akkor a fordítás hibával leáll. Ha szándékosan akar átdefiniálni egy parancsot, akkor a

```
\RenewDocumentCommand
```

parancsral tegye. Ha csak abban az esetben akar egy parancsot definiálni, ha az még nem létezik, ugyanakkor, ha létezik, akkor nincs szándékában átdefiniálni, akkor használja a

```
\ProvideDocumentCommand
```

parancsot. Ha egy parancsot minden esetben akar definiálni, akár létezik már akár nem, akkor használja a

```
\DeclareDocumentCommand
```

parancsot. Ezeket pontosan úgy kell használni, mint a `\NewDocumentCommand` parancsot.

Amennyiben erős helyett gyenge parancsot akar definiálni, mint a `\newcommand`, akkor az előző parancsok neveiben a `Document` előtt be kell szűrni a `Expandable` szót:

```
\NewExpandableDocumentCommand
\RenewExpandableDocumentCommand
\ProvideExpandableDocumentCommand
\DeclareExpandableDocumentCommand
```

Ezeket pontosan úgy kell használni, mint a `\NewDocumentCommand` parancsot.

Mutatunk egy összetett példát, ami a `\section` parancsot úgy definiálja át, hogy külön rövid cím kerülhessen a tartalomjegyzékbe és a fejlécbe:

```
\NewCommandCopy{\oldsection}{\section}
\let\oldsectionmark\sectionmark
\RenewDocumentCommand{\section}{ s 0{#3} m 0{#2} }{%
    \IfBooleanTF{#1}{\oldsection*{#3}}{%
        \IfNoValueTF{#4}{\oldsection[#2]{#3}}{%
            \def\sectionmark##1{\oldsection[#2]{#3}}%
            \let\sectionmark\oldsectionmark\sectionmark{#4}}}}
```

Fontos, hogy az `\oldsectionmark` parancs bevezetése nélkül az utolsó előtti sort blokkba kellene zártani, ami azért nem jó megoldás, mert akkor a környező térközök nem lesznek megfelelőek. Ezután a `\section` parancsot a következő módon lehet használni:

```
\section[<tartalomjegyzék>][<szöveg>][<fejléc>]
```

A `<tartalomjegyzék>` alapértéke `<szöveg>`, illetve a `<fejléc>` alapértéke `<tartalomjegyzék>`. Ennek hatására a `<szöveg>` jelenik meg szakaszcímként a szövegben, `<tartalomjegyzék>` jelenik meg szakaszcímként a tartalomjegyzékben és `<fejléc>` jelenik meg szakaszcímként a fej-

lében. Értelemszerű módosításokkal hasonlóan át lehet definiálni a `\chapter` illetve `\subsection` parancsokat is.

22.8.10. Az etoolbox csomag

Erős parancsok definiálására használható eszköz az `etoolbox` csomag. Itt csak megemlíünk néhány lehetőséget a teljesség igénye nélkül.

`\newrobustcmd{<parancs>}[<paraméterek száma>][<alapérték>]{<kód>} ∈ etoolbox`

Ez pontosan úgy használható, mint a `\DeclareRobustCommand`, de ez hibával leáll a fordítás során, ha létező parancsot definiál át. Ennek is van csillagos verziója ugyanazzal a céllal, mint a `\newcommand*` esetén. A definiált `\<parancs>` erős lesz, de a védelmi mechanizmusa más, mint a `\DeclareRobustCommand` parancs esetén.

`\renewrobustcmd{\<parancs>}[<paraméterek száma>][<alapérték>]{<kód>} ∈ etoolbox`

A létező `\<parancs>`-ot lehet ezzel átdefiniálni. Az átdefiniált parancs erős lesz. Ennek is van csillagos verziója.

`\providerobustcmd{<parancs>}[<paraméterek száma>][<alapérték>]{<kód>} ∈ etoolbox`

Korábban még nem definiált `\<parancs>` esetén ugyanazt csinálja, mint a `\newrobustcmd`. Ha `\<parancs>` már létezik, akkor nem csinál semmit. Ennek is van csillagos verziója.

`\robustify{<parancs>} ∈ etoolbox`

A `\<parancs>`-ot átalakítja az ε-TEX védelmi mechanizmusával ellátott erős parancsá. Ha van a `\<parancs>`-nak argumentuma, azt ebben nem szabad megadni.

`\protecting{<kód>} ∈ etoolbox`

A `<kód>`-ban található gyenge parancsok mindegyikét ellátja `\protect` előtaggal.

`\csdef{<makrónév>} ∈ etoolbox`

Azt csinálja, mint a `\@namedef{<makrónév>}`, de ez erős parancs.

`\csgdef{<makrónév>} ∈ etoolbox`

Ugyanaz, mint a `\csdef`, de globális hatású.

`\csedef{<makrónév>} ∈ etoolbox`

Az `\edef{<makrónév>}` helyett kell használni, ha a `<makrónév>`-re ugyanaz vonatkozik, mint a `\@namedef` esetében. Ez erős parancs.

`\csxdef{<makrónév>} ∈ etoolbox`

Ugyanaz, mint a `\csedef`, de globális hatású.

`\cslet{<makrónév>}{<parancs>} ∈ etoolbox`

Ugyanaz a funkciója, mint a `\let` parancsnak, de itt a `<makrónév>`-re ugyanaz vonatkozik, mint a `\@namedef` esetében. Ez erős parancs. Globális hatáshoz elő kell tenni a `\global` parancsot.

`\letcs{<parancs>}{<makrónév>} ∈ etoolbox`

Hasonló, mint a `\cslet`, de ebben a második argumentum a `<makrónév>`.

`\csletcs{<makrónév1>}[<makrónév2>] ∈ etoolbox`

Hasonló, mint a `\cslet`, de ebben minden két argumentum egy makrónév.

`\undef{<parancs>} ∈ etoolbox`

A `<parancs>`-ot definiálatlanítja teszi. Ez erős parancs.

```
\gundef{parancs} ∈ etoolbox
```

Ugyanaz, mint az \undef, de globális hatású.

22.8.11. Létező parancs bővítése

Egy létező parancsot a \g@addto@macro globális hatású belső parancssal lehet bővíteni. Például

```
\def\EKKE{Eszterházy Károly}
\EKKE\
\makeatletter
\g@addto@macro\EKKE{ Katolikus Egyetem}
\makeatother
\EKKE
```

Eszterházy Károly
Eszterházy Károly Katolikus Egyetem

Hasonló feladatra alkalmas az

```
\apptocmd{parancs}{kód}{sikeres}{sikertelen} ∈ etoolbox
```

parancs, amely a *parancs* definíciójának végéhez fűzi a *kód*-ot. Ha ez sikeresen megtörtént, akkor még kifejti a *sikeres* kódot is, ellenkező esetben pedig a *sikertelen* kódot. Például

```
\def\EKKE{Eszterházy Károly}
\EKKE\
\apptocmd{\EKKE}{ Katolikus Egyetem}{}{}
\EKKE
```

Eszterházy Károly
Eszterházy Károly Katolikus Egyetem

Létező parancs definíciójának az elejére is tudunk kódot fűzni a

```
\pretocmd{parancs}{kód}{sikeres}{sikertelen} ∈ etoolbox
```

parancssal. Például

```
\def\EKKE{Katolikus Egyetem}
\EKKE\
\pretocmd{\EKKE}{Eszterházy Károly }{}{}
\EKKE
```

Katolikus Egyetem
Eszterházy Károly Katolikus Egyetem

22.8.12. Létező parancs egy részletének cseréje

Erre a következő parancs ad lehetőséget:

```
\patchcmd{parancs}{mit}{mire}{sikeres}{sikertelen} ∈ etoolbox
```

Ez a *parancs* definíciójában szereplő első *mit* kódot kicseréli a *mire* kódra. Ha ez sikeresen megtörtént, akkor még kifejti a *sikeres* kódot is, ellenkező esetben pedig a *sikertelen* kódot. Például

```
\def\EKKE{Eszterházy Károly Főiskola}
\EKKE\
\patchcmd{\EKKE}{Főiskola}{Katolikus Egyetem}{}{}
```

Eszterházy Károly Főiskola
Eszterházy Károly Katolikus Egyetem

```
\xpatchcmd*{\parancs}{\mit}{\mire}{\sikeres}{\sikertelen} ∈ regexpatch
```

Ez a *(parancs)* definíciójában szereplő összes *(mit)* kódot kicseréli a *(mire)* kódra. Ha ez sikeresen megtörtént, akkor még kifejti a *(sikeres)* kódot is, ellenkező esetben pedig a *(sikertelen)* kódot. Arra ügyelni kell, hogy – eltérően az *etoolbox* csomagtól – amennyiben a *(parancs)* definíciója tartalmaz @ karaktert, akkor az *\xpatchcmd* parancsot *\makeatletter* és *\makeatother* közé kell zárni. Például

```
\def\EKKE{Eszterházy Károly Katolikus Egyetem}
\EKKE\
\xpatchcmd*{\EKKE}{E}{\textbf{E}}{}{}
```

$$\xpatchcmd*{\EKKE}{K}{\textbf{K}}{}{}$$

Eszterházy Károly Katolikus Egyetem
Eszterházy Károly Katolikus Egyetem

22.8.13. Új verbatim parancs

Láttuk korábban, hogy verbatim típusú parancsot definiálni a *\NewDocumentCommand* parancssal lehetséges. Egy másik lehetőség a *newverbs* csomag használata. Például

```
\newverbcommand{\myverb}{\color{red}\rmfamily}{}{}
```

Ezután a *\myverb* parancs pontosan úgy használható, mint a *\verb*. Például

```
\myverb|# # \| \myverb*|# # \|
```

\| # # \|

22.8.14. Új overlay specifikációval rendelkező parancs

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, ..., 9), akkor az kibővül egy *n* + 1-edikkel, melyben az overlay specifikáció adható meg. Például

```
\newcommand<>{\textblue}[1]{\textcolor{blue}{#1}}
```

vagy

```
\newcommand<>{\blue}{\color{blue}}
```

Az így definiált *\textblue* illetve *\blue* parancsok overlay specifikációinak alapértéke <1>. Használatuk:

```
\textblue<\spec>{\szöveg}
\blue<\spec>\szöveg
```

22.8.15. Új `kulcs=érték` típusú opcióval rendelkező parancs

Ilyen parancs például a következő:

```
\includegraphics[width=5cm]{imagefile}
```

Szokás ilyenkor a `width=5cm` opcióban a `width`-et kulcsnak, az `5cm`-t pedig a kulcs értékének nevezni.

Ilyen típusú parancsot 2022. június 20. után telepített/frissített T_EX-rendszer esetén csomag használata nélkül definiálhatunk, melyhez a következő parancsok ismerete szükséges:

```
\DeclareKeys[\család]{\deklarációs lista}
```

Ezzel a lokális hatású parancssal tud `kulcs=érték` típusú opciókat deklarálni.

`\család` A deklarált opciók ebbe az családba fognak tartozni. Alapértéke annak a fájlnak a neve kiterjesztés nélkül, amelyben a `\DeclareKeys` parancs végrehajtódik. Az új `kulcs=érték` típusú opcióval rendelkező parancs definíciójában az opciók deklarálását érdemes egy blokkon belül megtenni, mert így a lokális hatás miatt nem keveredik más családdal. Ilyenkor elég az alapérték használata, azaz `\DeclareKeys{\deklarációs lista}`.

`\deklarációs lista` A deklarált opciók listája vesszővel elválasztva. A listaelemek szintaxisa:

```
\kulcs.\típus = érték
```

`\kulcs` A kulcs neve, melyben az angol ábécé betűi, számok és szóközök lehetnek.

`\típus` A lehetséges típusok:

`code` Ekkor az `\érték` egy kód, amely a `\SetKeys[\család]{\opciók}` (lásd később) parancs kiadásakor kifejtődik. Ebben a kódban a `\kulcs` értékére #1-ként kell utalni. Arra ügyeljen, hogy amennyiben ez egy paraméteres parancs definíciójában szerepel, akkor #1 helyett ##1 kell, hogy ne keveredjenek a változók. Ha a kód független a `\kulcs` értékétől, akkor az opciók megadásánál a `\kulcs`-nak nem kell adni értéket, azaz a `\kulcs` illetve `\kulcs=bármí` ugyanazt jelenti.

`if` Ekkor létrejön egy `\if\érték` feltétel. A `\kulcs` értéke `true` vagy `false` lehet. Az opciók megadásánál `\kulcs=true` helyett írható röviden `\kulcs` is. Ezután az

```
\if\érték \igaz\else \hamis\fi
```

kifejtése aszerint `\igaz` vagy `\hamis`, hogy a `\kulcs` értéke `true` vagy `false`.

