Často kladené otázky o TEXu a odpovědi na ně ($\mathcal{C}\mathcal{S}$TUG FAQ)

14. listopadu 2004

POZNÁMKA

Části A–U tohoto dokumentu jsou překladem originálu, který napsal Bobby Bodenheimer. Poprvé byl publikován v prosinci 1994 v časopise Baskerville *vydávaném britským UK TUGem, a od té doby je UK TUGem udržován. Část V byla doplněna a obsahuje informace specifické pro $\mathcal{C}\mathcal{S}$TUG.*

Obsah

A. Úvod

První verze tohoto dokumentu vznikla překladem z článku vytvořeného komisí UK TEX Users Group (UKTUG)[1]. Ten vznikl z často kladených otázek (FAQ) na *Usenet*ové skupině `comp.text.tex`, kterýžto dokument udržoval Bobby Bodenheimer (`bobby@hot.caltech.edu`).

Usenet je mechanismus pro výměnu informací mezi lidmi, které sdružují společné zájmy a potřeby, na celosvětové síti Internet. Protože valná většina otázek (a odpovědí) se často pravidelně opakuje a zatěžuje jak linky, tak dlouhodobé čtenáře dané skupiny, někteří lidé se dali do sepisování často kladených otázek a odpovědí na ně.

Protože někteří členové CSTUGu nemají přístup k Internetu, byly Často kladené otázky a odpovědi na ně zařazeny do zvláštního čísla 1996/3 Zpravodaje sdružení (viz Otázku 310).

Často kladené otázky jsou rovněž na CD distribuci TEX Live 3.

Jako východisko byla vzata anglická verze udržovaná UK TUGem a jsme vděčni za svolení k jeho překladu.[2] Překlad první československé verze provedli Leo Hadacz, Michal Kočer, Karol Nemoga, Petr Sojka a Jiří Zelinka. V roce 2001 provedl rozsáhlou aktualizaci Petr Matyáš v rámci bakalářského projektu, kdy do textu doplnil přes 50 nových odpovědí na otázky, a v dalších udělal změny, odstranil slepé odkazy, apod. V roce 2004 byla další rozsáhlá aktualizace. Radovan Panák přidal řadu dalších otázek a aktualizoval starší, čímž rozsah FAQ přibližně zdvojnásobil. Vše dal dohromady, ujednotil, vysázel a udržuje Libor Škarvada (`cstug-faq@cstug.cz`), jemuž posílejte případné náměty na vylepšení či návrhy otázek a odpovědí k vložení do příští verze. Tomáš Hudec a Aleš Vítek vytvořili skripty pro generování WWW-verze dokumentu na `http://www.cstug.cz/csfaq/` a rozhraní pro jeho prohlížení a vyhledávání v něm.

Kde najít soubory zmíněné v tomto článku

Pokud není řečeno jinak, všechny soubory zmíněné v tomto dokumentu jsou dostupné z archívu CTAN a z jeho zrcadel. Blíže o archívech CTAN viz Otázku 51, kde se dozvíte jak získat soubory. Pokud nemáte přístup k Internetu, v Otázce 60 se dozvíte o nabídkách a dostupnosti CD-ROM, které obsahují kopii archívu.

Čtenář by též neměl přehlédnout, že prefix specifikovaných cest se liší pro různá zrcadla CTAN. Pro `ftp://ftp.cstug.cz/` je to `/pub/tex/CTAN` Abychom předešli nedorozuměním, též vynecháváme tečku za větou, která končí URL.

B. Původ

1. Co je to TEX

TEX je sázecí systém napsaný Donaldem E. Knuthem, který v úvodu ke své knize o TEXu (viz Otázku 23) řekl, že TEX je *„určený pro tvorbu hezkých knih — a obzvlášť knih obsahujících spoustu matematiky“*.

Knuth je emeritním profesorem počítačového programování na Standfordské univerzitě v Kalifornii, USA. První verzi TEXu vyvinul v roce 1978, aby se vypořádal s revizí své knihy *The Art of Computer Programming*. Tato idea se osvědčila a Knuth vyprodukoval v roce 1982 druhou verzi, která je základem toho, co používáme dnes.

Knuth vytvořil k napsání TEXu systém „dokumentovaného programování“[3] a dokumentovaný zdrojový kód TEXu poskytl k volnému užívání spolu s programy pro zpracování zdrojového textu (`web`) do něčeho, co lze zpracovat překladačem, a do něčeho, co se dá vytisknout. Činnost TEXu není

[1] Pro rok 1995–96: Peter Abbott, Kaveh Bazargan, David Carlisle, Malcolm Clark, Robin Fairbairns, Carol Hewlett, Alan Jeffrey and Sebastian Rahtz

[2] Na tvorbě dokumentu UK TUGu, ze kterého jsme vycházeli, se dále podíleli: Rosemary Bailey, Barbara Beeton, Karl Berry, Damian Cugley, Michael Downes, Jonathan Fine, John Hobby, Berthold Horn, Werner Icking, Ted Nieland, Pat Rau, Chris Rowley, Piet van Oostrum, Joachim Schrod, Philip Taylor, Ulrik Vieth, Rick Zaccone a Reinhard Zierke.

[3] v originále „literate programming“, což se také někdy překládá jako „kultivované“ nebo „gramotné“ programování

nikdy zahalena tajemstvím. Mimoto systém WEB poskytuje mechanismus, jak implementovat TEX na nových operačních systémech a počítačích. V zájmu zachování důvěry k různým implementacím dodal Knuth testovací soubor, podle kterého lze posuzovat věrnost dané implementace systému TEX (tzv. trip-trap test). TEX a jeho dokumenty jsou tak velmi dobře přenositelné.

TEX je makroprocesor, který nabízí svým uživatelům mocné programovací schopnosti. Z tohoto důvodu je TEX sám o sobě pěkně složitá potvora, se kterou bychom se museli potýkat, a tak Knuth napsal balík maker pro použití s TEXem nazývaný `plain` TEX. `plain` TEX je minimální množina maker, která lze efektivně použít s TEXem, společně s několika demonstrativními verzemi příkazů na vyšší úrovni (které je lepší považovat za vzory, než je používat tak jak jsou). Když člověk řekne, že „programuje v TEXu“, většinou tím myslí, že programuje v `plain` TEXu.

2. Jak mám vyslovovat „TEX“

Písmeno „X“ znamená řecké písmenko chí (χ), a anglickými mluvčími je vyslovováno buď trochu jako „ch“ ve slově „loch“ ([x] v IPA[4]) nebo jako „k“. Rozhodně není vyslovováno jako „ks“. V českém a slovenském jazyce se většinou používá výslovnost [tech].

3. Co to je METAFONT

METAFONT byl napsán Knuthem jako souputník TEXu. Zatímco TEX definuje rozložení znaků na stránce, METAFONT definuje tvary znaků (glyfy) a vztahy mezi nimi. METAFONT přesně vymezuje velikosti glyfů a grafický rastr k jejich reprezentaci. Informaci o velikosti glyfů využívá TEX, zatímco rastr využívají programy, které produkují tisknutelný výstup, což je další zpracování toho, co vyprodukuje TEX.

Jazyk METAFONTu pro definici fontu dovoluje vyjádřit několik věcí: za prvé (ovšem) geometrické tvary glyfů, za druhé vlastnosti tiskového zařízení, pro které je určen výstup, a za třetí, „meta“-informaci, pomocí které lze rozlišit různé velikosti stejného fontu nebo rozdíly mezi dvěma fonty, které patří do stejné (nebo příbuzné) rodiny.

Knuth (a ostatní) navrhli velmi mnoho fontů s použitím METAFONTu, ale návrh fontů pomocí METAFONTu je mezi uživateli rozšířen mnohem méně, než psaní TEXovských maker. Dokonalý uživatel TEXu nicméně potřebuje vědět o METAFONTu a být schopný spouštět METAFONT ke generování vlastních souborů s fonty v různých velikostech a rozlišeních.

4. Čo je to METAPOST

METAPOST je systém, ktorého jazyk je veľmi podobný METAFONTu až na to, že jeho výstup je PostScript a nie bitové mapy. METAPOST je silný jazyk pre kreslenie obrázkov do dokumentov, ktoré budú tlačené na PostScriptovej tlačiarni. Poskytuje prístup ku všetkým možnostiam PostScriptu a umožňuje spojenie textu a obrázkov. (D. Knuth povedal, že nepoužíva nič iné na kreslenie diagramov, keď píše nejaké texty.)

Aj keď PDFLATEX bežne nevie spracovať postscriptovú grafiku, výstup METAPOSTu je dostatočne jednoduchý, takže ho PDFLATEX môže spracovať priamo použitím kódu vypožičaného z CONTEXTu (viď Otázku 105).

Veľká časť zdrojového kódu METAPOSTu bola s povolením D. Knutha kopírovaná z METAFONTu.

Pre METAPOST existuje mailing list, prihlásiť sa môžete cez NTG *mailman* rozhranie na `http://www.ntg.nl/mailman/listinfo/metapost`

5. Ako si môžem byť istý, že je to skutočne TEX?

TEX (tiež METAFONT a METAPOST) sú napísané v programovacom jazyku Web (viď Otázku 74), ktorý je navrhnutý prenositeľným na mnoho systémov. Ako je potom kontrolovaná nová verzia TEXu?

[4]International Phonetic Alphabet, mezinárodní abeceda pro zápis výslovnosti.

Každý rozumný človek implementujúci softvér má samozrejme vlastnú sadu testov na kontrolu svojho softvéru. Tí, čo portujú TEX a príbuzné programy na iné platformy, naozaj takúto kontrolu uskutočňujú.

Knuth poskytuje „test konformity“ pre TEX (`trip`) a METAFONT (`trap`). Sám tieto testy charakterizuje ako „mučivé“. Nie sú stavané na kontrolu bežných vecí, ktoré overia bežné dokumenty a fonty, ale na vyhľadávanie malých odchýlok od hlavnej cesty kódom TEXu. Pre bežného čitateľa sú nezrozumiteľné.

Keď implementácia TEXu prešla testom `trip` alebo implementácia METAFONTu prešla testom `trap`, môže byť rozumným spôsobom distribuovaná ako fungujúca verzia.

6. Co je to LaTeX

LaTeX je balík maker TEXu napsaný Leslie Lamportem, který představuje systém pro zpracování dokumentu. LaTeX dovoluje popsat strukturu dokumentu pomocí značkování tak, aby uživatel nebyl nucen přemýšlet o výsledném vzhledu. S použitím dokumentových tříd a přídavných balíků může být tentýž dokument vysázen v mnoha různých podobách.

Lamport říká, že LaTeX „představuje vyváženost mezi funkčností a snadností použití“. Toto se objevuje jako nepřetržitý konflikt vedoucí k potřebě článků podobných tomuto: LaTeX může splnit většinu uživatelských požadavků, ale vědět jak není jednoduché.

7. Co je to LaTeX 2ε

Poslední Lamportova verze LaTeXu (LaTeX 2.09 z roku 1992) byla o dva roky později nahrazena novou verzí (LaTeX 2ε) napsanou skupinou pro tvorbu tohoto systému (viz Otázku 295). LaTeX 2ε je nyní jediná snadno dostupná verze LaTeXu. Spojuje dohromady několik proudů jeho vývoje. Velikost této verze se proti LaTeXu 2.09 poněkud zvýšila, což je však vzhledem k získané kontinuitě a stabilitě oproti tomu, co tým původně očekával, zanedbatelné. LaTeX 2ε je částečně kompatibilní s některými staršími verzemi – například většina souborů napsaných pro LaTeX 2.09 funguje i v LaTeXu 2ε (i když v poněkud redukovaném provedení). Rozdíly mezi oběma verzemi LaTeXu jsou popsány v sérii „průvodcovských“ souborů, které jsou dostupné v každé distribuci LaTeXu. Nejzajímavější z těchto souborů jsou dostupné na adrese `http://www.tex.ac.uk/tex-archive/macros/latex/doc/html/usrguide`, ty popisují hlavně rozdíly viditelné běžným uživatelům LaTeXu.

8. Jak mám vyslovovat „LaTeX“, „LaTeX 2ε“

Lamport nikdy nedoporučil, jak sa má vyslovovat LaTeX, ale mnoho lidí jej vyslovuje „La TEX“ nebo snad „LejTEX“ (kde TEX je vyslovován jako samotný program, viz Otázku 2).

O písmenku „epsilon“ v LaTeX 2ε se předpokládá, že bude naznačovat malé zlepšení starého LaTeXu 2.09. Nicméně většina lidí vyslovuje toto jméno jako „LaTeX-dvě-é“.

9. Mám používat `plain`TEX nebo LaTeX

Na tuto otázku neexistuje přímá odpověď. Mnoho lidí nedá dopustit na `plain`TEX a s jeho použitím produkuje velmi obdivuhodné dokumenty (Knuth je samozřejmě příkladem). Ale stejně tak mnoho lidí rádo přenechá rozhodování o designu někomu jinému a přistoupí na malou ztrátu flexibility výměnou za ušetření duševního výkonu.

Úvahy kolem tohoto tématu mohou vyvolat vášnivé diskuse, aniž by vnesly do této věci jasno. Nejlepší je zjistit, co používají ti okolo, a jít s davem. Později se člověk vždycky může přiklonit k něčemu jinému. Nemá cenu se tím trápit.

Jestliže připravujete rukopis pro vydavatele nebo časopis, zeptejte se, jaké značkování se požaduje, ještě předtím, než vyvinete svoje vlastní. Mnoho velkých vydavatelů vyvinulo své vlastní LaTeXovské styly pro časopisy a knihy a trvají na tom, aby se autoři jejich značkování pevně drželi.

10. Aký je vzťah medzi LaTeXom a plainTeXom?

LaTeX je program napísaný v programovacom jazyku TeX (v tomto zmysle je každý LaTeXovský dokument tiež program, ktorý beží „v LaTeXu").

`plain` TeX je taktiež program napísaný v jazyku TeX.

Oba existujú, pretože písanie dokumentov v TeXu by znamenalo znovu vynaliezať koleso pre každý dokument. Oba slúžia hlavne na spríjemnenie tvorby dokumentov, pričom LaTeX poskytuje v tomto smere oveľa väčšiu pomoc.

LaTeX má blízko k označeniu nadmnožina `plain` TeXu. Mnoho dokumentov `plain` TeXu bude s drobnými zmenami fungovať v LaTeXu (aj keď niektoré budú vyžadovať podstatne viac zmien).

Interpretácia každého TeXovského/LaTeXovského programu zahŕňa dátové elementy, ktorých má LaTeX viac než `plain` TeX. Výsledkom je, že ich mapovanie z LaTeXu do `plain` TeXu je oveľa menej jasné, než v opačnom smere.

11. Co to je CONTEXT?

CONTEXT je balík maker vytvořený Hansem Hagenem původně sloužící potřebám holandské firmy Pragma. K jeho vytvoření vedly prakticky stejné důvody jako ke vzniku LaTeXu, avšak (protože je mladší) odráží více modernějších trendů značkování, atd. Zvláště CONTEXT umožňuje přizpůsobit značkování mateřskému jazyku (v současné době jsou nabízeny moduly pro holandštinu, angličtinu a němčinu). Do CONTEXTu je integrována i možnost hypertextového značkování podobně jako v PDFTeXu (viz Otázku 299). Základní instalace používá rozšíření PDFTeXu i ε-TeXu (viz Otázku 298) a vylepšuje jejich používání. CONTEXT nemá zatím tak velkou vývojářskou komunitu jako LaTeX, ale zdá se, že jeho aktivní vývojáři mají ohromnou energii. Můžete si ho zkusit zkopírovat z `macros/context`.

12. Co to jsou AMS balíky (AMS-TeX, atd.)

AMS-TeX je TeXovský balík maker původně napsaný Michaelem Spivakem pro Americkou matematickou společnost (AMS) během let 1983 – 1985. Je popsán v knize *The Joy of TeX* od Michaela D. Spivaka (druhé vydání, AMS, 1990, ISBN 0-821-82997-1). Je založen na `plain`TeXu, ale poskytuje mnoho možností pro tvorbu, které vypadají více profesionálně a méně zatěžují autory. Věnuje se pozornost jemnějším detailům nastavování velikosti a umísťování, na které dbají vydavatelé matematické literatury. Je brán ohled na víceřádkové samostatně sázené rovnice, číslování rovnic, tečky, matice, dvojité akcenty, víceúrovňové indexování, konrolu syntaxe (rychlejší zpracování při počátečních průchodech TeXu pro kontrolu chyb), a jiné věci.

Jakmile LaTeX získal na popularitě, autoři chtěli posílat AMS články v LaTeXu, a tak AMS vyvinula AMS-LaTeX. Ten zahrnuje LaTeXovské balíky a třídy a nabízí autorům většinu schopností AMS-TeXu.

13. Co je to LAMS-TeX

LAMS-TeX byl spojením AMS-TeXu a LaTeXu od Michaela Spivaka. Jeho silné stránky jsou:

- balík pro komutativní diagramy, který produkuje velmi uspokojivý výstup;
- samostatný program *dvipaste* pro přípravu složitých tabulek odděleně od hlavního dokumentu (čímž se předejde problémům s překročením kapacity hlavní paměti TeXu);
- rozsáhlá kontrola na uživatelské úrovni týkající se formátování automaticky generovaných čísel.

Nicméně LAMS-TeX se objevil v době, kdy byl vydán AMS-LaTeX, a tak se LAMS-TeX nikdy skutečně neuchytil.

14. Co je to Eplain

Balík maker Eplain rozšiřuje a doplňuje definice v `plain` TeXu. Eplain není zamýšlen k tomu, aby poskytoval „generické typografické nástroje", tak jako to činí LaTeX nebo Texinfo (viz Otázku 16).

Namísto toho poskytuje definice, které mají být užitečné bez ohledu na příkazy vyšší úrovně, které se vlastně používají při přípravě rukopisu.

Např. Eplain nemá příkaz `\section`, který by sázel záhlaví oddílů „vhodným“ způsobem, tak jako `\section` v LaTeXu. Filosofie Eplainu je, že někteří lidé budou vždy potřebovat nebo chtít jít za názor tvůrce makra na to, co je „vhodné“. Taková konzervovaná makra jsou dobrá — dokud je člověk ochoten akceptovat výsledný výstup. Pokud výsledek není vyhovující, nebo se člověk pokouší o jiný formát, narazí na těžkosti.

Na druhé straně, takřka všichni by uvítali takové možnosti, jako je vytváření křížových referencí návěštími tak, aby nebylo třeba dávat skutečná čísla stránek do rukopisu. Karl Berry, autor Eplainu, říká, že neví o žádných obecně dostupných balících maker, které by nevnucovaly jejich typografický styl autorovi a přesto poskytovaly takové možnosti.

15. Co je to Lollipop

Lollipop je balík maker napsaný Victorem Eijkhoutem. Byl použit při přípravě knihy *„TEX by Topic“* (viz Otázku 23). Manuál o něm říká:

> Lollipop je „usnadněný TEX“. Lollipop je balík maker, který funguje jako nástroj pro psaní TEXovských maker. Bylo mým úmyslem udělat psaní maker tak jednoduché, aby implementace zcela nového vzhledu dokumentu v TEXu se stala záležitostí méně než hodiny pro průměrný dokument a aby byla úkolem, který by mohl být vykonán někým, kdo má pouze velmi základní znalosti programování v TEXu.
> Lollipop je pokusem zpřístupnit formátování strukturovaného textu pro prostředí, ve kterých předtím mohly být používány pouze WYSIWYG balíky, protože přizpůsobení vzhledu dokumentu je s nimi o mnoho jednodušší než s tradičními TEXovskými balíky maker.

Manuál pokračuje rozpravou o ambicích „získat část trhu LaTeXu“. Je to velmi důmyslný balík, ale je pozorovatelný určitý náznak přebírání z LaTeXu... Článek o Lollipopu vyšel v TUGboatu 13(3), 1992.

16. Co je to Texinfo

Texinfo je dokumentační systém, který používá jeden zdrojový soubor k vytváření on-line informace i tištěného výstupu. Namísto psaní dvou různých dokumentů, jednoho pro on-line help a druhého pro tištěný manuál, je třeba napsat pouze jeden zdrojový soubor dokumentu. Když je práce revidována, je potřeba revidovat pouze jeden dokument. Je možné číst on-line informaci, známou jako „soubor Info“, pomocí programu Info pro čtení dokumentace. Podle konvence jména zdrojových souborů Texinfo končí s příponou `.texi` nebo `.texinfo`. Je možné psát a formátovat Texinfo soubory do Info souborů v GNU *emacs*u a číst je pomocí zabudovaného Info prohlížeče. Pokud není k dispozici *emacs*, je možné formátovat Texinfo soubory do Info souborů pomocí programu *makeinfo* a číst je pomocí programu *info*.

Množina TEXovských maker pro formátování Texinfo souborů je dostupná v `macros/texinfo`

17. Je-li TEX tak dobrý, jak to, že je volně šiřitelný

Je volně šiřitelný z rozhodnutí jeho autora D. Knutha. Nicméně podle všeho Knuth není zrovna šťastný, že ostatní sklízejí peníze za to, že poskytují služby a produkty založené na TEXu. Některé hodnotné nástroje a balíky týkající se TEXu jsou pod ochranou autorských práv podle tzv. GNU General Public Licence („Copyleft“), ale TEX sám takto chráněn není.

Jsou k dostání komerční verze TEXu. Některé uživatele uklidňuje, pokud mají placenou podporu. Navíc některé komerční implementace mají schopnosti, které volně šiřitelné verze nemají. (Ale platí to i naopak: některé volně šířené implementace nabízejí věci, které žádná komerční neumí.)

Obvykle tento článek nepopisuje komerční verze. Otázka 63 podává přehled hlavních dodavatelů.

18. Jaká je budoucnost TEXu

Knuth prohlásil, že už nebude TEX dále vyvíjet. Chce pouze opravovat všechny chyby, o kterých se dozví (ačkoliv těchto chyb je málo). Toto rozhodnutí udělal krátce poté, co byla vydána verze TEXu 3.0. Při každém dalším vydání, ve kterém jsou opraveny některé chyby, je číslo verze doplněno o jednu další číslici tak, že konverguje k číslu π. V době vzniku tohoto dokumentu byla aktuální verze 3,14159. Knuth chce, aby po jeho smrti byl TEX zmrazen na verzi π Po tomto okamžiku už nesmějí být prováděny žádné další změny v Knuthově zdrojovém kódu. (Podobné pravidlo je aplikováno na METAFONT. Jeho číslo verze konverguje k číslu e.)

Existují projekty (některé z nich jsou dlouhodobé, viz např. Otázky 295 a 296) na vybudování od základu nového balíku maker založeného na TEXu. Pro ještě delší časové období existují různé projekty na vybudování *následovníka* TEXu. Viz Otázky 297 a 298.

19. Prečo nie je TEX wysiwyg systém?

WYSIWYG je marketingový výraz („What you see is what you get“ - „Čo vidíte, to dostanete“) pre určitý druh textových procesorov. WYSIWYG systémý charakterizujú 2 tvrdenia: píšete to, čo chcete tlačit, a to, čo vidíte na obrazovke pri písaní, má veľmi blízko tlačenej podobe textu.

Jednoduchou odpoveďou na otázku je, že TEX vznikol dávno pred týmto výrazom. Prečo sa však neudialo nič, čo by tento „zázračný textový procesor“ prispôsobilo modernému vnímaniu?

Existujú dve odpovede: jednoduché veci *boli* spravené (ale neovládli svet TEXu) a existujú filozofické dôvody, prečo pre TEX nie je WYSIWYG systém vhodný (komplexnosť TEXu takmer znemožňuje získat ekvivalent TEXovského výstupu bez spustenia TEXu).

Systém poskytujúci „WYSIWYG prostredníctvom TEXu“ vzišiel z projektu VorTEX: pár pracovných staníc Sunu pracovalo v tandeme, jedna sa starala o používateľské rozhranie, kým druhá bola vyťažená spracovávaním výslednej sadzby na pozadí. VorTEX bol na svoju dobu obdivuhodný projekt, ale kombinácia pracovných staníc mala oveľa menší výkon než súčasný priemerný osobný počítač a kód projektu sa ukázal byť neprenositeľný (nikdy sa neuskutočnila posledná veľká zmena verzie TEXu na verziu 3). Moderné sytémy s podobným prístupom sú „Lightning Textures“ (rozšírenie pôvodného TEX systému pre systém Macintosh od Blue Sky) a Scientific Word (môže spolupracovať so systémom počítačovej algebry). Oba tieto systémy sú komerčne dostupné (viď Otázku 63).

Táto otázka začala posledné roky priťahovať pozornosť vývojárov TEXu a v súčasnosti prebiehajú práce na viacerých projektoch zaoberajúcich sa problémom TEXovského prostredia pre prípravu dokumentov (viď 302).

TEXovský svet dlho blokoval myšlienku WYSIWYG. Okrem jednoduchej arogancie („my sme dobrí, my to nepotrebujeme“) existuje koncepčný rozdiel medzi modelom sveta „word-processor“ programov a modelom podľa programov LATEX a CONTEXT — nápad „značkovania“. „Čisté“ značkovanie vyjadruje logický model dokumentu, kde každý objekt je značkovaný podľa toho, čo predstavuje, a nie podľa toho, ako má vyzerať. Výzor je odvodený z vlastností daného typu objektu. Pri správnom použití predstavuje značkovanie cennú pomoc pre opätovné použitie dokumentov.

Zavedené WYSIWYG systémy pokladajú interpretovanie takýchto štruktúrovaných značiek za zložité. Značkovanie sa však z dvoch dôvodov *začína* vyskytovať v zozname požiadaviek zo strany komerčného sveta: značkovanie pomáha na dokument aplikovať štýly (komerční používatelia sú posadnutí jednotnosťou štýlov) a čoraz viac prenikajúce používanie úložných formátov odvodených z XML si to vyžaduje.

Rovnaké výzvy musia byť adresované aj schémami pre TEX-ovskú podporu prípravy dokumentov, takže pozorujeme určitý stupeň konfluencie požiadaviek dvoch komunít.

20. Co je to TUG a TUGboat

TUG je zkratka za TEX Users Group (česky skupina uživatelů TEXu). TUGboat je název hlavního časopisu organizace TUG, který osahuje užitečné články o TEXu a METAFONTu. TUG rovněž vydává pro členy noviny TEX and TUG News, každý rok organizuje konferenci, provádí školicí kursy, prodává většinu knih věnovaných TEXu a distribuuje na disketách TEXovský software pro mikropočítače. TUG má Technickou Radu (Technical Council) pro koordinaci činností kolem TEXu (Viz Otázku 22).

Webovská stránka TUGu je cenným zdrojem informací týkajících se TEXu, jako jsou TEXovský software a seznamy dodavatelů TEXu a TEXovských konzultantů. Na stránku si také pomalu nacházejí svou cestu články z *TUGboat*u.

Dotazy mohou být směrovány na:

TEX Users Group
1466 NW Front Avenue, Suite 3141
Portland, OR 97209
USA

Tel: (+1) 503-223-9994
Fax: (+1) 503-223-3960
E-mail: `tug@tug.org`
Web: `http://www.tug.org/`
CTAN: `usergrps/tug`

21. Jsou také národní skupiny uživatelů TEXu

Následující skupiny publikují informace z jejich členské základny elektronicky v archívech CTANu:

DANTE, Deutschsprachige Anwendervereinigung
TEX e.V.
Postfach 10 18 40
D-69008 Heidelberg
Germany

Tel: (+49) 06221 2 97 66
Fax: (+49) 06221 16 79 06
E-mail: `dante@dante.de`
Web: `http://www.dante.de/`
CTAN: `usergrps/dante`

GUTenberg,
BP 10,
93220 Gagny principal,
France

`gut@irisa.fr`
Web: `http://www.gutenberg.eu.org`
CTAN details: `usergrps/gut`

NTG
Postbus 394, 1740AJ Schagen,
The Netherlands

E-mail: `ntg@ntg.nl`
Web: `http://www.ntg.nl`
CTAN details: `usergrps/ntg`

UK TEX Users' Group,
c/o Peter Abbott,
1 Eymore Close,
Selly Oak,
Birmingham B29 4LB
UK

Tel: (+44) 0121 476 2159
E-mail: `UKTuG-Enquiries@tex.ac.uk`

Web: http://www.tex.ac.uk/UKTUG
CTAN details: usergrps/uktug

Seznam všech známých skupin uživatelů je hledejte v dokumentu usergrps/info/usergrps.tex, informace o CSTUGu viz Otázku 304.

22. Technické pracovní skupiny TUGu

TUG (viz Otázku 20) má autonomní Technickou Radu, která dohlíží na několik pracovních skupin spolupracujících na společném zájmu TeXového společenství. Rada má tři členy, kteří mají spojení s vedoucími každé pracovní skupiny. Každá skupina si určuje svoje vlastní pracovní metody i členství a kdokoliv se zájmem spolupracovat může kontaktovat vedoucího. Návrhy na nové skupiny mají být adresovány na tech-council@tug.org. Seznam všech aktuálních skupin naleznete na adrese http://tug.org/committees.html.

Následuje stručný přehled aktivních skupin:

WG-92-00 (IRP-TWG) *Nezávislý výzkumný projekt TWG.*
Pro uznání důležitých projektů nezávislých na TUGu, ale týkajících se celé TeXové obce.
Kontakt: Alan Hoenig (ajhjj@cunyvm.cuny.edu)

WG-92-01 *Rozšířené kódování matematických fontů v TeXu.*
Pro vytváření standardů na kódování matematických fontů, které se používají v TeXovských systémech.
Kontakt: Barbara Beeton (bnb@math.ams.org)

WG-92-03 *Vícejazyčné dokumenty.*
Primárním účelem této pracovní skupiny je získat pro TeXovské systémy konzistentní prostředky pro implementaci, přístup a popis fontů, pravidel pro slitky, dělících vzorů a jiných speciálních požadavků na danou lingvistickou skupinu.
Kontakt: Yannis Haralambous (Yannis@pobox.com)

WG-92-04 *TeX pro postižené.*
Primárním účelem této pracovní skupiny je fórum pro ty, kteří se zajímají o používání a/nebo vylepšení TeXu, aby uspokojili potřeby lidí se zrakovou nebo jinou vadou.
Kontakt: T.V. Raman (raman@adobe.dec.com)

WG-92-05 *Směrnice pro archív TeXu.*
Účelem této technické pracovní skupiny je vyvinout směrnice pro efektivní management a používání hlavních TeXovských archívů a iniciovat komunikaci mezi správci existujících archívů za účelem koordinace a synchronizace.
Kontakt: George Greenwade (bed_gdg@shsu.edu)

WG-94-07 *Struktury adresářů TeXu.*
Primárním účelem této technické pracovní skupiny je najít univerzální strukturu adresářů pro makra, fonty a jiný TeXovský software tak, aby mohla být přijímána doporučení pro všechny dodavatele TeXovského softwaru.
Kontakt: Norm Walsh (norm@ora.com)

WG-94-08 *Otázky implementace a standardizace DVI ovladačů.*
Hlavním cílem by mělo být zkoumání nových požadavků na DVI ovladače vynucených změnami potřeb a technologií a dále přijímání doporučení pro implementaci a standardizaci takových ovladačů, aby se zvýšila jednotnost jejich použití. Práce bude zahrnovat, ale nebude se omezovat, na prověřování použitelnosti, syntaxe a sémantiky příkazů \special{...}.
Kontakt: Michael Sofka (sofkam@rpi.edu)

WG-94-09 *TeX a SGML.*
Hlavním cílem je zkoumat požadavky a těžkosti ve vývoji technologie rozhraní pro TeX a SGML.
Kontakt: Ken Dreyhaupt (kend@springer-ny.com)

WG-94-10 *TeX a lingvistika.*
Hlavním cílem je studovat a diskutovat požadavky pro sázení lingvistiky v TeXu a prostředky identifikace, prověřování, testování a porovnávání maker, fontů, souborů se styly a jiných pomůcek pro sázení lingvistiky.

Kontakt: Christina Thiele (`cthiele@ccs.carleton.ca`)

C. Dokumentace a nápověda

23. Knihy o TeXu a příbuzná literatura

Knuthova kniha je vyčerpávající referenční příručkou pro TeX, ale existují i jiné knihy pokrývající TeX:

The TeXbook od Donalda Knutha (Addison-Wesley, 1984, ISBN 0-201-13447-0, brožované ISBN 0-201-13448-9)

A Beginner's Book of TeX od Raymonda Seroula and Silvia Levyho, (Springer Verlag, 1992, ISBN 0-387-97562-4)

TeX by Example: A Beginner's Guide od Arvinda Bordeho (Academic Press, 1992, ISBN 0-12--117650-9)

Introduction to TeX od Norberta Schwarze (Addison-Wesley, 1989, ISBN 0-201-51141-X)

A PlainTeX Primer od Malcolma Clarka (Oxford University Press, 1993, ISBNs 0-198-53724-7 tvrdá vazba, a 0-198-53784-0 brožované)

A TeX Primer for Scientists od Stanleyho Sawyera a Stevena Krantze (CRC Press, 1994, ISBN 0-849-37159-7)

TeX by Topic od Victora Eijkhouta (Addison-Wesley, 1992, ISBN 0-201-56882-9)

TeX for the Beginner od Wyntera Snowa (Addison-Wesley, 1992, ISBN 0-201-54799-6)

TeX for the Impatient od Paula W. Abrahamse, Karla Berryho a Kathryn A. Hargreavesové (Addison-Wesley, 1990, ISBN 0-201-51375-7)

TeX in Practice od Stephana von Bechtolsheima (Springer Verlag, 1993, 4 svazky, ISBN 3-540--97296-X celek, nebo Vol. 1: 0-387-97595-0, Vol. 2: 0-387-97596-9, Vol. 3: 0-387-97597-7, a Vol. 4: 0-387-97598-5). jednotlivě.

TeX: Starting from $\boxed{1}$[5] od Michaela Dooba (Springer Verlag, 1993, ISBN 3-540-56441-1)

The Joy of TeX od Michaela D. Spivaka (druhá edice, AMS, 1990, ISBN 0-821-82997-1)

The Advanced TeXbook od Davida Salomona (Springer Verlag, 1995, ISBN 0-387-94556-3)

K dispozici je také kolekce Knuthových publikací týkajících se typografie:

Digital Typography od Donalda Knutha (CSLI and Cambridge University Press, 1999, ISBN 1-57586-011-2, paperback ISBN 1-57586-010-4).

a koncem roku 2000 byla vydavatelstvem Addison Wesley publikována „Millennium Boxed Set“ sada 5 dílů Knuthovy knižní série „*Computers and Typesetting*“ (o TeXu a METAFONTu):

Computers & Typesetting, Volumes A–E Boxed Set od Donalda Knutha (Addison-Wesley, 2001, ISBN 0-201-73416-8).

K LaTeXu viz:

LaTeX, a Document Preparation System od Leslieho Lamporta (druhé vydání, Addison Wesley, 1994, ISBN 0-201-15790-X)

A guide to LaTeX od Helmuta Kopky a Patricka W. Dalyho (Addison Wesley, ISBN 0-201-56889-6)

The LaTeX Companion od Michela Goossense, Franka Mittelbacha, a Alexandera Samarina (Addison Wesley, 1993, ISBN 0-201-54199-8)

The LaTeX Graphics Companion: Illustrating documents with TeX and PostScript od Michela Goossense, Sebastiana Rahtze a Franka Mittelbacha (Addison-Wesley, 1997, ISBN 0-201--85469-4)

The LaTeX Web Companion: Integrating TeX, HTML and XML od Michela Goossense a Sebastiana Rahtze (Addison-Wesley, 1999, ISBN 0-201-43311-7)

TeX Unbound: LaTeX and TeX strategies for fonts, graphics, and more od Alana Hoeniga (Oxford University Press, 1998, ISBN 0-19-509685-1 hardback, ISBN 0-19-509686-X paperback)

Math into LaTeX: An Introduction to LaTeX and AMS-LaTeX od George Grätzera (třetí edice Birkhäuser and Springer Verlag, 2000, ISBN 0-8176-4431-9, ISBN 3-7643-4131-9)

[5] To znamená „Starting from Square One“.

K dispozici je errata pro první vydání: `http://www.springer-ny.com/catalog/np/jan99np/0-387-98708-8.html`

First Steps in LaTeX od George Grätzera (Birkhäuser, 1999, ISBN 0-8176-4132-7)

LaTeX Notes: Practical Tips for Preparing Technical Documents od J. Kennetha Shultise (Prentice Hall, 1994, ISBN 0-131-20973-6)

LaTeX Line by Line od Antoniho Dillera (John Wiley & Sons, 1993, ISBN 0-471-93471-2)

LaTeX for Scientists and Engineers od Davida J. Buergera (McGraw-Hill, 1990, ISBN 0-070-08845-4)

LaTeX for Linux: A Vade Mecum od Bernice Sackse Lipkina (Springer-Verlag, 1999, ISBN 0-387-98708-8, dotisk)

Math into TeX: A Simplified Introduction using AMS-LaTeX od George Grätzera (Birkhäuser, 1993, ISBN 0-817-63637-4)

LaTeX pro začátečníky od Jiřího Rybičky (třetí vydání, KONVOJ, 2003, ISBN 80-7302-049-1)

Z tohoto seznamu pouze Lamport a Goosens s Mittelbachem a Samarinem pokrývají LaTeX 2_ε, nicméně jsou zprávy o tom, že se připravují aspoň některá nová vydání.

Seznam pro METAFONT je o dost kratší:

The METAFONTbook od Donalda Knutha (Addison Wesley, 1986, ISBN 0-201-13445-4)

TeX Unbound Alana Hoeniga obsahuje příklady použití METAFONTu.

Kniha pokrývající široké spektrum témat (včetně instalace a udržování) je:

Making TeX Work od Normana Walshe (O'Reilly and Associates, Inc, 1994, ISBN 1-56592-051-1)

Kniha je citelně zastaralá a momentálně je rozebraná. Kopie je však dostupná prosřednictvým `sourceforge` a na CTANu v sekci „tutoriály“ (viz. Otázku 28).

24. Knihy o písme

Nasleduje neúplný zoznam kníh venujúcich sa typografii obecne, Bringhurstova kniha je najčastejšie odporúčaná.

The Elements of Typographic Style od Roberta Bringhursta (Hartley & Marks, 1992, ISBN 0-88179-033-8)

Finer Points in the Spacing & Arrangement of Type od Geoffreyho Dowdinga (Hartley & Marks, 1996, ISBN 0-88179-119-9)

The Thames & Hudson Manual of Typography od Ruariho McLeana (Thames & Hudson, 1980, ISBN 0-500-68022-1)

The Form of the Book od Jana Tschicholda (Lund Humphries, 1991, ISBN 0-85331-623-6)

Type & Layout od Colina Wheildona (Strathmore Press, 1995, ISBN 0-9624891-5-8)

The Design of Books od Adriana Wilsona (Chronicle Books, 1993, ISBN 0-8118-0304-X)

Typografický manuál od Vladimíra Berana a kolektívu (tretie vydanie, Kafka design, 2003)

Existuje mnoho vzorníkov písiem, ale nasledujúce knihy poskytujú zaujímavejší pohľad na písma a ich históriu všeobecne.

Alphabets Old & New od Lewisa F. Daya (Senate, 1995, ISBN 1-85958-160-9)

An Introduction to the History of Printing Types od Geoffreyho Dowdinga (British Library, 1998, UK ISBN 0-7123-4563-9; USA ISBN 1-884718-44-2)

The Alphabet Abecedarium od Richara A. Firmagea (David R. Godine, 1993, ISBN 0-87923-998-0)

The Alphabet and Elements of Lettering od Fredericka Goudyho (Dover, 1963, ISBN 0-486-20792-7)

Anatomy of a Typeface od Alexandra Lawsona (David R. Godine, 1990, ISBN 0-87923-338-8)

A Tally of Types od Stanleyho Morisona (David R. Godine, 1999, ISBN 1-56792-004-7)

Counterpunch od Freda Smeijersa (Hyphen, 1996, ISBN 0-907259-06-5)

Treasury of Alphabets and Lettering od Jana Tschicholda (W. W. Norton, 1992, ISBN 0-393-70197-2)

A Short History of the Printed Word od Warrena Chappella a Roberta Bringhursta (Hartley & Marks, 1999, ISBN 0-88179-154-7)

25. Kde nájdem FAQ?

Článok Bobbyho Bodenheimera, z ktorého vznikla anglická verzia FAQ, bol mesačne uverejňovaný v usenetových skupinách `comp.text.tex`, `news.answers` a `comp.answers`. Jeho posledná (pomerne zastaralá) kópia je na CTANe a naďalej nie je udržovaná v archíve skupiny `news.answers`.