`store` Ekkor az `\érték` egy még nem definiált parancs. A `\kulcs` értéke ebben a parancsban lesz tárolva a `\SetKeys[\család]{\opciók}` (lásd később) parancs kiadásakor.

`usage` Ez akkor használható, ha ez előtt a `\kulcs` már deklarálva volt az előző három `\típus` valamelyikével. Ha az `\érték` helyére `preamble` kerül, akkor a `\kulcs` értékének a megadása csak preambulumban lehetséges. A másik lehetséges `\érték` a `load`, de erről a 23. fejezetben lesz szó.

\SetKeys[⟨család⟩]{⟨opciók⟩}

Ezzel lehet megadni a ⟨család⟩ néven deklarált opciók értékeit az ⟨opciók⟩-ban. Hatása lokális és többször is kiadható. Elsőnek meg kell adni az alapértelmezett értékeket. Ha egy használatakor egy opció értékét nem adja meg, akkor a korábban felvett értéke marad. A ⟨család⟩ alapértéke annak a fájlnak a neve kiterjesztés nélkül, amelyben a \SetKeys parancs végrehajtódik. Az új `kulcs=érték` típusú opcióval rendelkező parancs definíciójában az opciók deklarálását és ennek a parancsnak a megadását érdemes egy blokkon belül megtenni, mert így a lokális hatás miatt nem keveredik más családdal. Ilyenkor elég az alapérték használata, azaz \SetKeys{⟨opciók⟩}.

Mindezek könnyebben megérthetők a következő példán keresztül.

```
</> \newcommand\userframebox[2] [] {%
    \begingroup
    \DeclareKeys{
        sep.code      = \setlength{\fboxsep}{##1},
        line width.code = \setlength{\fboxrule}{##1},
        style.store   = \userframeboxstyle,
        inline.if     = userframeboxinline }
    \SetKeys{sep=3pt,line width=0.5pt,style=\bfseries,inline,#1}%
    \ifuserframeboxinline\else\begin{center}\fi
    \fbox{\userframeboxstyle#2}%
    \ifuserframeboxinline\else\end{center}\fi
    \endgroup}
```

Ez definiál egy \userframebox parancsot, melynek következő a hatása:

```
\userframebox{szöveg1}
\userframebox[sep=5pt,style=\itshape]{szöveg2}
\userframebox[style=\huge,inline=false]{szöveg3}
\userframebox[line width=2pt]{szöveg4}
```



Amennyiben régebbi a TeX-rendszer és még nincs definiálva a kernelben \SetKeys, akkor használhatja a következő csomagok valamelyikét: `xkeyval`, `expkv`, `expkv-def`, `expkv-cs`, `simplekv`. Most csak az utóbbit foglalkozunk. A használatához a következő parancsok ismerete szükséges:

\setKVdefault[⟨család⟩]{⟨kulcs1⟩=⟨érték1⟩,⟨kulcs2⟩=⟨érték2⟩,...} ∈ simplekv

Ez ⟨család⟩ néven deklarálja a ⟨kulcs1⟩, ⟨kulcs2⟩ stb. kulcsokat, rendre ⟨érték1⟩, ⟨érték2⟩ stb. alapértelmezett értékekkel. A ⟨család⟩, ⟨kulcs1⟩, ⟨kulcs2⟩ stb. csak angol betűket tartalmazhat, de a kulcsokban lehetnek szóközök is. Ha az ⟨érték1⟩, ⟨érték2⟩ stb. valamelyike `true` vagy `false`, akkor az adott kulcs logikai kulcsként funkcionál. Az `=true` el is hagyható. Az előbbi parancs után

```
\useKV[⟨család⟩]{⟨kulcs1⟩} ∈ simplekv
\useKV[⟨család⟩]{⟨kulcs2⟩}
```

...

eredményei rendre $\langle\text{érték1}\rangle$, $\langle\text{érték2}\rangle$ stb. Ha például a $\langle\text{kulcs1}\rangle$ logikai kulcs, akkor $\text{\ifboolKV}[\langle\text{család}\rangle]\{\langle\text{kulcs1}\rangle\}\{\langle\text{igaz}\rangle\}\{\langle\text{hamis}\rangle\} \in \text{simplekv}$ eredménye aszerint $\langle\text{igaz}\rangle$ vagy $\langle\text{hamis}\rangle$, hogy az $\langle\text{érték1}\rangle$ true vagy false. A kulcsok alapértékei átállíthatók a $\text{\setKV}[\langle\text{család}\rangle]\{\langle\text{kulcs1}\rangle=\langle\text{érték1}\rangle,\langle\text{kulcs2}\rangle=\langle\text{érték2}\rangle,\dots\} \in \text{simplekv}$

parancssal. Ha valamelyik kulcs nem szerepel az opciók között, akkor annak megmarad az éppen aktuális értéke. Például

```
\newcommand\userframebox[2] [] {%
    \setKVdefault[frame]{sep=3pt,line width=0.5pt,style=\bfseries,inline}%
    \setKV[frame]{#1}%
    \setlength{\fboxsep}{\useKV[frame]{sep}}%
    \setlength{\fboxrule}{\useKV[frame]{line width}}%
    \ifboolKV[frame]{inline}{}{\begin{center}}%
    \fbox{\useKV[frame]{style}}#2\%
    \ifboolKV[frame]{inline}{}{\end{center}}}
```

definiál egy \userframebox parancsot, melynek hatása pontosan az lesz, mint az előző példában.

22.9. Környezetek definiálása

22.9.1. A \newenvironment parancs

A L^AT_EX már meglévő környezetei mellé sajátokat is definiálhat a \newenvironment parancssal:

```
\newenvironment{\langle név\rangle}[\langle argumentumszám\rangle][\langle alapérték\rangle]{\langle nyitókód\rangle}{\langle végkód\rangle}
```

$\langle\text{név}\rangle$ A környezet neve.

$\langle\text{argumentumszám}\rangle$ Az argumentumok száma. Az $\langle n \rangle$ -edik argumentumra $\#\langle n \rangle$ hivatkozik, ahol $\langle n \rangle = 1, 2, \dots, 9$.

$\langle\text{alapérték}\rangle$ Az első argumentum alapértéke. Ha ez adott, akkor az első argumentum opció lesz.

$\langle\text{nyitókód}\rangle$ A környezet megnyitásakor hajtódkik végre.

$\langle\text{végkód}\rangle$ A környezet bezárásakor hajtódkik végre. Ebben nem lehetnek $\#\langle n \rangle$ hivatkozások, azaz a környezet argumentumai.

A $\langle\text{név}\rangle$ környezetet nem definiálhatja, ha már létezik ilyen környezet, vagy létezik $\backslash\langle\text{név}\rangle$ parancs. Viszont definiálható, ha már létezik $\langle\text{név}\rangle$ számláló. Ha egy már létező környezetet akar átdefiniálni, akkor azt a \renewenvironment parancssal teheti meg. Ennek használata megegyezik a \newenvironment használatával.

Amikor a \newenvironment parancssal definiál például egy $\langle\text{név}\rangle$ 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 $\langle\text{végkód}\rangle$ -ban miért nem lehet $\#\langle n \rangle$ hivatkozás.

Példák ♦ Tekintsünk néhány példát.

```
\newenvironment{rend}[1]{\begin{array}{|r|p{5cm}|}\hline}
{\hline\end{array}\par}
```

Ezután

```
\begin{rend}
6:30 & Kelés után reggeli torna, mosakodás és étkezés.\\
7:30 & Munkába indulás.\\
\ldots & \ldots
\end{rend}
```

Napirend

6:30	Kelés után reggeli torna, mosakodás és étkezés.
7:30	Munkába indulás.
...	...

illetve

```
\begin{rend}[Óraterv]
8:00 & Hetes jelentése, napló beírása.\\
8:02 & Házi feladat ellenőrzése, értékelése.\\
\ldots & \ldots
\end{rend}
```

Óraterv

8:00	Hetes jelentése, napló beírá- sa.
8:02	Házi feladat ellenőrzése, ér- tékelése.
...	...

```
\newenvironment{rend}[1]{\begin{array}{|r|p{5cm}|}\hline}
{\hline\end{array}\par}
```

Ezután

```
\begin{rend}[Óraterv]
8:00 & Hetes jelentése, napló beírása.\\
8:02 & Házi feladat ellenőrzése, értékelése.\\
\ldots & \ldots
\end{rend}
```

Óraterv

8:00	Hetes jelentése, napló beírá- sa.
8:02	Házi feladat ellenőrzése, ér- tékelése.
...	...

22.9.2. A `\NewDocumentEnvironment` parancs

Talán a legrugalmasabb megoldás környezet definiálására a

```
\NewDocumentEnvironment{<név>}{<argumentum specifikáció>}{<nyitókód>}{<végkód>}
```

parancs, melynek használatához 2020. október előtt telepített rendszerek esetén be kell tölteni az `xparse` csomagot. Az `<argumentum specifikáció>`-ban pontosan úgy lehet megadni az argumentumokat, mint a `\NewDocumentCommand` parancs esetén. Az argumentumokra – ellentétben a `\newenvironment` parancccsal – nemcsak a `<nyitókód>`-ban, hanem a `<végkód>`-ban is lehet hivatkozni. Az argumentumokat a `\begin{<név>}` után kell megadni. A `<név>` végére * is írható, így egy környezet csillagos verziója is definiálható. Ha a környezet belsejét (testét) is argumentumként akarja kezelni, akkor az `<argumentum specifikáció>`-ban utolsóként írja be a `b` (illetve több bekezdés esetén a `+b`) specifikációt. Például

```
\NewDocumentEnvironment{teszt}{ m 0{\today} +b }
  {\fbox{\parbox{4cm}{\textbf{\#1}\par\#3\par(\#2)}}}{}}
\NewDocumentEnvironment{teszt*}{ m 0{\today} +b }
  {\fbox{\parbox{4cm}{\underline{\#1}\par\#3\par(\#2)}}}{}}
```

után

```
\begin{teszt}{Cím}
Szöveg\par Szöveg
\end{teszt}
\begin{teszt*}{Cím}[tegnap]
Szöveg\par Szöveg
\end{teszt}
```

Cím	Cím
Szöveg	Szöveg
Szöveg	Szöveg
(2025. október 4.)	(tegnap)

illetve

```
\begin{teszt*}{Cím}
Szöveg\par Szöveg
\end{teszt*}
\begin{teszt*}{Cím}[tegnap]
Szöveg\par Szöveg
\end{teszt*}
```

Cím	Cím
Szöveg	Szöveg
Szöveg	Szöveg
(2025. október 4.)	(tegnap)

Ha létező környezetet akar `\NewDocumentEnvironment`-tel definiálni, akkor a fordítás hibával leáll. Ha át akar definiálni egy környezetet, akkor a `\RenewDocumentEnvironment`-tel tegye. Ha csak abban az esetben akar egy környezetet definiálni, ha az még nem létezik, ugyanakkor, ha létezik, akkor nincs szándékában átdefiniálni, akkor használja a `\ProvideDocumentEnvironment` parancsot. Ha egy környezetet minden esetben akar

definiálni, akár létezik már akár nem, akkor használja a `\DeclareDocumentEnvironment` parancsot. Ezeket pontosan úgy kell használni, mint a `\NewDocumentEnvironment` parancsot.

22.9.3. Új verbatim típusú környezet

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` ∈ `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 definíálásához használja a

```
\lstnewenvironment listings
```

parancsot, melynek használata megegyezik a `\newenvironment` használatával. Például

```
\lstnewenvironment{delphi}[1] []
{\lstset{language=Delphi,numbers=left,numberstyle=\tiny,\#1}}{}
```

Ezután a

```
\begin{delphi}[(opción)]
...
\end{delphi}
```

és

```
\begin{lstlisting}[language=Delphi,numbers=left,numberstyle=\tiny,(opción)]
...
\end{lstlisting}
```

kódok hatása ugyanaz.