Aktuálna verzia anglického FAQ je prístupná cez World-Wide Web na URL `http://www.tex.ac.uk/faq` a jej zdrojové súbory sú prístupné cez CTAN.

Ďalším skvelým anglickým zdrojom informácií je TEX/LATEX navigator: `http://tex.loria.fr`.

Frankofónna skupina používateľov TEXu Gutenberg a Československé združenie používateľov TEXu CSTUG publikovali vlastné preklady anglickej verzie FAQ (práve čítate preklad CSTUGu) spolu s rozšírením o informácie relevantné pre ich jazyk.

Skvelý dokument je LATEX tips and tricks (`http://www.educat.hu-berlin.de/~voss/LaTeX/`) od Herberta Voßa. Poskytuje vynikajúce rady vo väčšine oblastí (aj keď sa nedá striktne označiť ako FAQ) a je veľmi odporúčaný pre použitie bežnými smrteľníkmi.

Open Directory Project (ODP) udržuje zoznam zdrojov podpory pre TEX/LATEX, vrátane mnohých FAQ (viď `http://dmoz.org/Computers/Software/Typesetting/TeX/`).

Neanglické FAQ:

nemecké Pravidelne publikované v diskusnej skupine `de.comp.tex` a archivované na CTANe (súbor `help/de-tex-faq`). FAQ sa objavuje taktiež na `http://www.dante.de/faq/de-tex-faq/`.

francúzske Pravidelne publikované v skupine `fr.comp.text.tex` a archivované samozrejme na CTANe (`help/LaTeX-FAQ-francaise`).

španielske Viď `http://apolo.us.es/CervanTeX/FAQ/`

české a slovenské Viď `http://www.fi.muni.cz/cstug/csfaq/`

26. Kde najdu pomoc?

Nejprve si přečtěte tyto FAQ. Ambiciózní projekt podobný FAQ, který sbírá veškeré informace o TEXu, existuje na URL `http://ctv.es/USERS/irmina/TeEncontreX.html`. Podobně jako každé FAQ, i tento projekt potřebuje podporu a pomoc všech uživatelů – doposud nabízí nekompletní sadu odpovědí na potenciální otázky. Zdrojové kódy této aplikace (včetně sady HTML souborů) jsou dostupné v `info/spanish/TeEncontreX`. Tutoriál a další on-line dokumentaci (viz Otázku 28) můžete použít, pokud s TEXem začínáte. Pro hlubší porozumnění je lepší získat a přečíst přinejmenším jednu z mnoha dobrých knih s touto tématikou (viz Otázku 23).

Pokud sháníte jednotlivý program nebo balík, podívejte se nejprve ve svém systému, můžete ho tam klidně mít – lepší verze LATEXu totiž nabízejí navíc širokou škálu souvisejících materiálů.

Jestliže máte přístup k Internetu a jednotlivým diskusním skupinám, pak můžete diskuse o LATEXu, METAFONTu a METAPOSTu sledovat v diskusní skupině `comp.text.tex`. Nejlepší je si důkladně pročíst všechna vlákna, než se na něco zeptáte. Reakce v této skupině jsou běžně poměrně rychlé, ale její čtenáři nejsou příliš nadšeni, když se někdo ptá na otázku, která byla zodpovězena před pár dny. Na adrese `http://www.deja.com/usenet` se nachází systém pro archivaci usenetových skupin, kde můžete najít i `comp.text.tex`. Archív je stále příležitostně nápomocný, zvláště při prohlížení novějších vláken.

Archiv usenetových skupin můžete najít na `http://groups.google.com/` (najdete tady i `comp.text.tex`). Archiv Google obsahuje taky informace týkající se dávné minulosti (z dob před vznikem `comp.text.tex`) a často je nenahraditelným pomocníkem při hledání odpovědi, na kterou si už nikto nemůže spomenout. Taktéž umožňuje položiť vlastní otázku, prípadne odpověď.

Pokud váš přístup na Internet nezahrnuje přímý přístup diskusním skupinám, můžete se účastnit diskuse na Deja nebo podobných službách. Lidé, kteří nemají přístup k webovému rozhraní ani k diskusním skupinám, avšak mohou používat elektronickou poštu, si pomáhají pomocí mailing-listů. Mezi takovéto služby patří TEXhax, což je moderovaná elektronická diskuse. K němu se připojíte zasláním zprávy „subscribe texhax“ na adresu `texhax-request@tex.ac.uk`. Jeho odezva není příliš rychlá, ale předložené otázky se většinou dočkají odpovědí. Dalším mailing-listem je `info-tex`, do něhož se přihlásíte zasláním zprávy „`subscribe info-tex <vaše jméno>`“ na `listserv@shsu.edu`. Mnoho mailing-listů pokrývá některé malé části TEXové problematiky. Dobrý zdroj odkazů lze nalézt na `http://www.tug.org`.

Oznámení o TeXu příbuzných instalacích umístěných v archívu CTAN jsou posílány do mailing listu `ctan-ann`. Do něho se přihlásíte posláním zprávy „`subscribe ctan-ann <vaše jméno>`“ na adresu `listserv@urz.Uni-heidelberg.de`. Problémy METAFONTu (a stále více i METAPOSTu) jsou diskutovány na mailing listu `metafont`, do něhož se přihlásíte překvapivě zprávou s testem „`subscribe metafont <vaše jméno>`“ poslanou na adresu `listserv@ens.fr`. Několik dalších mailing listů o TeXu je přístupných přes mail `listserv@urz.uni.heidelberg.de`. Stačí poslat pouze mail obsahující řádek „help“ v těle zprávy na tuto adresu.

Usenetová skupina pro dokumentované programování (viz Otázku 74) `comp.programming.literate` je zrcadlena do elektronické konference `litprog`. Přihlášení se provede zasláním zprávy „`subscribe litprog <vaše jméno>`“ na `listserv@shsu.edu`.

27. Dokumentace k BibTeXu

BibTeX je program původně určený k produkci bibliografií ve spojení s LaTeXem. Je popsán v oddíle 4.3 a příloze B v příručce o LaTeXu od Leslieho Lamporta (viz Otázku 23). Úplnější popis podává dokument *BibTeXing* obsažený v souboru `btxdoc.tex`. *The LaTeX Companion* (viz Otázku 23) rovněž obsahuje informace o BibTeXu a psaní BibTeXových stylů.

Dokument *Designing BibTeX Styles* je v souboru `btxhak.tex` a popisuje postfixový zásobníkový jazyk, který se používá pro psaní BibTeXovských stylů (soubory `.bst`). Soubor `btxbst.doc` je předlohou pro čtyři standardní styly (`plain`, `abbrv`, `alpha`, `unsrt`). Obsahuje též jejich dokumentaci. Úplná kolekce dokumentace k BibTeXu (zahrnující i zmíněné soubory) je v `biblio/bibtex/distribs/doc`

Existuje též unixová manuálová stránka BibTeX v balíku *web2c* (viz Otázku 61). Nepoužívejte kopie zastaralých manuálových stránek, napsaných v roce 1985, dříve než se objevily dokumenty *BibTeXing* a *Designing BibTeX Styles*.

28. Tutoriály k TeXu nebo LaTeXu

Některé dobré tutoriály byly napsány až několik let po vydání TeXu. Dlouhou dobu je přístupný stabilní tutoriál *Gentle Introduction to PlainTeX* Michaela Dooba. Podívejte se na `info/gentle/gentle.pdf`. Zevrubnější je tutoriál *(Not so) Short Introduction to LaTeX 2ε* který je pravidelně aktulizován na podněty čtenářů (`info/lshort/english/lshort.pdf`).

Nejmodernější tutoriál je *Simplified Introduction to LaTeX* od Harveye Greenberga. Ten byl napsán pro přednáškový cyklus a je dostupný (pouze v postscriptu, bohužel) na adrese `info/simplified-latex/simplified-intro.ps`.

Příjemným čtením je také *Beginner's LaTeX* Petera Flynna (také vznikl jako studijní materiál). Jeho kopie existuje na CTANu (`info/beginlatex/beginlatex.a4.pdf`), je možné jej také najít na (`http://www.tex.ac.uk/tex-archive/info/beginlatex/html/`).

TUG India pracuje na sérii LaTeXovských tutoriálů, které je možno jen doporučit. Vybírejte si po kapitolách z `http://www.tug.org.in/tutorials.html`. Zatím se skládá jenom ze dvou částí: Text a Grafika.

Další skvělý dokument, který není ani FAQ, ani tutoriálem, je LaTeX tips and tricks (`http://www.educat.hu-berlin.de/~voss/LaTeX/`) Herberta Voße.

Zajímavý (a velice praktický) je tutoriál *l2tabu* nebo *A list of sins of LaTeX 2ε users* typu „jak ne“ Marka Trettina (přeložený do angličtiny Jürgenem Fennem). Tutoriál je dostupný skrze CTAN (`info/l2tabu/english/l2tabuen.pdf`, nebo jej můžete také získat v podobě zdrojového souboru `info/l2tabu/english/l2tabuen.tex`).

AMS publikuje *Short Math Guide for LaTeX* (dostupné v různých formátech na `http://www.ams.org/tex/short-math-guide.html`).

Herbert Voß pracuje na dalším užitečným dokumentu, jež je částí jeho výše uvedených *tipů a triků* (`http://www.perce.de/LaTeX/math/Mathmode-TeX.pdf`).

Keith Reckdahl přispívá dokumentem *Using Imported Graphics in LaTeX 2ε* sloužícím jako výborný úvod do používání grafiky (ikdyž je mírně zastaralý a zmiňuje jenom cestu *dvips*). Je dostupný na CTANu (`info/epslatex.pdf`, `info/epslatex.ps`), leč bohužel bez zdrojových souborů.

Nedocenitelnou příručkou typu krok za krokem na vytvoření „work flow" prostřednistvým vašeho TEXovského systému, aby se výstup objevil ve správné velikosti a pozici na papíru standardních rozměrů a aby byla kvalita postačující, je *testflow* od Mike Shella. Tutoriál je sestaven z velkého plaintextového dokumentu, dodáván je taky podpůrný LATEXovský soubor spolu se správným výstupem v PostScript i PDF (pro A4 i „letter" formáty papíru). Kompletní sada je dostupna na CTANu (distribuována s makrami autora pro dokumenty pro IEEE publikace).

Pro `plain` TEXovské příkazy existuje hezká rychlá referenční minipříručka od Johna W. Shipmana (`http://www.nmt.edu/tcc/help/pubs/texcrib.pdf`).

Některé sekce univerzit uvolňují svou dokumentaci na webu. Je to většinou překlad existující dokumentace do HTML (např. INFO dokumentace TEXovské/LATEXovské instalace na `http://www.tac.dk/cgi-bin/info2www?(latex)`)

Některé navíc mají nadšené lidi zabývající se dokumentací na podporu TEXu. Kupříkladu Tim Love udržuje stránky Cambridge University Engineering Department na `http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/` a Mimi Burbank stránky School of Computer Science & Information Technology at the University of Florida na `http://www.csit.fsu.edu/~mimi/tex/`.

Mnozí se dlouze přeli o dostupnost TEXovských knížek na webu. Kniha *TEX by Topic* Victore Eijkhouta (publikována vydavatelstvem Addison-Wesley, již delší dobu ale rozebrána) byla tímto způsobem poskytnuta o Vánocích 2001. Aktuálně k dispozici na `http://www.eijkhout.net/tbt/` (není vhodná pro začátečníky, ale je to dobrá příručka; přispěvatelé do knihy jsou vítáni).

Addison-Wesley taktéž uvolnila copyright *TEX for the Impatient* Paula W. Abrahamse, Karla Berryho and Kathryn A. Hargreavesové, další knihy, jejíž nedostupnost byla mnohými oplakána. Autoři knihu znovu vydali pod GNU general documentation licence, je dostupná na CTANu (`info/impatient`).

Kniha Norma Walshe *Making TEX Work* je také zdarma k dispozici na `http://makingtexwork.sourceforge.net/mtw/`; zdrojové soubory webových stránek jsou jako vždy dostupné na CTANu (`info/makingtexwork/mtw-1.0.1-html.tar.gz`).

Svého času byla skvělým zdrojem informací, ale teď je již mírně zastaralá. Každopádně však má ještě své využití a je vítaným přírůstkem do seznemu online zdrojů. Projekt na její aktualizaci je zřejmě na cestě.

29. Dokumentácia k balíkom

Dokument ako tento často odporúčajú na riešenie konkrétnych problémov pre konkrétne balíky. Odpoveď však často neposkytuje presný postup. Ako má ale užívateľ získať presný postup na riešenie problému?

V lepšom prípade je potrebný balík obsiahnutý už vo vašej inštalácii TEXu. V ešte lepšom prípade používate distribúciu poskytujúcu dokumentáciu k balíkom v dobre prezerateľnej forme. V systéme založenom na teTEXu je napríklad veľmi užitočný príkaz *texdoc*:

```
texdoc footmisc
```

je príkaz, ktorý otvorí dokumentáciu k balíku *footmisc* v okne *xdvi*. *texdoc* podľa prípony vyhľadá a spustí program *xdvi*, *ghostscript* alebo PDF prehliadač. Ak dokumentáciu nenájde, môže spustiť internetový prehliadač na prezretie vlastnej kópie katalógu CTANu.

V horšom prípade musíte nájsť dokumentáciu inými prostriedkami. V tomto prípade si precvičíte svoj dôvtip, pretože si dokumentáciu musíte nájsť sami. Nasledujúci text ponúka niekoľko možných spôsobov ako uspieť.

Najobvyklejšia forma dokumentácie k doplnkom LATEXu je `.dtx` súbor, v ktorom je distribuovaný zdrojový kód (viď Otázku 44). Tieto súbory by mali byť spracovateľné samotným LATEXom, avšak na ceste k čitateľnej dokumenácii sa môžu vyskytnúť občasné zádrhele. Častým problémom je potreba samotného balíku na spracovanie jeho dokumentácie (balík musí byť pred spracovaním rozbalený) alebo nemožnosť spracovania `.dtx` súboru LATEXom. V druhom prípade `.ins` súbor zvyčajne vyprodukuje `.drv` (alebo podobne pomenovaný) súbor. Tento súbor následne necháte spracovať LATEXom (občas spomenie autor tento fakt v súbore `README` daného balíku).

Osobitný súbor s dokumentáciou je ďalšou často sa vyskytujúcou formou. Ak je totiž balík rozsiahly (potrebuje teda rozsiahlu dokumentáciu), bola by táto zbytočnou záťažou pre `.dtx` súbor. Príkladom môže byť trieda *memoir*, ktorej dokumentácia (*memman*) je vychvaľovaná ako úvod do zvyklostí sadzby, sada *KOMA-script*, ktorej vývojári vytvárajú dokumentáciu v angličtine aj nemčine alebo balík *fancyhdr*, ktorého dokumentácia je odvodená z tutoriálu v matematickom žurnáli. Aj keď nie je existencia oddelenej dokumentácie spomenutá v `README` súbore, mala by byť ľahko identifikovateľná.

Treťou častou formou je dokumentácia v balíku samotnom. Bežne sa vyskytuje v komentároch na začiatku súboru, aj keď ju aspoň jeden autor umiestňuje za príkaz `\endinput`. Kedže `\endinput` je logickým koncom súboru, je takéto správanie prijateľné (TEX/LATEX sa totiž ďalej nepozerá, takže nás dokumentácia nestojí zvýšenie času potrebného na nahratie balíku).

Ak napriek uvedeným radám nemôžete dokumentáciu nájsť, je možné, že autor sa s dokumentáciou neobťažoval (podľa filozofie „ťažko sa to písalo, malo by sa to ťažko používať“). Väčšina bežných smrteľníkov v takomto prípade hladá pomoc u skúsenejších používateľov, je však možné uspieť čítaním zdrojového kódu (takto to zrejme autor zamýšľal).

30. Jak psát nové LATEXové třídy a balíčky

V příkazech používaných při psaní balíčků není žádná magie, jednoduše můžete vložit LATEXové příkazy `\(re)newcommand` a `\(re)newenvironment` do souboru `package.sty` a máte balík hotový.

Nicméně složitější balíčky vyžadují více důvtipu. Některé detaily o této problematice lze nalézt v materiálu *LATEX 2ε for class and package writers* umístěném na adrese `http://www.ubi.pt/~dfis-wg/sgml/tex/lc4/clsguide.html` (LATEXový zdroj tohoto dokumentu je obsažen v LATEXové distribuci). Mimoto se hodí i dobrá znalost samotného TEXu. S touto znalostí pak můžete používat dokumentované LATEXové zdroje jako refenční materiály (nadšení autoři se samozřejmě se zdroji seznámí). Kompletní sada dokumentovaných zdrojů LATEXu je dostupná v souboru `macros/latex/base/source2e.tex` v LATEXové distribuci.

Vytváření nových tříd není nic jednoduchého; užitečné je nejprve si projít některé již existující třídy (např. některé standardní, třeba `macros/latex/base/classes.dtx`). Třídy, které nejsou součástí distribuce, jsou běžně postaveny na těch, které jsou, a spouští se nahráním standardní knihovny `\LoadClass` – příklad této techniky můžete vidět v `macros/latex/base/ltxguide.cls`.

31. Tutoriály k METAFONTu a METAPOSTu

Kromě Knuthovy knihy je zatím veřejně k dispozici jenom jeden tutoriál pro METAFONT (`http://metafont.tutorial.free.fr/`) od Christophea Grandsirea (ke stažení je kopie v PDF). Existuje však pár článků zabývajících se během programu. Geoffrey Tobin poskytl *METAFONT for Beginners*, který je dostupný v souboru `info/metafont-for-beginners.tex` (viz Otázku 91). Ten popisuje, jak METAFONT funguje a jak se vyhnout častým chybám. Další článek, jehož autorem je tentokrát Peter Wilson, a který se jmenuje *Some Experiences in Running METAFONT and METAPOST*, je dostupný v souboru `info/metafp.ps` (postscriptová verze) nebo `info/metafp.pdf` (PDF verze). Tento článek se, co se týče METAFONTu, se zaměřuje spíše na instalaci a testování nových fontů než na systém jako celek. Mimo jiné popisuje použití METAPOSTových ilustrací v LATEXových a PDFLATEXových souborech s důrazem na používání příslušných fontů pro libovolný text či matematiku.

Hans Hagen (známý z CONTEXTu) poskytuje METAPOSTový tutoriál MetaFun, který se soustředí na použití METAPOSTu v CONTEXTu. Můžete jej nelézt na METAPOSTových stránkách jeho společnosti (`http://www.pragma-ade.com/metapost.htm`).

Jiné METAPOSTové tutoriály jsou anglický tutoriál Andrého Hecka (`http://remote.science.uva.nl/~heck/Courses/mptut.pdf`), a francouzský (je zde uveden jenom proto, že je napsán dostatečně srozumitelně, aby jej pochopil autor původní verze FAQ) Laurenta Chénoa (`http://pauillac.inria.fr/~cheno/metapost/metapost.pdf`).

Obrovská množina vzorových METAPOSTovských souborů od Vincenta Zoonekynda je dostupná na CTANu (`info/metapost/examples`). Zahrnuty jsou *perl*ovské skripty pro konverzi do html, soubory si můžete také prohlídnout na webu (`http://zoonek.free.fr/LaTeX/Metapost/metapost.html`). Ikdyž tyhle soubory nejsou tutoriálem, jsou jistě hodnotným učebným materiálem. Urs Oswald poskytuje PDF soubor (`http://www.ursoswald.ch/metapost/tutorial.pdf`) psaný více formou dokumentu.

32. Kde nájdem symbol pre...

V TEXu existuje veľa symbolov, ktoré môžete použiť, ale mnoho z nich nie je v knihách o TEXu/LATEXu zobrazených (dokonca ani spomenutých). *The Comprehensive LATEX Symbol List* (Vyčerpávajúci zoznam symbolov LATEXu) od Scotta Pakina a kol. ilustruje viac ako 2000 symbolov a popisuje potrebné príkazy a balíky. Dostupný je na CTANe (`info/symbols/comprehensive`).

Iné otázky v tomto FAQ poskytujú pomoc pri špecifických symboloch:

- Fonty pre matematiku (Otázka 138)
- Fonty pre číselné množiny (Otázka 149)
- Sadzba symbolu hlavnej hodnoty integrálu (Otázka 220)

33. Manuál k programu PICTEX

PICTEX je soubor maker pro kreslení diagramů a obrázků. Vytvořil ho Michael Wichura. Tato makra jsou volně dostupná v `graphics/pictex`. Nicméně manuál k PICTEXu není volně šiřitelný. Bohužel, TUG již není oprávněn ho šířit. Je k dostání za 30 $ (35 $ i s disketou) u TEX Users Group (viz Otázku 20). Zisky z prodeje jdou autorovi PICTEXu.

34. Hledání TEXovských/LATEXovských balíků maker

Před tím, než začnete hledat TEXová makra nebo LATEXové třídy nebo soubory, abyste zprovoznili „to a to", zkuste prohledat katalog Grahama Williamse (`help/Catalogue/catalogue.html`). Katalog můžete prohledat přes `help/Catalogue/catalogue.html`help/Catalogue/catalogue.html web, nachází se na adrese `http://www.tex.ac.uk/cgi-bin/texfaq2html?question=50`.

Pokud jste se dozvěděli o souboru, který vypadá zajímavě, zkuste ho vyhledat v archívu CTAN (viz Otázku 35). Balíky zmiňované v The LATEX Companion vyhledáte jednoduše, protože zde je uvedena i cesta.

35. Hledání souborů v archívu CTAN

K hledání softwaru na stránkách CTAN můžete použít anonymní `ftp` příkazem

```
quote site index <term>
```

nebo vyhledávacím skriptem na adrese `http://www.dante.de/cgi-bin/ctan-index`. Pro snadnější použití tohoto `ftp` si stačí zapamatovat, že `<term>` je regulární výraz, nikoliv pevně daný řetězec, a že některé soubory jsou distribuovány v podobě zdrojového kódu s příponou, která se liší od přípony finálního souboru (např. LATEXové balíčky jsou častěji distribuovány v podobě zdrojového kódu s příponou `.dtx` než v hotovém tvaru s příponou `.sty`).

Člověk by měl napsat natolik všeobecný výraz, aby byl vyhledán soubor, který opravdu hledá, ale ne příliš všeobecný, protože `ftp` rozhraní vrací pouze prvních 20 položek. Následující příklad krásně ilustruje tyto případy. K nalezení LATEXového balíku `caption` můžete použít příkaz:

```
quote site index caption.sty
```

ale ten nevyhledá žádaný soubor (který je distribuován jako `caption.dtx`) avšak vrátí nechtěné soubory (jako třeba `hangcaption.sty`). Ačkoliv to tento případ neukazuje, tečka v názvu souboru je brána jako součást regulárního výrazu – to je náhrada za libovolný znak, proto např.

```
quote site index doc.sty
```

vrátí nechtěné soubory typu `language/swedish/slatex/doc2sty/makefile`.

Jistě, pokud víte, že daný balík je uložen v podobě `.dtx`, můžete hledat jeho jméno, avšak nemůžete to použít, pokud nevíte, v jaké podobě je v archívu uložen. Řešením tohoto problému je přidání „`/`" před název balíku a „`\\.`" na jeho konec. To pak vyhledá pouze soubory, které obsahují dané jméno souboru mezi oddělovačem adresářů a příponou. Tyto dva příkazy

```
quote site index /caption\\.
quote site index /doc\\.
```

provedou dostatečně úzké vyhledávání. V případě `docu` je nalezeno několik dalších souborů, ale přesto tento příkaz vrátí na výstup natolik malý seznam, že je jednoduché se v něm orientovat.

Pokud je vyhledávací řetězec příliš obecný a je nalezeno příliš mnoho souborů splňujících danou podmínku, pak je výstupem pouze prvních 20 nalezených položek. Pokud znáte důkladně adresářový strom CTANu, pak můžete výběr dostatečně zúžit. Např. předpokládejme, že chcete najít kopii driveru `dvips` pro MS-DOS. Můžete použít příkaz

```
quote site index dvips
```

výsledkem však bude zkrácený seznam, který nebude obsahovat soubor, který vy hledáte (kdyby tento seznam nebyl zkrácen, pak by obsahoval 412 položek!). Nicméně však příkazem

```
quote site index msdos.*dvips
```

můžeme tento dotaz zúžit na hledání driverů pod MS-DOS. Příkaz vrátí jenom relevantní řádky jako `systems/msdos/dviware/dvips/dvips5528.zip`.

Při vyhledávání mají některé znaky svůj zvláštní význam:

- většina znaků nahrazuje sama sebe, např. „a“ se vyhodnotí jako „a“.
- „.“ nahrazuje libovolný znak
- „[abcD-F]“ nahrazuje jeden z množiny znaků {„a“, „b“, „c“, „D“, „E“, „F“}
- „*“ za výrazem nahrazuje jeho žádný nebo více výskytů, např „a*“ značí „a“ nebo „aaaaa“ a „[A-Za-z]*“ značí libovolné slovo
- „\“ před speciálním znakem ruší jeho funkci, např. „`\.`“ se nahradí „.“
- „`^`“ značí začátek řádku
- „`$`“značí konec řádku.

Z technických důvodů tedy musí být ve výše zmíněném případě příkazu `quote site index` opravdu dvě zpětná lomítka „`\\.`“. Hledání výrazu „caption“ nebo „CAPTION“ dá stejný výsledek.

36. Vyhledávání balíků přes webové rozhraní

Servery CTAN nabízejí dva webové vyhledávače: `http://www.tex.ac.uk/search` a `http://www.ctan.org/search`; s jejich pomocí však můžete vyhledávat pouze soubory, jejichž jména již znáte (podobně jako příkazem `quote site index`, viz Otázku 35), nebo můžete na základě klíčového slova prohledávat aktuální katalog.

Tyto vyhledávací skripty dávají na výstup URL souborů, které splňují námi zadaná kritéria. Daná URL směřují do archívu CTAN nebo do jejich zrcadel, která si vyberete; když poprvé spustíte tento skript, nejprve se vás zeptá na server, v němž má vyhledávat, a detaily uloží na vašem počítači. Vyberte server, který je vám nejbližší, abyste redukovali zátěž sítě.

Katalog, který je používán vyhledávacím skriptem, můžete procházet nezávisle na vyhledávání. Neleznete jej v archivu samotném na `http://www.ctan.org/tex-archive/help/Catalogue/`, `http://www.dante.de/tex-archive/help/Catalogue/` a `http://www.tex.ac.uk/tex-archive/help/Catalogue/`.

37. Hledání balíků LaTeX 2_ε

Uživatel může být zmaten tím, že nenajde na CTANu žádné `.sty` nebo `.cls` soubory příslušející k balíku nebo třídě, která mu byla doporučena. Například existuje mnoho balíků, které jsou uvedeny v knize *The LaTeX Companion* (viz Otázku 23) bez podrobností o jejich umístění na CTANu. Soubor `info/companion.ctan` by mohl pomoci při hledání takových souborů, ale není to univerzální řešení.

Balíky a třídy LaTeX 2_ε se kanonicky distribuují jako `.dtx` soubory (viz Otázku 44). Pokud zadáte příkaz „`site index`“ s parametrem „`package.sty`“, nic nenajdete — musíte prostě hledat „`package`“ (zadaný řetězec se bere jako podřetězec vyhledávaných jmen).

V případě, že to, co hledáte, je opravdu tímto způsobem distribuováno, měli byste si přenést celý adresář obsahující hledaný `.dtx` soubor, čímž získáte všechny doprovodné soubory, především jeden nebo více `.ins` souborů (installation). Zpracováním „vhodného“ `.ins` souboru pomocí LaTeX 2_ε se vygenerují všechny soubory, které potřebujete.

D. TEXové drobnosti

38. Čo je to DVI súbor

DVI súbor (teda súbor, ktorý má príponu `.dvi`) je hlavný výstupný súbor TEXu (použijúc TEX v širšom zmysle, zahŕňajúc v tom aj LATEX a ďalšie verzie a varianty TEXu). DVI znamená device independent, čo je nezávislý od zariadenia, a znamená to, že tento súbor by mal byť vytlačený rovnako, bez ohľadu na typ zariadenia, na ktorom bol vyhotovený. DVI súbor je čítaný ovládačom zariadenia (pozri Otázku 39), ako je tlačiareň (napr. LaserJet), alebo je vstupom pre prehliadač na obrazovke monitora. V DVI súbore je použité TEXovské vnútorné kódovanie. Z TEXovského vstupného súboru by sme mali urobiť vždy ten istý DVI súbor bez ohľadu na implementáciu TEXu, ktorú použijeme.

DVI súbor obsahuje všetky informácie, ktoré sú potrebné pre tlač alebo prezeranie, ale nie konkrétne bitové mapy alebo iné popisy písma. Snáď ešte materiál, ktorý je uvedený v príkaze `\special` (pozri Otázku 43).

Základný odkaz na štruktúru DVI súboru je zdrojový text *dvitype*, ktorý môžete nájsť na CTANe (`systems/knuth/texware/dvitype.web`).

39. Čo je to ovládač

Ovládač (driver) je program, ktorý má za vstup DVI súbor (pozri Otázku 38) a (obvykle) produkuje súbor, ktorý môže byť poslaný do typografického výstupného zariadenia, napríklad tlačiarne.

Ovládač je obvykle špecifický pre danú tlačiareň alebo triedu tlačiarní, aj keď každá PostScriptová tlačiareň by mala vytlačiť výstup PostScriptového ovládača.

Ovládač potrebuje DVI súbor, ako aj informáciu o písme. Písma môžu byť zadávané vo forme bitových máp alebo odkazom na písmo, ktoré je priamo obsiahnuté v tlačiarni. Každý ovládač očakáva informáciu o písme v špecifickej podobe. Viac informácií o formách písem môžete nájsť v Otázkach 40, 41, 47, 45 a 94.

40. Čo sú to PK súbory

PK súbory (od slov packed raster) obsahujú bitmapy písiem. Výstup z METAFONTu (pozri Otázku 91) obsahuje GF súbor (generic font). Tento konvertuje program *gftopk* do PK súboru. Existuje mnoho PK súborov, pretože pre každé písmo potrebujeme jeden. Dokonca pre každú bodovú veľkosť aj zväčšenie (magnification), každý typ každej rodiny písem. Navyše, PK súbor pre jednu tlačiareň nemusí nevyhnutne byť dobrý pre druhú tlačiareň. V takom prípade je treba mať celú množinu PK súborov aj pre túto ďalšiu tlačiareň. Výsledkom sú často komplikované adresárové štruktúry alebo FLI (font library file) súbory pre väčšiu usporiadanosť prístupu.

41. Čo sú to TFM súbory

TFM je skratka pre TEX font metric, čiže TFM súbor obsahuje informácie o rozmeroch charakterov (písmeniek), o ligatúrach (typografické spojenie dvoch alebo viacerých znakov) a o medziznakových medzerách v danom písme. Pre každé písmo je potrebný jeden TFM súbor, dokonca pre každú bodovú veľkosť. Ale jeden TFM súbor je pre všetky zväčšenia (magnifications), preto je vo všeobecnosti TFM súborov menej ako PK súborov. Dôležité je, že TFM súbory používajú programy TEX, LATEX, ale nie sú vo všeobecnosti potrebné napríklad pre ovládače tlačiarní.

42. Virtuální fonty

Virtuální fonty pro TEX jako první implementoval David Fuchs už na začátku jeho historie, ale pro většinu lidí začaly existovat, až když Knuth v roce 1989 změnil formát a napsal několik podpůrných programů (toho času napsal relevatní článek do CTANu - `info/virtual-fonts.knuth`). Použití virtuálních fontů je způsob, jak říci TEXu o něčem složitějším, než je vzájemně jednoznačné mapování znaků. Entity, které se definují ve virtuálním fontu, vypadají jako TEXovské znaky (objevují se s popisem jejich velikosti a souborem metriky fontu), ale `dvi` procesor je může expandovat do něčeho zcela odlišného. Tato schopnost se dá využít pro přeskládání znaků, pro vytvoření fontu poskládaného z glyfů pocházejících z různých zdrojových souborů, nebo obecně k vytváření různě složitých efektů — virtuální font může obsahovat cokoliv, co je přípustné v `dvi` souboru. Ve většině praktických případů se virtuální fonty používají k přeskládání postscriptových fontů (viz Otázku 96) nebo k vytváření „nepravých" matematických fontů.

Je důležité si uvědomit, že samotný TEX virtuální fonty *nevidí.* Pro každý virtuální font, který čte `dvi` ovladač, existuje odpovídající `tfm` soubor, který čte TEX. Virtuální fonty jsou normálně vytvořeny v jednom ASCII souboru (`vpl`, Virtual Property List), který obsahuje oba druhy informace. Program *vptovf* pak vytvoří binární `tfm` a `vf` soubory. Nejběžnějším způsobem (v dnešní době) jak vygenerovat `vpl` soubor, je použít programový balík *fontinst*, který je detailně popsán v Otázce 96. Dalším prostředkem pro ad-hoc vytváření virtuálních fontů je `fonts/utilities/qdtexvpl` od Eberharda Mattese.

43. Příkazy typu `\special`

TEX poskytuje prostředy k vyjádření věcí, které mohou dělat ovladače zařízení, ale o kterých TEX sám neví nic. Například TEX neví nic o tom, jak zařadit postscriptový obrázek do dokumentu, nebo jak nastavit barvu tisknutého textu. Ale některé ovladače zařízení to umějí.

Takové věci jsou umožněny prostřednictvím příkazů `\special`. Jediná věc, kterou s takovým příkazem TEX udělá, je, že expanduje jeho argumenty a pak příkaz přepíše do `dvi` souboru. Ve většině případů jsou k dispozici balíky (často i s ovladačem), které poskytují srozumitelné rozhraní k příkazům `\special`. Například při vkládání obrázku do textu je nutno vynechat patřičnou mezeru. Anebo změna barvy může být náročnější operace vyžadující znalost triků. LATEX 2$_\varepsilon$ má standardní balíky pro grafiku a barvu, s jejichž pomocí jsou vkládání souborů, rotace, změny měřítka a barvy prostřednictvím příkazů `\special` jednoduché.

Argumenty povolené pro příkaz `\special` jsou závislé na ovladači zařízení. Kromě předchozích příkladů existují příkazy `\special` pro ovladače z emTEXu (např. *dvihplj*, *dviscr*, *atd.*, které kreslí přímky s libovolným sklonem, a příkazy pro ovladač *dvitoln03*, které umožňují, aby stránka byla vysázena naležato).

44. Dokumentované LATEXovské zdroje (`.dtx` soubory)

LATEX 2$_\varepsilon$ a spousta podpůrných balíků maker jsou nyní psány metodou dokumentovaného programování (viz Otázku 74), se zdrojovým kódem i dokumentací ve stejném souboru. Autorem tohoto formátu, který je znám jako „doc", je Frank Mittelbach. Dokumentované zdroje mají podle konvence koncovku `.dtx` a dokumentace by z nich měla být před používáním odstraněna. Případně lze soubor `.dtx` zpracovat LATEXem, čímž vznikne pěkně zformátovaná verze dokumentovaného kódu. Většinou je k dispozici instalační dávka (s příponou `.ins`), která potřebuje standardní balík LATEX 2$_\varepsilon$ *docstrip.* Ten odstraňuje všechny poznámky, jejichž obsahem je právě ta dokumentace. V průběhu instalace se (kromě jiných věcí) odstraňuje ze souborů jejich dokumantace, aby se zvýšila rychlost čtení těchto souborů běžícím LATEXem. Do jednoho `.dtx` souboru může být zahrnuto více balíků, s podmíněnými sekcemi a prostředky pro indexy maker atd. Soubory `.dtx` může psát každý. Jejich formát je popsán v *The LATEX Companion* (viz Otázku 23). Zatím neexistují programy podporující psaní `.dtx` souborů.

Poté, co byly z `.dtx` souborů vytvořeny `.sty` nebo `.cls` (a další) soubory, nejsou už soubory `.dtx` LATEXem používány. Nemusí být uchovávány s fungujícím systémem. Nicméně pro mnoho balíků jsou primárním zdrojem dokumentace, takže je někdy dobré si je ponechat.

Zajímavým je také balík *docmfp*, který rozširuje model balíku *doc* do METAFONTu a METAPOSTu (viz Otázku 3 a viz Otázku 4), čímž umožnil distribuci dokumentovaných balíků obsahujících kód pro METAFONT a METAPOST společně s LATEXovským kódem.

45. Čo sú to kódovania?

Začneme definíciou dvoch pojmov: *znak* (*character*) a *glyf* (*glyph*). Znak je abstraktný pojem pre „atóm" nejakého jazyka alebo iného dialógu (môže to teda byť písmeno abecedy pre jazyk založený na abecede, slabika pre jazyk založený na slabikách alebo ideogram pre jazyk ideogramov). Glyf je značka na obrazovke alebo papieri reprezentujúca znak. Aby bolo možné čítanie daného jazyka, musí byť dohodnutý vzťah medzi glyfom a znakom, takže kým presný tvar glyfu môže byť ovplyvnený mnohými faktormi (možnosti média, štýl, ...), podstata konkrétneho znaku musí byť zachovaná.

Vždy, keď má počítač zobrazovať nejaké znaky, niekto musí zaviesť vzťah medzi množinou čísel a znakmi, ktoré reprezentujú. Toto je podstata kódovania (mapovanie množiny čísel na to, čo majú reprezentovať, a opačne).

TEX pracuje s kódovanými znakmi stále. Znaky na vstupe sú v nejakom kódovaní a TEX sám produkuje kódované znaky vo svojom DVI alebo PDF výstupe. Tieto kódovania môžu mať rôzne vlastnosti.

V čase, keď Knuth prvýkrát implementoval TEX, bol jeho vstupný prúd dosť spurný. Knuth pripravoval dokumenty na termináloch, ktoré produkovali divné znaky, čo malo za následok, že TEX obsahoval opatrenia na preklad vstupného kódovania do niečoho poriadneho. Dnes sa o preklad stlačenia klávesu do kódovania zodpovedajúceho používateľovmu jazyku stará operačný systém. Použité kódovanie je často národný alebo medzinárodný štandard (aj keď mnoho operačných systémov používa „kódové stránky" definované MicroSoftom). Tieto štandardy a kódové stránky obsahujú znaky, ktoré nie sú vo vstupnom prúde TEXu povolené. S týmito znakmi sa treba nejako vysporiadať. Znak ako „é" musí byť TEXom interpretovaný tak, aby aspoň pripomínal spôsob, ako je interpretované „`\'e`".

Výstupný prúd TEXu je v inej situácii. Znaky v ňom budú použité na výber glyfov z použitých fontov. Takže kódovanie výstupného prúdu predstavuje kódovanie fontu (aj keď použitý font môže byť virtuálny – viď Otázku 42). Časť výstupného prúdu môže byť priamym prepisom vstupu, ale môže obsahovať aj výsledok použitia príkazov alebo preklad vstupov ako napríklad ligatúry (`fi`⇒„fi").

Kódovania fontov sa stali horúcou témou, keď sa objavilo corkovské kódovanie (viď 47), pre možnosť potlačenia príkazov `\accent` vo výstupnom prúde (a tým zvýšenie kvality rozdeľovania slov, ktoré je prerušené príkazmi `\accent` — viď Otázku 46). Na využitie znakov s diakritikou vo fontoch je potrebné zaistiť, že pri každom vložení sekvencie príkazov „`\'e`" (explicitne alebo implicitne prostredníctvom spomenutého mapovania vstupu) bude použitý znak kódujúci pozíciu glyfu „é".

Takže znak s diakritikou vo vstupnom prúde TEXu je preložený do TEXovských príkazov generujúcich niečo vyzerajúce ako vstupný znak. Táto sekvencia TEXovských príkazov je pri vytváraní výstupu následne preložená späť do jediného glyfu s diakritikou. Práve o tieto veci sa starajú LATEXovské balíky *inputenc* a *fontenc* (ak pracujú v tandeme na (väčšine) znakov vo vstupnom kódovaní ISO Latin-1 alebo ISO Latin-2 a T1 kódovania fontu). Na prvý pohľad vyzerá zvláštne nechať prvý balík niečo spraviť a druhý nechať zmeny vrátiť späť. Nie vždy sa to však deje takto. Väčšina kódovaní fontov nesedí so zodpovedajúcim vstupným kódovaním úplne presne a práve tieto dva balíky poskytujú potrebnú mieru symetrie pre LATEX.

46. Ako funguje v TEXu rozdeľovanie slov?

Každý vie, čo je rozdeľovanie slov. Vidíme to vo väčšine kníh, ktoré čítame, a (ak sme pozorní) často môžeme zbadať smiešne, nesprávne rozdelenia (istý čas boli britské noviny celkom plodným zdrojom).

Štýly rozdeľovania slov sú regionálne závislé (jeden jazyk môže byť rozdielne rozdeľovaný v rôznych krajinách — napríklad britská a americká angličtina). Dôsledkom toho musí sádzací systém, ktorý nie je pri použití jedného regionálneho nastavenia obmedzený na jeden jazyk, byť schopný z času na čas zmeniť parvidlá pre delenie slov.

TEX má celkom dobrý systém delenia slov (pôvodne vytvorený Frankom Liangom). Aj keď sa mu občas podarí minúť zjavné miesta rozdelenia, málokedy vyberie úplne zlé. Algoritmus kontroluje kandidátov na rozdelenie voči množine rozdeľovacích vzorov. Kandidátmi na rozdelenie musia byť postupnosti písmen (alebo iné jednotlivé znaky, ktoré bude TEX považovať za písmená) — veci ako TEXovské primitivum `\accent` zabraňujú rozdelovaniu.

Množiny rozdeľovacích vzorov sú zvyčajne odvodené z analýzy zoznamu správnych rozdelení (odvodzovanie, vykonávané pomocou balíku *patgen*, nie je obvykle participačný šport).

Vzory pre jazyky, s ktorými sa TEXovský systém stretne, môžu byť nahraté pri jeho inštalácii. Pre zmenu množiny jazykov je nutná čiastočná reinštalácia (viď 221).

TEX poskytuje dva príkazy na „používateľskej úrovni" pre kontrolu rozdeľovania slov: `\language` (vyberá štýl rozdeľovania) a `\hyphenation` (dáva rozdeľovaciemu stroju explicitné inštrukcie, ktoré majú vyššiu prioritu ako vzory).

Bežný používateľ LATEXu si nemusí lámať hlavu s `\language`, kedže je veľmi dobre spravovaný balíkom *babel*. Použitie `\hyphenation` je preberané v Otázke 197.

47. Čo sú EC fonty?

Font pozostáva z množstva *glyfov*. Aby sme ich mohli tlačiť, musia byť kódované (viď Otázku 45). Kódovanie sa používa ako index do tabuliek vo fonte. Knuth si z rôznych príčin pre rodinu svojich fontov Computer Modern vybral veľmi zvláštne kódovanie (vybral si rôzne kódovania pre rôzne fonty, takže aplikácia, ktorá ich používa, musí pred výberom konkrétneho glyfu vedieť, ktorý font používa).

S príchodom TEXu verzie 3 sa vytratila väčšina dôvodov ospravedlňujúcich Knuthove kódovanie a na corkovskom stretnutí TUGu bolo definované kódovanie množiny 256 glyfov pre TEXovské texty. Zámer bol, aby kódovanie pokrylo „väčšinu" európskych jazykov používajúcich latinku (v zmysle zahrnutia všetkých potrebných znakov s diakritikou). Knuthove CMR fonty neobsahovali napríklad veci potrebné pre polštinu (medzi nešťastlivcov patrili ale aj obyvatelia Islandu), corkovské kódovanie ich už obsahovalo. Ani corkovské kódovanie však nepokrývalo všetko (smolu mali minimálne Rumuni, Welšania a Laponci; corkovské kódovanie situáciu severných Laponcov zlepšuje). LATEX sa na corkovské kódovanie odkazuje ako na T1 kódovanie a poskytuje prostriedky na použitie takto kódovaných fontov (čím sa vyhýba problémom s diakritikou a rozdeľovaním slov — viď Otázku 233).

Jediné METAFONTovské fonty vyhovujúce corkovskému kódovaniu sú EC fonty. Vyzerajú ako CM, aj keď ich metriky sa od metrík CM fontov líšia. Fonty sú považované za „stabilné" (v rovnakom zmysle ako CM fonty: ich metriky sa zrejme už meniť nebudú). Vážnymi problémami pre bežného používateľa sú ich veľkosti (každý EC font je približne dvojnásobný oproti zodpovedajúcemu CM fontu) a ich počet (je ich oveľa viac než CM fontov). Počet fontov býval demotiváciou pre tvorbu Adobe Type 1 verzií fontov, no mnoho komerčných dodávateľov poskytuje EC alebo EC-ekvivalentné fonty v type 1 alebo TrueType formáte — viď Otázku 63 (voľne dostupné verzie sa tiež dajú získať — viď 301). Kým sa nevyrobia zodpovedajúce fonty pre matematiku, CM fonty musia zostať zachované, pretože niektoré matematické symboly sú vykresľované z fontov v CM kódovaniach.

EC fonty sú distribuované so sadou „Text Companion" (TC) fontov poskytujúcich glyfy bežne používané v texte. TC fonty sú kódované podľa LATEX TS1 kódovania a nie sú považované za stabilné.

Corkovské kódovanie je implementované virtuálnymi fontami systému PSNFSS (viď Otázku 94), pre PostScriptové fonty a tiež balíkmi fontov *txfonts* a *pxfonts* (viď Otázku 99).

CTAN: `fonts/ec`.

48. Co je to TDS?

TDS je zkratka pro TEX Directory Structure, což je standardní způsob organizace TEXových souborů ve vašem sytému. Specifikace TDS je k dispozici v archívu CTAN v adresáři `tds`.

Moderní distribuce se většinou přizpůsobují TDS, které stanuvuje standardní i lokální hierarchii. TDS si vyhrazuje jméno `texmf` pro kořenový adresář hierarchie. Soubory nabízené jako součást hierarchie jsou uspořádány do adresářů podle této standardní hierarchie. Umístění kořenového adresáře je závislé na systému – na UNIXech to však standardně bývá jeden z adresářů `/usr/local/`

texmf, /usr/local/share/texmf nebo /opt/texmf nebo jiná podobná cesta, v každém případě jsou však všechny soubory umístěny v podadresáři texmf. Mohou zde však být mnohonásobné lokální hierarchie, v nichž mohou být uloženy další soubory. V extrémním případě může mít TEXová instalace jednu lokální hierarchii, ale každý uživatel může mít také svoji individuální lokální hierarchii. Jejich umístění není pouze závislé na systému, ale na jednotlivých uživatelích. Avšak přesto platí, že všechny soubory jsou umístěny v lokálním adresáři texmf.

TDS je publikováno jako výstup TUG Technical Working Group (viz Otázku 22). Můžete si projít online verzi na http://tug.org/tds/ standardu a získat kopie v různých formátech (včetně zdrojového kódu) na CTANu.

49. Čo je to Encapsulated PostScript

PostScript sa stal štandardným jazykom výkonných tlačiarní. Kedže PostScript je tiež mocným grafickým programovacím jazykom, je často používaný ako výstup pre kresliace (a iné) balíky.

Keďže je však PostScript silným jazykom, musia byť niektoré pravidlá podvrhnuté, aby mohla byť výstupná kresba vložená v dokumente ako obrázok bez „pretekania" (teda bez zničenia okolitých prvkov dokumentu, prípadne neschopnosti vykreslenia).

Príloha H PostScript Language Reference Manual (referenčný manuál k PostScriptu; druhé a nasledovné vydania) špecifikuje pravidlá pre použitie PostScriptu pre obrázky týmto spôsobom. Dôležité vlastnosti sú:

- sú vyžadované určité „štruktúrované komentáre"; dôležité sú komentáre identifikujúce typ súboru a informáciu o minimálnom boxe uzatvárajúcom obrázok (tzv. *bounding box*);
- niektoré príkazy sú zakázané — napríklad príkaz showpage zapríčiní zmiznutie obrázku vo väčšine prostredí TEXovského výstupu;
- je povolená informácia o náhľade (napríklad pre textové procesory, ktoré nevedia samy správne vykresliť PostScript) — táto informácia môže byť v ľubovolnom systémovo špecifickom formáte (každý prehliadací program sa môže rozhodnúť ignorovať ju).

O postscriptovom obrázku splňujúcom tieto pravidlá sa hovorí, že je vo formáte *Encapsulated PostScript* (zapuzdrený PostScript). TEXovské/LATEXovské balíky na vkladanie PostScriptu sú štruktúrované na použitie zapuzdreného PostScriptu, čo samozrejme vedie k humorným situáciam, pri ktorých nahnevaní používatelia TEXu/LATEXu priam bojujú s výstupom kresliaceho softvéru, ktorého autori tieto pravidlá nepoznajú.

50. Čo je to schéma názvov „Berry"

V starých časoch boli distribúcie TEXu obmedzovaný slabou schopnosťou súborových systémov reprezentovať dlhé názvy. (Súborový systém MS-DOS bol pekným strašiakom, no našťastie všetky aktuálne Microsoft systémy dovoľujú pri špecifikácii názvov súborov väčšiu slobodu. Štandard ISO 9660 pre štruktúru CD-ROMov má bohužiaľ podobné nedostatky, ktoré ale takisto boli modifikované rôznymi rozšíreniami.)

Jednou z oblastí, v ktorých toto pôsobilo problémy, boli názvy súborov Type 1 fontov. Tieto fonty sú distribuované ich dodávateľmi s nezmyselnými krátkymi názvami a existujú prirodzené ambície zmeniť názov na niečo, čo font identifikuje presnejšie. Bohužiaľ, názvy ako „BaskervilleMT" už sú ďaleko za schopnosťami typických slabších súborových systémov, pridajte špecifikáciu tvaru alebo varainty fontu a ťažkosti sa začnú vymykať kontrole.

Takže vznikla schéma názvov Berry.

Základom schémy je kódovanie rozličných častí špecifikácie súboru extrémne stručným spôsobom, aby mohlo byť vyjadrené dostatočné množstvo názvov fontov dokonca aj v chudobnejšom priestore názvov súborov. Kódovanie používa jedno písmeno pre „zlievareň" (foundry) fontu, dve pre názov typu písma, jedno pre duktus atď. Celá schéma je vykreslená v distribúcii balíku *fontname*, ktorá zahŕňa rozsiahlu dokumentáciu a sadu tabuliek fontov, ktorých názvy boli upravené.

fontname: info/fontname

E. Získávání softwaru

51. Co je CTAN

Za účelem snadné archivace a získávání TEXovského softwaru vyvinula pracovní skupina TUGu tzv. Comprehensive TEX Archive Network (CTAN). Každý počítač zapojený v síti CTAN obsahuje stejný materiál a udržuje jeho aktuální verze. Tato data jsou dosti rozsáhlá. Především všechno, co je zmíněno v těchto otázkách, je většinou na CTANu archivováno.

Servery s CTANem jsou např. `ctan.dante.de` (Mainz, Nemecko), `cam.ctan.org` (Cambridge, VB) nebo `tug.ctan.org` (Colchester, Vermont, USA).

Struktura TEXových souborů je na všech těchto místech identická a začíná v `tex-archive`. Soubory, které nejsou volně k dispozici (definice tohoto pojmu je ovlivněna hlavně Debian Free Software Guidelines viz `http://www.debian.org/social_contract#guidelines`), jsou udržovány v separátním stromu `nonfree`. Uživatelé archivu by si měli být před použitím baliku stromu `nonfree` jisti, že jsou k tomu oprávněni.

Na každý uzel CTANu můžete přistupovat přes Web na `http://www.dante.de/tex-archive`, `http://www.tex.ac.uk/tex-archive` a `http://www.ctan.org/tex-archive`. Ne všechny zrcadla CTANu jsou přístupné přes Web. Pro snížení zatížení počítačové sítě používejte prosím vám nejbližší CTAN počítač nebo jeho kopii — „zrcadlo“ (viz Otázku 312). Kompletní aktuální seznam CTAN počítačů a zrcadel obdržíte příkazem `finger ctan@<stroj>`, kde `<stroj>` je adresa jednoho z výše jmenovaných počítačů. Tento seznam je také dostupný jako soubor `CTAN.sites`

Pro hledání softwaru na CTANu lze použít anonymní `ftp` (viz Otázku 34).

E-mailové servery `ftpmail@dante.ctan.org` a `ftpmail@tug.ctan.org` nabízejí rozhraní, pomocí nehož můžete získávat soubory podobně jako přes `ftp`. Chcete-li znát detaily použití tohoto rozhraní, stačí poslat na jednu z výše uvedených adres zprávu obsahující řádek „help“

Konečně, uživatel TEXu, který nemá žádný přístup k síti, si samozřejmě může zakoupit kopii archívu na CD-ROM (viz Otázku 60).

52. Instalace nových balíčků

Chcete-li přidat nový balíček do vašeho LATEXu, musíte ho nejprve najít (viz Otázku 34) a stáhnout (viz Otázku 35). Běžně můžete stáhnout celý distribuovaný adresář; jediný případ, kdy není celý adresář nutný, je, pokud stahujete z LATEXových adresářů na CTANu; tyto adresáře obsahují sadu jednotlivých souborů, které jsou samy o sobě kompletní. Malé balíčky mohou být obvykle samostatné soubory s příponou `.sty` (typickým příkladem je `smallpack.sty`) s návodem k použití obsaženém uvnitř nebo v samostatném manuálu nebo v souboru `README`. Typicky však balíček (`pack`) bývá nabízen jako dvojice souborů `pack.ins` a `pack.dtx`. Jejich kód musí být načten z těchto souborů. Pokud je součástí jejich distribuce též soubor `README`, použijte ho tak, jak název říká! Uživatelský manuál a dokumentovaný zdrojový kód balíčku je obsažen v souboru `.dtx`, zatímco soubor `.ins` obsahuje LATEXové instrukce říkající, který kód má být ze souboru `.dtx` „vytažen“.

Balíček `pack` rozbalíte následovně:

- Spusťte LATEX na `pack.ins`. Tím se vygeneruje jeden nebo více souborů (běžně `pack.sty`, avšak, jedná-li se o zvláštní balíček, mohou zde být i jiné související soubory).
- Spusťte LATEX na soubor `pack.dtx`, čímž získáte uživatelský manuál a v některých případech i komentovanou verzi kódu.
- Spusťte LATEX ještě jednou na soubor `pack.dtx`, čímž můžete vyřešit některé odkazy a vygenerovat tabulku obsahů, pokud byly vyžádány.
- LATEX při této činnosti může zahlásit chybu „No file `pack.ind`“; to je pro index příkazů; pokud chcete tento index vytvořit, zpracujete tento materiál příkazem `makeindex -s gind.ist pack` a spustíte LATEX znovu.
- Vytiskněte a přečtěte si soubor `pack.dvi`.

Občas bývá uživatelský manuál nabízen odděleně v souboru `.dtx`. Téměř posledním krokem, který je nutný udělat, je překopírování souboru tam, kde ho LATEX najde. Kde se nachází toto kouzelné místečko, záleží na LATEXové verzi, kterou máte k dispozici, a na tom, jak je systém nastaven (viz Otázku 53). Nakonec ještě musíte říct LATEXu, že jste mu přidali nový soubor nebo soubory a kde ho (je) má najít. Většina volně šiřitelných LaTeXů má jakousi databázi jmen a umístění souborů, které využívá ke své činnosti, pro rychlejší vyhledávání. V těchto případech pak ona databáze musí být pravidelně aktualizována pomocí skriptů, které bývají pro tento účel distribuovány.

teTeX, fpTeX — spusťte *texhash*

web2c — v běžných distribucích by měl fungovat `texhash`, pokud nefunguje, pak spouštějte `mktexlsr`

MikTeX — V MikTeXu verze nižší než 2.0 proveďte toto: Start —> Programs —> MikTeX —> Maintenance —> Refresh filename database nebo otevřete DOSové okno a spusťte inittexmf --update-fndb V MikTeXu verze 2.0 a vyšší proveďte start —> Programs —> MikTeX 2 —> MikTeX Options a stiskněte tlačítko Update filename database.

PAMATUJTE, že příkaz `\usepackage{pack}` musí být v preambuli každého souboru, který balíček `pack` používá.

53. Kam mám uložit nové soubory?

Kam máte správně uložit soubory, které jste si stáhli, závisí na TeXové distribuci, kterou vlastníte. Nicméně předpokládejme, že vlastníte verzi, která vyhovuje TDS (např. teTeX, fpTeX nebo MikTeX). Zde platí následující pravidla, jimiž se musíte řídit:

(1) Vždy instalujte nové soubory do adresáře `texmf` nebo jeho podadresářů. Kořenový adresář bývá často pojmenován následovně:

```
teTeX{}:    /usr/share/texmf.local/
pfTeX{}:    c:\fptex\texmf.local\
mikTeX{}:   c:\localtexmf\
```

Dále používejme pro tento adresář zkratku `$TEXMF`

(2) Ve vašem lokálním adresáři `texmf` napodobte adresářovou strukturu z vašeho hlavního stromu. Zde jsou příklady toho, kam soubor s danou příponou uložit:

```
.cty .cls nebo .fd      $TEXMF/tex/latex/package/
.dvi .ps nebo .pdf      $TEXMF/doc/latex/package/
.tfm                    $TEXMF/fonts/tfm/supplier/font/
.vf                     $TEXMF/fonts/vf/supplier/font/
.afm                    $TEXMF/fonts/afm/supplier/font/
.pfb                    $TEXMF/fonts/type1/supplier/font/
.ttf                    $TEXMF/fonts/truetype/supplier/font/
```

Kde `<font>`, `<package>` a `<supplier>` závisí na každém souboru.

54. „Dočasné“ inštalácie súborov TeXu alebo LaTeXu

Operačné systémy a aplikácie potrebujú vedieť, kde hľadať súbory. Mnoho súborov, ktoré potrebujú, majú uvedené iba meno — používateľ nevie, *kde* sa nachádzajú, ale vie sa na ne spýtať. Najbežnejším prípadom sú príkazy, ktorých mená píšete do shellu (áno, aj vo Windows). Používajú shell na čítanie toho, čo napíšete. Mnoho príkazov zahŕňa nahranie a vykonanie súboru. Premenná `PATH` povie shellu, kde tieto súbory nájsť.

Moderné distribúcie TeXu obsahujú mnoho zabudovaných ciest na prehľadávanie (cesta=path). Vo väčšine prípadov sú tieto cesty správne, ale niekedy ich potrebujete rozšíriť, aby zachytili aj súbory na neobvyklých miestach (napríklad si môžeme chcieť vyskúšať balík pred jeho inštaláciou — viď Otázku 52). Musíme teda zmeniť relevantnú cestu tak, ako ju TeX vníma. Nechceme ale zahodiť zabudovanú cestu TeXu (TeX by si zrazu nevedel s ničím rady).

Pre *rozšírenie* TeXovskej cesty definujeme premennú prostredia operačného systému vo formáte cesty vynechajúc medzeru, ktorú TeX vyplní hodnotou jeho zabudovanej cesty. Najčastejšie chceme naše rozšírenie umiestniť pred cestu, aby boli nové veci preferované (takže „medzeru na zaplnenie“ vynecháme na konci premennej). Syntax je jednoduchá (aj keď závisí na tom, ktorý shell používate): na systémoch podobných Unixu, používajúcich *bash* shell, môžete postupovať nasledovne:

```
export TEXINPUTS=/tmp:
```

kým na systéme Windows, v okne MS-DOS, to bude:

```
set TEXINPUTS=C:/temp;
```

V oboch prípadoch žiadame TEX, aby nahrával súbory z dočasného (temporary) adresára na koreňovom disku. V prípade Unixu je „prázdne miesto“ vyznačené oddeľovačom cesty ':' na konci riadku, pre Windows je postup rovnaký, ale oddeľovač je ';'.

Všimnite si, že pri oboch druhoch systémov zmena ovplyvní iba inštancie TEXu spustené zo shellu, v ktorom bola premenná nastavená. Ak spustíte TEX z iného okna, použije pôvodnú cestu. Pre zmenu platnú pre všetky okná nastavte premennú prostredia vo svojom prihlasovacom skripte alebo profile na systéme Unix, následne sa odhláste a opäť prihláste. V systéme Windows môžete premennú nastaviť v súbore `autoexec.bat` a reštartovať systém.

Zatiaľ sme uviedli postupy pre TEX súbory makier, avšak postup je aplikovateľný na ľubovolný druh súboru, ktorý je čítaný ľubovolným TEXu príbuzným programom — existuje mnoho takýchto ciest a im príslušných premenných prostredia. V systéme založenom na *web2c* vám v tomto pomôžu bohaté poznámky v súbore `texmf.cnf`.

55. Inštalácia nového fontu

Fonty sú v skutočnosti iba ďalším balíkom, takže by mali byť inštalované rovnakým spôsobom ako balíky. Fonty však zvyknú byť komplikovanejšie než priemerné balíky, dôsledkom čoho je občas ťažké vidieť celkovú štruktúru.

Súbory fontov sa môžu objaviť vo veľkom počte formátov. Každý formát má v TEXovskom systéme odlišnú funkciu a každý je uložený v adresári svojho vlastného podstromu inštalačného stromu TDS. Všetky tieto podstromy majú ako svoj koreň adresár `$TEXMF/fonts`. Nasledujúce odpovede popisujú inštaláciu fontov v konkrétnych formátoch.

56. Inštalácia fontov v podobe zdrojových súborov METAFONTu

Metafontovské fonty sú (v porovnaní s inými druhami fontov) potešujúco jednoduché. V súčasnosti sú väčšinou distribuované iba ako zdrojové súbory METAFONTu, kedže moderné distribúcie TEXu sú schopné vyprodukovať všetko potrebné „za behu“. Ak však distribúcia *neobsahuje* TFM súbory, nainštalujte ich tiež (šetria trochu času a zaberajú málo miesta). Nikdy neverte distribúciam bitmapových súborov PK fontov: neexistuje spôsob, ako z nich zistiť tlačiareň, pre ktorú boli generované, menné schémy pod rozličnými operačnými systémami sú ďalším zdrojom zmätku.

57. Inštalácia fontu postscriptovej tlačiarne

Existuje štandardná sada fontov, ktorá sa objavuje v každej postscriptovej tlačiarni od jeho druhej generácie. Tieto fonty (8 rodín štyroch textových fontov a tri fonty na špeciálne účely) sú samozrejme používané širokým okruhom ľudí, pretože sú jednoducho dostupné. Sada pozostáva z:

- `Times` rodina (4 fonty)
- `Palatino` rodina (4 fonty)
- `New Century Schoolbook` rodina (4 fonty)
- `Bookman` rodina (4 fonty)
- `Helvetica` rodina (4 fonty)
- `Avant Garde` (4 fonty)
- `Courier` rodina (4 fonty)
- `Utopia` rodina (4 fonty)
- Zapf `Chancery` (1 font)
- Zapf `Dingbats` (1 font)
- `Symbol` (1 font)

Všetky tieto fonty sú podporované (pre používateľov LATEXu) sadou metrík `psnfss` a podpornými súbormi zo súboru `lw35nfss.zip` na CTANe. Takmer každý aspoň vzdialene moderný TEXovský systém má nenainštalovanú nejakú verziu `psnfss`. Používatelia by si však mali pamätať, že posledné verzie majú podstatne lepšie pokrytie matematiky v rámci fontov `Times` a `Palatino`, ako aj spoľahlivejšiu sadu metrík.

Archív `lw35nfss.zip` je navrhnutý podľa TDS, takže inštalácia v princípe pozostáva iba z rozbalenia súboru do koreňového adresára `texmf` stromu.

Dokumentácia balíku *psnfss* je poskytovaná v súbore psnfss2e.pdf v distribúcii. Otázka 53 udáva, kam sa majú súbory umiestniť. Neporiadok prinášajú rodiny fontov, ktorých autori navrhujú pravidlá pre automatické generovanie zdrojových súborov METAFONTu pre generovanie fontov konkrétnych veľkostí. Inštalácia musí o pravidlách vedieť, pretože inak nemôže generovať súbory fontov.
psnfss: macros/latex/required/psnfss

58. Jak umístit vlastní příspěvek do archívů

Použijte anonymní ftp na jakýkoliv CTAN počítač (viz Otázku 51) a stáhněte si README.uploads v kořenovém adresáři. Obsahuje instrukce pro umísťování souborů a informování lidí, kteří mají místní archív na starosti.

Můžete jít taky cestou Webu: všechny hlavní servery CTANu poskytují stránku pro upload — máte nasledující možnosti: http://www.ctan.org/upload.html, http://www.dante.de/CTAN/upload.html nebo http://www.tex.ac.uk/upload.html. Stránky vás samy provedou potřebným procesem.

Nemůžete-li použít anonymní ftp, pošlete svůj příspěvek e-mailem na ctan@dante.de. Všem ulehčíte život, jestliže vyberete pro svůj produkt trefné a jednoznačné jméno, takže by bylo dobré předem zkontrolovat, jestli se jméno vašeho .sty souboru již náhodou nepoužívá, což lze prostřednictvím příkazu „site index“ (viz Otázku 51). Mějte na paměti, že veřejně lze vystavovat pouze software, jehož autoři proti tomu nemají námitek.

59. Hledání nových fontů

Podrobný seznam METAFONT fontů posílá do usenetových skupin comp.fonts a comp.text.tex zhruba každých šest týdnů Lee Quin (lee@sq.sq.com). Je též dostupný jako info/metafont-list

Většina nových fontů jsou ve formě zvětšovatelných obrysových fontů nebo jiné (viz Otázku 99) a jsou téměř všechny distribuovány v rámci komerčných podmínek.

60. TEX na CD-ROMoch

Ak nemáte prístup k Internetu, určite vás zaujíma kolekcia TEXu na CD-ROMoch. Dokonca aj tí, čo prístup k Internetu majú, môžu zistiť, že mať veľké množstvo s TEXom spojených súborov poruke je veľmi príjemné.

Na TEX Live CD-ROMe je systém pripravený na okamžité spustenie. Jeho ôsma edícia bola vydaná v roku 2003. CD-ROM bol pôvodne vyvíjaný pod záštitou konzorcia používateľských skupín (hlavne TUG, UK TUG a GUTenberg). Všetci členovia rozličných používateľských skupín dostávajú kópie zadarmo. Niektoré skupiny taktiež predávajú dodatočné kópie: kontaktujte vašu miestnu skupinu alebo TUG (viď 20).

Detaily týkajúce sa TEX Livu sú dostupné z vlastnej stránky na serveri TUG: http://www.tug.org/texlive.html.

Alternatívou k systému pripravenému na okamžité spustenie je „snapshot“ archívu CTANu. Čakalo by sa, že takýto systém sa zložitejšie používa, ale množstvo zdrojov túto nepríjemnosť vyvažuje. Kedysi existovali na tomto poli komerčné ponuky, ale dnes je poskytovaný „snapshot“ (dodávaný raz ročne) jediným zdrojom svojho typu.

F. Systémy TEX

61. TEX/LATEX pro různé počítače

Uvádíme seznam volně šiřitelných nebo shareware balíků. Informace o komerčních balících viz Otázku 63.

Unix Instrukce pro získání balíku unixové distribuce TEXu *web2c* (systems/web2c) přes anonymní ftp jsou v souboru unixtex.ftp (systems/unix/unixtex.ftp). V současnosti již ale rozumný instalátor veme (a případne upraví) jednu z distribucí balíků jako teTEX (`systems/unix/teTeX/current/distrib`), který má obvykle zabudovanou čerstvější verzi *web2c*, jak byla vypustěná „do světa", nebo TEX Live CD-ROM (viz Otázku 60).

Pro skompilování úplné teTEXovské distribuce potřebujete soubor `.tar.gz` pro `teTeX-src`, `teTeX-texmf` a `teTeX-texmfsrc`.

Na CTANu nejsou poskytovány žádné binárky teTEXu, ale kompilace teTEXu je stabilní na veliké množine platforem. Nemáte-li prostřiedky pro kompilaci teTEXu, můžete najít kompilované verze na většině podpůrných stránek v jejich „volné oblasti". Disky TEX-live taky obsahují široký výběr binárních distribucí.

Počas období, když je teTEX vyvíjen, je k dispozici „teTEX-beta" (`systems/unix/teTeX-beta`). Před rozhodnutím se pro β verzi, skontrolujte soubory `ANNOUNCE` v dvou adresářích na CTANu. Je totiž možné, že vám β neposkytne nic nové, co potřebujete.

Uživatelé MacOS X by se měli řídit dole uvedenými informacemi (položka Mac).

Linux Existují přinejmenším dvě poměrně úplné implementace TEXu pro Linux. Distribuce Slackware zahrnuje NTEX (dostupný na CTANu v `systems/unix/ntex`), který obsahuje víc TEXovského materiálu, než by si člověk mohl přát. Nejnovější teTEX (dostupný jako `systems/unix/teTeX/current/distrib`) je založen na prohledávacím mechanismu cest od Karla Berryho a je víc kompaktní než NTEX, ale stále dost podrobný.

Vyvarujte se distribuci teTEXu na CD-ROMech Slackware '96, ta totiž obsahuje verzi CM fontů, která hluboce urazila Donalda Knutha (porušuje podmínky distribuce, které Knuth stanovil). Slackware aktualizuje nyní nabízený teTEX podobně jako většina Linuxových distribucí.

Nejnovější ponukou je volná verze komerčního VTEXu (viz Otázku 63), který se kromě jiného, specializuje na přímou produkci PDF. CTAN: `systems/vtex/linux`, `systems/vtex/common`.

PC; MS-DOS nebo OS/2 Balík emTEX pro PC s operačním systémem MS-DOS nebo OS/2 zahrnuje LATEX, BIBTEX, prohlížeče a ovladače a je k dispozici v `systems/msdos/emtex` jako řada archivů zip. Autorem je Eberhard Mattes. Dokumentace je k dispozici jak v němčině, tak v angličtině. Odpovídající paměťový manažér pro použití emTEXu s procesory 386 (a lepšími) a pod Windows jsou zahrnuty v distribuci. EmTEX bude pracovat taky pod Windows, ale uživatelé Windows by raději měli používat distribuci určenou pro prostředí Windows.

PC; MS-DOS Nejnověji je nabízená implementace Web2C7.0, která používá GNU překladač *djgpp* (ten je dostupný v `systems/msdos/djgpp`).

PC; OS/2 Uživatelé OS/2 můžou taky využívat volně dostupné verze komerčního VTEXu (viz Otázku 63).

PC: Win32 Nejlepším balíkem pod Windows je fpTEX, což je verze teTEXu od Fabricea Popineaua. Jako takový je zejména vhodný pro ty, kteří potřebují pracovat jak pod MS-DOSem, tak i pod Windows a pro administrátory, kteří potřebují udržovat obě verze (fpTEX totiž může vyžívat stejný `texmf` strom jako teTEX). fpTEXový DVI prohlížeč *windvi* je založen na *xdvi*, využívá však výhody prostředí Win32. *windvi* může tisknout buď přímo, nebo se tisk může, podobně jako pod UNIXem, provádět pomocí *dvips*. fpTEX je dostupný na CTANu v `systems/win32/fptex`.

Další TEXovou distribucí pod Windows je MikTEX Christiana Schenka vyvinutý odděleně z teTEXu. Má svůj vlastní prohlížeč – *Yap* – který sám o sobě umožňuje tisk dokumentů, přesto distribuce obsahuje *dvips*. Aktuální verzi si můžete soubor po souboru stáhnout z `systems/win32/miktex/2.0`. HTML soubory v tomto adresáři vám poskytují pokyny, které soubory si máte stáhnout, aby vám TEX fungoval. Předem zabalená verze celého adresáře je v `systems/win32/miktex/2.0.zip`.

Návod na instalaci MikTEXu do českého/slovenského prostředí lze najít na `http://measure.feld.cvut.cz/usr/staff/smid/miktex/` Tento návod se však vztahuje ke starší verzi – 1.20.

TUG (a některé další pracovní skupiny) nabízejí všechen volně dostupný software pro PC. Katalog je volně k dispozici od TUG (viz Otázku 20).

Window NT, jiné platformy Verze MikTEXu pro Power PC je dostupná z adresáře na CTANu (`systems/win32/miktexppc`), `AXP` verzi si můžete stáhnout z `systems/win32/miktex-AXP`. Obě verze již nebyly od verze MikTEXu 2.0 aktualizovány, proto vás nemusejí zcela uspokojit.

Další (volně dostupnou) volbou je balík CygWin (`http://www.cygwin.com`), jež reprezentuje unixové prostředí nad Win32, X-windows server je také k dispozici. Máte-li na svém stroji s Windows taky CygWin, máte taky možnost používat TEX, too (budete potřebovat X-server pro spuštění *xdvi*). Komponenty teTEXu se budou tvářit jako unixové aplikace (ale to jste zřejmě chtěli), ale budou taky o něco pomalejší než nativní Win32 implementace jako MikTEX nebo fpTEX. TeTEX (`systems/unix/teTeX/current/distrib`) je dostupný jako část distribuce CygWinu. Můžete si taky vytvořit vlastní kopii z aktuálních zdrojových kódů.

BaKoMa TEX (`nonfree/systems/win32/bakoma`) od Basile Malysheva je obsáhlá (shareware) distribuce, která je zaměřena na podporu pro Acrobat. Distribuce se dodává s balíkem Type 1 fontů zabalených pro práci s BaKoMa TEXem.

Mac OzTEX je sharewarová verze TEXu pro Macintosh. Obsahuje i `DVI` prohlížeč a ovladač pro PostScript. Měla by běžet na jakémkoliv Macintosh Plus, SE, II, nebo novějším modelu, ale nebude běžet na 128K nebo 512K Macovi. Napsal ji Andrew Trevorrow, a je dostupná na disketách od TUG (viz Otázku 20). Tento software lze také získat v placených CTAN stromech na adrese `nonfree/systems/mac/oztex`.

UK TUG předplácí sharewarový poplatek, takže členové UK TUGu mohou získat tento software bez dalších plateb. Otázky k OzTEXu mohou být směřovány na `oztex@midway.uchicago.edu` Jiný částečně sharewarový program je CMacTEX, který dal dohromady Tom Kiffe. Ten se více podobá unixovské implementaci (např. používá *dvips*). CMacTEX obsahuje také port poslední verze Omegy (viz Otázku 297).

Velice užitečným zdrojem pro uživatele TEXu pro Macintosh může být stránka `http://www.esm.psu.edu/mac-tex`, která mimo jiné obsahuje sekci novinek a nápověd.

OzTEX i CMacTEX jsou navíc k dispozici na `http://www.esm.psu.edu/mac-tex/versionsX.html` (MacOS X), ale uživatelé OS X mají taky možnost buildu teTEXu od Gerbena Wierdu. Ten je přirozeně použitelný z příkazové řádky, jako jakýkoliv jiný unixový systém, může však být použit v stylu Mac jádro systému pod TEXShopem Richarda Kocha (volně dostupný), jež je integrovaným TEXovským editorem a prohlížečem.

Užitečným zdrojem pro uživatele Macu se dá najít na `http://www.esm.psu.edu/mac-tex/`. Obsahuje sekce news (novinky) a help (nápověda) a taky detaily o systémech a nástrojích.

OpenVMS TEX pro OpenVMS můžete najít na CTANu (`systems/OpenVMS/TEX97_CTAN.ZIP`).

Atari TEX je dostupný pro Atari ST v `systems/atari` Nemáte-li přístup k `ftp`, pošlete zprávu obsahující slůvko „help" na `atari@atari.archive.umich.edu`

Amiga Úplné implementace TEXu 3.1 (PasTEX) a METAFONT 2.7 jsou dostupné v `systems/amiga` Můžete si také objednat CD-ROM obsahující tento software a jiný software pro Amigu od Walnut Creek CDROM, telefon (+1) 510-947-5997.

TOPS-20 TEX byl původně napsán na počítače DEC-10 pod systémem WAITS a tak byl jednoduše portován na TOPS-20. Distribuce, která běží pod TOPS-20 je k dispozici anonymním `ftp` z `ftp.math.utah.edu` (128.110.198.34) v `pub/tex/pub/web`

62. Editory a shelly vhodné pro TEX

Pro většinu operačních systémů existují dobrá prostředí a editory pro psaní v TEXu. Některé jsou níže popsány, ale je to jen osobní výběr:

Unix Vyzkoušejte GNU *emacs* a AUCTEX mód (`support/auctex`). Poskytuje položky menu a řídící sekvence pro běžné činnosti, kontroluje syntax, provádí vhodné značkování, dovoluje volat TEX a ovladače z prostředí editoru a všechno možné, na co člověk pomyslí. Složité, ale velmi mocné.

Téměř všichni, kterým se nelíbí *emacs*, preferují *vim* (`http://vim.sourceforge.net`), vysoce konfigurovatelný editor (dostupný taktéž pro systémy Windows a Macintosh). Existuje pro něj velké množství zásuvných modulů, které různými spůsoby pomáhají uživatelům TEXu/LATEXu (zvýrazňování syntaxe, volání TEXovských programů, automatické vkládání a doplňování kódu, procházení LATEXé dokumentace). Skripty `auctex.vim` a `bibtex.vim` jsou doporučovány nejčastěji.

Editor *NEdit* (`http://nedit.org/`) je také volně dostupný a programovatelný (existuje ve verzi pro Unix). Sadu rozšíření podobných AUC-TEXu najdete na CTANu v adresáři `support/NEdit-LaTeX-Extensions`.

LaTeX4Jed poskytuje hodně rozšířenou LATEXovskou podporu pro editor *jed* (`http://www.jedsoft.org/jed/`). *LaTeX4Jed* je podobný AUC-TEXu: menu, skratky, šablony, zvýraznění syntaxe, obrysy dokumentu, integrované ladění, doplňování symbolů, plná integrace s externími programy, … Při jeho tvorbě bylo myšleno na začátečníky i pokročilé.

MS-DOS Zde je několik možností:

- TEXshell (`systems/msdos/texshell`) je jednodušší, snadněji nastavitelné prostředí, ve kterém může být používán editor dle vlastní volby.
- Eddi4TEX (`systems/msdos/e4t`, také shareware) je speciálně napsaný editor pro TEX, který je charakterizován inteligentním obarvováním, kontrolou závorek, kontrolou syntaxe, on-line nápovědou a schopností zavolat TEX přímo z editoru. Je vysoce přizpůsobitelný, obsahuje silný makro jazyk.

Můžete též použít GNU *emacs* a AUCTEX pod MS-DOSem, nebo MS-DOSovou implementaci *jed*u.

OS/2 *Emacs* i *jed* fungují i pod OS/2. Dobrý je editor *EPM* (`http://www.sci.muni.cz/~mikulik/os2/pmCSTeX.html`). Makra jsou též součástí modulu `os2.zip` z distribuce CSTEXu (`http://math.feld.cvut.cz/olsak/cstex`).

Pod OS/2 také funguje Eddi4TEX (`systems/os2/epmtex`), který obsahuje shell specifický pro OS/2.

Windows '9x, NT *Winedt*, sharewarový balík nabízí shell, který umožňuje používání TEXu a jiných podobných programů stejně dobře jako výkonný uživatelsky dobře nastavitelný editor.

4AllTEX CD-ROM (viz Otázku 60) obsahuje další mocný shell pro Windows.

Macintosh Komerční program Textures poskytuje vynikající integrované prostředí Macintoshe s vlastním editorem. Jako editor je však stále lepší shareware *Alpha*, který je dost rozšiřitelný na to, aby se dala provádět téměř jakákoliv činnost s TEXem. Pracuje dobře s OzTEXem.

Vim je k dispozici i pro systémy Macintosh.

Uživatelé počítačů Atari, Amiga a NeXT též mají hezká prostředí. Uživatelé LATEXu, kteří mají rádi příkaz „make" mohou najít pomoc *např.* v `support/latexmk`

Existuje rovněž balík pomocných programů pro manipulaci s databází BIBTEXu.

63. Komerční implementace TEXu

Existuje mnoho komerčních implementací TEXu. První se objevila nedlouho poté, co se objevil TEX sám. Nejdéle přežili (od poloviny nebo začátku 80. let) dodavatelé ArborText (dříve Textset) a Personal TEX.

To, co následuje, je asi neúplný seznam. Zahrnutím dodavatele do tohoto seznamu přirozeně není garantována jeho užitečnost. Pro možnost ověření je poskytnut i zdroj informace.

Obecně jsou komerční implementace „úplné", tzn. s vhodnými prohlížeči a ovladači tiskáren. Většinou též obsahují obsáhlou dokumentaci (tj. nejenom TEXbook!) a některé druhy služeb podpory. V některých případech to znamená bezplatné telefonní číslo (použitelné asi jen v USA nebo Kanadě), ale jinak ostatní mají e-mail a normální telefon a fax.

PC; TrueTEX Běží na všech verzích Windows.