22.9.4. Új comment típusú környezet

Tegyük fel, hogy definiált egy `megjegyzes` téretszerű 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 L^AT_EX-fordító figyelmen kívül hagyja. Ezt tudja elérni az `\excludecomment` ∈ `comment` parancssal, a következő módon:

```
\renewenvironment{megjegyzes}{}{}
\excludecomment{megjegyzes}
```

Tegyük fel, hogy az előző példában a `megjegyzes` környezet együtt számozódik egy `tetel` téretszerű környezettel. Ekkor a dokumentum eredeti verziójában és az előző kóddal ellátott verzióban a tételek számozása nem fog megegyezni, hiszen a kiiktatott

megjegyzésekkel a sorszám nem emelkedik. Ha ezt nem szeretné, akkor a megoldás a `\processcomment` parancs:

```
\renewenvironment{megjegyzes}{}{}
\processcomment{megjegyzes}{\stepcounter{tetel}\def\ThisComment##1{}{}{}}
```

Ha ezen felül még a megjegyzések helyét például egy — jellet akarja megjelölni, akkor ez a megoldás:

```
\renewenvironment{megjegyzes}{}{}
\processcomment{megjegyzes}
{\stepcounter{tetel}\def\ThisComment##1{}{}{---}{}}
```

Ugyanez a hatás a `comment` csomag nélkül is elérhető a következő módon:

```
\RenewDocumentEnvironment{megjegyzes}{ +b }{\stepcounter{tetel}---}{}
```

22.9.5. Új overlay specifikációval rendelkező környezet

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 $n + 1$ -edikkal, melyben az overlay specifikáció adható meg. Például

```
\newenvironment<>{boldornormal}
{\begin{altnav}\#1\begin{bfseries}\end{bfseries}\end{altnav}\end{altnav}}
```

Az így definiált `boldornormal` környezet overlay specifikáció alapértéke `<1->`. Használata:

```
\begin{boldornormal}<(spec)>
<szöveg>
\end{boldornormal}
```

22.9.6. Új listakörnyezet

Saját listakörnyezet is definiálható a következő parancccsal:

```
\newlist{<listanév>}{{<listatípus>}}{<maximális szintszám>} \in enumitem
```

A `<listatípus>` lehet `enumerate`, `itemize` vagy `description` aszerint, hogy számozott, számozatlan vagy leíró listát szeretnénk definiálni. A `<maximális szintszám>` azt adja meg, hogy a definiált listát hány szint mélységgig lehessen egymásba ágyazni. Ezzel a parancccsal `<maximális szintszám>` darab számláló is létrejön `<listanév>i`, `<listanév>ii`, `<listanév>iii`, `<listanév>iv`, `<listanév>v`, stb. néven, melyek a különböző szintek számlálói lesznek. Például

```
\newlist{steps}{enumerate}{2}
```

definiál egy `steps` nevű számozott listakörnyezetet, amely maximum két szint mélységgig ágyazható egymásba. Ezután ennek beállítása a `\setlist` parancccsal pontosan úgy történhet, mint a többi listakörnyezet esetén (lásd a 7.4. szakaszban). Például

```
\setlist[steps,1]{label=\textbf{\arabic*. lépés.},align=left}
\setlist[steps,2]{label=(\alph*)}
```

Az első szint számlálója `stepsi` a másodiké pedig `stepsii` lesz.

22.10. Kapcsok

A kapcsok olyan pontok a parancsok vagy környezetek kódjában, ahová további kódokat tudunk utólag beszúrni. minden ilyen kapocsnak egyedi neve van.

```
\AddToHook{<kapocsnév>}{<kód>}
```

A *<kapocsnév>* nevű kapocshoz beszúrja a *<kód>*-ot. Ha ugyanahhoz a *<kapocsnév>*-hez többször is beszűr kódokat az \AddToHook parancsal, akkor ezek a kódok nem írják egymást felül, hanem az adott sorrendben lesznek összefűzve.

```
\AddToHookNext{<kapocsnév>}{<kód>}
```

Ugyanazt csinálja, mint az \AddToHook, de ezután csak az első *<kapocsnév>* nevű kapocsnál fejti ki a *<kód>*-ot, utána törlődik.

```
\RemoveFromHook{<kapocsnév>}
```

A *<kapocsnév>* nevű kapocshoz beszúrt kódokat eltávolítja.

A kapcsok kezeléséről az itt leírtaktól többet is megtudhat a következő parancsor segítségével megnyitott pdf fájlokban:

```
texdoc lthooks ltpara ltshipout ltfilehook ltcmdhooks-code
```

22.10.1. Környezetkapcsok

A *<környezet>* nevű környezethez tartozó kapcsok a következők:

```
env/<környezet>/before
env/<környezet>/begin
env/<környezet>/end
env/<környezet>/after
```

Megjegyezzük, hogy az

```
\AddToHook{env/<környezet>/before}{<kód>}
\AddToHook{env/<környezet>/begin}{<kód>}
\AddToHook{env/<környezet>/end}{<kód>}
\AddToHook{env/<környezet>/after}{<kód>}
```

parancsok rendre ekvivalensek a következőkkel:

```
\BeforeBeginEnvironment{<környezet>}{<kód>} ∈ etoolbox
\AtBeginEnvironment{<környezet>}{<kód>} ∈ etoolbox
\AtEndEnvironment{<környezet>}{<kód>} ∈ etoolbox
\AfterEndEnvironment{<környezet>}{<kód>} ∈ etoolbox
```

Ezeknek a kapcsoknak a helye a következő kóddal szemléltethető:

```
\AddToHook{env/<környezet>/before}{<kód1>}
\AddToHook{env/<környezet>/begin}{<kód2>}
\AddToHook{env/<környezet>/end}{<kód3>}
\AddToHook{env/<környezet>/after}{<kód4>}
<kód5>
\begin{<környezet>}
<kód6>
\end{<környezet>}
```

Ennek eredménye ugyanaz lesz, mintha a következőt használta volna:

```
<kód5>
```

```
(kód1){(kód2)\begin{(környezet)}
(kód6)(kód3)
\end{(környezet)}}(kód4)
```

Például

```
\AddToHook{env/center/begin}{\bfseries}
\begin{center}
Első
\end{center}
\AddToHook{env/center/begin}{\itshape}
\begin{center}
Második
\end{center}
```

Első

Második

```
\AddToHook{env/center/begin}{\bfseries}
\begin{center}
Első
\end{center}
\RemoveFromHook{env/center/begin}
\begin{center}
Második
\end{center}
```

Első

Második

```
\AddToHook{env/center/begin}{\itshape}
\AddToHookNext{env/center/begin}{\bfseries}
\begin{center}
Első
\end{center}
\begin{center}
Második
\end{center}
```

Első

Második

22.10.2. Dokumentumtest-kapcsok

A `document` környezet nagyon speciális, ezért érdemes külön erre a környezetre kifejlesztett kapcsokat használni:

```
begindocument/before
begindocument
begindocument/end
enddocument
```

Megjegyezzük, hogy az

```
\AddToHook{begindocument/before}{<kód>}
\AddToHook{begindocument}{<kód>}
\AddToHook{enddocument}{<kód>}
```

parancsok rendre ekvivalensek a következőkkel:

```
\AtEndPreamble{<kód>} \in etoolbox
\AtBeginDocument{<kód>}
\AtEndDocument{<kód>}
```

Ezeknek a kapcsoknak a helye a következő kóddal szemléltethető:

```
\AddToHook{begindocument/end}{<kód1>}
\AddToHook{begindocument/before}{<kód2>}
\AddToHook{begindocument}{<kód3>}
\AddToHook{enddocument}{<kód4>}
<kód5>
\begin{document}
<kód6>
\end{document}
```

Ennek eredménye ugyanaz lesz, mintha a következőt használta volna:

```
<kód5>
<kód2>
\begin{document}
<kód3><kód1><kód6>
<kód4>
\end{document}
```

22.10.3. Parancskapcsok

`cmd/<parancsnév>/before`

A `\<parancsnév>` parancsot definiáló kód előtt.

`cmd/<parancsnév>/after`

A `\<parancsnév>` parancsot definiáló kód után.

Például

```
\newcommand{\EKKE}{Katólikus}
1. \EKKE\
\AddToHook{cmd/EKKE/before}{Eszterházy Károly }
2. \EKKE\
\AddToHook{cmd/EKKE/after}{ Egyetem}
3. \EKKE
```

1. Katolikus
2. Eszterházy Károly Katolikus
3. Eszterházy Károly Katolikus Egyetem

22.10.4. Oldalkapcsok

`shipout/before`

Minden oldal befejezése után, a következő oldal megkezdésekor.

shipout/firstpage

Az első oldal megkezdésekor, még az oldalparaméterek beállítása előtt.

shipout/lastpage

Az utolsó oldal megkezdésekor, még az oldalparaméterek beállítása előtt.

shipout/foreground

Minden oldal előtere. Például

```
\AddToHook{shipout/foreground}{\put(\hossz1,-\hossz2){\kód}}
```

kiadása után minden oldal előterében megjelenik a `\kód` által létrehozott doboz, melynek bal alsó sarka `\hossz1` távolságra van az oldal bal szélétől és `\hossz2` távolságra van az oldal tetejétől.

shipout/background

Minden oldal háttere.

22.10.5. Bekezdéskapcsok

para/before

A bekezdések eleje, még mielőtt áttérne horizontális módba.

para/begin

A bekezdések eleje, miután horizontális módba lépett.

para/end

A bekezdések vége, még mielőtt áttérne vertikális módba.

para/after

A bekezdések vége, miután vertikális módba lépett.

22.10.6. Fájlkapcsok

file/before

Minden fájl betöltése előtt.

file/<fájlnév>/before

A `<fájlnév>` nevű fájl betöltése előtt. A `<fájlnév>`-ben a kiterjesztést is meg kell adni. Fontos, hogy

```
\AddToHook{file/<fájlnév>/before}{\kód1}
\AddToHook{file/before}{\kód2}
```

esetén a `<fájlnév>` nevű fájl betöltése előtt először a `\kód2` érvényesül és csak utána a `\kód1`. Azaz először az összes fájlról vonatkozó kódokat fejti ki és csak utána a `<fájlnév>` fájlról vonatkozókat.

file/after

Minden fájl betöltése után.

file/<fájlnév>/after

A `<fájlnév>` nevű fájl betöltése után. A `<fájlnév>`-ben a kiterjesztést is meg kell adni. Fontos, hogy

```
\AddToHook{file/after}{kód1}
\AddToHook{file/fájlnév/after}{kód2}
```

esetén a *fájlnév* nevű fájl betöltése után először a *kód2* érvényesül és csak utána a *kód1*. Azaz először a *fájlnév* fájlról vonatkozó kódokat fejti ki és csak utána az összes fájlról vonatkozókat.

Az \include parancshoz tartozó kapcsok

include/before

Amikor egy fájlt betölt az \include parancs, akkor közvetlenül a fájl betöltése előtt és után végrehajt egy \clearpage parancsot. Ez a kapcs az \include parancs esetén a fájl betöltése előtt, de a \clearpage parancs után helyezkedik el.

include/*fájlnév*/before

Ugyanaz a helye, mint az előző kapocsnak, de csak a *fájlnév* nevű fájl esetén.

include/end

Minden \include parancs esetén a fájl betöltése után, de még a \clearpage parancs előtt.

include/*fájlnév*/end

Ugyanaz a helye, mint az előző kapocsnak, de csak a *fájlnév* nevű fájl esetén.

include/after

Minden \include parancs esetén a fájl betöltése és a második \clearpage parancs után.

include/*fájlnév*/after

Ugyanaz a helye, mint az előző kapocsnak, de csak a *fájlnév* nevű fájl esetén.

Csomagkapcsok

package/before

Csomagbetöltés előtt.

package/*csomagnév*/before

Ugyanaz a helye, mint az előző kapocsnak, de csak a *csomagnév* nevű csomag betöltekéor. A *csomagnév*-ben a kiterjesztést nem szabad megadni, azaz például `geometry` és nem `geometry.sty`.

package/after

Csomagbetöltés után.

package/*csomagnév*/after

Ugyanaz a helye, mint az előző kapocsnak, de csak a *csomagnév* nevű csomag betöltekéor.

Dokumentumosztály-kapcsok

class/before

Dokumentumosztály betöltése előtt.

class/*osztálynév*/before

Ugyanaz a helye, mint az előző kapocsnak, de csak az `\osztálynév` nevű dokumentumosztály betöltésekor. Az `\osztálynév`-ben a kiterjesztést nem szabad megadni, azaz például `aricle` és nem `article.cls`.

```
class/after
```

Dokumentumosztály betöltése után.

```
class/\osztálynév/after
```

Ugyanaz a helye, mint az előző kapocsnak, de csak az `\osztálynév` nevű dokumentumosztály betöltésekor.