Richard J. Kinch
TrueTeX Software
6994 Pebble Beach Court
Lake Worth FL 33467
USA

Tel: +1 561-966-8400
Email: `kinch@truetex.com`
Web: `http://www.truetex.com/`

Zdroj: pošta od Richarda Kinche, říjen 2001.

Unix; TEX Silicon Graphics Iris/Indigo, Solaris 2.1, IBM RS/6000, DEC/RISC-Ultrix, HP 9000.

„Všechny TeXovské balíky. Připraveno k použití, úplná dokumentace a podpora.“

ArborText Inc
1000 Victors Way
Suite 400
Ann Arbor MI 48108
USA

Tel: (+1) 313-996-3566
Fax: (+1) 313-996-3573

Zdroj: *TUGboat*, **15**(1), 1994

VAX/VMS; Convergent TeX Úplný systém pro počítače VAX/VMS (připravuje se verze pro procesor Alpha). Zahrnuje LaTeX, mezinárodní podporu, METAFONT a Web.

Northlake Software, Inc.
812 SW Washington, Ste 1100
Portland, OR 97201
USA

Tel: (+1) 503-228-3383
Fax: (+1) 503-228-5662
E-mail: `rau@nls.com`

Zdroj: E-mail, odesilatel Pat Rau, listopad 1994

PC; Y&Y TeX “Bitmap free TeX for Windows.”

Y&Y, Inc.
106 Indian Hill, MA 01741-1747
USA

Tel: 800-742-4059 (within North America)
Tel: +1 978-371-3286
Fax: +1 978-371-2004
Email: `sales-help@YandY.com` a `tech-help@YandY.com` Web: `http://www.YandY.com/`

Zdroj: Potvrzeno Y&Y, únor 2003

pcTeX Dlouze vžitý: nyní má implementaci pro Windows.

Personal TeX Inc
12 Madrona Street
Mill Valley, CA 94941
USA

Tel: 800-808-7906 (within the USA)
Fax: +1 415-388-8865
Email: `texsales@pctex.com` a `texsupp@pctex.com`
Web: `http://www.pctex.com/`

Zdroj: Pošta od Personal TeX Inc, září 1997

PC; VTeX DVI, PDF a HTML backendy, Visual Tools a Type 1 fonty.

MicroPress Inc
68-30 Harrow Street
Forest Hills, NY 11375
USA

Tel: (+1) 718-575-1816
Fax: (+1) 718-575-8038
E-mail: `support@micropress-inc.com`
Web: `http://www.micropress-inc.com/`

Zdroj: stránka MicroPress, duben 1996

PC; microTeX MicroTeX a TeX tools.

Micro Programs, Inc.
251 Jackson Ave.
Syosset, NY 11791

USA

Tel: (+1) 516-921-1351
E-mail: sales@microprograms.com

Zdroj: AMS listing, listopad 1994

PC; Scientific Word Scientific Word a Scientific Workplace nabízejí mechanismus pro skoro WYSIWYG vstup LaTeXovských dokumentů. Dodávají se s balíkem TrueTeX od firmy Kinch (viz výše). Objednávky v rámci UK mají být adresovány na Scientific Word Ltd., v dalších zemích přímo na vydavatele: MacKichan Software Inc.

Dr Christopher Mabb
Scientific Word Ltd.
49 Queen Street
Peterhead
Aberdeenshire, AB42 1TU
UK

Tel: 0845 766 0340 (v rámci UK)
Tel: +44 1779 490500
Fax: 01779 490600 (v rámci UK)
Email: christopher@sciword.demon.co.uk
Web: http://www.sciword.demon.co.uk

MacKichan Software Inc.
600 Ericksen Ave. NE, Suite 300
Bainbridge Island WA 98110
USA

Tel: +1 206 780 2799
Fax: +1 206 780 2857
Email: info@mackichan.com
Web: http://www.mackichan.com

Zdroj: Pošta od Christophera Mabbe, květen 1999

Macintosh; Mac-TeX http://www.esm.psu.edu/mac-tex/

Macintosh; Textures „Systém TeX pro zbytek světa“. Též poskytuje implementaci METAFONTu a několik utilit pro manipulaci s fonty.

Blue Sky Research
534 SW Third Avenue
Portland, Oregon 97204
USA

Tel: 800-622-8398 (within the USA)
Tel: (+1) 503-222-9571
Fax: (+1) 503-222-1643
E-mail: sales@bluesky.com
Web: http://www.bluesky.com/

Zdroj: TUGboat, **15**(1), 1994

AmigaTeX Úplná implementace pro počítač Commodore Amiga. Zahrnuje úplnou podporu pro obrazovku i tiskárnu pro PostScript, grafiku a fonty, rastrovou grafiku IFF, automatické generování fontů a všechna standardní makra a utility.

Radical Eye Software
PO Box 2081
Stanford, CA 94309
USA

Zdroj: Dopis, odesilatel Tom Rokicki, listopad 1994

G. DVI ovladače a prohlížeče

64. Konverzní programy z DVI do PostScriptu

Nejlepší volně šiřitelný konverzní program z `DVI` do PostScriptu, který běží pod mnoha operačními systémy, je *dvips* Toma Rokickiho. *dvips* je napsán v C a snadno se přenáší do jiných operačních systémů; je dostupný v `dviware/dvips`. Veškerý vývoj probíha v kontextu knihovny *kpathsea* Karla Berryho, zdrojové kódy jsou dostupné z TEX live.

Verze pro VMS jsou dostupné přes knihovnu DECUS (viz Otázku 51) a také z CTANu: `systems/OpenVMS/TEX97_CTAN.ZIP`.

Přeloženou verzi tohoto ovladače pro systémy MS-DOS lze najít uloženou na CTANu v souboru `systems/msdos/dviware/dvips`

Verze *dvips* od Karla Berryho (nazvaná *dvipsk*) má konfigurační popis a kód pro prohledávání cesty podobné jako jeho další programy (např. *web2c*); je dostupná z `dviware/dvipsk`

Další dobrý přenositelný program je *dvitops* od Jamese Clarka, který je také napsán v C, lze přeložit pro Unix, MS-DOS, VMS a Primos, nepodporuje ale virtuální fonty. Je možné jej získat v `obsolete/dviware/dvi2ps`

Uživatelé Macintoshů mohou použít znamenité ovladače zabudované do OzTEXu nebo Textures, či verzi *dvips* v balíku CMacTEX.

65. DVI ovladače pro HP LaserJet

Balík emTEX (viz Otázku 61) obsahuje ovladač *dvihplj* pro LaserJet.

Verze 2.10 Beebeho ovladačů podporuje LaserJet. Tyto ovladače lze přeložit pod Unixem, VMS, na Atari ST s DEC-20 a jsou dostupné z `dviware/beebe`

Program Karla Berryho *dviljk*, který má stejný kód pro prohledávání cesty jako jeho *dvipsk* (viz Otázku 64), lze získat v `dviware/dviljk`

66. Výstup na „iné“ tlačiarne

V ranom období TEXu existovali masy DVI ovládačov pre ľubovolný (vtedy) predstaviteľný druh tlačiarní. Iniciatíva sa ale z trhu tvorby týchto ovládačov pre formáty závislé na tlačiarni vytratila. Existuje veľa dôvodov, ale hlavným bol nedostatok formátov podporujúcich flexibilitu poskytnutú PostScriptom, mnoho tvorcov ovládačov pre DVI výstup zrejme prešlo ku *ghostscript*u (CTAN: `nonfree/support/ghostscript`).

Odporúčaným postupom je vygenerovanie PostScriptu (viď 64), následné spracovanie PostScriptu *ghostscript*om a vygenerovanie formátu pre vlastnenú tlačiareň. Ak používate nejaký druh systému Unix, je obecne veľmi jednoduché vložiť *ghostscript* do spoolovacieho procesu.

67. DVI prohlížeče

EmTEX pro PC s MS-DOSem nebo OS/2, MikTEX a fpTEX pro PC s Windows a OzTEX pro Macintosh přicházejí s prohlížeči, jež mohou být použity na těchto platformách. Komerční balíky TEXu pro PC (viz Otázku 63) mají dobré Windows a Macintosh prohlížeče.

Pro systémy Unix existuje „kanonický“ prohlížeč *xdvi* (`dviware/xdvi`). *Xdvik* (`dviware/xdvi`) je verze *xdvi* používající knihovny *web2c*. Unixové distribuce TEXu (jako teTEX nebo NTEX) obsahují verzi *xdvik*, která používá stejnou verzi *web2c* jako zbytek distribuce.

Alternativy pro prohlížení jsou:

- konverze do „podobného“ ASCII textu (viz Otázku 77) a použití běžného prohlížeče texte,
- generování PostScriptové verze vašeho dokumentu a jeho prohlížení prohlížečem založeným na *Ghostscript*e (viz Otázku 97) a
- generování PDF výstupu a jeho prohlížení programem *Adobe Reader* nebo nějakou jeho náhradou.

68. Generovanie bitových máp z DVI

Počas poslednej analýzy generuje bitové mapy (bitmapy) každý DVI ovládač alebo prehliadač. Sú to bitmapy pre umiestnenie malých bodov na papier prostredníctvom laserovej alebo atramentovej tlačiarne, prípadne bitmapy na zaplnenie určitej oblasti vašej obrazovky. Je však ťažké extrahovať nejakú zo spomenutých bitmáp inou cestou, než zosnímaním obrazovky (screen capture), pričom rozlíšenie výsledku je väčšinou žalostné.

Na čo môže byť potrebné separovanie bitmáp? Najčastejšie sa požaduje niečo, čo môže byť vložené do HTML vygenerovaného z TeXovského/LaTeXovského zdrojového kódu — nie všetko, čo píšeme v TeXu/LaTeXu, sa dá preložiť do HTML (platí to aspoň pre prenositeľné HTML, ktoré môžete prehliadať vo väčšine prehliadačov). Najčastejším riešením je generovanie bitmapy chýbajúcich častí. Príkladom môže byť matematika (matematické rozšírenia rodiny *ML sú dostupné, ale nie sú často používané) a exotické skripty (tie, o ktorých neviete s istotou povedať, že ich majú aj vaši čitatelia). Ďalším príkladom je generovanie vzorových bitmáp a generovanie pre vloženie do zobrazenia inej aplikácie — vkladanie rovníc do Microsoft PowerPointu, ...

V minulosti bol najbežnejším spôsobom generovania bitmáp vygenerovanie postscriptového súboru z DVI a následne použitie *ghostscript*u na vytvorenie požadovaného formátu bitmapy (možno cestou PNM formátu alebo niečoho podobného). Táto procedúra je pomerne nepríjemná (je veľmi pomalá a vyžaduje dva alebo tri kroky), ale slúžila nám dlhý čas.

Používatelia TeXu/LaTeXu môžu využiť dva bitmapové ovládače. Najdlhšie existujúci, *dvi2bitmap* (`dviware/dvi2bitmap`), generuje formáty XBM a XPM, dávno zavrhovaný formát GIF (ktorý je už zastarávajúci, avšak v lete 2003 bola uvolnená patentová ochrana LZW kompresie, ktorú používa) a tiež moderný (ISO štandardizovaný) PNG formát.

Program *Dvipng* (`dviware/dvipng`) bol navrhnutý pre rýchlosť do prostredia generujúceho množstvo PNG súborov: súbor `README` spomína programy *preview-latex*, *LyX* a pár prostredí orientovaných na web. Za povšimnutie stojí, že *dvipng* vytvára vysoko kvalitný výstup, keď jeho interné operácie sú optimalizované na rýchlosť.

H. Podpůrné balíky pro TeX

69. Fig, TeXu přátelský kreslicí balík

Fig (*XFig*) je nástroj ovládaný pomocí menu, který umožňuje kreslit objekty na obrazovce X Windows pracovní stanice. *transfig* je sada nástrojů převádějících kód, jenž produkuje *fig*, do dalších grafických jazyků včetně PostScriptu, Encapsulated PostScriptu, METAFONT, METAPOST, PDF a LaTeXového prostředí picture. Jsou dostupné v `graphics/xfig` a `graphics/transfig`

Fig ošetřuje Micah Beck (`beck@cs.cornell.edu`) a *transfig* je spravován Brianem Smithem (`bvsmith@bl.gov`). Dalším prostředkem pro konverzi z *fig* je *fig2mf*, který generuje METAFONT kód ze vstupu, jenž dává *fig*. Lze jej získat v `graphics/fig2mf`

70. TeXCAD, kreslicí balík pro LaTeX

TeXCAD je program pro PC, který umožňuje uživateli vytvářet kresby na obrazovce myší nebo kurzorovými klávesami za pomoci obrazovkové nabídky dostupných základních obrázků. Výstupem je kód pro LaTeXovské prostředí picture. Volitelně se dá nastavit, aby šlo vkládat čáry pod libovolnými úhly za použití rodiny ovladačů emTeXu (`\specials` — viz Otázku 43). TeXCAD je součástí distribuce emTeXu.

Verze pro Linux, `nonfree/graphics/xtexcad/xtexcad-2.4.1.tar.gz`, by měla fungovat i na jiných operačních systémech Unix.

71. Korektory překlepů pro práci s TeXem

Pro Unix je pravděpodobně nejlepší program *ispell*. Je dostupný v `support/ispell`. Pozor na verzi s číslem `4.x` — tahle verze reprezentuje odchýlenou verzi, jež postrádá mnoho užitečných rysů série 3.x.

Konkurencí *ispell*u začíná být *aspell*. Byl navržen jako jeho nástupce a dosahuje lepších výsledků na většine metrik, avšak stále zostávají určité otázky ohledně jeho výkonu na zdrojových kódech TEXu/LATEXu. Volba *Aspell*u se objevuje ve většině distribucí *Linux*u, není však dostupný na CTANu — viz `http://aspell.sourceforge.net/` pro podrobnosti.

Pro Windows existuje dobrý korektor překlepů zabudovaný do WinEDT a 4AllTEX (viz Otázku 62). Korektor z 4AllTEX je taky dostupný jako osobitní balík *4spell* na CTANu (`support/4spell`).

Pro MS-DOS existuje několik programů. *amspell* může být vyvolán z editoru (dostupný jako `support/amspell`). *jspell* je rozšířená verze programu *ispell* (lze získat v `support/jspell`).

Pro Macintosh je možností program *Excalibur*. Běží v módu native na obou druzích Macu a je dostupný na CTANu v adresáři `systems/mac/support/excalibur/Excalibur-4.0.2.sit.hqx` (v tomtéž adresáři jsou další slovníky).

Korektor překlepů pro VMS lze najít v `support/vmspell`

Korektor pro češtinu je pro členy CSTUGu součástí distribuce CSTEXu. České korektory bývají také standardní složkou textových editorů komerčně distribuovaných v České republice.

72. Balík VORTEX

VORTEX (dostupný v `obsolete/support/vortex`) je balík programů napsaný na Kalifornské Univerzitě v Berkley a byl popsán Michalem A. Harrisonem v článku „*News from the VORTEX project*“ v TUGboat 10(1), s. 11–14, 1989. Zahrnuje několik pěkných prohlížečů a několik *emacs* módů pro TEX a BIBTEX. Distribuce VORTEXu není spravována a v současnosti je zastaralá (nebyl udělán upgrade pro TEX, verzi 3).

73. Kolik slov jsme napsali?

Občas se stane, že člověk musí vytvořit dokument (např. referát nebo disertační práce) určité délky. Rozumní lidé většinou určují maximální počet stran dokumentu, avšak existují lidé, kteří vám dokument omezí daným počtem slov.

Jednoduché řešení tohoto požadavku spočívá v určení počtu slov na plné stránce a určení počtu celých stránek dokumentu a ve vzájemném vynásobení těchto čísel. Nejjednodušší metoda však spočívá v odstranění (La)TEXových značek a použití nástroje k určení tohoto počtu. Na UNIXu můžete použít *detex* (`support/detex`) a vestavěný program `wc`:

```
detex <filename> | wc -w
```

Ve Windowsech k tomu
slouží *Winedt* (viz Otázku 62). Odstranění (La)TEXových značek však není zcela spolehlivé, ale samo značkování může přispívat k sazbě slov, a to může být problém (např. značka „\TeX{}“ vysází slovo TEX). Balíček na počítaní slov (`macros/latex/contrib/wordcount`) obsahuje Bourne Shellovský (to jest typický UNIXový) skript, který po spuštění na TEXový či LATEXový soubor zobrazí počet slov v logovém souboru. Toto je relativně nejspolehlivější automatická metoda pro určení počtu slov dokumentu.

I. Dokumentované programování

74. Co je dokumentované programování

Dokumentované programování je kombinace dokumentace a zdrojového textu do společné formy vhodné ke čtení lidmi. Stručně řečeno, dokumentované programy sdružují zdrojový text a dokumentaci do jednoho souboru. Nástroje dokumentovaného programování pak zpracováním tohoto souboru vytvoří buď čitelnou dokumentaci nebo kompilovatelný kód. Styl dokumentovaného programování WEB vytvořil D. E. Knuth při vývoji svého sázecího systému TEX.

Diskuse o dokumentovaném programování je v usenetové skupině `comp.programming.literate`, která je přístupná i jako elektronická konference na adrese `litprog@shsu.edu`. (viz Otázku 26).

FAQy o dokumentovaném programování jsou uloženy v `help/LitProg-FAQ`. Dalším skvělým zdrojem informací je URL `http://www.literateprogramming.com/`.

75. WEB pro různé jazyky

TEX je napsán v programovacím jazyku WEB. Jedná se o prostředek, který implementuje koncept „dokumentovaného programování“. Knuthova původní implementace bude v každé solidní distribuci TEXu, zdrojové kódy nástrojů (*tangle* a *weave*) společně s manuálem popisujícím samotnou techniku programování jsou k mání na CTANu.

CWEB, WEB pro programy v C, který vytvořil Silvio Levy, je dostupný jako `web/c_cpp/cweb`

Spidery WEB podporuje mnoho jazyků včetně Ada, awk a C. Byl napsán Normanem Ramseyem a ačkoli není volně šiřitelný, lze jej používat bezplatně. Dá se získat v `web/spiderweb`. V současnosti je už nahrazován programem *noweb* (taktéž od Normana Ramsayho). Autor se poučil z věcí, se kterými se setkal při implementaci spidery WEBu, a udělal z *noweb*e jednodušší, ale stejně silný nástroj (CTAN: `web/noweb`).

FWEB je verze pro Fortran, Ratfor a C napsaná Johnem Krommesem. Je dostupná v `web/fweb`

SchemeWEB je Unixový filtr, který překládá SchemeWEB do zdrojového kódu LATEXu nebo Scheme. Napsal ho John Ramsdell a je dostupný v `web/schemeweb`

APLWEB je verze WEBu pro APL a dá se získat v `web/apl/aplweb`

FunnelWeb je verze WEBu, která je jazykově nezávislá. Je dostupná v `web/funnelweb`

Další jazykově nezávislé verze WEBu jsou *nuweb* (napsaná v ANSI C), dostupná v `web/nuweb`, a *noweb*, dostupná na `web/noweb`

J. Formátové konverze

76. Konverze mezi TEXem/LATEXem a ostatními

troff *troff-to-latex* (k dispozici jako `support/troff-to-latex`), jenž napsal Kamal Al-Yahya na Standfordské universitě (Kalifornie, USA), napomáhá při překladu *troff* dokumentu do formátu LATEXu. Rozlišuje většinu `-ms` a `-man` maker, k tomu většinu *eqn* a některé *tbl* příkazy preprocesoru. Složitější věci je potřeba dodělat ručně. Jsou podporovány dva styly souborů. Existuje také manuálová stránka (která se velmi dobře konvertuje do LATEXu). Program má copyright, ale je zdarma. Rozšířená verze tohoto programu, *tr2latex*, je dostupná v `support/tr2latex`

Distribuce DECUS TEX (viz Otázku 51) také obsahuje program konvertující *troff* do TEXu.

Scribe Mark James (`jamesm@dialogic.com`) vlastní kopii programu *scribe2latex*, kterou nemohl vyzkoušet, ale kterou přenechá každému, kdo projeví zájem. Program napsal Van Jacobson z Lawrence Berkeley Laboratory.

WordPerfect Program *wp2latex* byl nedávno značně rozšířen a je nyní díky svému správci (Jaroslav Fojtik) dostupný pro MSDOS nebo systémy Unix.

PC-Write `pcwritex.arc` je k dispozici v `support/pcwritex`

Jedná se o ovladač tisku pro PC-Write, který „tiskne“ dokument formátu PC-Write V2.71 do souboru kompatibilního s TEXem. Napsal ho Peter Flynn z University College, Cork, Irská republika.

runoff Konverzní program Petera Vanroose (jeho adresa je `vanroose@esat.kuleuven.ac.be`) je napsán ve VMS Pascalu. Zdrojové texty a běhuschopný kód pro VAX jsou dostupné v `support/rnototex`

refer/tib Existuje několik programů pro konverzi bibliografických dat mezi BIBTEXem a formátem *refer/tib*. Lze je nalézt v adresáři `biblio/bibtex/utils/refer-tools`

Ačkoli to jeho název nenapovídá, adresář také obsahuje dávku pro převod BIBTEXu do *refer*. Kolekce není udržována.

RTF Program pro konverzi z Microsoft Rich Text Format do TEXu je k mání na `support/rtf2tex`. Napsal a spravuje jej Robert Lupton (`rhl@astro.princeton.edu`). Existuje také konvertor do LATEXu od Erwina Wechtla v `support/rtf2latex`. Nejnovějším konvertorem (od Ujwale

Sathyama a Scotta Prahla) je *rtf2latex2e*. Tento systém dosahuje celkem dobré kvality a je stále vylepšován.

Do RTF můžete dokument přeložit pomocí *TeX2RTF*, který umí vytvořit prosté RTF i Windows Help RTF (stejně dobře jako HTML, viz Otázku 79). Existují verze programu *TeX2RTF* pro různé UNIXové platformy i pro Windows verze 3.11 a nižší. Program je dostupný z `support/tex2rtf`.

Microsoft Word Často používaný způsob je konvertovat dokument do formátu RTF a použít některý z RTF konvertorů zmiňovaných výše. Program pro přímý převod z MS-Wordu do LaTeXu je *wd2latex* pro MS-DOS (v `dviware/wd2latex`). Pavel Nygrýn (`nygryn@fi.muni.cz`) napsal program zohledňující české a slovenské znaky: `http://www.fi.muni.cz/~nygryn/w2t/` Poslední verze programu *mswordview* umí taktéž export do LaTeXu, a to z velmi široké škály verzí Wordu.

Word2TeX a *TeX2Word* jsou sharewarové překladače od Chikrii Softlab (`http://www.chikrii.com/`). Správy od používatelů jsou velice příznivé.

FAQ Wilfrieda Henningse (`http://www.tug.org/utilities/texconv/index.html`) zabívající se specielně konverzemi mezi formáty na bázy TeXu a formáty textových procesorů poskytuje detailní informace jakožto i tabulky umožňující rychlé srovnání vlastností.

Excel *Excel2Latex* konvertuje soubor *Excel*u do LaTeXovského prostředí `tabular`. Dodává se jako `.xls` soubor definující makra *Excel*u pro vytvoření výstupu v novém formátu.

Skupina na Ohio State University (USA) pracuje na společném dokumentovém formátu založeném na SGML s cílem, aby se libovolné formáty daly vzájemně převádět. „Vstupní filtry“ poskytuje *FrameMaker* s cílem překladu z cizího formátu (pravděpodobně včetně TeXu) do vlastního.

Další informace viz `http://www.kfa-juelich.de/isr/1/texconv/texcnv.html`

77. Konverze z TeXu/LaTeXu do holého ASCII

Zde je snahou emulovat unixový *nroff*, který formátuje text na obrazovce, jak jen to jde nejlépe, a to ze stejného vstupu jako unixový sázecí program *troff*.

Ralph Droms (`droms@bucknell.edu`) má stylový soubor a program, který způsobuje, že je LaTeX rovnocenný s *nroff*, ale neporadí si dobře s tabulkami a s matematikou. Software je dostupný v `support/txt`.

Jinou možností je použít *screen.sty* z CTANu (`macros/latex209/contrib/misc/screen.sty`). Použijte některý z programů typu *dvi2tty*, můžete také zkusit `dviware/crudetype` nebo *catdvi* (`dviware/catdvi`). Poslední jmenovaný program je schopen generovat výstup v Latin-1 nebo UTF-8. *catdvi* bylo počato jako náhrada *dvi2tty*, ale zatím nemůže být na tento účel doporučeno.

Použít můžete konverzní program z LaTeXu do ASCII, *l2a* (`support/l2a`), ačkoliv je to skutečně hodně „odTeXovávací“ program.

Základní odTeXovávací program je *detex* (`support/detex`), který před zápisem na výstup smaže všechny komentáře a řídicí sekvence ze vstupu. Původním záměrem bylo připravit vstup pro jednoduchý korektor překlepů.

78. Převod z HTML či SGML do TeXu

SGML je velice zajímavý nástroj pro správu dokumentů a jejich výměnu, neklade však důraz na formátování, pro formátování dokumentů byl navržen ISO standard DSSSL (`http://www.jclark.com/dsssl`), ten však nebyl zatím implementován. Některé autorské SGML systémy (např. *SoftQuad Author/Editor*) umožňují formátování, a jsou mezi nimi velice profesionální high-endové typografické SGML systémy (např. *Geneva* od Miles33). Nicméně většinou uživatelé SGML, pokud potřebují svůj dokument vytisknout, ho raději transformují do již existujících typografických systémů. Zde můžete zaujmout jeden ze tří přístupů k tomuto problému:

- Můžete napsat jednoduchý překladač, který používá nástroje jako třeba *yacc* nebo *lex*. To je však velice obtížné, protože SGML je velice komplexní a složitý nástroj.
- Použijte speciální jazyk, který byl vytvořen pro transformaci SGML. Nejlepší budou pravděpodobně *Omnimark* a *Balise*. Jsou drahé, ale účinné. Slučují SGML problémy s transformačními schopnostmi.

- Vytvořte nový překladač založený na již existujícím SGML parseru. Zdaleka nejlepší známý (a zdarma!) je parser Jamese Clarka *nsgmls*, který produkuje mnohem jednodušší výstup zvaný `ESIS`, který může být přímo parsován. Tuto metodu využívají dva veřejné balíky:
  - *sgmlspm* vytvořený Davidem Megginsonem v Perlu 5. Je dostupný z `http://www.perl.com/CPAN/modules/by-module/SGMLS`
  - *stil* autorů Joachima Schroda a Christine Detiga napsaný v Common Lispu. Ten si můžete stáhnout na adrese `ftp.th-darmstadt.de/pub/text/sgml/stil`.

  Oba umožňují uživateli napsat si vlastní handlery pro každý SGML element, hojnými přístupy k atributům, entitám a informacím v dokumentovém stromu. Pokud tyto balíky nesplňují vaše požadavky, zřejmě budete pro vytváření běžných SGML dokumentů potřebovat větší komerční vybavení.

Protože HTML je zvláštním případem SGML, nepotřebujeme pro převod z něho do TEXu nebo LATEXu zvláštní systém. Přesto však Nathan Torkington (`Nathan.Torkington@vuw.ac.nz`) vyvinul z HTML ovladače v balíku Xmosaic od NCSA *html2latex*. Program načítá HTML soubor a generuje z něj soubor LATEXový. Konverzní kód je předmětem omezení NCSA, ale kompletní zdrojový text lze získat na `support/html2latex`

Michel Goossen a Janne Saarela vydali velice užitečný souhrn SGML a veřejně dostupných nástrojů pro tvorbu manipulaci s SGML soubory. Vyšel v TUGboatu 16(2). Viz též Otázku 300.

79. Konverze do HTML

Převod z TEXu do HTML není triviální. TEX není značkovací, ale sázecí systém, takže se koncepčně od HTML velmi liší. S LATEXem máte víc naděje, ale ani zde není převod zadarmo. Je dobré poznamenat že a) chcete-li skutečně kvalitní Webový dokument, je lepší přepsat ho znovu, a b) HTML (i HTML3) má velmi chudé sázecí schopnosti a cokoliv netriviálního (matematika, kterou ale TEX zvládá výborně; budoucnost pro metamatiku — viz Otázku 300) transformuje do grafiky.

Je nutné uvědomit si, že HTML dokument nespecifikuje svoje přesné vykreslení a toto je ponecháno do určité míry na prohlížeči. Proto, když chcete čtenářům zobrazit přesnou kopii vzhledu vašeho dokumentu, používejte formáty jako PDF.

V současnosti máme k dispozici tyto možnosti:

LaTeX2HTML (`support/latex2html`) je balík (většinou *perl* dávky), který rozkládá LATEXovský dokument na jednu či více komponent, které pak spojuje dohromady, takže se dají číst přes World-Wide Web jako hypertextový dokument. Definuje zobrazení mezi LATEXovskými vnitřními odkazy a hyperodkazy a rozšiřuje referenční mechanismus LATEXu, aby se zpřístupnily odkazy na jiné připojitelné dokumenty a další prostředky Internetu. Překládá akcentované a další znaky LATEXu (jak jen to jde nejlépe) do formy, kterou dokáže zobrazit World-Wide Web prohlížeč, a zprostředkovává mechanismus, kterým se matematika a další věci, jež prohlížeč nezvládá, dají převést do obrázku, který může být začleněn do hypertextového dokumentu.

LaTeX2HTML vyžaduje *Perl*, PBM utility, *dvips*, *GhostScript* a jiné programy. Běží pod Unixem. V *TUGboatu* **16**(2) je publikován podrobný popis programu LATEX2HTML a návod na jeho konfiguraci. Původní verze byla napsána Nikosem Drakosem pro systémy Unix, dnes je již ale spravován více autori a je k dispozici i pro Windows.

TtH je vykompilovaný program podporující LATEX nebo čistý TEX používající techniky font/tabulka pro reprezentaci matematiky. Byl napsaný Ianom Hutchinsonom používajíc *flex*. Distribuce je tvořena jediným zdrojovým souborem jazyka C (nebo vykompilovaným spustitelným souborem), který se snadno instaluje a je dost rychlý.

Tex4ht je vykompilovaný program napsaný Eitanem Gurarinem podporující LATEX nebo čistý TEX. Funguje na principu spracování DVI souboru a používa bitmapy pro matematiku. Když je to vhodné, může využívat i jiních technologií. Program parsuje DVI soubor generovaný, když nad vaším souborem obsahujícím *tex4ht* makra spustíte TEX/LATEX. Výsledkem je, že je celkem robustní vůči makrám ve vašich dokumentech a je taky celkem rychlý.

TeXpider je komerční program od MicroPress (viz Otázku 63), popsaný na `http://www.micropress-inc.com/webb/wbstart.htm` (pro rovnice používá bitmapy)

Hevea je vykompilovaný program podporující jenom LATEX používající techniku font/tabula pro rovnice (jeho přístup je podobný programu *TtH*). Je napsaný v jazyku Objective CAML Lucem Marangetem. *Hevea* není archivována na CTANu, detaily (vrátaně míst pro download) jsou na `http://pauillac.inria.fr/~maranget/hevea/`

Zajímavá sada příkladů (včetně konverze stejného textu zmíněnými čtyřmi programy) je přístupná na `http://www.mayer.dial.pipex.com/samples/example.htm`. Odkazovaná stránka poskytuje seznam za a proti formou porovnání

80. Vytváranie hypertextových dokumentov v TeXu

Ak chcete vytvoriť hypertextový dokument pomocou LaTeXu, napríklad pre World-Wide Web, uvážte štyri technológie (prekrývajúce sa).

1. Skúste priamu konverziu z LaTeXu do HTML (pozri Otázku 79);
2. Prepíšte svoj dokument použijúc Texinfo (pozri Otázku 16) a skonvertujte ho do HTML;
3. Pozrite sa na Adobe Acrobat, elektronický systém na uchovávanie a prezeranie dokumentov. Tento systém môže zachovať úplne sadzbu vášeho dokumentu. (Pozri Otázku 81.)
4. Využite hyperTeX konvencie (štandardizované príkazy `\special`). Existujú makro-balíky, ktoré ich podporujú pre plain TeX a LaTeX.

Projekt HyperTeX má za cieľ rozšíriť funkčnosť všetkých odkazových príkazov pre LaTeX (vrátane obsahu) vytváraním príkazov `\special`, ktoré sú sledované DVI procesormi a vytvárajú hypertextovú štruktúru. Poskytujú všeobecné hypertextové spojenia, vrátene napojenia na vonkajšie dokumenty.

HyperTeXovská špecifikácia určuje, že príslušný prezerač/prekladač musí rozpoznávať nasledujúce príkazy `\special`:

href: `html:<a href = "href_string">`
name: `html:<a name = "name_string">`
end: `html:</a>`
image: `html:<img scr = "href_string">`
base_name: `html:<base href = "href_string">`

Príkazy *href*, *name* a *end* sa používajú na definovanie základných hypertextových operácií spojenia jednotlivých častí dokumentu.

Ďalšie detaily sú dostupné na `http://arXiv.org/hypertex/`. Existujú zatiaľ len dve všeobecne používané implementácie špecifikácií: modifikované *xdvi* a modifikované *dvips*. Výstup z modifikovaného *dvips* môže byť použitý modifikovaným programom *GhostScript* alebo programom Acrobat Distiller.

81. Vytváranie PDF dokumentov v TeXu

Existujú tri možnosti: previesť PostScript do PDF programom *Distiller*, konvertovať DVI súbor alebo použiť program pre priamy prevod z TeXu do PDF (*pdftex* Han The Thanha alebo VTeX od spoločnosti MicroPress - viď Otázku 63 a Otázku 61).

Pre jednoduché dokumenty (bez hyper-referencií) môžete:

- spracovať dokument normálnym spôsobom, vyprodukovať výstup vo formáte PostScript a na ten použiť program *Distiller*,
- (na systéme Windows alebo Macintosh s nainštalovanými zodpovedajúcimi nástrojmi od Adobe) predať výstup cez PDFwriter na mieste ovládača tlačiarne (toto je však slepá ulica, pretože PDFwriter nevytvára odkazy),
- spracovať dokument normálnym spôsobom a vygenerovať PDF priamo s DVI programom *dvipdfm*, alebo
- spracovať dokument priamo do PDF pomocou PDFTeXu alebo VTeXu. PDFTeX má výhodu dostupnosti pre široký zoznam platforiem, VTeX (dostupný komerčne pre Windows, zadarmo pre Linux a OS/2) má širšie grafické možnosti pri spracovaní zapuzdreného a in-line PostScriptu.

Aby sme preložili všetky krížové odkazy v LaTeXu do Acrobatovských spojení, potrebujeme LaTeXovský balík, ktorý vhodne predefinuje vnútorné príkazy LaTeXu. Existujú dva takéto balíky pre LaTeX 2_ε, obidva založené na HyperTeXovských špecifikáciách (pozri Otázku 80). Prvý je *hyperref* od Sebastiana Rahtza (dostupný z `macros/latex/contrib/hyperref` a druhý je *hyper* od Michaela Mehlicha (`macros/latex/contrib/hyper`). Balík *hyperref* používa konfiguračný súbor pre zistenie, akým spôsobom bude generovať hypertext. Môže pracovať pomocou primitív PDFTEXu, hyperTEXovských príkazov typu `\special` alebo príkazov typu `\special` špecifických pre konkrétny DVI ovládač. *dvips* aj *DVIPSONE* spoločnosti Y&Y prekladajú DVI za pomoci týchto príkazov `\special` do PostScriptu akceptovaného Distillerom, *dvipdfm* má vlastné `\special` príkazy.

Neexistuje voľne dostupná implementácia všetkých funkcií *Adobe Distiller*u, ale posledné verzie *ghostscript*u poskytujú celkom spoľahlivý proces „destilácie" (vyvarujte sa však problémom preberaným v Otázke 82). Sám *Distiller* je v skutočnosti celkom lacný (aspoň pre akademické prostredie).

Na prezeranie (a tlač) výsledných súborov je dostupný Adobe *Reader* (predtým známy ako Adobe *Acrobat Reader*) na celkom slušný rozsah platforiem. Na platformách, pre ktoré dostupný nie je, vie PDF súbory zobraziť kombinácia aktuálnych verzií programov *ghostscript* a *ghostview*, prípadne *GSview*.

Mnohokrát je *ghostscript* v kombinácii s prehliadačom preferovaný oproti programu Adobe Reader. Adobe Reader pre Windows napríklad zamkýňa `.pdf` súbor, ktorý zobrazuje. Toto znemožňuje tradičný (a veľmi efektívny) vývojový cyklus dokumentu TEXu/LATEXu (úprava — spracovanie — prezretie). *GSview* túto chybu neopakuje.

82. Kvalita PDF z PostScriptu

Každý rozumný postscriptový dokument (vrátane výstupu *dvips*) sa dá skonvertovať do PDF, napríklad použitím dostatočne novej verzie *ghostscript*u, programu *PStill* Franka Siegerta (shareware; `http://www.pstill.com/`) alebo komerčného programu *Distiller* od Adobe.

Avšak aj keď sa to dá spraviť takmer vždy, výsledky často nie sú akceptovateľné. Najčastejším problémom je zlá reprezentácia glyfov znakov, ktoré dopĺňajú dokument. Nasledujúce odpovede ponúkajú riešenie týchto (a iných) problémov zlej prezentácie. Zodpovedané otázky sú:

- nesprávny typ použítých fontov (viď Otázku 84), ktorý je najčastejšou príčinou neostrého textu,
- stará verzia *ghostscript*u (viď Otázku 85), čo tiež môže zapríčiniť neostrý text,
- zmena na T1 kódovanie fontov (viď Otázku 86)takisto môže zapríčiniť neostrý text,
- ďalší problém (chýbajúce znaky) je zapríčinený starou verziou *Adobe Distiller*u (viď Otázku 87),
- obvyklý zmätok nastávajúci pri použití konfiguračného súboru *dvips* `-Ppdf` (viď Otázku 100).

Je dobré poznamenať, že *Adobe Reader* 6 (vydaný v polovici roku 2003) túto neostrosť neprejavuje. Je to dobrá správa, ale určite bude trvať dlhší čas, kým bude každý používateľ vybavený touto (alebo novšou) verziou, takže riešenia spomenutých problémov tu ešte nejaký čas zostanú.

83. *Hyperref* a opakované čísla strán

Trieda `book` (a príbuzné) automaticky mení zobrazenie čísle strán na začaitku dokumentu na malé rímske číslice. Toto nevadí ľudským čitateľom, ale je mätúce pre *hyperref*, kedže existujú straný, ktoré (pre *hyperref*) majú rovnaké čísla strán. Našťastie existujú konfiguračné voľby umožňujúce, aby *hyperref* „urobil správnu vec".

Dotyčné dve voľby sú:

`plainpages=false` Vynúti pomenovanie ukotvenia strán v arabskej forme absolútneho čísla strany (namiesto formátovanej formy). S touto voľbou bude *hyperref* zapisovať rozličné ukotvenia pre strany „ii" a „2".

`pdfpagelabels` Nastaví PDF návestia strán, t.j. zapíše hodnotu `\thepage` do PDF súboru, aby *Acrobat Reader* mohol zobraziť číslo strany ako (napríklad) „ii (4 of 40)" namiesto „4 of 40".

Uvedené dve voľby by mali byť používané kedykoľvek je číslovanie strán iné než iba „`1..`*n*"; môžu byť použité nezávisle, ale zvyčajne nie sú.

Návod nie je bezchybný: spolieha sa na to, že `\thepage` je rozdielne pre každú stranu v dokumente. Bežný problém vyvstáva pri neočíslovanej titulnej strane, po ktorej sú čísla strán resetované:

varovanie PDFTEXu o duplicitných cieľoch nastanú v tomto prípade bez ohľadu na voľby.
`hyperref.sty`: `macros/latex/contrib/hyperref`

84. Nesprávny typ fontov v PDF

Toto je zdaleka najčastejší problém: symptómom je neostro vyzerajúci text dokumentu.

Väčšina ľudí používa na prezeranie svojich PDF súborov *Adobe Reader*: *Reader* je distribuovaný zdarma a je dobre dostupný, má však mnoho nedostatkov. Jedným z nich je neschopnosť spracovania bitmapových fontov (minimálne vo verziách skorších než nedávno vydaná verzia 6).

Bitmapové fonty teda v našich PostScriptoch nechceme: s nimi sa znaky v zobrazení programom *Reader* objavujú ako rozmazané machule, z ktorých sa často nedajú rozoznať pôvodné písmená a často sú zle umiestnené na riadku. Napriek tomu má väčšina TEXovských systémov nakonfigurované *dvips* na použitie `.pk` súborov (viď Otázku 40) vo svojom výstupe. Dokonca aj PDFTEX použíje `.pk` súbpry, ak nevidí inú alternatívu pre font v spracúvanom dokumente.