22.11. Csomag betöltésének megakadályozása

Amikor egy dokumentumosztály betölt egy olyan csomagot, amit a felhasználó nem szeretne, akkor ezt a `\documentclass` parancs kiadása előtt a következő kóddal akadályozhatja meg:

```
\RequirePackage{scrlfile}  
\PreventPackageFromLoading{\csomag}
```

Amennyiben több csomag betöltését is meg akarja akadályozni, akkor a `\csomag` helyére a csomagok neveit sorolja fel vesszővel elválasztva.

Ha a preambulum egy adott pontján a letiltott csomagot mégis szeretné betölteni, akkor ezt a következő módon teheti meg:

```
\ResetPreventPackageFromLoading  
\RequirePackage{\csomag}
```

23. 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 TeX-disztribúciók részévé tegye. Erre vonatkozólag itt talál információt: <dtxtut.pdf>. A beadás itt lehetséges: <https://ctan.org/upload>.

23.1. 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 ékezes betűket parancsként gépelje be. Ez azért kell, hogy bármilyen kódolású főfájlba be lehessen tölteni a \usepackage parancsal.
- A csomag forrásfájljába minden olyan parancs írható, amely a főfájl preambulumában szerepelhet, egyedül a \usepackage helyett használjon \RequirePackage parancsot. A \RequirePackage annyiban különbözik a \usepackage-tól, hogy az a dokumentumosztály betöltése elő is kerülhet.
- A belső parancsok csomagban a \makeatletter és \makeatother parancsok nélkül is működnek.

Egy sty kiterjesztésű fájl szerkezete a következő:

```
\NeedsTeXFormat{LaTeX2e}[\dátum1]
\ProvidesPackage{csomagnév}[\dátum2] verzió leírás
tartalom
\endinput
```

A *dátum1* a csomag használatához szükséges L^AT_EX verzió dátuma éééé/hh/nn formáumban, pl. 1999/12/01. A *csomagnév* 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 *csomagnév* helyére *sajat* kerül. A *dátum2* a csomag publikálásának dátuma ugyanolyan formáumban, mint az előbb. A *verzió* a csomag verziószáma, pl. v1.0. A *leírás* 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}[2025/10/04 v1.0 Ez a csomag csak egy pelda]
\DeclareOption{opción1}{kód1}
\DeclareOption{opción2}{kód2}
\ExecuteOptions{opción1}
\ProcessOptions\relax
\endinput
```

Ekkor a *sajat* csomagnak két opciója lesz: *<opción1>* (alapopció) és *<opción2>*. Ha az *<opción2>* opciót használja, akkor a *<kód2>* kód lesz érvényben. Viszont akár kiadja akár nem az *<opción1>* opciót, a *<kód1>* mindenképpen érvényben lesz. Ha az \ExecuteOptions parancsban több opciót is megad alapopcióként, akkor azokat vesszővel kell elválasztani:

```
\ExecuteOptions{opción1,opción2,...}
```

Egy csomag opciója örökíthető a saját csomagunkra is a

```
\PassOptionsToPackage{opción}{csomag}
```

parancsal. Legyen például a *sajat.sty* tartalma:

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{sajat}[2025/10/04 Ez a csomag csak egy pelda]
\DeclareOption{hidelinks}{\PassOptionsToPackage{hidelinks}{hyperref}}
\ProcessOptions\relax
\RequirePackage{hyperref}
\endinput
```

Ekkor a *sajat* csomag opciójaként használható a *hyperref* csomagnak a *hidelinks* opciója.

Egy opciónak értéket (számláló, hossz, sztring, logikai érték) is adhat. Ezek úgynevezett *kulcs=érték* típusú opciók. Például a *geometry* és a *hyperref* csomagokban is vannak ilyen opciók:

```
\usepackage[width=150mm]{geometry}
\usepackage[linktocpage=false,linkcolor=blue]{hyperref}
```

A *kulcs=érték* típusú opciók 2022. június 20. után telepített/frissített TEX-rendszer esetén csomag használata nélkül definiálhatók, melyhez a következő parancsok ismerete szükséges:

```
\DeclareKeys[csomagnév]{deklarációs lista}
```

Ezzel a lokális hatású parancsal tud *kulcs=érték* típusú opciókat deklarálni, ahol a *<csomagnév>* a csomagfájl neve kiterjesztés nélkül és a *<deklarációs lista>* a deklárt opciók listája vesszővel elválasztva. A listaelemek szintaxisa:

```
<kulcs>.<típus> = <érték>
```

<kulcs> A kulcs neve, melyben az angol ábécé betűi, számok és szóközök lehetnek.

<típus> A lehetséges típusok:

code Ekkor az *<érték>* egy kód, amely a \SetKeys[*csomagnév*]{*opción*} (lásd később) parancs kiadásakor kifejtődik. Ebben a kódban a *<kulcs>* értékére #1-ként kell utalni. Ha a kód független a *<kulcs>* értékétől, akkor az opciók megadásánál a *<kulcs>*-nak nem kell adni értéket, azaz a *<kulcs>* illetve *<kulcs>=bármi* ugyanazt jelenti.

if Ekkor létrejön egy \if*<érték>* feltétel. A *<kulcs>* értéke **true** vagy **false** lehet. Az opciók megadásánál *<kulcs>=true* helyett írható röviden *<kulcs>* is. Ezután

```
\if〈érték〉 〈igaz〉\else 〈hamis〉\fi
```

kifejtése az szerint 〈igaz〉 vagy 〈hamis〉, hogy a 〈kulcs〉 értéke `true` vagy `false`.

`store` Ekkor az 〈érték〉 egy még nem definiált parancs. A 〈kulcs〉 értéke ebben a parancsban lesz tárolva a \SetKeys[〈csomagnév〉]{〈opciók〉} (lásd később) parancs kiadásakor.

`usage` Ez akkor használható, ha ez előtt a 〈kulcs〉 már deklarálva volt az előző három 〈típus〉 valamelyikével. Ha az 〈érték〉 helyére `preamble` kerül, akkor a 〈kulcs〉 értékének a megadása csak preambulumban lehetséges. Ha az 〈érték〉 helyére `load` kerül, akkor a 〈kulcs〉 értékének a megadása csak a csomag betöltésekor a \usepackage opciójában lehetséges.

\SetKeys[〈csomagnév〉]{〈opciók〉}

Ezzel lehet megadni a deklarált opciók értékeit az 〈opciók〉-ban. Hatása lokális és többször is kiadható, akár a tex fájlból is. Elsőnek meg kell adni az alapértelmezett értéket. Ha egy használatakor egy opció értékét nem adja meg, akkor a korábban felvett értéke marad.

\ProcessKeyOptions

A \DeclareKeys és \SetKeys kiadása után írja ezt a parancsot. Ezzel a csomagopciók elérhetővé válnak a betöltésükkor a \usepackage opciójában.

Legyen például a `sajat.sty` tartalma:

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{sajat}[2025/10/04 v1.0 Ez a csomag csak egy pelda]

\DeclareKeys[sajat]{
  redemph.if      = sajat@redemph,
  redemph.usage   = load,
  institute.store = \Institute,
  parindent.code   = \setlength\parindent{\#1},
  no hyphen.code  = \hyphenpenalty10000\tolerance10000 }

\SetKeys[sajat]{redemph=false,institute=EKKE,parindent=0pt}
\ProcessKeyOptions

\RequirePackage{xcolor}
\ifsajat@redemph\DeclareEmphSequence{\color{red}\itshape,\upshape}\fi
\endinput
```

Ezután, ha a dokumentumban alapopciókkal tölti be a `sajat.sty` csomagot

```
\usepackage{sajat}
```

módon, akkor az alapopciók által megadott kódok érvényesülnek, azaz az \emph parancs nem lesz átdefiniálva, az \Institute parancs kifejtése „EKKE” lesz, a bekezdések előtti behúzás mértéke 0 pt és lesznek sorvégi szóelválasztások. De ha például így tölti be

```
\usepackage[redemph,institute={Eszterházy Károly Katolikus Egyetem},
           parindent=5pt,no hyphen]{sajat}
```

akkor az \emph parancs dőlt piros betűkkel emel ki, az \Institute parancs kifejtése „Eszterházy Károly Katolikus Egyetem” lesz, a bekezdések előtti behúzás mértéke 5 pt és nem lesznek sorvégi szóelválasztások. Ugyanezek az opciók a

```
\SetKeys[sajat]{〈opciók〉}
```

parancsban is megadhatók a `sajat` csomag betöltése után a `\redemph` opciót kivéve.

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:

```
\PackageError{<csomagnév>}{<hibaüzenet>}{<segítség>}
```

Az üzenetek szövegében a `\MessageBreak` parancssal tud sort törni.

```
\AtEndOfPackage{<kód>}
```

A csomagfájlban ezt a parancsot bárhol is adja ki, a `<kód>` a fájl végén lesz végrehajtva. Többször kiadva ezt a parancsot, a megadott kódokat egymás után fejti ki.

Amennyiben régebbi a TeX-rendszere és még nincs definiálva a kernelben `\SetKeys`, akkor `\kulcs=érték` típusú opciók definiálásához használhatja a `\kvoptions` csomagot. Legyen például a `sajat.sty` tartalma:

```
\NeedsTeXFormat[LaTeX2e][1999/12/01]
\ProvidesPackage{sajat}[2025/10/04 v1.0 Ez a csomag csak egy pelda]
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=sajat,prefix=sajat@}
\DeclareVoidOption{<opcion1>}{<kód1>}
\DeclareBoolOption[true]{<opcion2>}
\DeclareComplementaryOption{<opcion3>}{<opcion2>}
\DeclareStringOption[<kód4>]{<opcion4>}
\ProcessKeyvalOptions{sajat}
\if{sajat@<opcion2>} {<kód2 igaz>} \else {<kód2 hamis>} \fi
\endinput
```

Ez a kód az alábbiak szerint működik:

opcion	hatása
<code><opcion1></code>	<code><kód1></code>
<code><opcion2>, <opcion2>=true, <opcion3>=false</code> (alapopció)	<code><kód2 igaz></code>
<code><opcion3>, <opcion3>=true, <opcion2>=false</code>	<code><kód2 hamis></code>
<code><opcion4>=<kód5></code> (alapopció <code><opcion4>=<kód4></code>)	<code>\def\sajat@<opcion4>{<kód5>}</code>

Ezután, ha a dokumentumban alapopciókkal tölti be a `sajat.sty` csomagot

```
\usepackage{sajat}
```

akkor az alapopciók által megadott kódok érvényesülnek:

- `<kód2 igaz>`
- `\sajat@<opcion4>` eredménye `<kód4>`.

Ha például így tölti be

```
\usepackage[<opcion1>,<opcion2>=false,<opcion4>=<kód5>]{sajat}
```

akkor a következő kódok érvényesülnek:

- `<kód1>`
- `<kód2 hamis>`
- `\sajat@<opcion4>` eredménye `<kód5>`.

A `sajat` csomag bármelyik opciója a

```
\setkeys{sajat}{<opcionk>}
```

parancsban is aktiválható, kivéve az `<opcion2>`.

23.2. Dokumentumosztály készítése

Dokumentumosztály esetén hasonló az eljárás, mint a csomagnál, csak a dokumentumosztály forrásfájljának kiterjesztése `cls` és a `\ProvidesPackage` helyett `\ProvidesClass` parancsot kell használni. Célszerű egy létező dokumentumosztályt betölteni alapnak a `\LoadClass` parancsal. Nézzük a következő példát. Legyen a `sajat.cls` tartalma:

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesClass{sajat}[2025/10/04 v1.0 Ez az osztaly csak egy pelda]

\DeclareKeys{sajat} {
  <opció>.if      = sajat@<opció>,
  <opció>.usage   = load }

\SetKeys{sajat}{<opció>}
\ProcessKeyOptions

\ifsajat@<opció> <kód igaz>\else<kód hamis>\fi

\LoadClass[12pt,a4paper]{article}
\RequirePackage[T1]{fontenc}
\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}
\RequirePackage[magyar]{babel}
\endinput
```

Ezután, ha a dokumentumban például

```
\documentclass[<opció>=false]{sajat}
```

módon tölti be a `sajat.cls` osztályfájlt, akkor egy 12 pt-os alap betűméretű, A4-es oldalméretű, magyar tipográfiájú dokumentumot kap az `article` osztálynak megfelelően, amelyben a `<kód hamis>` lesz kifejtve.

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}[2025/10/04 v1.0 Ez az osztaly csak egy pelda]
\DeclareKeys{sajat} {
  11pt.code  = \PassOptionsToClass{11pt}{article},
  11pt.usage = load,
  12pt.code  = \PassOptionsToClass{12pt}{article},
  12pt.usage = load }
\ProcessKeyOptions
\LoadClass[a4paper]{article}
\endinput
```

Ekkor a `sajat` dokumentumosztály opciójaként használható az `article` dokumentumosztály `11pt` és `12pt` opciói.

Lehetőség van arra, hogy adott esetben a fordításnál valamilyen figyelmeztetést küldjön a felhasználónak:

```
\ClassWarning{<osztálynév>} {<figyelmeztetés>}
```

Azt is megteheti, hogy adott esetben a fordítás leálljon egy hibaüzenettel:

```
\ClassError{\osztálynév}{\hibaüzenet}{\segítség}
```

Az üzenetek szövegében a `\MessageBreak` parancssal tud sort törni.

```
\AtEndOfClass{\kód}
```

Az osztályfájlban ezt a parancsot bárhol is adja ki, a `\kód` a fájl végén lesz végrehajtva. Többször kiadva ezt a parancsot, a megadott kódokat egymás után fejti ki.

24. fejezet

Fontok kiválasztása

Ebben a fejezetben tárgyalt fontkiválasztási metódus csak a `latex` illetve `pdflatex` forrótítok használatának esetére vonatkozik.



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](#) fájl is.

A L^AT_EX fontbetöltési mechanizmusa 2020. februárjától megváltozott (a neve NFSS, ami a „New Font Selection Scheme” rövidítése), így az itt leírtak jó része csak az ezután telepített vagy frissített rendszerek esetén működik (lásd [ltnews31.pdf](#)).

24.1. 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. Az elérhető fontkészletek folyamatosan bővülnek, így a következő listák sem lehetnek teljesek. A listák a következő linkekre kattintva érhetők el: [antikva](#); [groteszk](#); [írogép](#); [egyéb](#).

24.2. A forrásfájl fontkódolása és a L^AT_EX belső kódkészlete

A forrásfájlban található ASCII karaktereknek (lásd a 22.1. szakaszban) minden fontkódolás esetén ugyanaz a kódszámuk. Például az `0` 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 `Ó` karakter helyére berakja a `\H{Ó}` 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 `0` 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 `¡`, ami meglepő eredményhez vezet:

```
\documentclass{article}
\begin{document}
<0
\end{document}
```

jO

A L^AT_EX úgynevezett belső kódkészlete fogja azt meghatározni, hogy egy nem ASCII karakternek megfelelő parancsnak a pdf-ben a fontkészlet melyik kódszámú karakterre feleljen meg. Alapesetben a L^AT_EX az OT1 jelű belső kódkészletet használja, amely olyan fontkészletekhez lett kitalálva, amelyek nem tartalmaznak ékezesek karaktereket. A cmr10 is ilyen. Így ékezes betűk esetén nem egy fontkészletbeli elem lesz hozzárendelve, hanem kettő: az alapbetű és az ékezet külön. Például az Ő betű helyére beillesztett \H{O} 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: Ŷ.

```
\documentclass{article}
\begin{document}
Ő
\end{document}
```

Ő

Az ékezesek két karakterből történő összerakása a pdf-ben néhány gondot okoz:

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

További probléma, hogy az alapértelmezett fontcsaládok nem mindegyike tartalmaz külön ékezetkaraktereket, így például a következő eset hibás eredményt produkál:

```
\documentclass{article}
\begin{document}
\ttfamily Ő
\end{document}
```

Ő

Ennek az az oka, hogy az itt használt cmtt10 fontkészletben a 125 decimális számmal kódolt helyen nem ékezet, hanem } van, így ezt teszi az 0 karakterre.

A megoldás az, hogy a L^AT_EX-ben telepített fontkészletek közül olyat kell használni, amelyben vannak ékezesek karakterek. Ilyen például az ecrm1000. Ebben 142 kódossal 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{O} 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 L^AT_EX úgy van beállítva, hogy T1 belső kódkészletre áttérve, alapból az ecrm1000 fontkészletet használja. Így a megoldáshoz elég a következő:

```
\documentclass{article}
```

```
\usepackage[T1]{fontenc}
\begin{document}
Ó<
\end{document}
```

Ó<

Ráadásul ekkor már a családváltás sem okoz hibás eredményt:

```
\documentclass{article}
\usepackage[T1]{fontenc}
\begin{document}
\ttfamily Ó
\end{document}
```

Ó

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.

24.3. 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**).

A család-, testesség- és alakkód lehetséges értékeit megtalálja a 24.1. szakaszban. A betűméret beállítását lásd a 4.5. szakaszban.

Hacsak a forráskódban nincs erre más irányú utasítás, a L^AT_EX-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 betűmétre, melyek alapesetben m, n és 10pt. Az **ot1cmr.fd** fájlban ezekre vonatkozóan azt az utasítást fogja találni a L^AT_EX-fordító, hogy a **cml10** 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.

24.3.1. Család

```
\renewcommand{\rmdefault}{<cslád>}
```

Az antikva családkód alapértéke legyen *<cslád>*. Az \rmfamily és \textrm ezt aktiválja. A *<cslád>* alapértéke **cml** (Computer Modern Roman).

```
\renewcommand{\sfdefault}{<cslád>}
```

A groteszk családkód alapértéke legyen `\család`. Az `\sffamily` és `\textsf` ezt aktiválja. A `\család` alapértéke `cmss` (Computer Modern Sans Serif).

```
\renewcommand{\ttdefault}{\család}
```

Az írógép családkód alapértéke legyen `\család`. Az `\ttfamily` és `\texttt` ezt aktiválja. A `\család` alapértéke `cmtt` (Computer Modern Typewriter).

```
\renewcommand{\familydefault}{\család}
```

Az alapértelmezett családkód legyen `\család`. A `\normalfont` és `\textnormal` ezt a családkódot aktiválja. A `\család` alapértéke `\rmdefault`. Lehet még `\sfdefault` és `\ttdefault`.

24.3.2. Testesség

```
\DeclareFontSeriesDefault[\családtípus]{md}{\testesség}
```

Ezzel lehet beállítani, hogy a `\testesség` legyen a normál testességekód alapértéke, azaz az `\mdseries` és `\textmd` ezt a testességekódot aktiválja, ha az aktuális család a `\családtípus`. A `\családtípus` lehetséges értékei `rm`, `sf` vagy `tt` aszerint, hogy az `\rmfamily` (`\textrm`), `\sffamily` (`\textsf`), `\ttfamily` (`\texttt`) parancsok közül melyikkel van deklarálva. Mindhárom esetben a `\testesség` alapértéke `m` (normál). Például

```
\DeclareFontSeriesDefault[rm]{md}{lc}
```

esetén

```
\rmfamily\mdseries
```

az `lc` testességekódot aktiválja. Megjegyezzük, hogy a

```
\DeclareFontSeriesDefault[\családtípus]{md}{\testesség}
```

kód ekvivalens a

```
\renewcommand{\mdseries@{\családtípus}}{\testesség}
```

kóddal.

```
\DeclareFontSeriesDefault{md}{\testesség}
```

Ezzel lehet beállítani, hogy a `\testesség` legyen a normál testességekód alapértéke minden családtípus esetén, azaz az `\mdseries` és `\textmd` ezt a testességekódot aktiválja minden esetben. Ez a kód ekvivalens a

```
\renewcommand{\mddefault}{\testesség}
```

kóddal. Alapesetben az `\mddefault` értékét (alapértéke `m`, azaz normál) az `\mdseries` és `\textmd` parancsok akkor veszik figyelembe, ha a családtípus nem az `\rmfamily`, `\textrm`, `\sffamily`, `\textsf`, `\ttfamily`, `\texttt` parancsok valamelyikével lett megadva. Ha az `\mdseries` illetve `\textmd` parancsok azt érzékelik, hogy az `\mddefault` értéke megváltozott, mert a korábbi kóddal átállítottuk, akkor lokálisan az `\mdseries@rm`, `\mdseries@sf`, `\mdseries@tt` értékei az `\mddefault` értékét veszik fel.

```
\DeclareFontSeriesDefault[\családtípus]{bf}{\testesség}
```

Ezzel lehet beállítani, hogy a `\testesség` legyen a félkövér testességekód alapértéke, azaz a `\bfseries` és `\textbf` ezt a testességekódot aktiválja, ha az aktuális család a `\családtípus`. A `\családtípus` lehetséges értékei `rm`, `sf` vagy `tt` aszerint, hogy az `\rmfamily` (`\textrm`), `\sffamily` (`\textsf`), `\ttfamily` (`\texttt`) parancsok közül melyikkel van deklarálva. Mindhárom esetben a `\testesség` alapértéke `bx` (kiterjesztett félkövér). Például

```
\DeclareFontSeriesDefault[rm]{bf}{eb}
```

esetén

```
\rmfamily\bfseries
```

az `eb` testességkódot aktiválja. Megjegyezzük, hogy a

```
\DeclareFontSeriesDefault[⟨családtípus⟩]{bf}{⟨testesség⟩}
```

kód ekvivalens a

```
\renewcommand{\bfseries@⟨családtípus⟩}{⟨testesség⟩}
```

kóddal.

```
\DeclareFontSeriesDefault{bf}{⟨testesség⟩}
```

Ezzel lehet beállítani, hogy a `⟨testesség⟩` legyen a félkövér testességkód alapértéke minden családtípus esetén, azaz a `\bfseries` és `\textbf` ezt a testességkódot aktiválja minden esetben. Ez a kód ekvivalens a

```
\renewcommand{\bfdefault}{⟨testesség⟩}
```

kóddal. Alapesetben a `\bfdefault` értékét (alapértéke `b`, azaz félkövér) a `\bfseries` és `\textbf` parancsok akkor veszik figyelembe, ha a családtípus nem az `\rmfamily`, `\textrm`, `\sffamily`, `\textsf`, `\ttfamily`, `\texttt` parancsok valamelyikével lett megadva. Ha a `\bfseries` illetve `\textbf` parancsok azt érzékelik, hogy a `\bfdefault` értéke megváltozott, mert a korábbi kóddal átállítottuk, akkor lokálisan a `\bfseries@rm`, `\bfseries@sf`, `\bfseries@tt` értékei a `\bfdefault` értékét veszik fel.

```
\renewcommand{\seriesdefault}{⟨testesség⟩}
```

Az alapértelmezett testességkód legyen `⟨testesség⟩`. A `\normalfont` és `\textnormal` ezt a testességkódot aktiválja. A `⟨testesség⟩` alapértéke `\mddefault`.

24.3.3. Alak

```
\renewcommand{\updefault}{⟨alak⟩}
```

Az álló alakkód alapértéke legyen `⟨alak⟩`. Az `\upshape` és `\textup` ezt az alakkódot aktiválja. Az `⟨alak⟩` alapértéke `up` (álló).

```
\renewcommand{\ulcdefault}{⟨alak⟩}
```

A kis/nagybetűs alakkód alapértéke legyen `⟨alak⟩`. Az `\ulcshape` és `\textulc` ezt az alakkódot aktiválja. Az `⟨alak⟩` alapértéke `ulc` (kis/nagybetűs).

```
\renewcommand{\sldefault}{⟨alak⟩}
```

A döntött alakkód alapértéke legyen `⟨alak⟩`. Az `\slshape` és `\textsl` ezt az alakkódot aktiválja. Az `⟨alak⟩` alapértéke `sl` (döntött).

```
\renewcommand{\itdefault}{⟨alak⟩}
```

A dőlt alakkód alapértéke legyen `⟨alak⟩`. Az `\itshape` és `\textit` ezt az alakkódot aktiválja. Az `⟨alak⟩` alapértéke `it` (dőlt).

```
\renewcommand{\scdefault}{⟨alak⟩}
```

A kiskapitalis alakkód alapértéke legyen `⟨alak⟩`. Az `\scshape` és `\textsc` ezt az alakkódot aktiválja. Az `⟨alak⟩` alapértéke `sc` (kiskapitalis).

```
\renewcommand{\shapedefault}{⟨alak⟩}
```

Az alapértelmezett alakkód legyen $\langle alak \rangle$. A `\normalfont` és `\textnormal` ezt az alakot aktiválja. Az $\langle alak \rangle$ alapértéke `n` (normál, álló kis/nagybetűs). Lehet még `\sdefault`, `\itdefault` és `\scdefault`. Az alapérték azért `n` és nem `\updefault`, mert az `\upshape` és `\textup` parancsok dölt vagy döntött kiskapitális esetén csak az álló alakot kell hogy visszaállítsák, a kis/nagybetűs alakot nem, míg a `\normalfont` és `\textnormal` az alapértelmezett alakra tér vissza.