Naším riešením sú Adobe Type 1 verzie potrebných fontov. Kedže Adobe fonty Type 1 predáva, *Reader* bol od začiatku vytvorený tak, aby si s nimi vedel dobre poradiť.

Samozrejme, ak váš dokument používa iba fonty od Adobe (fonty ako `Times`, ktoré sú v takmer každej PostScriptovej tlačiarni, alebo Adobe `Sabon`, za ktorý si musíte priplatiť) potom nemáte problém.

Väčšina ľudí však na začiatok používa fonty `Computer Modern` a dokonca aj tí uvedomelí, čo používajú niečo tak exotické ako *Sabon*, sa často nechtiac nachádzajú v situácii, keď použíjú zvláštne znaky z CM. Dostupné sú našťastie celkom dobré verzie CM fontov od AMS (vďaka Blue Sky Research a Y&Y – viď Otázku 63).

Väčšina moderných systémov má nainštalované fonty pripravené na použitie a každý systém inštalovaný pred menej než 3 rokmi má *dvips* konfiguračný súbor „`pdf`“ signalizujúci použitie CM fontov a tiež nastavuje pár ďalších parametrov na vylepšenie výstupu *dvips*. Túto konfiguráciu použite nasledovne:

```
dvips -Ppdf myfile -o myfile.ps
```

Toto môže produkovať varovné hlásenie o nenájdení konfiguračného súboru:

```
dvips: warning: no config file for 'pdf'
```

alebo niečo podobné, alebo hlásenie o nenájdení súboru fontu:

```
dvips: ! Couldn't find header file cmr10.pfb
```

Každé zo zlyhaní signalizuje, že váš systém fonty neobsahuje.

Spôsob používania fontov, ktorý nezahŕňa sofistikovanosť mechanizmu `-Ppdf`, je jednoduché nahranie máp:

```
dvips -Pcmz -Pamz myfile -o myfile.ps
```

Môžete sa tu stretnúť s rovnakými varovnými hláseniami ako boli uvedené vyššie.

Ak váš systém fonty neobsahuje, budete potrebovať fonty nainštalovať.

85. Neostré fonty pre príliš starý *Ghostscript*

Spravili ste všetko, čo vám radilo FAQ, máte správne nainštalované potrebné fonty, zobrazujú sa vo výstupe *dvips*, ale *stále* dostávate po destilácii pomocou programu *ghostscript* výstup neostrých znakov.

Problém mohol vzniknúť z príliš starej verzie programu *ghostscript*, ktorú môžete používať priamo, alebo cez skript ako *ps2pdf* (distribuovaný so samotným *ghostscript*om), *dvipdf* alebo podobný. Hoci *ghostscript* mal schopnosť destilácie od verzie 5.50, uvedená verzia vedela produkovať iba bitmapový Type 3 výstup ľubovolného fontu iného než základných 35 fontov (`Times`, `Helvetica` atď.). Neskoršie verzie pridali „úplnú“ destiláciu, ale až do verzie 6.50 sa na ňu nedalo spoľahnúť pri použití v každodennej práci.

Ak teda váš PDF výstup stále vyzerá neostro v programe *Acrobat Reader*, aktualizujte *ghostscript*. Nová verzia by mala byť samozrejme aspoň verzia 6.50, je však dobrým pravidlom siahnuť po najnovšej verzii (verzia 8.12 v dobe písania anglickej verzie tejto otázky – rok 2003).

86. Fonty sa stané neostrými pri prechode na T1

Mali ste problémy s rozdeľovaním slov a niekto vám poradil, aby ste na ich vyriešenie použili príkaz „`\usepackage[T1]{fontenc}`“. Zrazu zisťujete, že vaše hotové PDF je neostré. Problém môže vzniknúť bez ohľadu na to, či používate PostScriptový výstup a následne destilujete, alebo pre celú prácu používate PDFTEX.

V skutočnosti sa jedná o podobný problém, aké sú mnohé ďalšie pri zhoršenej kvalite PDF výstupu (viď Otázku 82): opustili ste vaše predchádzajúce nastavenie používajúci Type 1 verzie CM fontov a *dvips* vložilo do výstupu vášho dokumentu Type 3 verzie EC fontov (poznamenajme len, že *kódovanie* fontov T1 nemá priamo nič spoločné s *typom* fontov Type 1).

Ako však je poznamenané v Otázke 88, Type 1 verzie fontov podobných CM v T1 (alebo ekvivalentnom) kódovaní sú dostupné ako sady „skutočných“, ako aj virtuálnych fontov. Jedným riešením je preto používanie jednej z týchto alternatív.

Ďalšou možnosťou je zmena celej rodiny fontov na niečo ako `Times` (štandardná voľba bez premýšľania) alebo na jeden z mnohých viac potešujúcich Adobe-kódovaných fontov. Štandardná akcia balíku *fontinst* pri vytváraní metrík pre takéto fonty je vytvorenie nastavení pre OT1 aj T1 kódovania, takže sa jedná o malú zmenu (na užívateľskej úrovni) dokonca aj pri zmene na T1 kódovanie.

87. Chýbajúce znaky v PDF výstupe

Ak používate *Acrobat Distiller* na tvorbu PDF výstupu, môžete nájsť určité chýbajúce znaky. Toto sa môže prejaviť ako nesprávne matematické rovnice (napríklad chýbajúce znaky '$-$') alebo chýbajúce časti väčších symbolov. Staré verzie *Distiller*u mali vo zvyku ignorovať pozície znakov 0–31 a 128–159 pri každom fonte. Fonty Adobe nikdy tieto pozície nevyužívajú, tak prečo by mal *Distiller*?

Odpoveď je: „Pretože Adobe neprodukuje všetky fonty na svete.“ Fonty ako `Computer Modern` existovali pred príchodom Adobe na scénu a používali pozície 0–31. Firma Adobe na sťažnosti typu predchádzajúcej vety nereagovala, ale vydala novú verziu svojich programov (*Distiller* od verzie 4.0 už rozpoznáva pozície fontov, ktorým sa predtým vyhýbal).

Používatelia TEXu so starou verziou *Distiller*u sa zatiaľ potrebovali vysporiadať so svojimi fontami. Pomáha im *dvips*, konkrétne prepínač `-G1` („premapuj znaky“), ktorý problémové znaky odstraňuje z cesty. Konfiguračný súbor PDF (`-Ppdf`) doporučovaný vyššie (v Otázke 84) tento prepínač obsahuje.

Ani tento prepínač však neprichádza bez problémov. Verzie *dvips* nižšie než 2003 ho aplikujú aj na Adobe fonty, čo zapríčiní hotový blázninec (viď Otázku 100). Problém je našťastie vysvetliteľný. Avšak dokument používajúci fonty CM i fonty špecifické pre Adobe má problém. Jediným skutočným riešením je buď aktualizácia *dvips*, alebo minúť peniaze na aktualizáciu *Distiller*u.

88. Hľadanie 8-bitových Type 1 fontov

Inde vám odpovede na tieto „FAQ“ odporučia používať 8-bitový font na povolenie diakritiky vo flektívnych jazykoch (viď Otázku 233) a použite Type 1 fontov na zaručenie toho, že dostanete kvalitné PDF (viď 84). Tieto odporúčania *boli* protichodné. Nebolo možné vymeniť voľne dostupné CM fonty za voľne dostupné Type 1 fonty v corkovskom (alebo podobnom) kódovaní. Prvý prístup, ktorý začal pomáhať s touto priamou výmenou, bol vývoj virtuálnych fontov, ktoré majú dobrý prístup ku corkovskému kódovaniu (viď nižšie). V súčasnosti však máme dostupné skutočné Type 1 fonty. Ako vždy máme k dispozícii „bohatý“ výber (tri voľne dostupné, dve komerčné a jednu shareware verziu).

`CM-super` (CTAN: `fonts/ps-type1/cm-super`) je automaticky trasovaná sada, ktorá zahŕňuje kódovania T1, TS1 a T2* (rodina kódovaní pokrývajúca jazyky založené na cyrilike). Tieto fonty sa celkom jednoducho inštalujú (pustup je zrozumiteľný), ale sú obrovské — nepokúšajte sa ich nainštalovať, ak máte málo voľného priestoru na pevnom disku.

`CM-LGC` (CTAN: `fonts/ps-type1/cm-lgc`) je podobná sada „super-fontov“, ale oveľa skromnejšej veľkosti. Pokrýva T1, TS1 a T2A kódovania (ako `CM-super`) a naviac aj LGR kódovanie (pre sadzbu gréčtiny, založené na zdrojových kódoch METAFONTu Claudia Beccariho). CM-LGC sa darí zachovať si malú veľkosť tým, že ide do opačného extrému než CM-super, ktorý zahŕňa fonty všetkých veľkostí podporovaných pôvodným EC (veľké množstvo). CM-LGC má jeden font pre každý rez, ostatné veľkosti získava škálovaním. Z tohoto postupu nevyhnutne vyplýva určitá strata kvality, ale zároveň sa šetrí miesto na pevnom disku.

Latin Modern (fonts/ps-type1/lm) je vyrobený pomocou programu *MetaType1*, ktorý (ako už názov naznačuje) prináša silu METAFONTu do prostredia produkcie Type 1 fontov. Sada Latin Modern obsahuje varianty v kódovaniach T1, TS1 LY1 (a taktiež variantu používajúcu poľské kódovanie QX). Obrysy pokrytých glyfov vyzerajú „čistejšie“ ako v CM-super. Balík Latin Modern má menšie požiadavky na diskový priestor ako CM-super a nezachádza do takých extrémov v podporovaných veľkostiach ako CM-LGC (Latin Modern fonty sú poskytované vo veľkostiach rovnakých ako CM fonty). S týmto rozhodnutím sa nedá prieť, Knuthov rozsah veľkostí odolal skúške času a je jedným zo základov, ktorým systém TEX vďačí za svoju excelentnosť.

Virtuálne fonty (viď Otázku 42) nám pomáhajú vysporiadať sa s týmto problémom, kedže nám dovoľujú mapovať časti DVI súboru na jednotlivé znaky vo virtuálnych fontoch. Môžeme teda vytvoriť znak „é“ pomocou znovuvytvorenia DVI príkazov, ktoré vyplývajú z príkazov „\'e“. Keďže však toto zahŕňa výber dvoch znakov z fontu, opatrenie postačuje na to, aby oklamalo *Adobe Reader* (nemôžete využívať prostriedky programu pre hľadanie textu obsahujúceho flektívne znaky a v prípade kopírovania textu z okna obsahujúceho tieto znaky sa stane niečo nečakané, obvykle dostanete diakritické znamienko a základný znak oddelené medzerou). Ak vám to však nevadí, sú virtuálne fonty dobrým a priamočiarym riešením.

Existujú dve ponuky virtuálnych fontov založených na 8-bitových CM fontoch. Sú to sady ae (almost EC; takmer EC) a zefonts. Môžete ich nájsť na CTANe: fonts/ae, fonts/zefonts. zefonts má lepšie pokrytie znakov (aj keď ae môže byť rozšírená balíkom aeguill; CTAN: macros/latex/contrib/aeguill). Žiadna zo sád neposkytuje znaky ako eth a thorn (používané napríklad v islandčine), ale balík aecompl využíva ae fonty na poskytnutie chýbajúcich znakov z EC fontov (napríklad ako bitmapy).

Komerčné fonty 8-bitové fonty typu CM fontov sú celkom lacné, ale stoja viac, než si tento autor môže bežne dovoliť... Y&Y poskytuje fonty „European Modern“, rozšírenie CM fontov, ktoré môže byť používané s kódovaním T1 alebo LY1. Tieto fonty sú z rovnakej stajne, ktorá nám zadarmo poskytla AMS/Blue Sky Research/Y&Y fonty. Sú mierne rozšírené, aj keď nepokrývajú všetky oblasti T1 kódovania a neposkytujú chaos veľkostí ako EC fonty). Micropress poskytuje úplnú EC sadu v Type 1 formáte (ako časť ich množstva obrysových fontov pôvodne distribuovaných v METAFONT formáte). Ďalšie informácie nájdete v Otázke 63.

Shareware distribúcia BaKoMa TEX (viď Otázku 61) poskytuje sadu Type 1 EC fontov (nonfree/systems/win32/bakoma/fonts) ako extra shareware voľbu (podľa znalostí autora sú tieto fonty zatiaľ dostupné *iba* používateľom BaKoMa TEXu; sú archivované vo formáte, ktorý nie je verejne dostupný).

Nakoniec môžete použiť jeden z nespočetne mnoho textových fontov v Type 1 formáte (so zodpovedajúcou PSNFSS metrikou pre T1 kódovanie, alebo metrikou pre nejaké iné 8-bitové kódovanie ako LY1). Ak však použijete niekoho textový font, budete musieť nájsť príslušnú rodinu matematických fontov, čo je netriviálna záležitosť (viď Otázku 99).

89. Používání TEXu k přímému čtení XML a SGML

Číst XML nebo SGML lze pomocí důmyslného makroprogramování. Dvěma příklady balíčků, které byly vyvinuty za tímto účelem a jsou volně dostupné, jsou *xmltex* (macros/xmltex/base) Davida Carlislea a *GMLbase* (http://www.active-tex.demon.co.uk). Při psaní je *xmltex* poněkud vyspělejší než *SGMLbase* a nabízí praktické řešení sazby XML dokumentů. *SGMLbase* byl vytvořen, aby se vypořádal s SGML stejně dobře jako s XML, ale v současnosti je to tak akorát příklad na hraní.

Využití TEXu umožňující vysázení XML souborů je backendový procesor pro formátovací objekty XSL serializované jako XML. PassiveTEX Sebastiana Rahtze používá balík xmltex aby tohoto dosáhl.

90. Preklad LaTeXu do čistého TeXu

Bohužiaľ neexistuje žiadny obecný postup, s ktorým by ste uspeli pri riešení tejto úlohy. Pre podrobnosti si pozrite Otázku 10.

Preklad dokumentu navrhnutého pre LaTeX do dokumentu pre čistý TeX obnáša opatrné zahrnutie (alebo inú re-implemenáciu) všetkých častí LaTeXu, ktoré dokument používa a nie sú poskytované čistým TeXom.

K. METAFONT

91. Jak dostat z METAFONTu to, co chcete

METAFONT vám umožňuje vytvořit si vlastní font, ale většina uživatelů TeXu ho nikdy nepoužívá. METAFONT, na rozdíl od TeXu, vyžaduje nekterá speciální nastavení. Každé výstupní zařízení, pro které se generuje font, potřebuje přidružený mód. Módy se definují použitím konvencí `mode_def` popsaných na straně 94 v *The METAFONTbook* (viz Otázku 23). Je potřeba mít soubor zpravidla nazvaný `local.mf` obsahující `mode_defs`, které budete používat. Jestliže `local.mf` ještě neexistuje, dobrým výchozím bodem může být kolekce módů od Karla Berryho, dostupná jako `fonts/modes/modes.mf`. Lze ji použít jako `local.mf` beze změn pro „dostatečně širokou" implementaci METAFONTu. Výpis nastavení pro nejrůznější výstupní zařízení bývá také pravidelně zveřejňován v TUGboatu (viz Otázku 20). Nyní vytvořte bázový soubor `plain` s použitím *inimf*, `plain.mf` a `local.mf`:

```
% inimf
This is METAFONT...
 **plain                    zadejte „plain"
 (output)
 *input local               zadejte toto
 (output)
 *dump                      zadejte toto
 Beginning to dump on file plain...
 (output)
```

Tím se vytvoří bázový soubor pojmenovaný `plain.base` (případně něco podobného; například v MS-DOSu to bude `PLAIN.BAS`), který je třeba přemístit do adresáře, jenž ve vašem systému obsahuje bázové soubory. (Některé systémy obsahují i dva nebo více takových adresářů, každý pro jinou „velikost" použitého METAFONTu.)

Nyní se potřebujete ujistit, že METAFONT při spuštění čte tuto novou bázi. Jestliže METAFONT načítá na vašem systému implicitně `plain` bázi, jste připraveni. Pod Unixem (za použití běžné distribuce *web2c*) to tak skutečně funguje, ale mohli bychom třeba definovat příkaz *mf*, který vykonává `virmf &plain` načtením `plain` bázového souboru.

Obvyklý způsob, jak vytvořit font s `plain` METAFONTem, je začít řádkem

```
\mode=<jméno módu>; mag=<zvětšení>; input <jméno fontového souboru>
```

jako odpověď na výzvu „`**`" nebo na příkazové řádce METAFONTu. (Pokud `<jméno módu>` je neznámé nebo neplatné, bude použit implicitní mód a METAFONT vyrobí výstupní soubor nazvaný `<jméno fontového souboru>.2602gf`) `<Zvětšení>` je číslo v plovoucí řádové čárce nebo „magstep" (magstepy jsou definovány v *The METAFONTbook* a v *The TeXbook*). Jestliže `mag=<zvětšení>` je neplatné, bude implicitně 1 (magstep 0). Například, pro generování cmr10 zvětšeného na 12pt pro tiskárnu epson byste měli napsat

```
mf \mode=epson; mag=magstep 1; input cmr10
```

Poznamenejme, že pod Unixem jsou znaky \ a ; obvykle v uvozovkách nebo se zpětným lomítkem, takže by to zpravidla mělo vypadat takhle:

```
mf '\mode=epson; mag=magstep 1; input cmr10'
```

v bázi, můžete vložit jeho příkazy do souboru (např. `ln03.mf`) a vyvolat ho za běhu příkazem `\smode`. Tak například, aby se vytvořil `cmr10.300gf` pro tiskárnu LN03 za použití souboru

```
% This is ln03.mf as of 2/27/90
```

```
% mode_def courtesy of John Sauter
proofing:=0;
fontmaking:=1;
tracingtitles:=0;
pixels_per_inch:=300;
blacker:=0.65;
fillin:=-0.1;
o_correction:=.5;
```

(všimněte si nepřítomnosti příkazů `mode_def` a `enddef`), je třeba napsat

```
mf \smode="ln03"; input cmr10
```

Tato technika není jediná, kterou byste správně měli používat, ale může se ukázat užitečná, jestliže máte novou tiskárnu a chcete experimentovat s parametry, či z nějakých jiných důvodů editujete používané parametry. Jakmile jste si jednou ujasnili přípustnou množinu parametrů, můžete je uplatnit při přebudování bázového souboru, který používáte. Souhrn výše uvedeného od Geoffrey Tobina a upozornění na nejčastější nástrahy a úskalí při používání METAFONTu lze nalézt v dokumentu `info/metafont-for-beginners.tex`

Jiné zdroje pomoci jsou zmíněnyv otázce 31.

92. Které fontové soubory je třeba si nechat

METAFONT produkuje při svém běhu tři soubory: metrikové (`tfm`) soubory, soubory s generickými fonty (`gf`) a soubory s popisem překladu (`log`). Všechny tyto soubory mají stejné bázové jméno jako vstup, (např. pokud vstupní soubor byl `cmr10.mf`, výstupem bude `cmr10.tfm`, `cmr10.nnngf`[6] a `cmr10.log`).

Pro práci TEXu s fontem je zapotřebí `tfm` soubor, takže si ho musíte nechat. Ale obyčejně chcete generovat tentýž font ve více velikostech a pokaždé se bude (automaticky) generovat další `tfm` soubor. Tyto soubory jsou stejné, proto stačí nechat si jediný z nich.

K výrobě obrazovkového nebo tiskového výstupu potřebuje `dvi` procesor rastrový soubor fontu; to je to, co poskytuje `gf` soubor. Avšak, zatímco dříve používané `dvi` procesory uměly pracovat s `gf` soubory, moderní procesory používají komprimované rastrové (`pk`) soubory. Proto je třeba generovat `pk` soubor z `gf` souboru. Tohle za vás udělá program *gftopk* a jakmile je to jednou hotovo, můžete soubor `gf` odstranit.

Pokud nenastanou nějaké problémy, neměl by být soubor `log` nikdy použit. Normálně tedy není třeba nechávat si jej.

93. Vytažení bitových map z archívu

Většina lidí, kteří začínají používat TEX s laserovou tiskárnou s rozlišením 300 dpi (dots-per-inch) a fonty Computer Modern pro toto rozlišení, je zásobena množstvím TEXovských balíků. Na CTAN jsou také dvě takové sady: `fonts/cm/pk/pk300.zip` (pro černý tisk) a `fonts/cm/pk/pk300w.zip` (pro bílý tisk). Ale někteří uživatelé chtějí poslat svou práci na vysoce kvalitní sázecí stroje (obyčejně s rozlišením 1270 dpi nebo více). Též se častěji používají 600 dpi laserové tiskárny. Proč archívy a knihovny neposkytují bitové mapy fontů v těchto velikostech? Důvody jsou dva:

1. Pokud je bitová mapa fontu vytvořena pomocí METAFONTu, je třeba znát charakteristiky zařízení; kdo může vědět, jaký 600 či 1270 dpi přístroj vlastníte? (Tahle námitka se samozřejmě stejně dobře týká 300 dpi tiskáren.)
2. Při vysokých rozlišeních jsou bitové mapy *objemné*. Kdo ví, které fonty a s jakou velikostí potřebujete?

Technologie distribucí TEXu/LATEXu naštěstí tyto problémy ukončila. Většina (když ne všechny) současných distribucí generuje bitmapové fonty, když jsou zapotřebí a ukládá je do cache pro opakované pozdější použití. Netrpezlivý uživatel, který si myslí, že všechny bitmapové fonty by měly být vytvořeny jenom jednou (a navždy), může najít podporu v skriptech jako *allcm* (distribuován přinejmenším s teTEXem); tato osoba by si měla taky pročíst Otázku 91).

[6] Všimněte si, že jméno souboru může být přetvořeno u takových operačních systémů jako MS-DOS, které nepřipouštějí dlouhé názvy souborů.

Na druhé straně, pokud vlastníte PostScriptové zařízení, uvažujte o fontech ve formátu Type 1. Je možné koupit všechny fonty Computer Modern v PostScriptovém tvaru od Blue Sky Research nebo od Y&Y (adresa je uvedena v odpovědi na Otázku 63), či použít veřejně přístupnou verzi Basila Malysheva v `fonts/cm/ps-type1` (kolekce Paradissa je kompletní, ale postupně je nahrazována lepší kolekcí BaKoMa). V posledních letech se ve formátu Type 1 stalo dostupnými množství jiných fontů METAFONTu. Obvykle není jiný důvod pro generování bitmapových fontů než za jiným účelem než vytvoření náhledu (viz Otázku 97).

Komerční dodavatelé fontů jsou vždy o krok vpřed vůči hnutí free software a poskytují Type 1 verze EC fontů, CM fontů cyriliky, jakožto i rozsah matematických fontů nahrazujících rodinu CM (viz Otázku 99).

L. PostScript a TEX

94. Použití PostScriptových fontů v TEXu

Ještě než začnete: aby mohl použít PostScriptové fonty, TEX potřebuje soubory s *metrikami* (zvané `tfm`). Několik sad metrik je dostupných v archívech; mechanismus generování nových je popsán v Otázce 96. Také potřebujete vlastní fonty. PostScriptové tiskárny se sice dodávají již se sadou fontů zabudovaných, ale pro rozšíření svého repertoáru si většinou stejně musíte koupit jednu z komerčních nabídek fontů (viz Otázku 99).

Pokud používáte LATEX 2_ε, nejlepším způsobem, jak dostat do svého dokumentu PostScriptové fonty, je použít balík PSNFSS, který spravují Walter Schmidt (`macros/latex/required/psnfss`). Je podporován projekčním týmem pro LATEX3, takže zprávy o chybách by mohly a měly být brány do úvahy. PSNFSS poskytuje sadu balíků pro změnu standardních roman, sans-serif a typewriter fontů. Např. `times.sty` nastaví fonty Times Roman, Helvetica a Courier místo Computer Modern, zatímco `avant.sty` změní rodinu fontů sans-serif na AvantGarde. Pro práci s těmito balíky jsou potřebné soubory s metrikou (pozor na problémy s kódováním! – viz Otázku 96) a soubor s popisem fontu (`.fd`) pro každou rodinu, kterou chcete použít. Výhodné je, že metriky pro společných 35 PostScriptových fontů pro většinu tiskáren jsou začleněny do PSNFSS, zkomprimovány jako `macros/latex/required/psnfss/lw35nfss.zip`

Pro starší verze LATEXu existují nejrůznější schémata, z nichž nejjednodušší pro použití jsou pravděpodobně makra PSLATEX distribuovaná s *dvips*.

Pro `plain`TEX můžete použít fonty, jaké chcete. Pokud kódování fontů není stejné jako u Computer Modern, je na vás, abyste si předefinovali různá makra a akcenty, anebo můžete použít mechanismus pro překódování fontů dostupný v mnoha ovladačích a v *ps2pk* a *afm2tfm*.

Rafinovaný balík Lollipop Victora Eijkhouta (`nonfree/macros/lollipop`) podporuje deklaraci rodin fontů a styly podobným způsobem jako LATEXovský NFSS, takže se dá snadno použít s PostScriptovými fonty.

Některé společné problémy, které byly zmíněny, jsou diskutovány i jinde (viz Otázku 98).

95. Prohlížení souborů s PostScriptovými fonty

Většina TEXovských prohlížečů zobrazuje pouze bitové mapy `PK` fontů. Nicméně většina běžně dostupných DVI prohlížečů nabízí automatické generování potřebných `PK` souborů (pomocí *gsftopk* nebo něčeho podbného). Pokud chcete prohlížet dokumenty s PostScriptovými fonty, máte tři možnosti:

1. Převést `DVI` soubor do PostScriptu a použít PostScriptový prohlížeč. Některé moderní Unixové X-ové implementace mají tohle zabudováno (jako třeba NeXT-step). Uživatelé Unixu (X11), Windows, OS/2 a MS-DOSu mohou použít volně dostupný GhostScript (`nonfree/support/ghostscript`), implementaci kompletního PostScriptu úrovně 2 nebo prohlížeč *gsview*, který je založený na *Ghostscriptu*.
2. Pod Windows na PC, nebo na Macintoshi umí Adobe Type Manager zobrazit PostScriptové fonty. TEXtures (Macintosh) pracuje podobně, pod Windows pak lze použít program *dviwindo* od Y&Y pro prohlížení bitových map. (Detaily těchto doplňků: viz Otázku 63.)

3. Jestliže máte PostScriptové fonty formátu Type 1, použijte *ps2pk* (`fonts/utilities/ps2pk`) nebo *gsftopk* (navržený pro použití s fonty programu *ghostscript*, `fonts/utilities/gsftopk`) pro výrobu `pk` bitových map fontů, kterým váš prohlížeč bude rozumět. Tak lze dosáhnout skvělých výsledků, které jsou vhodné i pro tisk na zařízeních bez PostScriptu. Prověřte si právní podmínky, pokud chcete tyto fonty získat. Nejznámější PostScriptové fonty jako Times a Courier se dodávají s formátem Type 1 na disku s Adobe Type Managerem (často dohromady s Windows a část s OS/2).

96. Soubory metrik TEXovských fontů pro PostScriptové fonty

Výrobci fontů takoví jako Adobe opatřují metrikovým souborem každý font, a to v `afm` (Adobe Font Metric) formě. Tento formát lze konvertovat do tvaru `tfm` (TEX Font Metric). CTAN archivy obsahují předem připravené metriky, které budou pro mnoho lidí více než dostatečné, ale můžete si udělat konverzi i sami, pokud máte speciální potřebu. Zajímavou otázkou je *kódování* fontů. Zatímco všichni více či méně souhlasíme s pozicemi 96 znaků ve fontech (základní ASCII sada), zbytek z (běžných) 256 se různí. Nejobvyklejší problémy jsou s plovoucími akcenty a se speciálními znaky, jako je třeba označení libry šterlinků. Existují tři způsoby, co s tím: buď změníte makra TEXu, která odkazují na znaky (nic moc a náchylné na chyby), nebo změníte kódování fontu (jednodušší, než si myslíte), anebo použijete virtuální fonty, (viz Otázku 42) abyste TEXu předstírali, že kódování je totéž, jaké používá on. Používání LATEXu dovoluje změnu kódování v TEXu. Podrobnosti si lze přečíst v *LATEX Companion* (viz Otázku 23). V praxi, pokud často píšete jinak než anglicky (ale latinkou), se silně doporučuje použití `fontenc` balíku s volbou „`T1`“ pro výběr T1 (také známé jako „Cork“ nebo „DC“) kódování. PSNFSS balík (viz Otázku 94) podporuje PostScript (s virtuálními fonty) v tomto kódování.

Balík *fontinst* Alana Jeffreye (`fonts/utilities/fontinst`) je konvertor z `afm` do `tfm` napsaný v TEXu. Používá se pro generování souborů s použitím LATEXovského PSNFSS balíku k podpoře užívání PostScriptových fontů. Jedná se o komplikovaný balík, nic pro slabé nátury, ale je dost silný pro uspokojení většiny potřeb. Mnoho z jeho síly se vztahuje na použití virtuálních fontů (viz Otázku 42).

Pro poněkud jednodušší problémy je použitelný rychlý a efektivní Rokickiho *afm2tfm*, distribuovaný s *dvips* (`dviware/dvips`). Poznamenejme, že metriky a styly, které se dodávají s *dvips*, *nejsou* kompatibilní se současným LATEXem.

Pro MacIntosh existuje program nazvaný *EdMetrics*, který se tímhle (a dalšími věcmi) zabývá. Dodává se společně s distribucí Textures, ale de facto je to volný software, který lze najít na CTANu v `systems/mac/textures/utilities/EdMetrics.sea.hqx`

Uživatelé systému Windows mohou koupit (viz Otázku 63) balík Font Manipulation Tools od Y&Y, který mezi jinými lahůdkami obsahuje silný program *afmtotfm*.

97. Prezeranie súborov využívajúcich Type 1 fonty

Až donedávna vedeli voľne dostupné TEXovské prehliadače zobrazovať iba bitmapové PK fonty. (Komerčný prehliadač firmy Y&Y DVIWindo – viď Otázku 63 – dlhú dobu používal *Adobe Type Manager* na zobrazovanie Type 1 fontov priamo a najnovšie vydania *xdvi* obsahujú renderer Type 1 fontov.) Iné prehliadače súčasnej generácie poskytujú automatické vytváranie potrebných PK súborov (na pozadí používajúc *gsftopk* alebo niečo podobné). Ak to váš prehliadač nedokáže, máte tri možnosti:

1. Konverttovať DVI súbor do PostScriptu a použiť PostScriptový prehliadač. Niektoré systémy poskytujú túto schopnosť štandardne, ale väčšina ľudí bude musieť použiť oddelený prehliadač ako *ghostscript* alebo na *ghostscript*e založené programy ako *ghostview*, prípadne sharewarový *GSview*.
2. Na PC pod Windows, alebo na Macintoshi fonty zobraziť Adobe Type Manager. Textures (Macintosh) pracuje podobne a pod Windows môžete použiť *dviwindo* od Y&Y pre náhľad bez bitmáp (pre detaily týkajúce sa spomenutých dodávateľov viď Otázku 63).
3. Ak máte PostScriptové fonty v Type 1 formáte, použite *ps2pk* alebo *gsftopk* (navrhnutý pre použitie s *ghostscript*ovskými fontami) na vytvorenie PK bitmapových fontov, ktorým bude

váš prehliadač rozumieť. Tento postup môže produkovať skvelé výsledky vhodné tiež na tlač pomocou ne-PostScriptových zariadení. Ak ste si fonty zakúpili, skontrolujte si legálnosť tohto postupu. Najbežnejšie PostScriptovské fonty ako Times a Courier v Type 1 formáte na disku s Adobe Type Managerom (často dodávaným spolu s Windows a čiastočne OS/2).

`ghostscript`: `nonfree/support/ghostscript`
`ghostview`: `support/ghostscript/gnu/ghostview`
`gsftopk`: `fonts/utilities/gsftopk`
`GSview`: `nonfree/support/ghostscript/ghostgum`
`ps2pk`: `fonts/utilities/ps2pk`
`xdvi`: `dviware/xdvi`

98. Problémy s použitím PostScriptových fontů

Před typickým uživatelem LaTeXu, zkoušejícího balík PSNFSS (viz Otázku 94), se často vynoří tři problémy.

První: máte oznámit `dvi` ovladači, že používáte PostScriptové fonty. V případě, že se jedná o *dvips*, znamená to přidání řádku do souboru `psfonts.map`. Jinak *dvips* zkusí najít `pk` soubory. Pokud font není zabudován do tiskárny, musíte jej nějak získat (v mnoha případech to znamená koupi od komerčního distributora).

Druhý: váš prohlížeč musí vědět, co má s fonty dělat (viz Otázku 97.

Třetí: natahování a smršťování (rozpal) mezi slovy je funkcí metriky fontu. Není specifikováno v `afm` souboru, takže rozdílné konvertory vybírají rozdílné hodnoty.

Když nejste pozorní, PostScriptová metrika, kterou přinášel PSNFSS, dělala celkem těsné nastavení, ale v roku 1995 byla revidována, aby produkovala kompromis mezi americkymi a evropskými praktikami. Sofistikovaní uživatelé nemusí být spokojeni ani s novými hodnotami a budou je chtít obejít. Dokonce i běžný uživatel se může střetnout s větším množstvým rozdělených slov nebo přetečených boxů, než produkuje Computer Modern, jenomže CM je extrémně štědrý.

99. Výběr zvětšovatelných obrysových fontů

Pokud se zabýváte samotným textem, můžete použít libovolný z 20 000 fontů(!) ve formátu Adobe Type 1 (zvané PostScriptové fonty ve světě TeXu a ATM fonty ve světě DTP), nebo některý z několika set fontů v TrueType formátu. To samozřejmě za podmínky, že váš prohlížecí a tiskový ovladač podporuje zvětšovatelné obrysové fonty.

TeX sám se stará *jenom* o metriky, nikoliv o skutečný vzhled znaků. Je potřeba pouze vytvořit TeXovský soubor s metrikou `tfm` použitím nějakého prostředku jako *afm2tfm* (případně v kombinaci s *vpt*), *afmtotfm* (od Y&Y, viz Otázku 63), *afm2tfm* (pokud možno v kombinaci s *vptovf*) nebo *fontinst*. Pro prohlížecí nebo tiskový ovladač potřebujete skutečné obrysové fonty (`pfa` pro zobrazení PostScriptu, `pfb` pro ATM na IBM PC, Macovské obrysové fonty na Macintoshi).

Jestliže také potřebujete matematiku, jste poněkud omezeni požadavky, jež má TeX na matematické fonty (detaily viz příspěvek B.K.P. Horna ve sborníku konference TUGu konané v roce 1993 v Astonu, otištěný v TUGboatu, **14**(3), 1994). Pro matematiku je tedy několik možností:

Computer Modern (75 fontů — optické zvětšování) Donald E. Knuth.
Poznamenejme, že CM jsou dostupné ve zvětšovatelné obrysové formě. Tyto fonty byly vytvořeny v METAFONTu. Existují komerční i volně dostupné verze, jsou jak Adobe Type 1, tak TrueType verze. Některé z nich mají „komerční úroveň“ s plně ručně laděným ovládáním, další vyhlížejí velmi uboze, zatímco jiné jsou pouze nekompatibilní s Adobe Type Managerem (ATM). AMS sada těchto fontů je volně dostupná z `fonts/cm/ps-type1/bluesky`

AMS fonty (52 fontů — optické zvětšování) AMS
Tato sada je jakýmsi doplňkem CM sady. Obsahuje dvě sady symbolových fontů (`msam` a `msbm`) a Eulerův textový font. Fonty jsou dostupné v Type 1 verzi na `fonts/amsfonts/ps-type1` (samozřejmě zdarma). Není to samostatná rodina fontů, ale zasluhuje si zde malou diskuzi. Balík *eulervm* dovoluje použít eulerovskou matematickou abecedu spolu s textovými fonty neposkytujícími matematické abecedy (například Adobe Palatino nebo Minion).

Computer Modern Bright (62 fontů — optrické zvětšování) Walter Schmidt

CM Bright je rodina bezpatkových fontů založených na Knuthových CM fontech. Zahrnuje fonty nezbytné pro sazbu matematiky včetně AMS symbolů i textových symbolů různých tvarů. Tato kolekce přichází s vlastní sadou souborů určených k použití v LaTeXu. CM Bright fonty jsou nabízeny firmou MicroPress, Inc. Další detaily (včetně vzorků) naleznete na `http://www.micropress-inc.com/fonts/brmath/brmain.htm`. Volně dostupné verze jsou k dispozici na CTANu — balík *hfbright* pro použití s textem v OT1 kódování a balík *cm-super* pro použití s T1.

Concrete Math (25 fontů — optické zvětšování) Ulrik Vieth

Concrete Math fonty byly odvozeny od řezu Concrete Roman vyvinutého Knuthem. Sada nabízí kolekci matematických kurzív, matematických symbolů, matematických rozšíření fontů a fonty AMS symbolů, které doplňují sadu Concrete tak, že Concrete může zcela nahradit sadu Computer Modern. Protože jsou fonty sady Concrete značně tmavší, jsou vhodné pro tiskárny s menším rozlišením a pro tisk na plakáty či transparenty. Concrete Math fonty jsou (podobně jako Concrete Roman fonty) nabízeny v podobě Type 1 firmou MicroPress na URL `http://www.micropress-inc.com/fonts/ccmath/ccmain.htm`.

BA Math (13 fontů) MicroPress Inc.

BA Math je rodina patkových fontů inspirovaných elegantním a graficky perfektním designem fontů Johna Baskervilla. Tato rodina zahrnuje fonty pro sazbu matematiky v normální i tučné podobě. Také navíc obsahuje všechny fonty různých tvarů v kódování T1 i OT1 i užitečné glyfy fontů v kódování TS1. S těmito fonty jsou také nabízena makra, která je umožňují použít v plain TeXu, LaTeXu i LaTeXu 2.09. Další informace a vzorky těchto fontů naleznete na adrese `http://www.micropress-inc.com/samples/bafonts`.

HV Math (14 fontů) MicroPress Inc.

HV Math je rodina bezpatkových fontů založených na Helvetice. Zahrnuje fonty nezbytné pro matematickou sazbu (matematickou kurzívu, matematické symboly atd.) v normální i tučné podobě. Také navíc obsahuje fonty různých tvarů v kódování T1 i OT1 i užitečné glyfy fontů v kódování TS1. S těmito fonty jsou také nabízena makra, která je umožňují použít v plainTeXu, LaTeXu i LaTeXu 2.09. Bitmapové kopie těchto fontů jsou dostupné zdarma z `fonts/micropress/hvmath`. Chcete-li další informace, tak navštivte stránku `http://www.micropress-inc.com/samples/hfonts`.

Informal Math (7 obrysových fontů) MicroPress Inc.

Informal Math je rodina pestrých fontů volně založených na rodině firmy Adobe – Tekton, která napodobuje ručně psané písmo. Zahrnuje fonty nezbytné pro matematickou sazbu (matematickou kurzívu, matematické symboly atd.) v normální podobě. Také navíc obsahuje fonty různých tvarů v kódování OT1 v kolmé i kosé podobě. S těmito fonty jsou také nabízena makra, která je umožňují použít v plainTeXu, LaTeXu i LaTeXu 2.09. Další informace a vzorky těchto fontů naleznete na adrese `http://www.micropress-inc.com/samples/ifonts`.

Lucida Bright a Lucida New Math (25 fontů) Chuck Bigelow a Kris Holmes.

Lucida je rodina příbuzných fontů včetně patkových, bezpatkových, bezpatkových s pevnou šířkou, kaligrafických, blackletter, faxových, ručně psaných spojovaných Holmesových fontů atd. Nejsou tak „vřetenovité“ jako Computer Modern s velkým x-ovým rozměrem, zahrnují rozsáhlejší sadu matematických symbolů, operátorů, relačních a ohraničovacích znaků, než CM (přes 800 na rozdíl od 384; mimo jiné obsahuje také AMS `msam` a `msbm` sadu symbolů). Plánované „Lucida Bright Expert“ (14 fontů) připojí patkové písmo s pevnou šířkou, další ručně psaný font, malé kapitálky, tučné matematické písmo, stojatou „matematickou kurzívu“ atd. Distribuční sada obsahuje podporu pro `plain` TeX a LaTeX 2.09. Podpora pro LaTeX 2_ε je zajištěna v PSNFSS (viz Otázku 94) díky Sebastianu Rahtzovi. Chcete-li si stáhnout vzorky nebo zjistit další informace o těchto fontech, přečtěte si `http://www.YandY.com/download/chironlb.pdf`.