```
\DeclareFontShapeChangeRule{\langle alak1 \rangle}{\langle alak2 \rangle}{\langle alak3 \rangle}{\langle alak4 \rangle}
```

Ha az aktuális alakkód $\langle alak1 \rangle$ és ezt akarjuk kombinálni az $\langle alak2 \rangle$ alakkóddal, akkor az $\langle alak3 \rangle$ lesz aktív, ha azt az aktuális fontkészlet támogatja, különben az $\langle alak4 \rangle$. Ha az $\langle alak4 \rangle$ sem támogatott, akkor az $\langle alak2 \rangle$ válik aktívvá. Ha egy $\langle alak1 \rangle$ – $\langle alak2 \rangle$ párhoz nincs definiálva $\langle alak3 \rangle$ illetve $\langle alak4 \rangle$ kimenet a `\DeclareFontShapeChangeRule` parancssal, akkor az $\langle alak2 \rangle$ válik aktívvá. Számos alakváltó szabály van előre definiálva:

$\langle alak1 \rangle$	$\langle alak2 \rangle$	$\langle alak3 \rangle$	$\langle alak4 \rangle$	$\langle alak1 \rangle$	$\langle alak2 \rangle$	$\langle alak3 \rangle$	$\langle alak4 \rangle$
<code>n</code>	<code>it</code>	<code>it</code>	<code>sl</code>	<code>scit</code>	<code>sw</code>	<code>scsw</code>	<code>sc</code>
<code>n</code>	<code>sl</code>	<code>sl</code>	<code>it</code>	<code>scit</code>	<code>sc</code>	<code>scit</code>	
<code>n</code>	<code>ulc</code>	<code>n</code>		<code>scit</code>	<code>ulc</code>	<code>it</code>	
<code>n</code>	<code>up</code>	<code>n</code>		<code>scit</code>	<code>up</code>	<code>sc</code>	
<code>it</code>	<code>sl</code>	<code>sl</code>	<code>it</code>	<code>scsl</code>	<code>it</code>	<code>scit</code>	<code>scsl</code>
<code>it</code>	<code>sc</code>	<code>scit</code>	<code>scsl</code>	<code>scsl</code>	<code>sl</code>	<code>scsl</code>	
<code>it</code>	<code>ulc</code>	<code>it</code>		<code>scsl</code>	<code>sw</code>	<code>scsw</code>	<code>sc</code>
<code>it</code>	<code>up</code>	<code>n</code>		<code>scsl</code>	<code>sc</code>	<code>scsl</code>	
<code>sl</code>	<code>it</code>	<code>it</code>	<code>sl</code>	<code>scsl</code>	<code>ulc</code>	<code>sl</code>	
<code>sl</code>	<code>sc</code>	<code>scsl</code>	<code>scit</code>	<code>scsl</code>	<code>up</code>	<code>sc</code>	
<code>sl</code>	<code>ulc</code>	<code>sl</code>		<code>scsw</code>	<code>it</code>	<code>scit</code>	<code>scsw</code>
<code>sl</code>	<code>up</code>	<code>n</code>		<code>scsw</code>	<code>sl</code>	<code>scsl</code>	
<code>sc</code>	<code>it</code>	<code>scit</code>	<code>scsl</code>	<code>scsw</code>	<code>sw</code>	<code>scsw</code>	
<code>sc</code>	<code>sl</code>	<code>scsl</code>	<code>scit</code>	<code>scsw</code>	<code>sc</code>	<code>scsw</code>	
<code>sc</code>	<code>sw</code>	<code>scsw</code>	<code>sw</code>	<code>scsw</code>	<code>ulc</code>	<code>sw</code>	
<code>sc</code>	<code>ulc</code>	<code>n</code>		<code>scsw</code>	<code>up</code>	<code>sc</code>	
<code>sc</code>	<code>up</code>	<code>n</code>		<code>sw</code>	<code>sc</code>	<code>scsw</code>	
<code>scit</code>	<code>it</code>	<code>scit</code>		<code>sw</code>	<code>ulc</code>	<code>sw</code>	
<code>scit</code>	<code>sl</code>	<code>scsl</code>	<code>scit</code>	<code>sw</code>	<code>up</code>	<code>n</code>	

Például alapesetben egy dokumentum kezdésekor a `\normalfont` lép érvénybe, ami a `\shapedefault` értékét, azaz az `n` alakkódot aktiválja. Vagyis ekkor $\langle alak1 \rangle = n$. Ha ezután kiadjuk az `\upshape` parancsot, akkor az aktiválja az `\updefault` értékét, ami alapesetben `up`. Vagyis ekkor $\langle alak2 \rangle = up$. Az előző táblázatból kiolvasható, hogy ekkor $\langle alak3 \rangle = n$, tehát ez az alakkód válik aktívvá. Ebben az esetben az $\langle alak4 \rangle$ nincs definiálva, mert `n` alakkód minden fontkészletben megtalálható.

Másik példaként elemezzük ki, hogy mi történik, ha alapesetben kiadjuk az `\scshape` parancsot egy üres dokumentum elején. Az előbb már láttuk, hogy ekkor $\langle alak1 \rangle = n$. Az `\scshape` aktiválja az `\scdefault` értékét, ami alapesetben `sc`, azaz $\langle alak2 \rangle = sc$. A táblázatban nincs ilyen variáció, ezért ekkor az $\langle alak2 \rangle = sc$ válik aktívvá. Ha ezután kiadjunk egy másik alakváltó parancsot, például `\itshape`, akkor annak értelmezésekor már $\langle alak1 \rangle = sc$ és $\langle alak2 \rangle = it$ (hiszen az `\itshape` aktiválja az `\itdefault` értékét, ami alapesetben `it`). Így a táblázat alapján először az $\langle alak3 \rangle = scit$ alakkódot próbálja aktiválni, ha az az adott fontkészletben definiált. Ha nem, akkor az $\langle alak4 \rangle = scsl$ lép érvénybe. Ha ez sem támogatott, akkor az $\langle alak2 \rangle = it$ aktiválódik.

24.4. Lokális beállítás

A következő parancsokkal egy adott helyen ideiglenesen áttérhetünk az alapbeállításoktól 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[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\ldots}
(William Shakespeare)
\end{document}
```

„Lenni vagy nem lenni: az itt a kérdés...” (William Shakespeare)

24.5. Testességek kombinálása

Korábban láttuk, hogy a különböző alakok kombinálhatóak, ha azt az aktuális fontkészlet támogatja. Ez a testességre is igaz. Például

```
{\usefont{T1}{antt}{c}{n} szöveg 1 \bfseries szöveg 2}
```

szöveg 1 szöveg 2

Itt a „szöveg 1” testességek kódja `c`, másrészett, mivel a család `\usefont` parancssal van deklarálva, ezért a `\bfseries` a `b` testességek kódot aktiválja. A „szöveg 2” testességek kódja ezek kombinálásával `bc` lesz. A kombinálási szabályokat a

```
\DeclareFontSeriesChangeRule{<testesség1>}{<testesség2>}{{<testesség3>}{<testesség4>}}
```

parancssal adhatjuk meg, ami csak a preambulumban adható ki. Ha az aktuális testességek kódai `<testesség1>` és ezt akarjuk kombinálni a `<testesség2>` testességek kódjával, akkor a `<testesség3>` lesz aktív, ha azt az aktuális fontkészlet támogatja, különben a `<testesség4>`. Ha a `<testesség4>` sem támogatott, akkor a `<testesség2>` válik aktívvá.

Ha a `<testesség1>`–`<testesség2>` variációhoz nincs `<testesség3>` illetve `<testesség4>` kimenet definiálva a `\DeclareFontSeriesChangeRule` parancssal, akkor a `<testesség2>` válik aktívvá.

Rengeteg testességeket kombináló szabály van előre definiálva. Ezeket itt terjedelmi okok miatt nem részletezzük.

24.6. Családkód deklarálása

24.6.1. Több családkód összevonása új kóddal

Akár több családkód is összevonható egy új kód alatt. Új családkód a következő paramccsal deklarálható:

```
\DeclareFontFamily{<kódolás>}{<új családkód>}{}{}
```

Ezután az `<új családkód>` alatt egy adott testességhoz és alakhoz a következő módon rendelhetünk egy korábban már definiált család-, testesség- és alakkódot:

```
\DeclareFontShape{<kódolás>}{<új családkód>}{<új testessékgód>}{<új alakkód>}{<->ssub*<család>/<testesség>/<alak>}{}{}
```

Fontos, hogy a `<kódolás>` ugyanaz legyen, mint a `<család>`-hoz tartozó belső kódkészlet. Például

```
\DeclareFontFamily{T1}{myrm}{}{}
\DeclareFontShape{T1}{myrm}{m}{n}{<->ssub*c1m/m/n}{}{}
\DeclareFontShape{T1}{myrm}{b}{sc}{<->ssub*fnc/b/sc}{}{}
```

Ezután

```
\usefont{T1}{myrm}{m}{n} szöveg
\usefont{T1}{myrm}{b}{sc} szöveg
```

és

```
\usefont{T1}{c1m}{m}{n} szöveg
\usefont{T1}{fnc}{b}{sc} szöveg
```

kódok ugyanazt eredményezik:

szöveg SZÖVEG

24.6.2. Új családkód deklarálása

Az előző alszakasz azt mutatta meg, hogyan lehet új családkódot létrehozni korábban definiáltak segítségével. De ezeket a családkódokat hogyan definiálták? Ennek illusztrációra nézzük meg, hogy T1 belső kódkészlettel az `lmr` családkód hogyan van definiálva `m` testessékgód 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` testessékgóddal 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 fontkészlet fogja betölteni. Azért töltenek be a különböző mérettartományokban más-más fontkészleteket, mert egy font 5 pt méretben lehet, hogy jól néz ki, de kinagyítva 20 pt méretre már nem biztos, hogy a legideálisabb. Ennek illusztrálására nézzük meg a „szöveg” kiszedését különböző fontkészletekkel és méretekben:

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 elő 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}{}  
A fontnevek elő 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:
```

24.7. 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. Érdemes elolvasni a csomagok leírásait is, mert egyesekhez opciók is tartoznak.

csomag	antikva	groteszk	írógép	mat.
anttor	antt	-	-	<input checked="" type="checkbox"/>
arev	fav	fav	fvm	<input checked="" type="checkbox"/>
bera	fve	fvs	fvm	-
cmbright	-	cmbr	cmtl	<input checked="" type="checkbox"/>
cyklop	cyklop	-	-	-
kpfonts	jkp	jkpss	jkptt	<input checked="" type="checkbox"/>
lmodern	lmr	lmss	lmtt	<input checked="" type="checkbox"/>
lxfonts	-	llcmss	llcmtt	<input checked="" type="checkbox"/>
mlmodern	mlmr	milmss	mlmtt	<input checked="" type="checkbox"/>
newpxtext, newpxmath	zpllf	qhv	npxtt	<input checked="" type="checkbox"/>
newtxtext, newtxmath	ntxtlf	qhv	ntxtt	<input checked="" type="checkbox"/>
pxfonts	pxr	pxss	pxtt	<input checked="" type="checkbox"/>
stix	stix	-	-	<input checked="" type="checkbox"/>
tgadventor	-	qag	-	-
tgbonum	qbk	-	-	-
tgchorus	qzc	-	-	-
tgcursor	-	-	qcr	-

Folytatás a következő oldalon!

csomag	antikva	groteszk	írógép	mat.
tgheros	-	qhv	-	-
tgpagella	qpl	-	-	-
tgschola	qcs	-	-	-
tgtermes	qtm	-	-	-
times	ptm	phv	pcr	-
txfonts	txr	txss	txtt	<input checked="" type="checkbox"/>

24.8. 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 megtérvezett 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 kódszámaival 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{\fontht}
\newlength{\fonthtd}
\newlength{\fonthtnext}
\newlength{\fonthtnextd}
\settoheight{\fontht}{\FONTS\char0}
\settodepth{\fonthtd}{\FONTS\char0}
\addtolength{\fontht}{\fonthtd}
\loop
```

```
\ifnum\value{currchar}<255
\stepcounter{currchar}
\settoheight{\fonthtnext}{\FONT\char\arabic{currchar}}
\settodepth{\fonthtnextd}{\FONT\char\arabic{currchar}}
\addtolength{\fonthtnext}{\fonthtnextd}
\ifdim\fonthtnext>\fontht\setlength{\fontht}{\fonthtnext}\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}
```

25. fejezet

LuaL^AT_EX és XeL^AT_EX

A **LuaL^AT_EX** a pdfL^AT_EX kiterjesztett változata, amely a **Lua**-t beágyazott szkriptnyelv-ként használja. Futtatása TeXstudióban **Eszközök** ▶ **Parancsok** ▶ **LuaLaTeX**, parancssorból

```
lualatex dokumentum.tex
```

Együttműködik a **latexmk**-val is a következő parancssorral:

```
latexmk -lualatex dokumentum
```

Hosszútávon érdemes megfontolni a pdflatex fordítóról való áttérést a lualatex használatára, mely számos előnnyel jár.



A **XeL^AT_EX** név az eXtended L^AT_EX kifejezésre utal. Kiejtése „zilatexh”. Ez is egy L^AT_EX fordító, mely a forrást xdv-be (extended dvi), majd az xdvipdfmx programmal az xdv-t pdf-be konvertálja. Ezután az xdv fájlt törli. Használata TeXstudióban **Eszközök** ▶ **Parancsok** ▶ **XeLaTeX**, parancssorból

```
xelatex dokumentum.tex
```

Együttműködik a **latexmk**-val is a következő parancssorral:

```
latexmk -xelatex dokumentum
```

A xelatex használata ma már nem javasolt, mert úgy tűnik, hogy 2020-ban megrekedt a fejlesztése.



25.1. Jellemzőik

- Ezekkel nemcsak a T_EX-rendszer saját TFM (TeX Font Metric) és Type 1 fontjai, hanem akár külső TrueType és OpenType fontok is betölthetők. A külső fontok használata esetén számolni kell azzal, hogy a dokumentum nem lesz hordozható, hiszen más gépen nem biztos, hogy a forrás által használt fontok telepítve vannak. Ez a probléma úgy oldható meg, hogy a használt fontkészletek fájljait a forrásfájl mellé kell tenni, és onnan kell betölteni.
- Csak UTF-8 kódolású forráskóddal működnek.
- minden karakter tömbnek számít, így például a következő kód lefordítása nem generál hibát (ellenértben a pdflatex-hel, lásd a 30. oldalon):
`\textit{ és }`
- A forrásfájlból az **inputenc** és **fontenc** csomagok helyett a **fontspec** csomagot kell betölteni.

- Alapból a Latin Modern belső fontkészletet tölti be TU belső kódkészlettel. (Ez a kódkészlet `latex` és `pdflatex` fordítók esetén nem használható.)
- Matematika fontok kezelése a `mathspec` vagy `unicode-math` csomagokkal lehetséges. Az `amsmath`, `amssymb`, `mathtools` csomagokat ezek előtt kell betölteni.
- Képek eps, pdf, jpg, png formátumban is betölthetőek. (A `luatex` az eps képeket először pdf-be konvertálja, majd azt tölti be, pont úgy, mint a `pdflatex`.)
- Az `f{}f` nem akadályozza meg a ligatúrát. Helyette: `f\mbox{}f`.

25.2. Fontok betöltése

Korábban láttuk, hogy a fontokat három családba oszthatjuk: antikva, groteszk, írógép. Ezeket rendre a

```
\setmainfont ∈ fontspec
\setsansfont ∈ fontspec
\setmonofont ∈ fontspec
```

parancsokkal 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 nem telepített fontot akar betölteni, akkor a fontfájlt rakja a forrásfájl mellé és a fájl nevét a kiterjesztésével együtt írja a font neve helyére. Például

```
\setmainfont[NotePlanner.ttf]
```

Ha ideiglenesen át akar térti egy ezektől különböző betűcsaládra, akkor használja a

```
\fontspec ∈ fontspec
```

parancsot. Például

```
\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 T_EX-rendszerből nem csak a Latin Modern fontkészlet tölthető be. Használhatóak például még az előző parancsokkal a következő fonttípusok is, melyek közül az első három a Computer Modern fontkészletet tölti be:

```
CMU Serif
CMU Sans Serif
CMU Typewriter Text
```

```
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}
```

Az elérhető fontok neveit kilistázhatja a következő parancssorral:

```
luafindfont -x "*"
```

25.3. Lua szkript használata lualatex fordítóval

A `lualatex` fordító nagy előnye a külső fontok használatán túl, hogy `Lua` szkriptek ágyazhatók be a forrásfájlba. Például

```
</> \documentclass{article}
\usepackage{fontspec,luacode}

\begin{luacode}
function rows(n)
    for i = 0, n do
        x = 0.1 * i
        y = math.sin(x)
        z = math.cos(x)
        tex.sprint(x .. " & " .. y .. " & " .. z .. " \\ \\
    end
end
\end{luacode}

\def\sincostable#1{%
\begin{tabular}{lll}
$x$ & $\sin(x)$ & $\cos(x)$ \\
\hline
\directlua{rows(#1)}
\end{tabular}

\begin{document}
\sincostable{10}
\end{document}
```

25.4. A fordító detektálása

Ha figyelmeztetni akarja a felhasználót, hogy `pdflatex`, `xelatex` vagy `lualatex` fordítót kell alkalmaznia, akkor rendre használja a

```
\RequirePDFTeX ∈ iftex
\RequireXeTeX ∈ iftex
\RequireLuaTeX ∈ iftex
```

parancsokat. Ha olyan forrást akar, amely többféle fordítóval is használható, akkor alkalmazza az

```
\iftutex ∈ iftex
```

feltételes utasítást. Például

```
\documentclass{article}
\usepackage{iftex}
\iftutex
  \usepackage{fontspec}
\else
  \usepackage[T1]{fontenc}
\fi
\begin{document}
...
\end{document}
```

esetén, ha `xelatex` vagy `lualatex` a fordító, akkor a `fontspec` csomagot tölti be, míg ellenkező esetben a `fontenc` csomagot.

26. fejezet

További információk

26.1. A TeX Live és a MiKTeX pdf tömörítési szintje

Ha ugyanazt a forrásfájlt TeX Live illetve MiKTeX rendszeren is lefordítja `pdflatex` fordítóval, akkor azt fogja tapasztalni, hogy bár a két pdf kinézetre teljesen egyforma, de TeX Live esetében kisebb méretű a fájl. Ennek az az oka, hogy a két rendszer alapbeállításaiban a pdf tömörítési szintje különböző értékre van beállítva. Ha ezt el akarja kerülni, akkor a forrásfájlba közvetlenül a dokumentumosztály betöltése után írja be a következőket:

```
\pdfcompresslevel=9  
\pdfobjcompresslevel=2
```

26.2. A generált pdf verziószáma

A `pdflatex` fordítóval generált pdf verziószáma alapértelmezésben 1.7. Ha ezt például 1.5-re szeretnénk módosítani, akkor a dokumentumosztály betöltése után írja be a következőket:

```
\pdfmajorversion=1 % Ezt nem kötelező beírni, mert ez az alapértelmezés.  
\pdfminorversion=5
```

A másik lehetőség, amely minden fordító esetén működik, hogy a dokumentumosztály betöltése előtt(!) írja be a következőt:

```
\DocumentMetadata{pdfversion=1.5}
```

26.3. Ha a magyar nem alapnyelvként van beállítva

Próbálja ki a következő kódot, melyben nem a magyar az alapnyelv.

```
\documentclass{book}  
\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 töltse be a `classmod=unchanged` opciót is.

26.4. Rendszerparancsok végrehajtása fordítás közben

Néha szükség lehet a tex fájl pdf-be konvertálása közben terminálon futtatható rendszerparancsok végrehajtására. Erre használható a

`\ShellEscape{<rendszerparancs>} ∈ shellessc`

parancs. Ennek használatára nézzük a következő összetett példát:

```
\documentclass{article}
\usepackage{graphicx, listings, shellessc}
\begin{document}

\begin{filecontents*}[overwrite]{minta.tex}
\documentclass{article}
\usepackage{lipsum}
\begin{document}
\lipsum[1-5]
\end{document}
\end{filecontents*}

\lstinputlisting{minta.tex}

\ShellEscape{pdflatex minta.tex}
\ShellEscape{del minta.tex} % unix rendszereken 'del' helyett 'rm'
\ShellEscape{del minta.log} % unix rendszereken 'del' helyett 'rm'
\ShellEscape{del minta.aux} % unix rendszereken 'del' helyett 'rm'

\begin{center}
\fbox{\includegraphics[width=10cm]{minta}}
\end{center}

\end{document}
```

Ennek lefordításához szükség van a fordító `-shell-escape` kapcsolójára is. Ha a forrásállomány például a `dokumentum.tex`, akkor parancssorba írja be, hogy

`pdflatex -shell-escape dokumentum.tex`

majd `Enter`. Ha kereszthivatkozásokat is használ, akkor célszerűbb a `latexmk` program használata `-shell-escape` kapcsolóval:

`latexmk -pdf -shell-escape dokumentum`

majd `Enter`. TeXstudióból történő fordításhoz alkalmazza az 1.10. szakasz 2. pontjának beállítását. Ezután `Eszközök` ▶ `Parancsok` ▶ `Latexmk`.

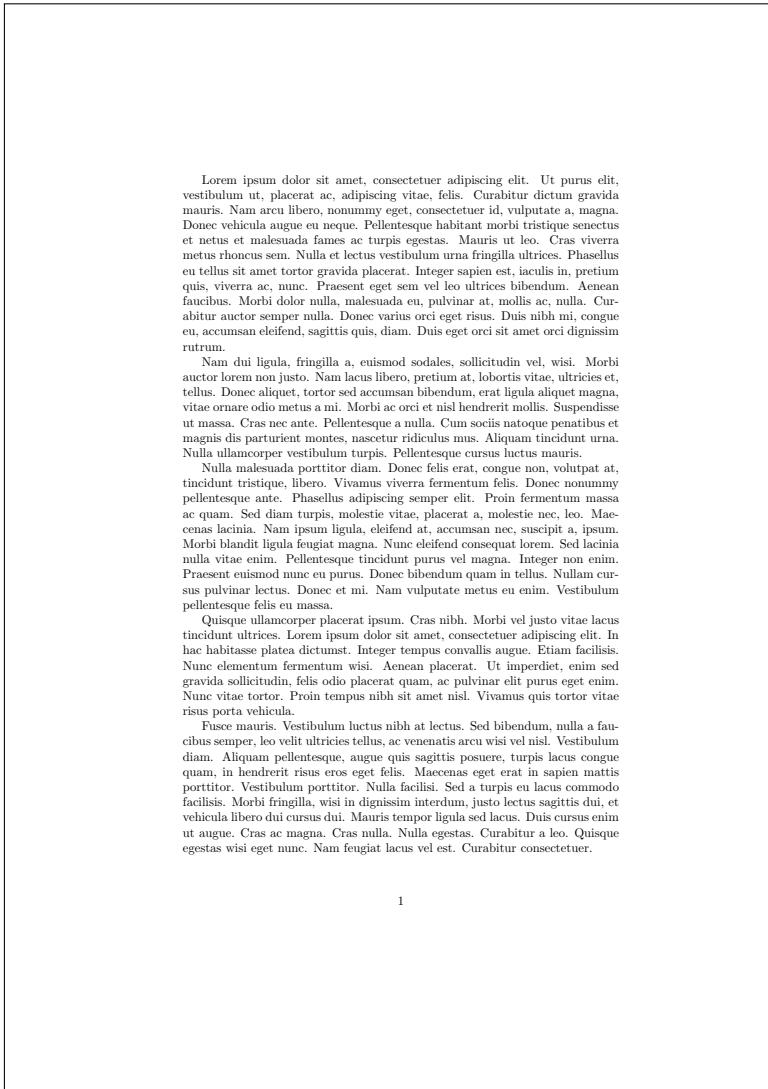
Az előző kód először a `filecontents*` környezet tartalmát kiírja egy `minta.tex` fájlba, majd annak tartalmát megjeleníti verbatim szövegként. Ezután futtatja a következő rendszerparancsokat:

```
pdflatex minta.tex
del minta.tex
del minta.log
```

del minta.aux

Ez először a `minta.tex` fájlt konvertálja `minta.pdf`-be, majd törli a `minta.tex` fájlt, továbbá a fordításkor keletkező `minta.log` és `minta.aux` munkafájlokat. (Ha Windows helyett unix rendszeren dolgozik, akkor `del` helyett írjon `rm` rendszerparancsot.) Végül egy 10 cm széles bekeretezett dobozban középre illesztve megjelenik képként a `minta.pdf` fájl. A kapott eredmény:

```
\documentclass{article}
\usepackage{lipsum}
\begin{document}
\lipsum[1-5]
\end{document}
```



26.5. Szöveg másolása pdf-ból

Ha pdf-ből kimásolva egy szövegrészt, majd azt egy editorba beszűrva rossz karaktereket kapunk, és valamiért fontos, hogy ez ne így legyen, akkor a forrásfájlban töltse

be a `cmap` csomagot a preambulum elején és az `upquote` csomagot a preambulum végén. 2021. júniusa után telepített rendszerekben a `cmap` csomag betöltésére már nincs szükség.