MathTime 1.1 (3 fonty) TeXplorators (Michael Spivak).

Sada obsahuje matematickou kurzívu, symboly a rozšíření fontů, navržené pro snadnou práci s Times-Roman. Jsou zpravidla používány s písmem Times, Helvetica a Courier (jež jsou rezidentní na mnoha tiskárnách a jsou podporovány některými verzemi PC). Navíc můžete tuto základní sadu doplnit fontem Adobe Times Smallcap a snad i sadou fontů Adobe „Math

Pi", která zahrnuje blackboard bold, blackletter a script. Distribuce obsahuje podporu pro plainTEX a LATEX 2.09 (včetně kódu pro spojení s Adobe Math Pi 2 a Math Pi 6). Podpora pro LATEX 2$_\varepsilon$ je zajištěna v PSNFSS (viz Otázku 94) díky Sebastianu Rahtzovi. Chcete-li si stáhnout vzorky, přečtěte si nejprve `http://www.YandY.com/download/chironmt.pdf`

MathTime Plus (12 fontů) Publish or Perish (Michael Spivak)
: Doplňuje předchozí rodinu o tučné a těžké verze základních matematických fontů právě tak jako o stojatou matematickou „kurzívu", jsou tam také řecká písmena určená pro sazbu základních fyzikálních a matematických výrazů a tučná i regulární podoba scriptových fontů. Obě matematické distribuce zahrnují podporu pro použití v plainTEXu a LATEXu 2.09. Podpora pro LATEX 2$_\varepsilon$ je zajištěna v PSNFSS (viz Otázku 94) díky Franku Mittelbachovi a Davidu Carlislovi. Chcete-li si stáhnout vzorky nebo zjistit další informace o těchto fontech, přečtěte si `http://www.YandY.com/download/mathplus.pdf`.

TM Math (14 fontů) MicroPress Inc.
: TM Math je rodina patkových fontů inspirovaná řezem Times. Zahrnuje fonty nezbytné pro matematickou sazbu (matematickou kurzívu, matematické symboly atd.) v normální i tučné podobě. Také navíc obsahuje fonty různých tvarů v kódování T1 i OT1 i užitečné glyfy fontů v kódování TS1. Bitmapové kopie těchto fontů jsou dostupné zdarma z `fonts/micropress/tmmath`. Chcete-li další informace, tak navštivte stránku: `http://www.micropress-inc.com/samples/tfonts.htm`.

Belleek (3 fonty) Richard Kinch
: Belleek je výsledkem Kinchovych úvah o tom, jak by mohl METAFONT být v budoucnu využíván. Fonty byly publikovány současne jako zdrojové soubory METAFONTu, Type 1 fonty a TrueType fonty. Fonty se chovají jako nevyžádaná náhrada základní sady MathTime (jako příklad, co všechno se dá udělat).
: Dokument načrtující Kinchove myšlenky (od úvah o „intelektuální" nadřazenosti METAFONTy, k hodnocení, proč jeho osvojování probíhá v tak omezeném počtu a co by se s tím dalo dělat) je dostupný na `http://truetex.com/belleek.pdf` (je to dobré čtení, ale vystavuje problém zmíněny v otázce 82 – nepokoušejte se ho přečíst na obrazovce v Adobe Readeru).

`mathpazo` version 1.003 (5 fontů) Diego Puga
: Fonty Pazo Math jsou rodina type 1 fontů vhodných pro sázení matematiky v kombinaci s rodinou textových fontů Palatino. Čtyři nebo pět fontů distribuce jsou matematické abecedy v kolmé verzi a kurzívě, v normální a tučné variantě. Pátý font obsahuje malý výběr znaků typu „blackboard bold" (tabulové tučné), jsou vybrány pro svůj matematický význam. Podpora pod systémem LATEX 2$_\varepsilon$ je dostupna v PSNFSS (viz Otázku 94). Fonty jsou licencovány pod GPL, s povolením používat je v publikovaných dokumentech.

`pxfonts` set version 1.0 (26 fontů) Young Ryu
: Sada `pxfonts` se skládá z:
  - virtuálních textových fontů využívajících Adobe Palatino (nebo URW náhradu užívanou *ghostscript*em) s modifikovanými znaky plus, rovná se a lomítko,
  - matematické abecedy používající times,
  - matematických fontů všech symbolů matematických fontů computer modern (`cmsy`, `cmmi`, `cmex` a řecké písmena `cmr`)
  - matematických fontů všech symbolů odpovídajících AMS fontům (`msam` a `msbm`),
  - dalsích matematických fontů s různými symboly.

  Textové fonty jsou dostupné v kódováních OT1, T1 a LY1. Symboly v kódování TS jsou také dostupné. Fonty sans serif a monospaced dodávané se sadou `txfonts` (viz níže) můžou být používany s fonty`pxfonts`. Sada `txfonts` by měla být nainstalována kdykoliv jsou nainstalovány fonty `pxfonts`. Zahrnuty jsou podpůrné soubory pro LATEX, *dvips* a PDFTEX. Dokumentace (`http://www.tex.ac.uk/tex-archive/fonts/pxfonts/doc/pxfontsdocA4.pdf`) je dostupná pro čtení.
: Jsou licencovány pod GPL s povolením používat je v publikovaných dokumentech.

PA Math PA Math je rodina serif fontů volně založených na řezu písma Palatino (TM). PA Math se skládá z fontů potřebných pro sázení matematiky (matematická kurzíva, matematika, kaligrafické a starodávné symboly a rozšíření) v normální a tučné variantě. Rahrnuje taky všechny textové fonty různých tvarů v kódování OT1, T1, jakožto i nejužitečnější glyfy kódování TS1.

Poskytovány jsou makra pro použití fontů v `plain` TEXu, LATEX 2.09 a současném LATEXu.

Por podrobnosti (a vzorky) viz

`http://www.micropress-inc.com/fonts/pamath/pamain.htm`

txfont sada verze 1.0 (22 fontů) Young Ryu

`txfont` sada se skládá z:

- virtuálních fontů používajících Adobe Times (nebo URW používaných *Ghostscriptem*) s modifikovanými znaky „+“, „/“ a „=“.
- matematickou abecedu využívající Times
- matematické fonty všech symbolů v moderních matematických fontech (`cmsy`, `cmmi`, `cmex` a řecká písmena `cmr`)
- matematickou podobu všech fontů odpovídající AMS fontům (`msam` a `msbm`)
- matematickou podobu různých symbolů

Textové fonty jsou dostupné v kódování OT1, T1 i TS. S těmmito fonty jsou nabízeny soubory podporující jejich použité v *dvips*, LATEXu a PDFTEXu. Dokumentaci naleznete v `http://www.tex.ac.uk/tex-archive/fonts/txfonts/doc/txfontsdocA4.pdf`. Fonty jsou dostupné pod LPPL a můžete je získat z `fonts/txfonts`.

Adobe Lucida, LucidaSans a LucidaMath (12 fontů)

Lucida a LucidaMath jsou obecně považovány za poněkud tučné. Tři matematické fonty obsahují pouze glyfy z matematické kurzívy CM, symboly a rozšířené fonty. Podpora pro použití LucidaMath v TEXu není moc dobrá, musíte si dát práci s překódováním fontů atd. (V jistém smyslu je tahle sada předchůdcem fontové sady LucidaBright a LucidaNewMath.)

Concrete, AMS matematické fonty atd. Donald E. Knuth a AMS.

Někdy jsou zmiňovány jako alternativa k CM, ale ve skutečnosti jsou doplňkem, poněvadž je třeba s nimi použít alespoň základní matematické fonty CM.

Proprietary fonts Různé zdroje.

Jelikož vysoce kvalitní sady fontů v obrysovém formátu fungujícím v TEXu poskytují vydavatelům skutečně konkurenceschopné dokumenty, zaplatili si mnozí hodně peněz za výrobu takových fontů na zakázku. Naneštěstí, tyto sady nejsou k dostání volně na trhu, ačkoli jsou pravděpodobně mnohem úplnější než sady jiné.

Mathptm (4 fonty) Alan Jeffrey.

Sada obsahuje matematickou kurzívu, symboly, rozšířené a virtuální Roman fonty, vytvořené z fontů Adobe Times, Symbol, Zapf Chancery a Computer Modern. Mathptm jsou volně dostupné a výsledné PostScriptové soubory mohou být volně sdíleny. Obsahují mnoho CM matematických symbolů. Podpora pro LATEX 2_ε v PSNFSS (viz Otázku 94) je díky Alanu Jeffreymu and Sebastianu Rahtzovi.

Některé sady fontů jsou dostupné ve formátech vhodných pro IBM PC/Windows, Macintosh a Unix/NeXT od Y&Y a od Blue Sky Research (detaily viz Otázku 63). Fonty MathTime lze také získat od:

TEXplorators, 1572 West Gray #377, Houston TX 77019, USA.

Velmi omezený výběr matematických fontů je přímým důsledkem toho, že musí být navrženy pro použití v TEXu, čímž ztrácí něco ze své přitažlivosti pro jiné trhy. Navíc, trh s komerčními fonty pro TEX je nepatrný (pro porovnání, například balíků fontů #1 Microsoft TrueType se prodalo asi 10 miliónů kopií během několika týdnů po vydání Windows 3.1!).

Textové fonty formátu Type 1 jsou dostupné od mnoha prodejců včetně Adobe, Monotype, Bitstream. Vyhněte se levným produktům neznámých výrobců: nejen že byste podporovali jejich neetické chování a poškozovali skutečné tvůrce písem, ale také velmi pravděpodobně byste pořídili zmetek. Fonty se nemusí dobře zarovnávat, nemusí mít standardní doplněk 228 glyfů, anebo třeba neobsahují soubory s metrikami (je třeba udělat TFM soubory). Také se vyhněte TrueType fontům ode všech možných prodejců kromě velkých.

TrueType fonty jsou o řád obtížněji „hintovatelné“ než fonty Type 1, a proto TrueType fonty odjinud než od Microsoft a Apple mohou být podezřelé. Navíc můžete narazit ještě na další problémy s TrueType fonty, třeba, že servisní střediska s nimi odmítají pracovat.

TrueType je nativním formátem pro Windows. Některé implementace TEXu jako TrueTEX (viz Otázku 63) používají TrueType verze Computer Modern a Times Maths fontů na vykreslení dokumentů TEX ve Windows dalšího systémového software jako ATM.

Při výběru fontů nemůže být vaším jediným kritériem prostředí vášho systému. Když budete někomu posílat hotové dokumenty na další použití, měli by ste zvážit zda použitý formát fontu nezpůsobí problémy s kompatibilitou. Vydavatelé můžou vyžadovat jenom TrueType fonty, protože jejich systémy jsou založeny na Windows, nebo jenom Type 1 fonty, protože jsou založeny na raní popularitě formatu ve vydavatelském průmyslu. Mnohým na tom však nezáleží, pokud je obdaříte hotovým tiskovým souborem (PostScript nebo PDF) pro jejich zařízení.

100. Zvláštne znaky vo výstupe *dvips*

Vygenerovali ste výstup programu *dvips* a objavili sa v ňom zvláštne zámeny (napríklad ligatúra `fi` sa objavila akosymbol £)? Toto je nechcený vedľajší efekt opatrenia pre generovanie PostScriptu pre PDF vysvetleného v Otázke 82. Prepínač `-G1` diskutovaný v spomenutej otázke je vhodný pre Knuthove textové fonty, ale nefunguje s textovými fontami, ktoré nedodržujú Knuthove vzory (ako fonty od Adobe).

Ak sa vyskytne problém, zdržte sa prepínaču `-G1`. Ak ste ho explicitne používali, *prestante*! Ak ste používali `-Ppdf`, pridajte `-G0` na potlačenie implicitného prepínaču v súbore pseudotlačiarne.

Problém bol opravený v *dvips* v 5.90 (verzia distribuovaná s CD-ROMom TEX-Live 7 a v iných novších distribúciach TEXu).

101. Používanie fontov „Concrete“

Fonty Concrete Roman (`fonts/concrete`) boli navrhnuté Donom Knuthom pre knihu „Concrete Mathematics“, ktorú napísal spoločne s Grahamom a Patashnikom (*tým* Patashnikom známym z BIBTEXu). Kedže kniha pre matematiku používala fonty Euler, Knuth navrhol iba fonty textové. Kniha bola vysádzaná `plain` TEXom s dodatočnými makrami, ktoré si môžete prezrieť v súbore `gkpmac.tex` (CTAN: `systems/knuth/local/lib/gkpmac.tex`). O pár rokov neskôr navrhol Ulrik Vieth fonty Concrete Math. Balíky *beton* (`macros/latex/contrib/beton`), *concmath* (`macros/latex/contrib/concmath`; `fonts/concmath`), *ccfonts* (`macros/latex/contrib/ccfonts`) sú LATEXovskými balíkmi meniacimi štandardné textové fonty z Computer Modern na Concrete. Balíky *beton* a *ccfonts* tiež mierne zvyšujú štandardnú hodnotu `\baselineskip`, aby sa vysporiadali s vyššou váhou fontov Concrete. Balíky *concmath* a *ccfonts* naviac zmenia štandardné fonty pre matematiku z Computer Modern na Concrete a používajú Concrete verzie AMS fontov (toto chovanie je v prípade balíku *concmath* voliteľné).

Neexistujú žiadne tučné verzie Concrete fontov, obecne sa ale uznáva názor, že fonty Computer Modern Sans Serif demibold condensed predstavujú adekvátnu náhradu. Ak používate balík *concmath* alebo *ccfonts* a chcete sa tejto rady držat, používajte balík s voľbou triedy `boldsans` (napriek tomu, že dokumentácia *concmath* ju nazýva voľba triedy `sansbold`). Ak používate *beton*, pridajte `\renewcommand{\bfdefault }{sbc}` do preambuly vášho dokumentu.

Sú dostupné aj Type 1 verzie fontov. Pre kódovanie OT1 sú dostupné od firmyMicroPress (viď Otázku 99). CM-Super fonty (viď Otázku 301 - `fonts/ps-type1/cm-super`) obsahujú Type 1 verzie fontov Concrete v kódovaní T1.

102. Používanie fontov Latin Modern

Nové (leto 2003) fonty *lm* (`fonts/ps-type1/lm`) sú skvelým doplnkom výzbroje používateľa TEXu/LATEXu (vysoko kvalitné obrysy fontov, ktoré boli až doposiaľ iba ťažko dostupné, v kompaktnom balíku, ktorý voľne dostupný je). Avšak sparťanský informačný súbor dodávaný s fontami upozorňuje: „Predpokladá sa, že prípadný použvateľ vie, čo s týmito súbormi robiť.“ Táto odpoveď má za úlohu informovať o požadovaných úkonoch. Postup nie je príliš zložitý.

Súbory fontu (a príbuzné súbory) sa objavujú na CTANe ako sada podstromov TDS (viď Otázku 48) — `fonts`, `dvips`, `tex` a `doc`. Podstrom `doc` nemusí byť skopírovaný (v skutočnosti je to pár vzorových súborov). Ostatné tri skopírujte do vášho existujúceho lokálneho `$TEXMF` stromu a aktualizujte databázu mien súborov (viď Otázku 52).

Teraz je treba začleniť fonty do sady hľadanej programami PDFLATEX, *dvips*, *dvipdfm*, vaším prehliadačom a konverznými programami Type 1/PK nasledovne:

- Na systémoch teTEX pred verziou 2.0 upravte súbor `$TEXMF/dvips/config/updmap` a vložte absolútnu cestu k súboru `lm.map` hneď za riadok začínajúci `extra_modules="` (a pred uzatvárajúce úvodzovky).
- Na systémoch teTEX verzie 2.0 (alebo neskoršej) spustite príkaz
  `updmap --enable Map lm.map`
- Na systéme MikTEX pred verziou 2.2 operácia „obnovenie databázy názvov súborov“, ktorú ste vykonali po inštalácii súborov, taktiež aktualizuje systémovú „databázu postscriptových zdrojov“.
- Na systéme MikTEX verzie 2.2 alebo vyššej aktualizujte súbor `updmap.cfg`, ktorý je popísaný v online dokumentácii MikTEXu (`http://www.miktex.org/2.2/mkfntmap.html`). Potom spustite program *MkFntMap* a všetko je hotové.

Pre používanie fontov v LATEXovskom programe by ste mali používať príkaz

```
\usepackage{lmodern}%
```

Tým sa stanú fonty štandardnými pre všetky tri rodiny fontov LATEXu („roman“, „sans-serif“ a „typewriter“). Taktiež budete potrebovať príkaz

```
\usepackage[T1]{fontenc} %
```

pre text a

```
\usepackage{textcomp}%
```

(ak chcete používať akýkoľvek symbol v TS1 kódovaní).

103. Vkládání PostScriptových obrázků v LATEXu

LATEX 2_ε (viz Otázku 294) obsahuje standardní balík umožňující včlenění grafiky, otáčení, práci s barvami a další rysy závislé na výstupních ovladačích. Balík je zdokumentován ve druhém vydání Lamportova manuálu, stejně tak i v *LATEX Graphic Companion* (viz Otázku 23). Tento LATEXový balík je také distribuován v podobe vhodné pro plainTEX. Dá se získat na CTANu v adresáři `macros/latex/required/graphics`. Jeho distribuce je doplněna dokumentací ve zdrojové i preložéné podobě `grfguide.ps`. Díky tomu si uživatel muže dokumentaci přečíst, aniž by musel balík nejprve nainstalovat.

graphicx — který umožnuje vhodněji změnit velikost obrázku i jinak s ním manipulovat. Obvykle musíte v souboru `.cfg` u těchto balíku nastavit, který DVI procesor používáte, avšak občas lze toto nastavit přímo volbou balíku. Rozsah typů obrázků, které lze použít záleží pouze na systému, který používáte.

Pokud nepoužíváte LATEX 2_ε, asi nejlepší bude použít makro `psfig`, které napsal Trevor Darrell, dostupné v `nonfree/graphics/psfig`

Budete také potřebovat konverzní program z `dvi` do PostScriptu, který podporuje `\special`. Ovladače zmíněné v Otázce 64 to umějí, a přinášejí s sebou verzi `psfig` připravenou pro použití s nimi. Makro `psfig` pracuje nejlépe s Encapsulovaným PostScriptem (EPS). Zejména `psfig` potřebuje, aby soubory obsahovaly BoundingBox (viz Apendix H v *PostScript Language Reference Manual*). Když nemáte EPS soubor, váš život se může zkomplikovat.

Další věcí kolem vkládání PostScriptového obrázku je, že obrázek není součástí `dvi` souboru, ale je včleněn až když použijete konvertor do PostScriptu. Důsledkem toho je, že většina `dvi` prohlížečů ukáže jenom prázdné místo rezervované TEXem pro váš obrázek, nikoliv obrázek samotný.

Príkaz `\special`, který zpracovává pointery na grafické soubory a další informace, muže být pro každý DVI procesor ponekud odlišný (to je také duvod, proc balíku *graphics* musíte ríct, který DVI procesor používáte).

Některé DVI prolížeče se nedokáží vypořádat s příkazy `\special`, které používáte, neboť nejsou jejich vlastní. Modernější z nich (zejména *xdvi*, *yap* nebo *windvi*) umí pomocí dobře nastaveného *ghostscript*u vyrenderovat z obrázku bitmapu tak, že ji lze na obrazovce prohlédnout.

Na CTANu se nacházejí dva dokumenty venující se zpracování obrázku v LATEXu:

- EpsLATEX Keitha Reckdahla, který pokrývá standardní LATEXové možnosti a zabývá se i balíky *subfigure* a *psfrag*
- Anil K. Goel napsal dlouhý dokument podrobně popisující, kterak vkládat obrazce, obrázky a ilustrace do LATEXovských dokumentů. Je dostupný jako `info/figsinltx.ps`

104. Importovanie grafiky v *dvips*

Dvips môže tak, ako bolo pôvodne počaté, importovať iba jediný grafický formát: zapuzdrený PostScript (`.eps` súbory, viď 49). *Dvips* si poradí aj s mierne nezvyčajným `.eps` vytvoreným METAPOSTom (viz Otázku 4).

Okrem skutočnosti, že množstvo aplikácii produkujúcich nekorektné EPS je priam deprimujúce, je toto celkom uspokojivé pre fungovanie vektorovej grafiky.

Na vloženie bitmapovej grafiky potrebujete spôsob, ako ju previesť do PostScriptu. Mnohé štandardné nástroje na manipuláciu s obrázkami (ako *convert* od *ImageMagick*u) odvedú pri tvorbe EPS celkom dobrú prácu. Aj keď používatelia *Unix*u by si mali dať pozor na tvrdenia programu *xv*: má tendenciu podvzorkovať vašu bitmapu na rozlíšenie vašej obrazovky.

Celkom dobré sú aj špeciálne aplikácie *jpeg2ps* (konvertuje JPEG súbory pomocou funkcionality PostScript level 2) a *bmeps* (konvertuje JPEG a PNG súbory). *Bmeps* obsahuje patche na vytvorenie vašej vlastnej verzie *dvips*, ktorá si poradí so súbormi JPEG a PNG priamo, použitím konverznej knižnice *bmeps*. Aplikácie môžete nájsť na CTANe: `nonfree/support/jpeg2ps`, `support/bmeps`.

105. Importovaná grafika v PDFLaTeXu

Sám PDFTeX podporuje široký zoznam formátov, ktoré vie natívne zabudovať do svojho výstupného PDF prúdu: JPEG (`.jpg` súbory) pre fotografie a podobné obrázky, PNG súbory pre umelo vytvorené bitmapové obrázky a PDF pre vektorové kresby. Staršie verzie PDFTeXu (pred verziou 1.10a) podporovali formát TIFF (`.tif` súbory) ako alternatívu k súborom PNG.

Štandardný balík PDFLaTeXu `graphics` okrem toho zaisťuje nahratie `supp-pdf` makier Hansa Hagena. Tieto makrá vedia prekladať výstup METAPOSTu do PDF za behu, teda výstup METAPOSTu (`.mps` súbory) tiež môže byť vložený do dokumentov PDFLaTeXu.

Najčastejším problémom, s ktorým sa stretávajú používatelia pri prechode z TeXu, je neexistencia priamej cesty pre vloženie EPS súborov. Kedže PDFTeX je svojím vlastným „ovládačom“ a neobsahuje spôsob konvertovania PostScriptu do PDF, neexistuje priamy spôsob, ktorým by sa to dalo spraviť.

Jednoduchým riešením je konverzia EPS do príslušného PDF súboru. Toto za vás spraví program *epstopdf* (`support/epstopdf`, `macros/latex/contrib/oberdiek`). Je dostupný ako spustiteľný súbor pre Windows alebo skript pre *Perl* pod Unix a iné podobné systémy. LaTeXovský balík `epstopdf` sa dá použiť na generovanie potrebných PDF súborov za behu. Je to pohodlný spôsob, ale vyžaduje, aby ste potlačili jednu z bezpečnostných kontrôl TeXu (nepoužívajte ho v súboroch zo zdrojov, ktorým plne nedôverujete).

Alternatívnym (a podľa autora plne uspokojujúcim) riešením je používanie *perl*ovského skriptu *purifyeps* (`support/purifyeps`) používajúceho dobré služby programu *pstoedit* (`support/pstoedit`) a METAPOSTu na konvertovanie vašeho zapuzdreného PostScriptu do „zapuzdreného PostScriptu z METAPOSTu“, takže môže byť vložený priamo.

106. Import grafiky z „iných“ zdrojov

Grafické príkazy ako `\includegraphics` štandardne hľadajú grafické súbory, ktoré chceme, aby používali, všade, kde sa dajú nájsť súbory TeXu. Toto chovanie znižuje flexibilitu, ak sa rozhodnete, že si budete udržiavať grafické súbory v normálnom adresári (mimo vašich zdrojových kódov TeXu/LaTeXu).

Najjednoduchším riešením je zmena cesty k TeXu zmenením štandardnej cesty. Na väčšine systémov sa štandardná cesta berie z premennej prostredia `TEXINPUTS` (ak existuje). Môžete si ju prispôsobiť tak, aby po úprave zahŕňala aj cesty, ktoré obsahovala pred ňou:

```
TEXINPUTS=.:<cesty ku grafike>:
```

na systéme Unix. Windows používa ako oddelovač namiesto znaku '`:`' znak '`;`'. Znak '`.`' zaručuje, že aktuálny adresár je prehľadávaný ako prvý, posledný znak '`:`' zasa vloženie hodnoty `TEXINPUTS` z konfiguračného súboru.

Táto metóda je efektívna (TeX/LaTeX vykonáva *všetky* prehľadávania, čo je rýchle), ale je to stále nešikovné a môže to byť (aspoň vo Windows) nepohodlné.

Alternatívou je použitie príkazu `\graphicspath` balíku `graphics`. Tento príkaz je samozrejme dostupný aj používateľom balíku `graphicx` a `epsfig`. Syntax jediného parametru `\graphicspath` je trochu zvláštna. Je to postupnosť ciest (typicky relatívnych), z ktorých každá je uzavretá v zátvorkách. Mierne zvláštny príklad je uvedený v dokumentácii k balíku `graphics`:

```
\graphicspath{{eps/}{tiff/}}
```

Ak to bezpečnostné kontroly vášho TEX/LATEX systému dovoľujú, cesta môže byť ľubovolná (nie iba striktne relatívna ako hore uvedené).

`\graphicspath` neovplyvní operácie grafických makier iné ako tie z grafického balíku — konkrétne makrá zastaralého balíku `epsf` a `psfig` sú imúnne.

Nevýhoda medódy `\graphicspath` je neefektívnosť. Balík bude volať TEX pre každú položku, čo samo o sebe spomaľuje. Vážnejší problém však je skutočnosť, že TEX si pamätá názov súboru, čím stráca pamäť vždy, keď vyhľadáva súbor. Dokument používajúci veľa grafických vstupov teda môže trpieť nedostatkom pamäte.

Ak je váš dokument rozdelený do viacerých adresárov a každý adresár má príslušnú grafiku, balík `import` (`macros/latex/contrib/misc/import.sty`) vám môže pomôcť (viď diskusiu v Otázke 193). `graphics`: `macros/latex/required/graphics`

107. Prenosná importovaná grafika

Častou potrebou je dokument vo viacerých formátoch (obvykle PostScript a PDF). Nasledujúce rady sú založené na príspevkoch niekoho so skúsenosťami s problémom zaobchádzania s EPS grafikou.

- Pri nahrávaní vašej verzie `graphics` nešpecifikujte ovládač. Schéma sa spolieha na schopnosť distribúcie rozhodnúť, ktorý ovládač sa použije. Voľba je v tomto prípade medzi *dvips* a PDFTEXom. Určite nepoužívajte voľby `dvips`, `pdftex` a `dvipdfm` (*dvipdfm* sa v tejto schéme nepoužíva, ale program na tvorbu PDF ho môže používať pre svoj výstup pred prepnutím na schému).
- Použite `\includegraphics[...]{filename}` bez uvedenia prípony (ani `.eps`, ani `.pdf`).
- Pre každý `.eps` súbor, čo budete vkladať, vytvorte `.pdf` verziu podľa postupu popísaného v otázke 105. Keď toto spravíte, budete mať z každej grafiky dve kópie (`.eps` a `.pdf` súbor) vo vašom adresári.
- Použite PDFLATEX (namiesto LATEX–*dvips*–destilácia alebo LATEX–*dvipdfm*) na tvorbu PDF výstupu.

Kúzlo *Dvipdfm* je teraz menej príťažlivé, samotný dokument musí byť pozmenený zo svojho štandardného (*dvips*) stavu predtým, než ho spracuje *dvipdfm*.

108. Obmedzenie šírky importovanej grafiky

Predpokladajme, že máte grafiku, ktorej nemusí stačiť šírka strany. Ak sa na stranu zmestí, chcete ju mať v pôvodnej veľkosti, ale inak ju chcete zmenšiť tak, aby sa na stránku zmestila.

Toto dosiahnete pátraním vo vnútornostiach grafického balíku (čo samozrejme vyžaduje trochu programovania vnútorností LATEXu):

```
\makeatletter
\def\maxwidth{%
  \ifdim\Gin@nat@width>\linewidth
    \linewidth
  \else
    \Gin@nat@width
  \fi
}
\makeatother
```

Toto definuje „premenlivú“ šírku, ktorá má vlastnosti, aké chcete. Ak máte na šírku grafiky iné obmedzenia, vymeňte príkaz `\linewidth`. Príkazy použite nasledovne:

```
\includegraphics[width=\maxwidth]{figure}
```

109. Importovaná grafika zarovnaná nahor

Keď TEX spracováva akýkoľvek riadok, uistí sa, že účiarie každého objektu na riadku je na rovnakej úrovni ako účiarie finálneho objektu (samozrejme okrem `\raisebox` príkazov...).

Väčšina importovanej grafiky má svoje účiarie nastavené na spodnú stranu obrázku. Pri používaní balíkov ako *subfig* je často požadované zarovnanie obrázkov nahor. Tento kúsok zvláštneho kódu sa o to postará:

```
\vtop{%
  \vskip0pt
  \hbox{%
    \includegraphics{figure}%
  }%
}
```

Primitívum `\vtop` nastavuje účiarie výsledného objektu na jeho „prvý riadok“. `\vskip` vytvára klam prázdneho riadku, takže `\vtop` nastaví účiarie na horný okraj boxu.

V prípadoch, kde sa má grafika zarovnať s textom, existuje spôsob ako znížiť účiarie o výšku „x“ pod horný okraj boxu:

```
\vtop{%
  \vskip-1ex
  \hbox{%
    \includegraphics{figure}%
  }%
}
```

V skutočnosti sa môžete sami rozhodnúť kde bude účiarie. Táto odpoveď vám iba ukazuje rozumné rozhodnutia, ktoré môžete uskutočniť.

110. Zobrazovanie výstupu METAPOSTu v *ghostscript*e

METAPOST obvykle očakáva, že jeho výstup bude zahrnutý v kontexte, kde budú „štandardné“ fonty METAFONTu (ktoré ste špecifikovali) už definované, napríklad ako obrázok v TEXovskom dokumente. Ak ladíte váš METAPOSTovský kód, možno si ho budete chcieť prezrieť v *ghostscript*e (alebo inom prehliadači PostScriptu). Postscriptový „stroj“ v *ghostscript*e však obvykle *nemá* nahrané fonty, takže sa stretnete s chybami ako:

```
Error: /undefined in cmmi10
```

V METAPOST existuje opatrenie pre vyhnutie sa tomuto problému. Na začiatku `.mp` súboru zadajte príkaz `prologues := 2;`.

Bohužiaľ PostScript, ktorý METAPOST vkladá do svojho výstupu za týmto príkazom, nie je kompatibilný s bežným použitím PostScriptu vo vloženiach do dokumentov TEXu/LATEXu, takže najlepšie je spraviť príkaz `prologues` voliteľným. METAPOST naviac používa prostoduchý prístup ku kódovaniu fontov. Kedže TEXovské kódovania fontov obvykle mätú aj sofistikované mysle, toto môže byť problematické. Ak trpíte takýmito problémami (symptómom je zmiznutie alebo zlá interpretácia znakov), jediným riešením je prehliadnutie „pôvodného“ metapostovského výstupu po spracovaní LATEXom a *dvips*.

Podmienená kompilácia môže byť zabezpečená vložením `MyFigure.mp` nepriamo jednoduchým wrapperom `MyFigureDisplay.mp`:

```
prologues := 2;
input MyFigure
```

alebo zadaním príkazu shellu ako

```
mp '\prologues:=2; input MyFigure'
```

(čo v prípade, že nepoužívate unixový shell, bude fungovať bez apostrofov).

Vhodná cesta pre LATEX by zahŕňala spracovanie súboru `MyFigure.tex`, ktorý obsahuje:

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
\thispagestyle{empty}
```

```
\includegraphics{MyFigure.1}
\end{document}
```

Spracovanie výsledného DVI súboru príkazom *dvips* command

```
dvips -E -o MyFigure.eps MyFigure
```

by potom poskytlo uspokojivý zapuzdrený postscriptový súbor. Táto procedúra môže byť zautomatizovaná perlovským skriptom *mps2eps* (`support/mps2eps`), čím by sme si ušetrili určité množstvo nudy.

Používateľ `plain` TEXu môžu použiť úpravu od Dana Lueckinga. Danova verzia `mpsproof.tex` (`graphics/metapost/contrib/misc/mpsproof.tex`) bude fungovať pod TEXom, aby vyprodukovala DVI súbor pre použitie s *dvips* alebo pod PDFTEXom, aby vyrobila PDF súbor priamo. Výstup je nastavený, aby vyzeral ako kontrolný hárok.

Skriptová aplikácia *mptopdf* je dostupná v každej novšej distribúcii TEXu/LATEXu.

111. Zrcadlově převrácený tisk

Pro offsetový tisk bývá často potřeba připravit zrcadlově převrácené předlohy. Postscriptovým řešením tohoto problému je napsat na začátek dokumentu v plain TEXu (nebo do preambule v LATEXu) řádek

```
\special{!userdict begin /bop-hook{hsize 0 translate -1 1 scale} def end}
```

Tím dosáhneme zrcadlového otočení kolem svislé osy.

Pokud chceme dokument převracet okolo vodorovné osy, vypadá řádek takto:

```
\special{!userdict begin /bop-hook{0 vsize translate 1 -1 scale} def end}
```

Komplexnější řešení (s možností zvětšování/zmenšování textu) poskytuje soubor `ftp://ftp.cstug.cz/pub/local/cstug/sojka/mirrorps/mirror.ps`, který můžete (po úpravě le svých potřeb) zařadit na začátek svého postscriptového dokumentu.

Pro častější používání je však vhodné mít program, který umí některé (nebo všechny) stránky postscriptového souboru zrcadlově převrátit, případně s nimi provést jiné úpravy. Univerzálním programem pro konverzi poscriptových souborů je *pstops* z balíku *psutils*. Program *pstops* umí otáčet jednotlivé stránky. Stanislav Brabec vytvořil záplatu, po jejíž aplikaci se *pstops* naučí stránky i zrcadlově převracet okolo horizontální či vertikální osy. Záplata je na `ftp://ftp.penguin.cz/pub/users/utx/psutils/psutils-flip.diff`

Pokud používáte emTEX s jeho ovladači (*dviscr*, *dvihplj*), lze jimi docílit zrcadlového převrácení i bez PostScriptu. Například převrácení kolem vertikální osy stránky zajistí volba `/tr7`.

M. Bibliografie

112. BIBTEX nerozumí mému seznamu jmen

BIBTEX má striktně vymezenou syntax pro psaní jmen autorů (nebo vydavatelů) v BIBTEXovém souboru; pokud tento seznam píšete ve zdánlivě přirozeném tvaru, pak je zde možnost, že tím BIBTEX zmatete a výsledek bude dost odlišný od toho, co jste očekávali. Jména se zadávají v jednom z následujících tvarů:

```
Jméno Příjmení
Příjmení, Jméno
Příjmení, Suffix, Jméno
```

a seznam autorů může být oddělen slovem „and“. Např. výraz

```
AUTHOR={Fred Q. Bloggs, John P. Doe & Fairbairns}
```

porušuje hned dvě pravidla: syntakticky významná čárka se vyskytuje na nesprávném místě a výraz „&“ zde nezastává funkci oddělovače jmen, jak by se mohlo zdát. Výstup výše uvedeného by mohl vypadat takto „John P. Doe & Robin Fairbairns Fred Q. Bloggs“, protože „John P. Doe & Robin Fairbairns“ se stává křestním jménem, zatímco „Fred Q. Bloggs“ je příjmením. Tento příklad by měl být syntakticky zapsán takto:

```
AUTHOR = {Fred Q. Bloggs and John P. Doe and Fairbairns}
```

Některé bibliografické styly s dlouhým seznamem autorů dokáží dělat divy. Násilně můžete seznam zkrátit tím, že použijete pseudojméno „others". Chce-li pan Bloggs ve výpisu autorů potlačit jména svých kolegů, stačí, aby do BIBTEXového souboru napsal:

```
AUTHOR = {Fred Q. Bloggs and others}
```

113. Lze zpracovat bibliografii pro každou kapitolu zvlášť

Bibliografie pro každou kapitolu zvlášť může být zpracována pomocí balíku *chapterbib*. Vhodnější je napsat tuto bibliografii do zvláštního externího souboru než přímo do zdrojového souboru k jednotlivým kapitolám dokumentu. Balíček *bibunits*, sváže bibliografii podle logických částí dokumentu. Tento balík pracuje se sekcemi a kapitolami (jak je definujete v LATEXu) a také umožňuje definovat zvláštní prostředí, kde si uživatel může zvolit svoji vlastní strukturu dokumentu. Výše zmiňované balíky naleznete na těchto místech:
`chapterbib`: `macros/latex/contrib/cite`
`bibunits`: `macros/latex/contrib/bibunits`

114. Viacnásobné bibliografie

Ak máte na mysli mnohonásobné bibliografie previazané s rôznymi časťami vašeho dokumentu (napríklad s rôznymi kapitolami), pozrite si odpoveď na Otázku 113.

Pre viac než jednu bibliografiu sú tri možnosti.

Balík `multibbl` poskytuje veľmi jednoduché rozhranie: použijete príkaz `\newbibliography` na definíciu „tagu" (značky) bibliografie. Balík znovu definuje iné príkazy bibliografie tak, aby ste pri každom použití ľubovolného z nich príkazu zadali značku bibliografie, kde chcete, aby sa citácie objavili. Príkaz `\bibliography` tiež akceptuje ďalší extra parameter, ktorý určuje, aký nadpis použiť pre výslednú sekciu či kapitolu (t.j. upravuje `\refname` a `\bibname` — viď Otázku 217 — spôsobom bezpečným pre `babel`). Takže môžete napísať:

```
\usepackage{multibbl}
\newbibliography{bk}
\bibliographystyle{bk}{alpha}
\newbibliography{art}
\bibliographystyle{art}{plain}
...
\cite[pp.~23--25]{bk}{milne:pooh-corner}
...
\cite{art}{einstein:1905}
...
\bibliography{bk}{book-bib}%
             {References to books}
\bibliography{art}{art-bib}%
             {References to articles}
```

(Všimnite si, že voliteľný parameter príkazu `\cite` je umietnený *pred* parametrom novej značky a príkaz `\bibliography` môže obsahovať viac než jeden `.bib` súbor — skutočne všetky príkazy `\bibliography` môžu obsahovať rovnaký zoznam súborov.)

Údaje príkazu `\bibliography` smerujú do súborov s menami <názov-značky>emph.aux, takže budete musieť po prvom spustení LATEXu spustiť

```
bibtex bk
bibtex art
```

aby ste dostali citácie na správne miesto.

Balík `multibib` dovoľuje definovať sériu „dodatočných tém", z ktorých každá má vlastnú sériu bibliografických príkazov. Môžete teda napísať:

```
\usepackage{multibib}
\newcites{bk,art}%
         {Odvolávky na knihy,%
          Odvolávky na články}
```

```
\bibliographystylebk{alpha}
\bibliographystyleart{plain}
...
\citebk[pp.~23--25]{milne:pooh-corner}
...
\citeart{einstein:1905}
...
\bibliographybk{book-bib}
\bibliographyart{art-bib}
```

Znova platí, že každý príkaz `\bibliography...` môže obsahovať ľubovolný zoznam `.bib` súborov.

BibTeXovské spracovanie s `multibib` je veľmi podobné spracovaniu s `multibbl`. S predchádzajúcim príkladom potrebujete:

```
bibtex bk
bibtex art
```

Všimnite si, že (na rozdiel od `multibbl`) `multibib` dovoľuje jednoduchú, nemodifikovanú bibliografiu (ako aj „témy").

Balík `bibtopic` umnožňuje oddelene citovať rozličné bibliografie. Na vhodné miesto dokumentu vložte sekvenciu prostredí `btSect` (z ktorých každé udáva databázu bibliografie, ktorá sa má preskúmať) pre sadzbu oddelených bibliografií. Môžete teda mať súbor `diss.tex` obsahujúci:

```
\usepackage{bibtopic}
\bibliographystyle{alpha}
...
\cite[pp.~23--25]{milne:pooh-corner}
...
\cite{einstein:1905}
...
\begin{btSect}{book-bib}
\section{Odvolávky na~knihy}
\btPrintCited
\end{btSect}
\begin{btSect}[plain]{art-bib}
\section{Odvolávky na~články}
\btPrintCited
\end{btSect}
```

Všimnite si rozličný spôsob špecifikovania štýlu bibliografie: ak chcete iný štýl pre konkrétnu bibliografiu, môžete ho zadať ako voliteľný parameter prostredia `btSect`.

Spracovanie BibTeXom v tomto prípade používa `.aux` súbory, ktorých názvy sú odvodené z názvu základného dokumentu. Takže v tomto príklade potrebujete:

```
bibtex diss1
bibtex diss2
```

Existuje aj príkaz `\btPrintNotCited`, ktorý poskytuje zbytok obsahu databázy (ak nebolo z databázy nič citované, je tento príkaz ekvivalentný LaTeXovskému štandardnému `\nocite{*}`).

Skutočný rozdiel oproti `multibbl` a `mltibib` je, že výber toho, čo sa objavuje v každej sekcii bibliografie, je určené v `bibtopic` tým, čo je v `.bib` súboroch.

`bibtopic.sty`: `macros/latex/contrib/bibtopic`
`multibbl.sty`: `nonfree/macros/latex/contrib/multibbl`
`multibib.sty`: `macros/latex/contrib/multibib`

115. Výpis položek bibliografie do textu

Toto je základní požadavek pro časopisy či jiné publikace. Občas je třeba, aby se daná položka objevila v běžném textu dokumentu, ačkoliv ostatní styly požadují, aby se položky objevily jako poznámky pod čarou. Možnosti pro výstup položek v běžném textu jsou tyto:

- Balíček *bibentry* (`macros/latex/contrib/natbib`), který vyžaduje nepatrné omezení na formát položek generovaných vaším souborem `.bst`, ale nepožaduje žádný bibliografický styl.

- Balíček *inlinebib* (`biblio/bibtex/contrib/inlinebib`) požaduje, abyste používali soubor `inlinebib.bst`.
- Balíček *jurabib* (`macros/latex/contrib/jurabib`), který byl původně používán k sazbě německých právních dokumentů umožňující snadnou manipulaci s citacemi.

Pro výpis položek v podobě poznámek pod čarou složí dva balíčky:

- *Footbib* (`macros/latex/contrib/footbib`)
- a opět *jurabib*.

116. Triedenie a kompresia citácií

Ak dáte LATEXu `\cite{fred,joe,harry,min}`, jeho štandardné príkazy vám môžu dať niečo ako „[2,6,4,3]“ (vyzerá to hrozne). Samozrejme, môžete dať veci do poriadku preusporiadaním kľúčov v príkaze `\cite`. Komu sa ale chce robiť niečo podobné iba pre výsledok „[2,3,4,6]“?

Balík `cite` usporiada čísla a deteguje postupnosti po sebe idúcich čísel, čím vytvorí „[2–4,6]“. Balík `natbib` s voľbami `numbers` a `sort&compress` spraví to isté, ak bude pracovať so svojimi vlastnými numerickými štýlmi bibliografie (`plainnat.bst` and `unsrtnat.bst`).

Ak potrebujete spraviť hyperreferencie na vaše citácie, balík `cite` nie je vhodný. Ak pridáte balík `hypernat`:

```
\usepackage[...]{hyperref}
\usepackage[numbers,sort&compress]{natbib}
\usepackage{hypernat}
...
\bibliographystyle{plainnat}
```

balíky `natbib` a `hyperref` budú spolupracovať.

`cite.sty`: `macros/latex/contrib/cite`
`hypernat.sty`: `macros/latex/contrib/misc/hypernat.sty`
`hyperref.sty`: `macros/latex/contrib/hyperref`
`plainnat.bst`: distribuované v `macros/latex/contrib/natbib`
`unsrtnat.bst`: distribuované v `macros/latex/contrib/natbib`

117. Viacnásobné citácie

Konvencia občas používaná vo fyzikálnych žurnáloch je spojenie skupiny príbuzných citácií do jedného záznamu v bibliografii. BIBTEXu štandardne toto opatrenie nestačí, ale balík `mcite` sa s problémom vysporiada.

Balík preťažuje (nahradzuje) príkaz `\cite`, aby rozpoznal „`*`“ na začiatku kľúča, takže citácie tvaru:

```
\cite{paper1,*paper2}
```

sa v dokumente vyskytujú ako jediná citácia a sú príslušne zoradené v bibliografii samotnej. Nie ste limitovaní spojením iba dvoch referencií. Spojené referencie môžete miešať s „normálnymi“:

```
\cite{paper0,paper1,*paper2,paper3}
```

Tieto sa v dokumente objavia ako tri citácie „[4,7,11]“ (povedzme) — citácia „4“ bude odkazovať na paper 0, „7“ na kombinovaný záznam pre paper 1 a paper 2, a „11“ na paper 3.

Budete musieť spraviť malú zmenu súboru bibliografického štýlu (`.bst`), ktorý používate. Dokumentácia balíku `mcite` vám poradí, ako to spraviť.

`mcite.sty`: `macros/latex/contrib/mcite`

118. Triedenie zoznamov citácií

BIBTEX má triediacu funkciu a väčšina BIBTEXovských štýlov triedi zoznam citácií, ktorý produkuje; množstvo ľudí toto považuje za vhodné správanie.

je však možné napísať prostredie `thebibliography`, ktoré *vyzerá*, ako by pochádzalo z BIBTEXu. Veľa ľudí to aj robí (aby v krátkodobom hľadisku ušetrili čas).

Problém sa objaví, keď sa autori `thebibliography` rozhodnú, že ich citácie potrebujú zotriediť. Bežným nedorozumením je vloženie `\bibliographystyle{alpha}` (alebo niečoho podobného) a očakávanie nejakým magickým spôsobom zotriedeného vysádzaného výstupu. BibTeX takto nefunguje! — ak napíšete `thebibliography`, jeho obsah musíte zotriediť *vy*. BibTeX zotriedi obsah prostredia `thebibliography`, iba keď ho vytvára (na vloženie z `.bbl` súboru príkazom `\bibliography` command).

119. Výpis všetkých vašich BibTeXovkých záznamov

LaTeX a BibTeX spolupracujú, aby poskytli tejto požiadavke špeciálne zaobchádzanie. Príkaz `\nocite{*}` je ošetrený špeciálne a spôsobuje, že BibTeX generuje záznamy bibliografie pre každý záznam vo všetkýchh `.bib` súboroch uvedených vo vašom `\bibliography` výraze, takže po postupnosti LaTeX–BibTeX–LaTeX máte dokument s úplným výpisom.

Všimnite si, že LaTeX *ne*produkuje varovania „`Citation ... undefined`“ alebo „`There were undefined references`“ rešpektujúc `\nocite{*}`. Toto nepredstavuje problém, ak používate LaTeX „ručne“ (presne *viete* koľkokrát máte veci spustiť), môžu to ale spliesť automatické procesory prechádzajúce log súbor, aby zistili, či je potrebné ďalšie spustenie.

120. Vytváranie HTML bibliografie

Elegantné riešenie je poskytované bibliografickým štýlom `noTeX`. Tento štýl produkuje `.bbl` súbor, ktorý je v skutočnosti postupnosťou HTML „`P`“ elementov triedy `noTeX`, a preto môže byť vložený do HTML súboru. Pripravené sú opatrenia pre prispôsobenie vašej bibliografie (pri spracovaní prostredníctvom `noTeX` môže byť obsah rozdielny od obsahu spracovaného tradičným spôsobom).

Bežnejším prekladačom je *awk* skript *bbl2html*, ktorý prekladá vami vygenerovaný `.bbl` súbor: príklad výstupu skriptu si môžete pozrieť na webe (`http://rikblok.cjb.net/lib/refs.html`)

`bbl2html.awk`: `biblio/bibtex/utils/bbl2html.awk`

`noTeX.bst`: `biblio/bibtex/utils/noTeX.bst`

121. Vytváření BibTeXových stylů

Opravdu si můžete vytvořit svůj vlastní styl: standardní bibliografické styly jsou distribuované v komentované podobě s popisem jazyka (viz Otázku 27). Nicméně, musíme připustit, že jazyk, v němž se BibTeXové styly vytvářejí, je jaksi neprůhledný. Proto ho nemůžeme doporučit nikomu, kdo není dobrý programátor, byť i menší změny v existujícím stylu se mohou ve výsledku projevit nepříznivě. Pokud váš styl není „přehnaně moderní“, můžete ho pravděpodobně generovat pomocí některého z balíků pro vlastní bibliografii. Ten zahrnuje soubor `makebst.tex` (distribuovaný v `macros/latex/contrib/custom-bib`), který vás vede textovým menu k vytvoření souboru s instrukcemi, pomocí nichž můžete vytvořit vlastní `.bst` soubor. Tato technika však nepojednává o vytvoření nového stylu dokumentu (současný autor potřeboval pro svou dizertaci „standard committee papers“ a „ISO standards“, jiný často požadovaný typ je webovská stránka — viz Otázku 122).

122. Citování URL v BibTeXu

Neexistuje žádný vzor, jak sázet URL ve standardním BibTeXovém stylu, nicméně Oren Patashnik (autor BibTeXu) předpokládá vývoj dlouho očekávaného BibTeXu verze 1.0.

Aktuální informace, které musí být dostupné v citaci URL jsou rozebírány v některých on-line dostupných výtazích normy ISO 690-2 dostupných na adrese `http://www.nlc-bnc.ca/iso/tc46sc9/standard/690-2e.htm`; níže uvedené techniky však neuspokojí všechny požadavky normy ISO 690-2, alepsoň však nabízejí řešení dnešním běžným uživatelům. Do příchodu nové verze je nejjednodušším řešením použití pole `\howpublished` funkce standardního stylu `@misc`. Samozřejmně platí omezení sázení URL, takže položka pak bude vypadat takto:

```
@misc{...,
  ...,
  \howpublished = "\url{http://...}"
}
```

Alternativním přístupem je použití BibTeXovských stylů jiných než standardních, které mají vstupní typy URL. Vynikající jsou styly *natbib* (*plainnat*, *unsrtnat* a *abbrevnat*). Tyto styly jsou rozšířením standardních stylů, pro použití převážně s *natbib*, ale osvojily si URL a jiné moderní položky. *custom-bib* stejného autora také umí generovat styly, které respektují položky URL.

Jiným kandidátem je balík *harvard*. Jaho styly bibliografie obsahují ve své specifikaci pole „`url`“. Poskytnutá sazba je však tak trochu slabá (ikdyž rozpozná použití maker *LaTeX2HTML* na použití odkazů, když jsou makra dostupná).

Můžete také získat nové BibTeXovské styly užitím systému *urlbst* Normana Graye, který je založen na perlovských skriptech editujících existující BibTeXovské styly vytvářejíc nové. Nové styly mají vstupní typ `webpage` a poskytují podporu pro pole `url` a `lastchecked`. Skripty *perl*u jsou dodávány se sadou konvertovaných verzí standardních bibliografických stylů. Dokumentace je ve formě LaTeXovského zdrojového kódu.

Další možností je, že zdroj, z něhož jsme citovali (např. tecnická zpráva), je také dostupný přes Web. V takovém případě se běžně používá tato technika:

```
@techreport{...,
  ...,
  note = "Také dostupný na \url{http://...}"
}
```

V tomto kontextu je také vhodné použít makro `macros/latex/contrib/misc/url.sty` nebo `macros/latex/contrib/hyperref`, protože ignoruje mezery v argumentech. BibTeX obvykle rozděluje řádky, které považuje za příliš dlouhé, a pokud v něm nejsou žádná bílá místa pro přirozené rozdělení, pak BibTeX vkládá znak komentáře („%“), který je v URL přijatelný tak, že ho `\url` vysází. Způsob, jak tento problém obejít, spočívá ve vložení pomocných mezer v URL do souboru `.bib`, aby si BibTeX rozumně vybral místo, kde řádek zalomí. Uvažte, že verze `\url` přicházející s posledními verzemi *hyperref*, netrpí problémem '`%`-end of line': *hyperref* problém zjistí a potlačí nechtěné znaky.

N. Osobitné spôsoby sadzby

123. Nahradenie štandardných tried

Ľudia stále vymýšľajú triedy, ktoré majú nahradiť štandardné: súčasný autor vyrobil v roku 1980 triedu `ukart` používajúcu balík `sober` a pár vecí špecifických pre Britov (ako tie, čo sa objavujú v špecializácii `babel`u pre britskú angličtinu). Trieda sa stále príležitostne používa.

Podobné verejné snahy boli dostupné v dobe LaTeX 2.09: významným príkladom, ktorého potešujúci design sa zrejme časom veľmi nezmenil, je trida `ntgclass`. Každá zo štandardných tried je nahradená výberom tried pomenovaných v holandčine (občas s jednou pridanou číslicou). Takže máme triedy `artikel2`, `rapport1`, `boek3` a `brief`. Tieto dokumenty sú priemerne zdokumentované v angličtine.

Balík tried `KOMA-script` (triedy pomenované `scr...`) je v súčasnosti silným súťažiacim. Triedy sú aktívne podporované, sú obsiahle v pokrytí dôležitostí sadzby, produkujú dobre vyzerajúci výstup a sú dobre dokumentované v angličtine i nemčine (*scrguien* v anglickej distribúcii, *scrguide* v nemeckej).

Ďalšou porovnateľnou triedou je `memoir`. Táto má za cieľ priamo nahradiť triedy `book` a `report`. Podobne ako `KOMA-script` je obsiahla v pokrytí drobný problémov. Dokumentácia triedy `memoir` (*memman*) je často vychvaľovaná a jej dlhý úvod je pravidelne doporučovaný ako tutoriál pre sadzbu.

`balík KOMA-script`: `macros/latex/contrib/koma-script`
`memoir.cls`: `macros/latex/contrib/memoir`
`balík NTGclass`: `macros/latex/contrib/ntgclass`
`sober.sty`: `macros/latex209/contrib/misc/sober.sty`

124. Kreslenie s TEXom

V TEXu/LATEXu samotnom je veľa prostriedkov na kreslenie obrázkov (čo je lepšie ako importovať obrázky externé), od jednoduchých, ako je prostredie `picture`, trochu lepších, ako je balík *epic*, až po výborné (ale pomalé) kreslenie s PICTEXom. V závislosti na type kreslenia by jeden zo štyroch nasledujúcich systémov mohol byť pre Vás špičkový:

1. *pstricks*; umným využívaním príkazu `\special` dáva prístup k plnej sile PostScriptu z TEXu. Potrebujete slušný prekladač z DVI do PostScriptu, ale výsledok stojí za to. Dobre zdokumentovaný balík vám dáva možnosť použiť nielen príkazy nižšej úrovne, ktoré vykresľujú priamky, kruhy či kružnice, ale aj makrá vysokej úrovne na manipuláciu s objektami či textovými rámami, kreslenie stromov, či matíc, pre efekty 3D grafiky a ďalšie. Balík nájdete na `graphics/pstricks`. Používatelia PDFTEXu môžu používať `pdftricks`, ktorý (ako aj `epstopdf` — viď Otázku 105) generuje PDF súbory za behu z príkazov `pstricks`.
2. METAPOST; mali ste radi METAFONT, ale nikdy Vás neuchvátili súbory písiem? Vyskúšajte METAPOST (pozri Otázku 4). Má všetku silu METAFONTu, ale generuje PostScript. METAPOST je v súčasnosti súčasťou najdôležitejších distribúcií TEXu/LATEXu. D. Knuth ho používa pri svojej práci.
3. *Mfpic*; páči sa Vám METAFONT, ale nerozumiete jeho jazyku? Tento balík (`graphics/mfpic`) vytvorí pre Vás kód METAFONTu z jednoduchšie zvládnuteľných TEXovských makier. Nemá všetku silu METAFONTu, ale priateľskejší vzhľad.
4. Páči sa Vám PICTEX, ale nemáte dosť pamäti alebo času? Pozrite sa na makrá v CTAN adresári `graphics/dratex` od Eitana Guarariho, ktoré sú tak silné, ako väčšina ostatných balíkov, ale je to celkom nová implementácia, dobre čitateľná a plne zdokumentovaná.

125. Prostrkávaná sazba

Jednou ze základních metod zvýrazňování textu v dokumentu je změna vzdálenosti mezi písmeny. Tato technika se nazývá prostrkávaná sazba. Toto zvýraznění však není doporučováno, protože ubírá textu na čitelnosti, což je proti principům typografie. Proto Donald Knuth také neumožnil jednoduše měnit vzdálenosti mezi znaky.

Jeden z nejvetších typografů — Eric Gill — toto přirovnal ke „kradení oveček“, což v jeho době byl jeden z nejčastějších přestupků v Británii. Vkládání mezer mezi velká písmena je však považováno za menší „zločin“, než mezi malá písmena. Bežně se ale tato technika používá při zvýraznění textu psaného gotickým (nebo jiným podobným) písmem.

Na CTANu je balík `macros/generic/letterspacing.tex`, v němž je definováno makro `\letterspace`. Používá se podobně jako `\hbox` :

`\letterspace to 1.3\naturalwidth{Důkaz:}`

Lepší balík je `macros/latex/contrib/soul`, který je optimalizován pro použití v LATEXu, ale funguje i v plainTEXu. *Soul* také umožnuje dělení takto zvýraznených slov.

126. Formátovanie diplomových a disertačních prác v LATEXu

Štýl pre dizertáciu je obvykle veľmi špecifický pre danú univerzitu. Preto spravidla nie je príliš užitočné pátrať po takomto štýle mimo rámec univerzity alebo inštitúcie. Mnoho univerzít vo svojej excentricite stále používa pre dizertácie dvojité riadkovanie. Vtedy vás môžeme odkázať na Otázku 146. Ak chcete napísať vlastný štýl pre dizertácie, môže vám poslúžiť ako dobrý štart štýl University of California (dostupné v `macros/latex/contrib/ucthesis`), ale nemá cenu púšťať sa do ťažkostí. (Ak vedenie nepripustí štandardné typografické konvencie, aj tak nebudete schopný vytvoriť esteticky uspokojivý dokument.)

127. Obtekanie obrázkov v LATEXu

Existuje niekoľko LATEXovských balíkov, ktoré majú za cieľ toto umožňovať, ale všetky majú svoje ohraničenia, pretože TEX nebol navrhovaný na riešenie takejto úlohy. Piet van Oostrum, ktorý vytvoril prehľad dostupných balíkov, odporúča:

picins picins.sty je časť veľkého balíku umožňujúceho zahrnutie obrázkov (napr. so šedými boxami, rôznymi DOS formátmi, …). Nachádza sa v systems/msdos/picins/picins.zip. Príslušný príkaz je:

\parpic(*šírka*, *výška*)(*x-off*, *y-off*)[*Options*][*Pozícia*]{*Obrázok*}

Text odstavca

Všetky parametre okrem *Obrázok* sú voliteľné. Obrázok môže byť umiestnený vľavo alebo vpravo, orámovaný obdĺžnikom, oválom alebo čiarkovane. Môže mať popis (\caption), ktorý bude zahrnutý do zoznamu obrázkov.

Nanešťastie (pre tých, ktorí nerozumejú nemecky) je dokumentácia v nemeckom jazyku. Piet van Oostrum však napísal anglické resumé (macros/latex209/contrib/picins/picins.txt).

floatflt macros/latex/contrib/floatflt je zdokonalená verzia floatfig.sty (pre LaTeX 2_ε), a jeho syntax je:

\begin{floatingfigure}[*options*]{*šírka obrázku*}
  Obrázok
\end{floatingfigure}

Existuje aj (viacmenej podobné) prostredie floatingtable.

Tabuľky alebo obrázky môžu byť dané na pravú/ľavú časť strany a párnu/nepárnu stranu pri dokumentoch, ktoré rozlišujú párne a nepárne strany.

Tento balík umožňuje použiť aj balík *multicol*, ale nespolupracuje s ďalšími blízkymi prostrediami.

wrapfig macros/latex/contrib/wrapfig má syntax:

\begin{wrapfigure}[*výška obrázku v riadkoch*]{l,r,*atď*}[*presah*]{*šírka*}
  obrázok, popis obrázku atď.
\end{wrapfigure}

Syntax prostredia wraptable je podobná.

Výška môže byť vynechaná. Vtedy je dopočítaná automaticky programom. Program vždy berie väčšiu hodnotu z vypočítanej a udanej hodnoty. Parameter {l,r *atď*} môže byť tiež špecifikovaný ako i (inside – vnútro) alebo o (outside – vonkajšok), pre dokumenty, ktoré rozlišujú párne a nepárne strany. Keď použijeme veľké písmená, obrázok môže byť pohyblivý. Parameter *presah* dovoľuje presah obrázku do okrajov. Obrázok alebo tabuľka môžu byť zahrnuté do zoznamu, keď použijeme príkaz \caption.

Uvedené prostredia nepracujú vnútri zoznamov, ktoré končia pred dolným okrajom obrázku alebo tabuľky, ale môžu byť použité v prostrediach parbox, minipage a to aj pri dvojstĺpcovej sadzbe.

128. Alternatívne head- a footline v LaTeXu

Štandardný LaTeX zahŕňa v sebe niekoľko štýlov, ktoré špecifikujú riadok záhlavia a spodku strany (headline, footline). Možnosti, ktoré máme na zmeny, sú veľmi obmedzené. LaTeX umožňuje veľa, ale nie je jednoduché to urobiť. My to robiť nemusíme, urobil to Piet van Oostrum.

Balík je v adresári macros/latex/contrib/fancyhdr. Tento balík poskytuje jednoduchý mechanizmus na definovanie rôznych variantov záhlavia a spodnej časti strany. Adresár obsahuje aj celkom dobrú dokumentáciu a ďalší jeden alebo dva balíky. Balík *fancyhdr* umožňuje aj predefinovať štýl strany a úvodnej strany (pozri Otázku 236).

129. Doslovné (verbatim) vložení souboru v LaTeXu

Dobré řešení je použití verbatim.sty od Rainera Schöpfa, které poskytuje příkaz \verbatiminput, který určuje jméno vstupního souboru jako argument. Balík je součástí *2etools* (na CTANu v macros/latex/required/tools).

```
\usepackage{verbatim}
...
\verbatiminput{verb.txt}
```

Jiná metoda je použití prostředí alltt, které vyžaduje alltt.sty (ten je standardní částí LaTeXu).

Toto prostředí vkládá obsah souboru „vetšinou“ doslovně, provádí však TEXové příkazy, které v souboru objeví. Pro vložení souboru `verb.txt` stačí napsat:

```
\usepackage{alltt}
...
\begin{alltt}
 \input{verb.txt}
\end{alltt}
```

Jistě, tak málo stačí pro vložení souboru...

Balík *moreverb* (`macros/latex/contrib/moreverb`) rozšiřuje možnosti balíku *verbatim*, nabízí prostředí `listing` a příkaz `\listinginput`, který umožňuje číslovat řádky vkládaného souboru. Balík obsahuje také příkaz `\verbatimtabinput`, který respektuje TAB znaky na vstupu (prostředí `listing` a příkaz `\listinginput` taky respektují TAB znaky).

Balík *fancyvrb* obsahuje konfigurovatelné implementace všeho, co obsahují balíky *moreverb* a *verbatim* a ještě něco navíc. V současné době bystří sazeči sahají po tomto produktu nejraději, ale bohatství možností z něho dělá velice komplexní „potvoru“. Z tohoto důvodu si raději před jeho použitím důkladně prostudujte dokumentaci.

130. Vložení čísel řádků do výstupního souboru

Pro celkové číslování řádků existují dva balíky použitelné pod LATEXem: *lineno* (který dovoluje k jednotlivým řádkům výstupu připojit hesla) a *numline*. Oba balíky však neberou ohledy na LATEXový výstup, což může způsobit problémy. Na to by si měli dát uživatelé pozor.

Pokud však chcete číslovat doslovný text (`verbatim`), pak je nejlepší použít balíky (*moreverb* nebo *fancyvrb*, viz Otázku 129). Další balík – edmac (*edmac*) – nabízí pomoc v případech kritických edicí.

`lineno`: `macros/latex/contrib/lineno`
`numline`: `macros/latex/contrib/numline/numline.sty`
`moreverb`: `macros/latex/contrib/moreverb`
`fancyvbr`: `macros/latex/contrib/fancyvrb`
`edmac`: `macros/plain/contrib/edmac`

131. Generovanie registra v TEXu/LATEXu

Vytvorenie registra – indexu – nie je jednoduchá úloha. Čo to je index a ako ho robiť je ťažko rozhodnúť. Rovnako je ťažké urobiť všeobecnú dohodu, ako je správne index robiť. V každom prípade musíte označiť všetky položky v texte (obvykle príkazom `\index`), ktoré majú byť indexované.

Triedenie veľkého indexu priamo v TEXu nie je praktické, preto sa používajú postprocessing programy, ktoré spracovávajú index mimo TEXu a register je dostupný v ďalšom behu programu TEX.

Dostupné sú nasledujúce programy:

makeindex pre LATEX pod Unixom (ale pracuje aj pod inými operačnými systémami rovnako). Je v `indexing/makeindex`. Verzia pre Macintosh je dostupná na CTANe pod názvom *macmakeindex* (`systems/mac/macmakeindex2.12.sea.hqx`), pre MS-DOS je súčasťou distribúcie emTEXu a gTEXu (emTEX má verziu aj pre OS/2).
Dokumentácia pre *makeindex* predstavuje dobrý zdroj informácií ako vytvárať register. *Makeindex* môžeme použiť aj s niektorými balíkmi makier TEXu, ako je napríklad *Eplain* (pozri Otázku 14) alebo TEXsis. (`macros/texsis`)

idxtex pre LATEX pod VMS. Je dostupný (s „poznámkovačom“ `glotex` v `indexing/glo+idxtex`).

texindex Veľmi vtipná malá utilita pre LATEX pod operačným systémom Unix. Je dostupná v `support/texindex`.
Existujú aj iné programy s menom *texindex*, napríklad ten, ktorý je súčasťou distribúcie Texinfo (pozri Otázku 16).

xindy novo vyvinutý, navrhnutý ako rozsiahlo flexibilný (vrátane podpory viacjazičných indexov), založený na Common Lispe. Systém je dostupný na CTANe (`indexing/xindy`), ale je jednoduchšie prístupný internetovým prehliadačom na adrese `http://www.iti.informatik.th-darmstadt.de/xindy/`, pretože distribúcia obsahuje niekoľko odlišných implementácii.

132. Používanie BIBTEXu s plainTEXom

Súbor `macros/eplain/btxmac.tex` obsahuje makrá a dokumentáciu pre BIBTEX s `plain` TEXom alebo priamo s Eplainom (pozri Otázku 14). Pozrite si aj Otázku 27, ktorá obsahuje ďalšie informácie o BIBTEXu samotnom.

133. Sazba URL

V poslední době se URL prodlužují a obsahují znaky, které jim mohou přirozeně zabraňovat v hladkém dělení, pokud nejsou typicky označeny `\ttfamily`. Proto bez zvláštního ošetření mohou přečnívat za hranici sloupce a jejich výsledná sazba je hrozná.

Existují tři možnosti přístupu k tomuto problému:

- `macros/latex/contrib/misc/path.sty`, který definuje příkaz `\path`. Tento příkaz definuje každý potenciálně problematický znak jako `\discretionary` a umožňuje každému uživateli definovat svůj osobní seznam těchto znaků. Jeho hlavní nevýhoda je, že patří mezi křehké příkazy. Navzdory své dlohé historii už není doporučován.
- `macros/latex/contrib/misc/url.sty`, který definuje příkaz `\url` (mezi jinými také obsahuje vlastní příkaz `\path`). Tento příkaz přiřadí každému problematickému znaku matematický kód a pak URL vysází v matematickém režimu fontem, který si uživatel vybere. V LATEXovém prostředí umožňuje vytvářet robustní příkazy (viz Otázku 255). Všimněte si, že jelikož je operace provedena v matematickém módu, jsou mezery v URL parametru ingorovány (když neuděláte specielní kroky).

  Balík *url* je možné použít v `plain` TEXu s pomocí balíku *miniltx* (jež byl původně vyvíjen pro použití grafického balíku LATEXu v `plain` TEXu). Potřebná je ale malá úprava, požadovaná sekvence je tedy:

  ```
  \input miniltx
  \expandafter\def\expandafter\+%
                                  \expandafter{\+}
  \input url.sty
  ```
- balík `macros/latex/contrib/hyperref`, který používá sázecí kód balíku *url* v kontextu, kdy vysázený text tvoří ukontvění odkazu.

Autor této odpovědi preferuje (novější) balík *url* (přímo nebo nepřimo). Oba balíky *path* a *url* pracují skvěle v `plain` TEXu (ikdyž tam samozřejmě přitažlivé LATEXovské prostředky balíku *url* nemají co dělat). (*hyperref* není ve verzi pro `plain` TEX dostupný.)

134. Sadzba nôt v TEXu

V minulosti bol napísaný balík *mutex* od Angeliky Schoferovej a Andreu Steinbacha, ktorý dokazoval, že sadzba hudby je možná. Balík bol však veľmi obmedzený a naďalej nie je dostupný.

Výkonný balík, ktorý umožňuje sadzbu zložitých notových osnov, je MusicTEX, ktorý napísal Daniel Taupin (`taupin@rsovax.lps.u-psud.fr`). Je dostupný v `macros/musictex`, ale nie je naďalej odporúčaný. V nedávnej minulosti Daniel (ako aj mnohí ďalší, ako Ross Mitchell a Andreas Egler) pracovali na vývoji MusicTEXu, ktorý má názov MusiXTEX. MusiXTEX je trojprechodový systém (s predprocesorovým programom, ktorý počíta medzery pre prvky) a dosahuje jemnejšiu kontrolu, ako je možné dosiahnuť v nemodifikovanom MusicTEXu. Momentálne je vyvíjaná iba verzia Daniela Taupina (Andreas Egler mal tiež svoju verziu, ale v súčasnosti už pracuje a inom balíku) MusiXTEXu. Obe sú dostupné v `macros/musixtex/taupin` a `macros/musixtex/egler`.

Vstup pre MusixTEX je extrémne zložitá záležitosť. Preferovanou metódou vytvárania vstupu pre Taupinovu verziu je preprocesor Dona Simonsa *pmx*. *Pmx* veľmi uľahčuje používanie MusixTEXu, ale nepodporuje všetky prostriedky MusixTEXu priamo. Dovoľuje však in-line kód MusixTEXu v zrdojovom kóde *pmx*.

Fanúšikovia digitálnej hudby môžu sádzať notáciu použitím *midi2tex*, ktorý prekladá MIDI súbory do zdrojového kódu MusicTEXu. Je dostupný z `support/midi2tex`.

O dosť jednoduchší zápis ako MusicTEX má *abc2mtex*. Je to balík navrhnutý pre zápis melódie vo forme ASCII znakov (`abc` notácia). Táto notácia bola navrhnutá hlavne pre zápis ľudových melódií západoeurópskeho pôvodu (írskych, anglických a škótskych), ktoré môžu byť zapísané aj klasickým spôsobom na jednej notovej osnove. Ale môže byť použitá aj na mnoho iných typov hudby. Balík je dostupný v `support/abc2mtex`.

Pre diskusiu o problémoch sadzby nôt v systéme TeX využívajte vytvorený mailing list `TeX-music@icking-music-archive.org`. Pre subskripciu využite možnosti stránky `http://icking-music-archive.org/mailman/listinfo/tex-music/`.

135. Kreslenie Feynmanových diagramov v LATEXu

Makro-balík pre kreslenie Feynmanových diagramov v LATEXu od Michaela Lavine je dostupný v `macros/latex209/contrib/feynman`.

Môžeme tiež použiť `macros/latex/contrib/feynmf`, balík makier pochádzajúci od Thorstena Ohla, ktorý pracuje v kombinácii s METAFONTom (alebo METAPOSTom).

Makrá z balíkov *feynmf* alebo *feynmp* čítajú popis diagramu, ktorý je napísaný v TEXu, a zapisujú svoj kód. METAFONT (alebo METAPOST) potom môže vytvoriť v ďalšom behu LATEXu príslušné písma. Pre nových používateľov, ktorí majú možnosť použiť METAPOST, možno odporúčať PostScriptovú verziu pre lepšiu prenositeľnosť dokumentov, ako aj niektoré ďalšie dôvody.

Ďalšie balíky použiteľné na tento účel sú: `axodraw` Josa Vermaserena (`graphics/axodraw`) a *feyn* Normana Graya (`fonts/feyn`).

136. Nulové odsadenie odstavcov

Bežný spôsob sadzby textu nemá odstavce navzájom oddelené, iba prvý riadok každého odstavca je odsadený.

Konvenciou pre text písaný na písacom stroji naopak bolo neodsadzovanie odstavcov. Takýto štýl je vyžadovaný pre publikácie ako technické manuály a v štýloch vyžadujúcich originál písaný písacím strojom (ako napríklad oficiálne špecifikované formáty dizertácií).

Každý po chvíľke rozmýšľania vidí, že pri nulovom odsadení odstavcov musia byť tieto oddelené prázdnym miestom, inak bude nemožné rozoznať zlomy medzi odstavcami.

Jednoduchým prístupom je teda vynulovanie odsadenia odstavcov:

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{\baselineskip}
```

a v najjednoduchších textoch je to dobré riešenie.

Avšak nenulový `\parskip` prekáža zoznamom a podobnému formátovaniu, takže výsledky vyzerajú hrozne. Balík *parskip* upravuje veci tak, aby vyzerali rozumne. Nie je bezchybný, ale vysporiada sa s väčšinou problémov.

Sada tried skupiny používateľov z Holandska obsahuje ekvivatent triedy `article` (`artikel3`) a triedy `report` (`rapport3`), ktorých návrh zahŕňa nulové odsadenie a nenulový `\parskip`.
triedy `\acro{NTG}`: `macros/latex/contrib/ntgclass`
`parskip.sty`: `macros/latex/contrib/misc/parskip.sty`

137. Iniciálky

Bežným štýlom sadzby, v súčasnosti málo sa vyskytujúcim (okrem novín), je začatie odstavca (v knihách, zvyčajne prvého odstavca kapitoly) prvým písmenom dostatočne veľkým (rozsahu niekoľkých riadkov).

Tento štýl sa nazýva „iniciálky“ (dropped capitals) alebo (po francúzsky) „lettrines“. TEXovské prostriedky umožňujú jeho jednoduchú a priamočiaru implementáciu.

Balík *dropping* prácu vykoná jednoducho, má ale zvláštny postoj k výpočtom veľkosti fontu veľkých písmen (iniciálok). Príklady sú uvedené v dokumentácii, takže pred spracovaním súboru `.dtx` musíte mať balík už nainštalovaný. *dropping* má bohužiaľ chúlostivý vzťah k sade ovládačov zariadení dostupných v prvých verziách grafického balíku LaTeXu. Nemôžeme si teda byť istý, že bude pracovať s novinkami ako PDFTeX, VTeX alebo DVIpdfm.

V takýchto prípadoch pravdepodobne uspeje novší balík *lettrine*. Balík má dobre konštruované pole volieb a príklady (celkom pôsobivá sada) sú dodávané ako osobitný súbor distribúcie (dostupné tiež v PostScripte, aby mohli byť prezreté bez nutnosti inštalovať balík).

dropping: `macros/latex/contrib/dropping`
lettrine: `macros/latex/contrib/lettrine`

138. Psací písma pro matematický režim

Font vybraný příkazem `\mathcal` je jediným zabudovaným psacím písmem. Nicméně však v moderních TeXových distribucích jsou zabudovány i další kaligrafické fonty.

Euler `\usepackage{eucal}` Je součástí pouze několika nejmodernějších TeXových distribucí, nicméně je volně dostupný na CTANu archívu na adrese `fonts/amsfonts/latex/eucal.sty`. Je součástí AMS fontů. Nabízí nepatrně „zakroucenější" font. Tento balík mění fonty, které jsou vybrány příkazem `\mathcal`. Verze Type 1 těchto fontů jsou dostupné v distribuci AMS fontů (`fonts/amsfonts/ps-type1`).

RSFS `\usepackage{mathrsfs}` (součást `macros/latex/contrib/supported/jknappen`) používá opravdu ozdobné psací písmo (zkratka z „Ralph Smith's Formal Scripts"), které je již součástí nejmodernějších distribucí (můžete ho najít i ve CTANovském archívu v `fonts/rsfs`). Balík vytváří nový příkaz `\mathscr`. Tyto fonty ve verzi Type 1 byly zpřístupněny Tacem Hoekwaterem. Jsou přístupné na CTANu na adrese `fonts/rsfs/ps-type1/hoekwater`.

Zapf Chancery je standardní postscriptový kaligrafický font. Není spojen s žádnými balíky, ale snadno ho můžete zpřístupnit příkazem

```
\DeclareMathAlphabet{\mathscr}{OT1}{pzc}{m}{it}
```

ve vaší preambuli. Také ho můžete použít v jiné velikosti:

```
\DeclareFontFamily{OT1}{pzc}{}
\DeclareFontShape{OT1}{pzc}{m}{it}{<-> s * [0.900] pzcmi7t}{}
\DeclareMathAlphabeth{\mathscr}{OT1}{pzc}{m}{it}
```

Adobe Zapf Chancery (který výše zmíněný příklad používá) je distribuován s některými postscriptovými tiskárnami. V podstatě identický font z URW a je distribuován s Ghostscriptem.

Příklady dostupných stylů jsou předvedeny na `info/symbols/math/scriptfonts.pdf`.

139. „Vodoznak" na každé stránce

Občas je užitečné na pozadí každé stránky dokumentu umístit jistý text (např. „DRAFT"). Uživatelé LaTeXu toho dosáhnou použitím balíku `macros/latex/contrib/draftcopy`. Umí pracovat s mnoha typy DVI procesorů a umí přeložit slovo „DRAFT" do širokého množství jazyků (také si však můžete vybrat vlastní slovo). Vypracovanějších vodoznaků dosáhnete pomocí balíku `eso-pic`, který ještě využívá balíček `everyshi`. Oba jsou distribuovány v balíku `macros/latex/contrib/ms`. *Eso-pic* ke každé stránce připojí prostředí `picture`, do prostředí můžete vkládat věci. Balík poskytuje příkazy pro umísťování věcí do určitých užitečných míst obrázku (jako „text nahoru vlevo", ...), ale máte možnosť dělat, co se vám zlíbí.

140. Sazba textu naležato

Občas je nezbytné vysázet část textu dokumentu naležato. Abychom tohoto dosáhli, nestačí pouze změnit rozměry stránky, musíme ještě informovat tiskárnu, že má tuto stránku tisknout odlišně. Existují dva možné způsoby, jak toho docílit:

- Pokud máte objekt, který svou šířkou přesahuje šířku stránky a hodí se pouze na stránku naležato, pak můžete použít balík *rotating* (`macros/latex/contrib/rotating`, který definuje dvě různá prostředí `sidewaysfigure` a `sidewaystable`, které pracují pouze s celou stránkou.
- Pokud máte více těchto objektů, které potřebujete vysázet naležato (např. seznamy různých kódů, široké prostředí `tabbing` nebo velké tabulky sázené pomocí *supertabular* nebo *longtable*), pak můžete použít balík *lscape* (`macros/latex/required/graphics`), případně

pdflscape (když generujete PDF výstup; `macros/latex/contrib/oberdiek`). Ten definuje prostředí `landscape`, které vymaže běžnou stránku a zahájí znovu její sazbu naležato (a smaže stránku na konci prostředí před vrácením do režimu nastojato).

Žádné běžně dostupné balíky nevytvářejí přímé opatření pro sazbu v obou možných orientacích na jedné stránce. Pokud byste takové chování nutně potřebovali, použijte techniku popsanou v Otázce 127. Výsledek můžete otočit pomocí balíku *graphics* (`macros/latex/required/graphics`. Návrat k vertikální orientaci (portrait) je snazší: vertikálně orientovaná část stránky bude spodním floatem na konci horizontální (landscape) sekce s otočeným obsahem.

Pro vysázení celého dokumentu horizontálně můžete použít *lscape* kolem celého dokumentu. Lepší je volba `landscape` balíku *geometry*. Jestliže mu předáte také volbu `dvips` nebo `pdftex`, *geometry* vychrlý rotační instrukce, které spůsobí, že výstup bude správné orientován. Třída `memoir` v tomto ohledu poskytuje stejné prostředky jako *geometry*.

Na závěr ještě jedno varování: běžně dostupné prohlížeče TEXových souborů většinou zanedbávají požadavky na rotaci v `DVI` souborech. Pokud váš prohlížeč není schopný rotace, je nutno přeložit soubor do formátu `PS` nebo `PDF` a výsledek pak můžete prohlédnout vhodným programem.