26.6. 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}}
\begin{dirtree}%
.1/images\DTcomment{képek helye}.
.2/sources.
}
```

`bookcover` Könyvborító készítéséhez.

`empheq` Többsoros képlet keretezésére.

`fancypar` Bekezdések hátterének beállítása.

`fancy tooltips` Hivatkozások külön ablakban bukanjanak fel.

`fp` Fixpontos aritmetikánál használható, maximum 18 számjegyig.

`hfoldsty` Régi típusú számok.

`keystroke` Billentyűzet rajzolására.

`lwarpx` 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` A pdf-ben felbukkanó megjegyzések írása.

`pdfmarginpar` A pdf-ben felbukkanó megjegyzések írása.

`picinpar` Képek körbefuttatására.

`prettyref` A `\ref` parancs tudását bővíti.

`pst-3d` 3D árnyékoláshoz (csak `latex` fordítóval működik).

`pst-3dplot` 3D rajzokhoz (csak `latex` fordítóval működik).

`pst-fr3d` 3D dobozhoz (csak `latex` fordítóval működik).

`pst-text` Görbén vezetett szöveghez (csak `latex` fordítóval működik).

`refcheck` A pdf-be írja a kereszthivatkozások label-jeit széljegyzetként. Azt is mutatja, hogy melyekre hivatkoztunk, melyekre nem.

`relsizer` 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 íratjuk ki nagyon sok tizedesjeggyel.

`shadethm` Tételeszerű környezetek árnyékolására.

`sidecap` A kép címét a kép oldalára lehet rakni.

`sketch` A tikz csomagot kiegészíti 3D lehetőségekkel.

`splitindex` Több tárgymutató is készíthető egy dokumentumban.

spot Különlegesen lehet kiemelni.

subfigure Számoszott képeknél alszámoszás esetén.

stringstrings Sztringek kezelése

sverb Például `\begin{demo}{Cím}$$\frac{12}{\end{demo}}`

tablists Sorfolytonos számoszott 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 L^AT_EX konvertálása HTML illetve XML formátumba

textpos Szöveget az adott oldal tetszőleges pozíciójába rakhatunk.

tocloft Tartalomjegyzék stílus készítés.

todonotes Dokumentumban megjegyzésekkel lehet ezzel készíteni.

tram Szöveg hátterét kipontozza.

umoline Többsoros szöveg aláhúzásához.

varioref A `\ref` parancs tudását bővíti.

venndiagram Egyszerűen rajzolhatunk Venn-diagramokat.

vwcol Többhasábos szedést lehet csinálni úgy, hogy a hasábok különböző szélesek legyenek.

xargs Többopciónak definiálásának megkönnyítése

xstring Sztringek kezelése

27. fejezet

LATEX-kurzus videókkal

Ez a fejezet az Eszterházy Károly Katolikus Egyetem LATEX-kurzusának anyagát tartalmazza, amely önálló tanulásra is felhasználható. minden leckéhez ikonokkal jelölt linkek tartoznak:

- A leckéhez tartozó pdf. A gyakorlatok esetében ez az előállítandó dokumentum kimenete.
- A leckéhez tartozó dokumentum megoldásának LATEX-forrása zip fájlba csomagolva.
- Amennyiben az [Overleaf](#) oldalon van regisztrációja, akkor ezzel a linkkel közvetlenül is betöltheti oda a lecke megoldásának LATEX-forrását.
- A lecke itt tekinthető meg videón.

A kurzus anyaga

Bevezető

Telepítés

Tananyag

Az első dokumentum elkészítése

1. *gyakorlat* – bekezdések, központozás, betűméretek, betűtípusok, igazítások, listák, térközök

2. *gyakorlat* – számosztott listák testreszabása, táblázatok, úsztatás, kereszthivatkozások, lábjegyzetek

3. *gyakorlat* – URL címek, képek, úsztatás, kereszthivatkozások

4. *gyakorlat* – saját úsztatott környezetek készítése, dobozok, többhasábos szedés, színek kezelése

5. *gyakorlat* – matematikai képletek

6. *gyakorlat* – verbatim, programkódok szintaxis kiemelése (highlight)

7. gyakorlat – strukturált dokumentum készítése **article** dokumentumosztályban, té-
- telszerű környezetek    
8. gyakorlat – strukturált dokumentum készítése **report** dokumentumosztályban, oldal
- geometriája, tartalomjegyzék, fej- és lábléc, irodalomjegyzék, hivatkozás iroda-
- lomra    
9. gyakorlat – szakdolgozat **thesis-ekf** dokumentumosztályban
- Az Eszterházy Károly Katolikus Egyetem szakdolgozatsablonja    
10. gyakorlat – prezentáció készítése **beamer** dokumentumosztályban
- Beadandó feladat az Eszterházy Károly Katolikus Egyetem hallgatói részére*   

28. fejezet

Linkek

28.1. A könyvben található videók

- [TeX-rendszer telepítése Windowsra](#)
- [Az első L^AT_EX-dokumentum készítése](#)
- [Betűtípusok és -méretek, térközök, törések](#)
- [Bekezdések, lábjegyzetek, színek, kereszthivatkozások](#)
- [Listák](#)
- [Képek és táblázatok](#)
- [Irodalomjegyzék készítése `biblatex` csomaggal](#)
- [Szakdolgozat készítése az Eszterházy Károly Katolikus Egyetem sablonjával](#)

28.2. Sablonok

- | | |
|---|---|
| – LaTeX Templates | |
| – TeXample.net | |
| – Magyar nyelvű dokumentumalap |  |
| – Szakdolgozat – thesis-ekf |  |
| – Prezentáció |  |
| – Curriculum Vitae |  |
| – Levél |  |
| – Angol nyelvű cikk article osztályjal |  |
| – Angol nyelvű cikk amsart osztályjal |  |
| – Annales Mathematicae et Informaticae folyóirat cikksablonja |  |
| – Könyvbörítők |  |

28.3. TeX-rendszer

- [TeX Live \(svn – source – tlnet-archive – bug – bug report: `tex-live@tug.org`\)](#)
- [MacTeX](#) TeX Live alapú rendszer Macintoshra
- [Overleaf](#) Online működő TeX Live alapú rendszer
- [MiKTeX](#) (`github` – `ctan.org`)
- [TeXfireplace](#) MiKTeX-alapú rendszer Windowsra

28.4. \TeX -hez fejlesztett editorok

- [TeXstudio](#) (Sourceforge, GitHub)
- [Texmaker](#)
- [WinEdt](#) (shareware)
- [Kile](#)
- [TeXworks](#)
- [TeXnicCenter](#) (Sourceforge)
- [LyX](#) „Rich text” szerkesztő felületet biztosít, amely félíg „Amit látsz, azt kapod” típusú rendszer. Más szerkesztő által létrehozott \LaTeX -forrást nem tudja kezelni.
- [JabRef](#) A Bib \TeX használatát segítő editor.
- Editorok összehasonlítása

28.5. \LaTeX oldalak

- [TeX Users Group](#)
- [The Comprehensive TeX Archive Network](#)
- [The LaTeX Project](#)
- [BME Math LaTeX](#)

28.6. Leírások

- [The TeXbook](#)
- [LaTeX2e: An unofficial reference manual](#)
- [LaTeX Wikibook](#)
- [Dickimaw LaTeX Books](#)
- [TeXdoc Online](#)
- [TeX tips](#)
- [TeX FAQ – Frequently Asked Question List for TeX](#)
- [Learn \$\text{\LaTeX}.org\$](#)
- Egy nem túl rövid bevezető a LaTeX2e használatába avagy LaTeX2e 78 percben
- Magyar nyelvű műszaki-tudományos tipográfia
- Magyar nyelvű szöveg szedése Magyar \LaTeX -hel
- LaTeX – platformfüggetlen általános célú dokumentum készítő rendszer ([videó1](#), [videó2](#), [videó3](#), [videó4](#))

28.7. \LaTeX fórumok

- [TeX - LaTeX Stack Exchange](#)
- [LaTeX Community](#)

28.8. \LaTeX fontok

- [The \$\text{\LaTeX}\$ Font Catalogue](#)
- [Detexify - LaTeX symbol classifier](#)
- [The Comprehensive LaTeX Symbol List](#)

28.9. 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)
- [LaTeX Complex Table Editor](#) (L^AT_EX táblázat online)
- [Equation Editor](#) (L^AT_EX egyenletszerkesztő online)

Irodalomjegyzék

- [1] LaTeX2e: An unofficial reference manual.
URL: <https://latexref.xyz/dev/latex2e.pdf>
- [2] Wikibooks.org: LaTeX.
URL: <https://upload.wikimedia.org/wikipedia/commons/2/2d/LaTeX.pdf>
- [3] BUJDOSÓ GYÖNGYI, FAZEKAS ATTILA: *TeX kezdőlépések*, Budapest, 1997, Terra Kiadó.
- [4] JOHANNES BRAAMS, DAVID CARLISLE, ALAN JEFFREY, LESLIE LAMPORT, FRANK MITTELBACH, CHRIS ROWLEY, RAINER SCHÖPF: *The L^AT_EX 2 _{ε} Sources*.
URL: <https://mirrors.ctan.org/macros/latex/base/source2e.pdf>
- [5] GYURGYÁK JÁNOS: *Szerzők és szerkesztők kézikönyve*, Osiris Kiadó, Budapest, 2005.
- [6] DONALD ERVIN KNUTH: *The TeXbook*, Reading/Ma. etc., 1984, Addison-Wesley.
URL: <https://mirrors.ctan.org/systems/knuth/dist/tex/texbook.tex>
- [7] L^AT_EX3 PROJECT TEAM: L^AT_EX News.
URL: <https://www.latex-project.org/news/latex2e-news/ltnews.pdf>
- [8] L^AT_EX3 PROJECT TEAM: L^AT_EX3 News.
URL: <https://www.latex-project.org/news/latex3-news/l3news.pdf>
- [9] FRANK MITTELBACH: *How to influence the position of float environments like figure and table in L^AT_EX?* TUGboat volume 35, number 3, pp. 248–254, 2014.
URL: <https://www.latex-project.org/publications/2014-FMi-TUB-tb111mitt-float-placement.pdf>
- [10] TOBIAS OETIKER, HUBERT PARTL, IRENE HYNA, ELISABETH SCHLEGL: *Egy nem túl rövid bevezető a L^AT_EX 2 _{ε} használatába*.
URL: <https://math.bme.hu/latex/dl/latex78.pdf>
- [11] SZABÓ PÉTER: *Magyar nyelvű szöveg szedése MagyarL^AT_EX-hel*.
URL: <https://math.bme.hu/latex/magyarldf-doc.pdf>
- [12] TIBOR TÓMÁCS: *A class for book covers and dust jackets*.
URL: <https://mirrors.ctan.org/macros/latex/contrib/bookcover/bookcover.pdf>
- [13] TIBOR TÓMÁCS: *Draw rulers on the foreground or in the text*.
URL: <https://mirrors.ctan.org/macros/latex/contrib/fgruler/fgruler.pdf>

- [14] TIBOR TÓMÁCS: *Hungarian dummy text (Lórum ipse)*.
URL: <https://mirrors.ctan.org/macros/latex/contrib/hulipsum/hulipsum.pdf>
- [15] TIBOR TÓMÁCS: *Spelling cardinal and ordinal numbers*.
URL: <https://mirrors.ctan.org/macros/latex/contrib/numspell/numspell.pdf>
- [16] TIBOR TÓMÁCS: *The huaz package for automatic Hungarian definite articles*.
URL: <https://mirrors.ctan.org/macros/latex/contrib/huaz/huaz.pdf>
- [17] TIBOR TÓMÁCS: *Thesis class for the Eszterházy Károly Catholic University*.
URL: <https://mirrors.ctan.org/macros/latex/contrib/thesis-ekf/thesis-ekf.pdf>
- [18] WETTL FERENC, MAYER GYULA, SZABÓ PÉTER: *LATEX kézikönyv*, Budapest, 2004, Panem Könyvkiadó.
URL (első két fejezet): https://math.bme.hu/latex/lakk_free.pdf