141. Umiestňovanie na fixné pozície na stránke

TEXovský model sveta je (obecne povedané), že autor píše text a TEX a jeho makrá rozhodujú, ako sa to všetko zmestí na stránku. Toto nie je dobrá správa pre autora, ktorý požaduje, aby boli určité veci umiestnené na presné miesta na stránke.

Na stránke existujú miesta, na ktorých môžu byť „zavesené" veci. Dva LATEXovské balíky vám dovoľujú umiestňovanie relatívne k týmto miestam, čím poskytujú absolútne umiestnenie.

Balík *textpos* pomáha vytvárať stránky z bodov umiestnených vôkol stránky (ako pri plagáte). Vy mu zadáte lokáciu, on príslušne umiestni sádzací box.

eso-pic definuje „obrázok" pokrývajúci stránku. Používateľ do tohto obrázku môže pridávať príkazy módu `picture`, ktoré môžu obsahovať samozrejme aj umiestnenia boxov ako aj iných príkazov `picture` módu. (*Eso-pic* potrebuje služby balíku *everyshi*, ktorý teda musí tiež byť dostupný.)

`eso-pic.sty`: `macros/latex/contrib/eso-pic`
`everyshi.sty`: distribuované ako súčasť `macros/latex/contrib/ms`
`textpos.sty`: `macros/latex/contrib/textpos`

142. Zabránenie zlomu stránky medzi riadkami

Občas je potrebné, aby blok vysádzaného materiálu bol na jednej strane. Zabezpečenie tejto podmienky je prekvapivo zložité.

LATEX poskytuje prostredie `samepage`, ktoré o sebe tvrdí, že práve toto za vás spraví. Postupuje nastavením pokút (penalties) za všetky možné situácie vedúce k zlomu stránky na nekonečno, no v mnohých prípadoch nepomáha. Ak sa pokúšate udržať text pohromade, budete musieť ukončiť odstavec vo vnútri prostredia. Ak veci, čo chcete udržať pohromade, vkladajú vlastné náznaky na zlom stránky, `samepage` nad nimi nemá moc: dobrým príkladom sú položky zoznamu — navrhujú medzi sebou zlom stránky. Dokonca aj keď prostredie funguje, je pravdepodobné, že na spodku stránky nechá určité veci vyčnievať.

Vhodným trikom je vysádzať všetko dôležité do `\parbox` (v prípade, že obsahuje niečo, čo nemôže byť parametrom `\parboxu`, do prostredia `minipage`). Výsledný box sa medzi stranami určite „nezlomí", ale to neznamená, že spraví to, čo chcete: takisto môže presahovať spodný okraj stránky.

Prečo žiadna z týchto vecí nefunguje? Pretože TEX nerozlíši nekonečne „zlé" veci. Prostredie `samepage` spraví každý možný bod zlomu nekonečne zlým a boxy dokonca ani neposkytujú možnosť zlomu, ale ak je alternatívou ponechanie nekonečne zlých pár centimetrov prázdneho papieru na konci strany, TEX sa vyberie cestou najmenšieho odporu a nespraví nič.

Problém vzniká aj keď máte v platnosti `\raggedbottom`: TEX si nevšimne jeho hodnotu, pokiaľ neodošle stránku von. Jeden prístup je nestavenie:

```
\raggedbottom
\addtolength{\topskip}{0pt plus 10pt}
```

10pt naznačuje výstupnej rutine, že stĺpec možno roztiahnuť. Toto spôsobí, že TEX bude pri budovaní stránky tolerantnejší k potrebe roztiahnutia. Ak to robíte ako dočasné opatrenie, zrušte zmenu u \topskip:

```
\addtolength{\topskip}{0pt plus-10pt}
```

Mali by ste aj resetovať \flushbottom. Všimnite si, že 10pt sa v skutočnosti nikdy neukáže, pretože pri odosielaní stránky von je prekonané roztiahnuteľnosťou vloženou \raggedbottom; mohlo by však mať nejaký účinok, ak by účinkovalo aj \flushbottom.

Alternatívou (odvodenou z Knuthovho návrhu v TEXbooku) je balík *needspace* alebo trieda memoir, pričom v oboch je definovaný príkaz \needspace, ktorého parameter mu vraví, aký prestor je potreba. Ak priestor nie je dostupný, materiál, ktorý musí zostať spolu, je vložený na druhú stranu. Ak napríklad 4 riadky textu musia zostať pohromade, použite:

```
\par
\needspace{4\baselineskip}
% veci, čo musia zostať pohromade
<text generujúci riadky 1-4>
% a teraz veci, na ktorých nám nezáleží
```

Ďalší Knuthov trik je užitočný, ak máte postunosť malých blokov textu, ktoré individuálne potrebujú zostať na svojej vlastnej stránke. Vložte príkaz \filbreak pred každý malý blok a dosaihnete požadovaný efekt. Technika môže byť použitá v prípade postupností sekcií LATEXovského štýlu. Zabudujte \filbreak do definície príkazu. Jednoduchou a efektívnou úpravou je:

```
\let\oldsubsubsection=\subsubsection
\renewcommand{\subsubsection}{%
  \filbreak
  \oldsubsubsection
}
```

Trik funguje pre nasledujúce postupnosti blokov. V prípade, že postupnosť je prerušená vynúteným zlomom stránky (ako \clearpage, ktorý môže byť vložený príkazom \chapter alebo koncom dokumentu), je však trochu obtiažne dostať sa z takých sekvencii von. Ak postupnosť nie je prerušená, posledný blok bude pravdepodobne vytlačený na novú stranu bez ohľadu na to, či to skutočne potrebuje.

Ak akceptujete, že nie všetko sa dá dosiahnuť úplne automaticky, najlepším spôsobom je vysádzať dokument a skontrolovať problémové veci. V takomto prípade sa môžete rozhodnúť, ako sa s problémom vysporiadať pri poslednej kontrole čítaním. Možnosti sú vloženie príkazov \clearpage tam, kde sú potrebné, alebo použitie \enlargethispage. Predpokladajme, že máte jeden alebo dva zatúlané riadky. Zadajte príkaz \enlargethispage{2|\cs |baselineskip} a na aktuálnu stránku budú pridané dva riadky. Záleží na dokumente, či toto vyzerá nemožne alebo akceptovateľne, ale príkaz zostáva užitočnou položkou vo výzbroji.
memoir.cls: macros/latex/contrib/memoir
needspace.sty: macros/latex/contrib/misc/needspace.sty

143. Paralelná sadzba textu

Bežne je potrebné prezentovať text v dvoch jazykoch spoločne na jednej stránke alebo na opačných stranách dvojstranového rozloženia. Aby to bolo akceptovateľné, musí medzi textami existovať určité zarovnanie.

Balík *parallel* tieto potreby uspokojuje. Umožňuje sadzbu do dvoch stĺpcov (nie nutne rovnakej šírky) na jednej strane alebo do opačných strán dvojstranového rozloženia.
parallel.sty: macros/latex/contrib/parallel

144. Sadzba epigrafov

Epigrafy sú tie krásne citáty, ktoré autori vkladajú na začiatok kapitol (alebo dokonca na koniec kapitol, Knuth v TEXbooku vkladá na koniec kapitol).

Ich sadzba je tak trochu umenie, ale nie je nemožné spraviť ju vlastnoručne. Balík *epigraph* však spraví prácu za nás, dokonca aj v situáciach, kde je mimoriadne protivné spraviť to správne.

Balík definuje príkaz \epigraph na vytvorenie jedného epigrafu (ako na vrchu kapitoly):

```
\chapter{The Social Life of Rabbits}
\epigraph{Oh!  My ears and whiskers!}%
         {Lewis Carroll}
```

Ďalej definuje prostredie epigraphs na zadávanie viacerých epigrafov za sebou v istom druhu zoznamu tvoreného príkazmi \qitem:

```
\begin{epigraphs}
\qitem{What I tell you three times is true}%
      {Lewis Carroll}
\qitem{Oh listen do, I'm telling you!}%
      {A.A. Milne}
\end{epigraphs}
```

Príkaz \epigraphhead vám dovoľuje umiestniť epigraf *nad* hlavičku kapitoly:

```
\setlength{\unitlength}{1pt}
...
\chapter{The Social Life of Rabbits}
\epigraphhead[<distance>]{%
  \epigraph{Oh!  My ears and whiskers!}%
           {Lewis Carroll}%
}
```

<distance> (vzdalenosť) udáva, ako ďaleko nad nadpis kapitoly sa má epigraf umietniť; je to vyjadrené v zmysle \unitlength (používané v prostredí picture). Autor balíku odporúča hodnotu 70pt.

Balík tiež poskytuje rôzne triky na nastavenie rozloženia hlavičky kapitoly (potrebné, ak ste našli veľmi dlhú citáciu pre \epigraphhead), na úpravu bibliografie, na úpravu strán \part a tak ďalej. (Niektoré z týchto úprav vás povedú cez písanie vlastného balíku...)

Trieda memoir poskytuje všetky prostriedky balíku *epigraph*. Triedy Koma-script obsahujú príkazy \setchapterpreamble a \dictum, ktoré tieto prostriedky tiež poskytujú.

epigraph.sty: macros/latex/contrib/epigraph
balík KOMA script: macros/latex/contrib/koma-script
memoir.cls: macros/latex/contrib/memoir

145. Dokumenty s jinou základní velikostí písma

LaTeXové standardní třídy mají koncept základních velikostí dokumentových fontů. Tyto velikosti jsou základem, na němž jsou postaveny jiné fonty. Tyto třídy jsou postaveny na předpokladu, že nebudou používány v jiné velikosti, než kterou umožňuje TeX (10–12 bodů), avšak lidé často potřebují psát jinou velikostí. Typograficky správné řešení tohoto požadavku spočívá ve vytvoření nového designu dokumentu, avšak tato představa spoustu lidí odrazuje. Někteří lidé proto používají balíček macros/latex/contrib/extsizes, který umožňuje vytváření dokumentů třídy article, letter, report nebo book ve velikostech 8, 9, 14, 17 a 20 bodů stejně dobře jako ve velikostech 10–12.

146. Dvojité riadkovanie v dokumentoch LaTeXu

Rýchlym a jednoduchým spôsobom, ako získať medziriadkový priestor pre korektúry, je zmena \baselinestretch — \linestretch{1.2} (alebo \renewcommand{\baselinestretch }{1.2}) môže pomôcť. Všimnite si, že zmeny \baselinestretch sa neuplatnia, pokým si nevyberiete nový font, takže zmenu robte v preambule predtým, než si vyberiete font. Nepokúšajte sa zmeniť \baselineskip: jeho hodnota je zmenená po každom príkaze meniacom veľkosť, takže výsledky by boli nekonzistentné.

Prednosť dajte radšej (pri produkčnom dokumente ako dizertačná práca alebo článok) riadkovaciemu balíku. Jediným momentálne podporovaným balíkom je *setspace* (*nenechajte* sa zviesť balíkom *doublespace* — jeho výkon v aktuálnom LaTeX je prinajlepšom problematický). *Setspace* vypína dvojité riadkovanie na miestach, kde by aj najdrsnejší úradník spochybnil jeho užitočnosť (poznámky pod čiarou, popisky obrázkov atď.). Je veľmi ťažké toto spraviť konzistentne, ak manipulujete s `\baselinestretch` sami.

Skutočným riešením je samozrejme (pre iné účely než korektúra) *nepoužívať* dvojité riadkovanie vôbec. Chovanie univerzít vyžadujúcich dvojité riadkovanie v prácach je neospravedlniteľné: LaTeX je sádzací systém, nie náhrada písacieho stroja a vie spraviť (správnym použitím) text s jednoduchým riadkovaním dokonca čitateľnejší ako dvojito riadkovaný text. Ak máte na systém vašej univerzity akýkoľvek vplyv (napríklad ako vedúci práce), stálo by za to pokúsiť sa o zmenu pravidiel (aspoň povoliť formát dobre navrhnutej knihy —„well-designed book").

Dvojito riadkované príspevky sú tiež obvykle požadované na konferenciách alebo v žurnáloch. Našťastie (súdiac podľa otázok uživateľov) je táto požiadavka čoraz zriedkavejšia.

Dokumentácia *setspace* sa objavuje v samotnom balíku v podobe TeXovských komentárov.
`setspace.sty`: `macros/latex/contrib/setspace/setspace.sty`

O. Jak udělám X v TeXu nebo LaTeXu?

147. Prostredie dôkaz (proof)

Dlho sa považovalo za nemožné vytvoviť prostredie `proof`, ktoré by automaticky vkladalo symbol konca dôkazu. Niektoré dôkazy končia vysádzaním matematiky, iné nie. Ak vstupný súbor obsahuje `...\] \end{proof}`, potom LaTeX ukončí vysádzanú matematiku a je pripravený na nový riadok bez toho, aby prečítal akúkoľvek informáciu o ukončení dôkazu. Kód je teda celkom zložitý. Symbol *môžete* vložiť ručne, ale balík *ntheorem* tento problém rieši za používateľov LaTeXu: skutočne poskytuje automatický spôsob značenia konca dôkazu.

$\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeXovský balík *amsthm* tiež poskytoju prostredie `proof`, ktoré toto dokáže, hoci ak dôkaz končí rovnicou, musíte vložiť príkaz `\qedhere`:

```
\begin{proof}
  text...
 \begin{equation*}
  matematika... \tag*{\qedhere}
 \end{equation*}
\end{proof}
```

Konštrukcia `\tag*{\qedhere}` môže byť použitá v ľubovolnom číslovacom prostredí $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeXu.
`amsthm.sty`: distribuované ako súčasť sady $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX `macros/latex/required/amslatex`
`ntheorem`: `macros/latex/contrib/ntheorem`

148. Matematické věty a definice sázené antikvou

Chceme-li využít výhod příkazu `\newtheorem`, a přitom nechceme být omezeni na to, že obsah bude vysázen skloněným fontem (např. k vytvoření poznámek, příkladů, důkazů,. . .) pak použijeme styl `theorem.sty` (`macros/latex/required/tools`). Popřípadě můžeme definovat prostředí, např. `remark`, jehož obsah bude sázen antikvou (roman):

```
  \newtheorem{preremark}{Poznámka}
  \newenvironment{remark}%
    {\begin{preremark}\upshape}{\end{preremark}}
```

Balík *ntheorem* (`macros/latex/required/tools`) poskytuje matematické věty a definice sázené antikvou přímo.

149. Označení číselných množin a dvojitě psané písmo

Je dobré mít příkazy typu `\R` pro standardní množinu reálných či jiných čísel. Tradičně byly tyto množiny sázeny tučně. Matematici však obvykle nemají přístup k tlustým křídám, a proto si vymysleli speciální symboly, které se nyní běžně používají pro označení `\R`, `\C`, atd. Tyto symboly jsou označovány jako „zdvojená latinka" (blackboard bold). Dříve, než začneme trvat na použití těchto symbolů, bylo by dobré se zamyslet, zda by nebylo lépe se navrátit k původnímu označování číselných množin prostým tučným písmem, což by činilo věc daleko jednodušší.

Sada velkých písmen zdvojené latinky je součástí AMS-fontů „msam" (např. „msam10" pro velikost 10pt) a „msbm". Fonty obsahují veliké množství matematických symbolů, které podstatně rozšiřují sadu symbolů dodávanou ve standardní distribuci TEXu v Type 1 formátu. K dispozici jsou podpůrné soubory pro běh pod Plain TEXem a LATEXem (balíky *amssymb* a *amsfonts*).

Součástí balíku jsou i dva soubory, z nichž jeden usnadňuje vkládání fontů do dokumentu a ve druhém jsou definovány příkazy pro jednotlivé symboly. Oba lze použít jak v TEXu tak v LATEXu. Otázky a doporučení tykající se těchto fontů směrujte na `tech-support@math.ams.org`.

Jiná úplná sada fontů zdvojené latinky, rodina fontů bbold, je dostupná ve zdrojových souborech pro METAFONT v adresáři `fonts/bbold`. Zajímavostí je, že tyto fonty obsahují kromě velkých i malá písmena, které jsou na skutečných tabulích vidět jen zřídka. Zdrojový adresář fontu obsahuje také zdrojové soubory LATEXovského balíku umožňujícího používaní fontu. Fonty nejsou dostupny v Type 1 formátu.

Alternativním zdrojem Type 1 fontů se znaky zdvojené latinky můžou být pořád se rozšiřující sady kompletních rodin (jak komerční, tak volně dostupné), které byly připraveny pro použití s TEXem/LATEXem (99). Z volně dostupných sad poskytujý repliky *msam* and *msbm* rodiny *txfonts* a *pxfonts*, rodina *mathpazo* obsahuje „matematicky důležitý" výběr zdvojené latinky.

V případě, že font nemáme, lze vytvořit také jednoduchá makra pro tvorbu zdvojených znaků:

```
\newcommand{\R}{{\sf R\hspace*{-0.9ex}%
  \rule{0.15ex}{1.5ex}\hspace*{0.9ex}}}
\newcommand{\N}{{\sf N\hspace*{-1.0ex}%
  \rule{0.15ex}{1.3ex}\hspace*{1.0ex}}}
\newcommand{\Q}{{\sf Q\hspace*{-1.1ex}%
  \rule{0.15ex}{1.5ex}\hspace*{1.1ex}}}
\newcommand{\C}{{\sf C\hspace*{-0.9ex}%
  \rule{0.15ex}{1.3ex}\hspace*{0.9ex}}}
\newcommand{\openone}{\leavevmode\hbox{\small1\kern-2.8pt\normalsize1}}
```

která lze s úspěchem použít v případě, že okolní text je sázen `cmr10`. Avšak, jelikož se nejedná o řádné matematické fonty, nelze je užít v matematických indexech. Krom toho velikost a poloha vertikální linky je ovlivněna výběrem fontu okolního textu (makra obsahují `ex`).

150. Jak vysázím značku stupně v TEXu

Pomocí makra `\degree` definovaného takto:

```
\newcommand{\degree}[1][]{\ensuremath{{#1}^\circ}}
```

151. Jak vysázím „středoevropské" uvozovky

Uvozovky v češtině, slovenštině, jakož i jiných středoevropských jazycích, mají jiný tvar než anglické "", které TEX sází implicitně. Nejjednodušší způsob, jak vysázet text v „našich" uvozovkách, je pomocí makra `\uv`: `v~\uv{našich} uvozovkách`. Makro `\uv` je součástí balíků `czech.sty` a `slovak.sty`.

Lepším a obecnějším řešením je makro Tomáše Hály `\uviq`, které dovoluje uvozované texty vnořovat. Toto makro je součástí distribuce CSTEX.

152. Chci desetinnou čárku místo tečky

TEX implicitně předpokládá, že v zápisu desetinných čísel odděluje desetiny od jednotek desetinná tečka. Čárku zapsanou v matematickém módu TEX považuje za oddělovač položek v posloupnostech, množinách apod., a proto za ni sází malou mezeru.

Chceme-li jednorázově vysázet desetinné číslo s desetinnou čárkou, stačí čárku uzavřít do skupiny: `$3{,}14159$`.

Takové řešení je však nevýhodné pro texty, kde je desetinných čísel mnoho. V takovém případě lze nastavit globálně, že za čárkou v matematickém módu se mezera sázet nemá:

```
\mathcode`,="002C
$3,14159$
```

Pokud při tomto globálním nastavení přesto někde mezeru udělat potřebujeme, musíme ji napsat explicitně: `$(m,\,n)$` nebo `$\{0,\:1,\:2\}$`.

Jiným řešením je ponechat ve vstupním souboru desetinné tečky, ale přemapovat na čárky (to se hodí třeba tehdy, když máme velké soubory desetinných čísel z jiných výstupů).

```
\mathcode`.="002C
$3.14159$
```

Snad nejjednoduchším řešením je použití balíku *icomma*. Balík zajistí, že za čárkou nebude mezera sázena, pokud ji za čárku nenapíšete (jako například u `f(x, y)`), jinak se za čárku vysází obvyklá malá mezera.

`icomma.sty`: distribuováno jako součást `macros/latex/contrib/was`

153. Zalamovanie textových boxov

TEXovské/LATEXovské boxy nemôžu byť pri bežnom použití zalamované. Keď raz niečo vysádzate do boxu, zostane to tam a box bude prečnievať niektorý z okrajov stránky (v prípade, že sa nezmestí do oblasti, kam má byť vysádzaný).

Ak chcete mať podstatnú časť textu orámovanú (alebo zafarbenú), obmedzenia začínajú predstavovať skutočné bremeno. Našťastie sa problém dá obísť.

Balík *framed* poskytuje prostredia `framed` a `shaded`. Obe vkladajú svoj obsah do niečoho, čo vyzerá ako orámovaný (alebo farebný) box, ale podľa potreby sa na konci stránky láme. Prostrediam chýbajú poznámky pod čiarou, marginpar, head-line položky a nebudú fungovať s balíkom *multicol* alebo inými makrami na vyvažovania stĺpcov. Trieda `memoir` zahŕňa funkcionalitu balíku *framed*.

Balík *boites* poskytuje prostredie `breakbox`. Príklady použitia nájdete v distribúcii, `README` súbor obsahuje hutnú dokumentáciu. Prostredia môžu byť vnorované a môžu sa objaviť v prostrediach `multicols`. Plávajúce prostredia, poznámky pod čiarou a marginpar sa však stratia.

Pre používateľov `plain` TEXu môžu byť užitočné prostriedky balíku *backgrnd*. Tento balík prinúti výstupnú rutinu poskytnúť vertikálne čiary na označenie textu, makrá sú jasne označené a ukazujú, kde môžu byť zavedené farebné pozadia (toto vyžaduje balík *shade*, ktorý je distribuovaný ako TEXovské makrá a pre tieňovanie METAFONT nezávislý na zariadení). Autor *backgrnd* tvrdí, že balík pracuje v prostredí LATEX 2.09, ale existujú dôvody domnievať sa, že pri práci s aktuálnym LATEXom môže byť nestabilný.

`backgrnd.tex`: `macros/generic/backgrnd.tex`
`boites.sty`: `macros/latex/contrib/boites`
`framed.sty`: `macros/latex/contrib/misc/framed.sty`
`memoir.cls`: `macros/latex/contrib/memoir`
`shade.tex`: `macros/generic/shade.sty`

154. Realistické úvodzovky pre doslovné výpisy programov

Font `cmtt` má „okrúhle" úvodzovky (`‘takéto’`), ktoré lahodia oku, ale nesúhlasia s tým, čo môžeme vidieť na modernom termináli *xterm*.

Vzhľad je kritický vo výpisoch programov, hlavne shellovských skriptoch Unixu. Balík *upquote* modifikuje chovanie prostredia `verbatim` tak, aby bol výstup jasnejšou reprezentáciou toho, čo musí používateľ napísať (napr. do skriptu).

`upquote.sty`: `macros/latex/contrib/upquote/upquote.sty`

155. Predefinovanie `\the`-príkazov počítadiel

Kedykoľvek požadujete od LATEXu nové počítadlo, LATEX vytvorí mnoho „zákulisných" príkazov ako aj počítadlo samotné.

`\newcounter{fred}` okrem iného vytvorí príkaz `\the`*`fred`*, ktorý sa pri sádzaní expanduje na hodnotu počítadla „`fred`".

Definícia `\the`*fred* by mala vyjadrovať hodnotu počítadla. Je takmer vždy chybou používať príkaz na vytvorenie niečoho iného. Hodnota môže byť rozumne vyjadrená ako arabské, rímske, prípadne grécke číslo, ako abecedný výraz alebo dokonca ako postupnosť (alebo vzor) symbolov. Ak sa potrebujete rozhodnúť, či príkaz `\the`*fred* predefinovať, zvážte, čo by sa mohlo stať, ak sa tak rozhodnet učiniť.

Takže ak napríklad chcete, aby čísla sekcií boli ukončené bodkou, môžete upraviť `\thesection` tak, aby sa expandoval s ukončovacou bodkou. Takáto zmena `\thesection` však má vplyv na definúciu `\thesubsection`: budete musieť predefinovať kopu iných vecí. Radšej použite štandardné techniky na prispôsobenie a prezentáciu čísel sekcií (viď Otázku 273).

Teraz predpokladajme, že chcete, aby číslo stránky bolo na konci každej stránky obklopené pomlčkami („–nnn–“). Chcete to dosiahnuť predefinovaním `\thepage` (všetky čísla stránok v obsahu budú pravdepodobne rovnakého tvaru) alebo predefinovaním `\pageref` referencií? V tomto prípade je najlepšie spraviť modifikáciu predefinovaním štýlu strany, napríklad balíkom *fancyhdr*.

156. Formát obsahov, ...

Formát položiek obsahu je ovládaný niekoľkými internými príkazmi (spomínané v sekcii 2.4 *The LaTeX Companion* — viď Otázku 23). Príkazy `\@pnumwidth`, `\@tocrmarg` a `\@dotsep` ovládajú priestor pre čísla strán, odsadenie od pravého okraju a separáciu bodiek v bodkovaných vodiacich linkách. Séria príkazov `\l@\emph {xxx}`, kde *xxx* je meno nadpisu sekcie (ako `kapitola` alebo `sekcia`, ...), ovláda rozvrhnutie príslušného nadpisu vrátane miesta pre číso sekcie. Všetky tieto interné príkazy môžu byť individuálne predefinované, aby mali požadovaný efekt.

Balík *tocloft* poskytuje sadu príkazov používateľskej úrovne, ktoré môžete použiť na zmenu formátovania obsahu. Keďže pre zoznamy obrázkov a zoznamy tabuliek sú použité presne rovnaké mechanizmy, rozvrhnutie týchto sekcií môžete ovládať rovnakým spôsobom.

Triedy `KOMA-Script` poskytujú voliteľne variantu štruktúry obsahu a počítajú priestor potrebný pre čísla automaticky. Trieda `memoir` obsahuje funkcionalitu *tocloft*.

sada `KOMA script`: `macros/latex/contrib/koma-script`
`memoir.cls`: `macros/latex/contrib/memoir`
`tocloft.sty`: `macros/latex/contrib/tocloft`

157. Pekne číslované zoznamy

Balík *enumerate* vám dovoľuje kontrolovať zobrazenie počítadla zoznamu. Balík pridáva prostrediu `enumerate` voliteľný paramater, ktorý sa používa na špecifikáciu rozvrhnutia návestí. Parameter obsahuje typ číslovania (‘`1`’ pre arabské číslice, ‘`a`’ alebo ‘`A`’ pre abecedné „číslovanie“ a ‘`i`’ alebo ‘`I`’ pre rímske číslice) a dekorácie číslovania. Takže napríklad

```
\usepackage{enumerate}
...
\begin{enumerate}[(a)]
\item ...  ...
\end{enumerate}
```

začne zoznam, ktorého návestia sú (a), (b), (c), ..., zatiaľ čo

```
\usepackage{enumerate}
...
\begin{enumerate}[I/]
\item ...  ...
\end{enumerate}
```

začne zoznam, ktorého návestia sú I/, II/, III/, ...

Balík *paralist*, ktorého hlavným účelom je kompaktnosť zoznamov (viď Otázku 158), poskytuje rovnaké prostriedky pre svoje prostredia podobné `enumerate`.

Ak potrebujete netradičný dizajn balík, *enumitem* vám poskytne flexibilitu na vytvorenie vlastného. Hlúpy „rímsky“ príklad dosiahneme nasledovne:

```
\usepackage{enumitem}
...
```

```
\begin{enumerate}[label=\Roman{*}/]
\item ...  ...
\end{enumerate}
```

Všimnite si, že „*“ v hodnote kľúča značí počítadlo zoznamu tejto úrovne. Môžete tiež manipulovať s formátom odkazov na návestia položiek zoznamu:

```
\usepackage{enumitem}
...
\begin{enumerate}[label=\Roman{*}/,
  ref=(\roman{*})]
\item ...  ...
\end{enumerate}
```

aby formát odkazov na položky zoznamu vyzeral ako (i), (ii), (iii) a tak ďalej.

Trieda `memoir` obsahuje funkcie, ktoré zodpovedajú funkciám v balíku *enumerate* a obsahuje podobnú funkcionalitu aj pre zoznamy `itemize`.

`enumerate.sty`: distribuované ako súčasť `macros/latex/required/tools`
`enumitem.sty`: distribuované ako súčasť `macros/latex/contrib/bezos`
`memoir.cls`: `macros/latex/contrib/memoir`
`paralist.sty`: `macros/latex/contrib/paralist`

158. Ako zredukovať medzery v zoznamoch

Lamportova kniha (viď Otázku 23) udáva rozličné parametre pre rozvrhnutie zoznamu (ako napríklad `\topsep`, `\itemsep` a `\parsep`), ale nespomína už, že sú nastavované automaticky v zozname samotnom. Funguje to tak, že každý zoznam vykoná príkaz `\@list<hĺbka>` (hĺbka sa objavuje vo forme malých rímskych číslic). `\@listi` je naviac zvyčajne resetované pri zmene veľkosti fontu. Výsledok je, že pre užívateľa je dosť zložité kontrolovať medzery v zoznamoch (medzery medzi položkami zoznamu). Správnym riešením je samozrejme používanie dokumentovej triedy navrhnutej s miernejšími medzerami v zoznamoch, ale všetci vieme, že k takýmto veciam sa ťažko dostať. Trieda `memoir` nebola navrhnutá pre kompaktnejšie zoznamy, ale poskytuje kontrolu nad medzerami v zoznamoch.

Existujú balíky poskytujúce určitú kontrolu nad medzerami v zoznamov, málokedy však adresujú oddelenie od okolitého textu (definované príkazom `\topsep`). Balík *expdlist*, spomedzi všetkých svojich ovládacích prvkov vzhľadu `description` zoznamov poskytuje parameter kompaktnosti (viď dokumentáciu); balík *mdwlist* poskytuje príkaz `\makecompactlist` pre uživateľské definície zoznamov a používa ho na definície kompaktných zoznamov `itemize*`, `enumerate*` a `description*`. V skutočnosti môžete celkom jednoducho definovať podobné zoznamy — napríklad:

```
\newenvironment{itemize*}%
  {\begin{itemize}%
    \setlength{\itemsep}{0pt}%
    \setlength{\parskip}{0pt}}%
  {\end{itemize}}
```

Balík *paralist* poskytuje veľa prístupov pre kompaktnejšie zoznamy:

- jeho prostredie `asparaenum` formátuje každú položku, ako keby to bol odstavec zavedený počítadlovým návestím (čo v prípade, že texty položiek sú dlhé, šetrí miesto),
- jeho prostredie `compactenum` je rovnakým druhom kompaktného zoznamu ako v *expdlist* a *mdwlist*, a
- jeho prostredie `inparaenum` vytvára zoznam „v odstavci“, t.j. bez zlomov medzi položkami, čo veľmi dobre šetrí miesto, ak sú texty položiek krátke.

Balík manipuluje s návestiami svojho prostredia `enumerate` ako balík *enumerate* (viď Otázku 157).

Paralist taktiež poskytuje ekvivalenty `itemize` (`asparaitem`, ...) a `description` (`asparadesc`, ...).

Maximum v kompaktnosti (každého druhu) je poskytované balíkom *savetrees*, zahrnutá je aj kompaktnosť zoznamov. Hlavnou úlohou balíku je šetrenie miestom na každom kroku: ak máte nejaké dizajnové obmedzenia, nepoužívajte ho!

Balíky *expdlist*, *mdwlist* a *paralist* poskytujú iné prostriedky konfigurácie zoznamov. Ak potrebujete niektorý z balíkov pre inú konfiguráciu zoznamu než kompaktnosť, pravdepodobne by ste nemali skúšať nižšie uvedené postupy.

Balík *enumitem* pre maximálnu flexibilitu (vrátane manipulácie s `\topsep`) povoľuje prispôsobenie parametrov zoznamu pomocou formátu „*kľúč*=<hodnota>“. Môžete teda napísať

```
\usepackage{enumitem}
...
\begin{enumerate}[topsep=0pt, partopsep=0pt]
\item ...  ...
\end{enumerate}
```

čím potlačíte všetky medzery pod a nad vaším zoznamom. *Enumitem* tiež dovoľuje manipulovať s formátom návestia (základnejším spôsobom ako balík *enumerate* — Otázka 157).

`enumerate.sty`: distribuované ako súčasť `macros/latex/required/tools`
`enumitem.sty`: distribuované ako súčasť `macros/latex/contrib/bezos`
`expdlist.sty`: `macros/latex/contrib/expdlist`
`memoir.cls`: `macros/latex/contrib/memoir`
`mdwlist.sty`: distribuované ako súčasť `macros/latex/contrib/mdwtools`
`paralist.sty`: `macros/latex/contrib/paralist`
`savetrees.sty`: `macros/latex/contrib/savetrees`

159. Prerušenie číslovaných zoznamov

Často je vhodné mať vysvetľujúci text mimo zoznamu, medzi za sebou idúcimi položkami. V prípade `itemize` zoznamov to nie je problém, keďže neexistuje nič, čo by odlíšilo za sebou idúce položky. V prípade `description` zoznamov sú návestia položiek pod kontrolou užívateľa, takže automatická kontinuita neexistuje.

Pri `enumerate` zoznamoch sú návestia generované automaticky a sú citlivé na kontext, takže kontext (v tomto prípade je to stav počítadla) musí zostať zachovaný.

Jedným z prístupov je zapamätať si stav číslovania do vlastnej premennej a neskôr (pri pokračovaní číslovania) ho obnoviť:

```
\newcounter{saveenum}
 ...
\begin{enumerate}
  ...
  \setcounter{saveenum}{\value{enumi}}
\end{enumerate}
<vysvetľujúci text>
\begin{enumerate}
  \setcounter{enumi}{\value{saveenum}}
  ...
\end{enumerate}
```

Toto je rozumné v malých dávkach. Problémom (okrem „ukecanosti“ riešenia) je správne nastavenie úrovne („mám použiť čítač `enumi`, `enumii`, ...“) a zapamätanie si, že nemáme vnorovať prerušenia (t.j. nemať ďalší zoznam, ktorý je tiež prerušený) do vysvetľujúceho textu.

Balík *mdwlist* definuje príkazy `\suspend` a `\resume`, ktoré proces zjednodušujú:

```
\begin{enumerate}
  ...
\suspend{enumerate}
<vysvetľujúci text>
\resume{enumerate}
  ...
\end{enumerate}
```

Balík povoľuje zadať voliteľný názov (ako pri `\suspend[id]{enumerate}`), aby ste mohli identifikovať konkrétne prerušenie, a tým poskytuje možnosť manipulovania s vnorenými prerušeniami.

Ak prerušujete pekne číslované zoznamy (viď Otázku 157), musíte pri pokračovaní v zozname znovu dodať voliteľné parametre „item label layout“ (položka, návestie, rozvrhnutie) vyžadované balíkom *enumerate*, nezávisle na tom, ktorý z uvedených postupov ste použili. Úloha je v prípade balíku *mdwlist* trochu zdĺhavá, keďže voliteľný parameter musí byť celý zapuzdrený vo vnútri voliteľného parametru pre `\resume`, čo vyžaduje zátvorky naviac:

```
\begin{enumerate}[\textbf{Item} i]
  ...
\suspend{enumerate}
<vysvetlivky>
\resume{enumerate}[{[\textbf{Item} i]}]
...
\end{enumerate}
```

`enumerate.sty`: distribuované ako súčasť `macros/latex/required/tools`
`mdwlist.sty`: distribuované ako súčasť `macros/latex/contrib/mdwtools`

160. Dizajn tabuliek

V poslených rokov množstvo autorov argumentovalo, že príklady vytvorené Lamportom v jeho LaTeX manuálu (viď Otázku 23) obmedzovali štýl autorov a viedli k extrémne úbohému dizajnu tabuliek. V skutočnosti je ťažke čo i len zistiť, čo mnohé z príkladov Lamportovej knihy znamenajú.

Kritika sa zameriava na prehnané použitie liniek (horizontálnych aj vertikálnych) a na slabé vertikálne odstupy poskytované Lamportovymi makrami.

Problém vertikálnych odstupov je hocikým jednoducho viditeľný a je riešený v mnohých balíkoch — viď Otázku 162.

Argument týkajúci sa liniek je prezentovaný v skvelej eseji, ktorá tvorí predslov dokumentácie balíku *booktabs* od Simona Feara.

Lamportov LaTeX neflexibilne trval na tom, že popisky majú byť na spodku tabuľky. Kedže sa tabuľka môže rozprestierať na viacerých stranách, tradičná typografia umiestňuje popisky navrch plávajúceho prostredia tabuľky. Príkaz `\caption` bude mať zlú pozíciu (o `10pt`), ak napíšete:

```
\begin{table}
  \caption{Ukážková tabuľka}
  \begin{tabular}{...}
    ...
  \end{tabular}
\end{table}
```

Balík *topcapt* tento problém rieši:

```
\usepackage{topcaption}
...
\begin{table}
  \topcaption{Ukážková tabuľka}
  \begin{tabular}{...}
    ...
  \end{tabular}
\end{table}
```

Triedy `KOMA-script` poskytujú podobný príkaz `\captionabove`, obsahujú tiež voľbu triedy, ktorá zaistí, že v prostredí tabuliek `\caption` *znamená* `\captionabove`.

Manuálny postup je celkom jednoduchý: *topcapt* prepne hodnoty parametrov systému LaTeXe `\abovecaptionskip` (štandardne `10pt`) a `\belowcaptionskip` (štandardne `0pt`), takže:

```
\begin{table}
  \setlength{\abovecaptionskip}{0pt}
  \setlength{\belowcaptionskip}{10pt}
  \caption{Ukážková tabuľka}
  \begin{tabular}{...}
    ...
  \end{tabular}
\end{table}
```

spraví to, čo chceme. (Balík je sám o sebe mierne komplikovanejší. . .)

booktabs.sty: macros/latex/contrib/booktabs
sada KOMA script: macros/latex/contrib/koma-script
topcapt.sty: macros/latex/contrib/misc/topcapt.sty

161. Tabuľky pevnej šírky

Existujú dva základné spôsoby na tvorbu tabuliek pevnej šírky v LaTeXu: môžete roztiahnuť úseky (medzery) medzi stĺpcami, alebo môžete roztiahnuť určité bunky tabuľky.

Základný LaTeX môže roztiahnuť medzery. Prostredie `tabular*` si vezme jeden parameter navyše (pred parametrom rozvrhnutia `clpr`), ktorý akceptuje popis dĺžky (môžete vložiť veci ako „`15cm`“ alebo „`\columnwidth`“). V parametri rozloženia `clpr` musíte mať v direktíve `@{}` príkaz `\extracolsep`. Príklad:

```
\begin{tabular*}{\columnwidth}%
                {@{\extracolsep{\fill}}lllr}
```

`\extracolsep` je tiež použitý na všetky medzistĺpcové medzery napravo. Ak nechcete roztiahnuť všetky medzery, pridajte `\extracolsep{0pt}`, čím zrušíte pôvodné.

Balík *tabularx* definuje extra špecifikáciu `clpr` stĺpca, `X`. Stĺpce `X` sa správajú ako `p` stĺpce, ktoré sa rozširujú, aby zabrali dostupné miesto. Ak je v tabuľke prítomný viac než jeden `X` stĺpec, voľné miesto je medzi tieto stĺpce rozdelené .

Balík *tabulary* (od rovnakého autora) poskytuje spôsob „vyvažovania“ miesta zabraného stĺpcami tabuľky. Balík definuje špecifikácie stĺpcov zdieľajúcich priestor `C`, `L`, `R` a `J` so zarovnaniami na stred, vľavo, vpravo a zarovnaním na obe strany. Balík skúma, aký by bol každý zo stĺpcov prirodzene dlhý (t.j. na papiery nekonečnej šírky) a alokuje príslušné miesto pre každý stĺpec. Prítomné sú aj kontroly, ktoré dohliadajú na to, aby veľké položky nespôsobili kolaps zvyšných stĺpcov (pre každý stĺpec existuje maximálna šírka) a aby sa drobné veci nezmenšili pod špecifikovanú hodnotu minima. Táto práca samozrejme znamená, že balík musí vysádzať každý riadok niekoľkokrát, takže veci s vedľajšími efektami (napríklad čítač produkujúci číslo riadku) sú nespoľahlivé a nemali by radšej byť používané.

ltxtable kombinuje vlastnosti balíkov *longtable* a *tabularx*. Radšej si prečítajte dokumentáciu, pretože použitie balíku je značne odlišné.

ltxtable.sty: distribuované ako súčasť macros/latex/contrib/carlisle
tabularx.sty: distribuované ako súčasť macros/latex/required/tools
tabulary.sty: distribuované ako súčasť macros/latex/contrib/carlisle

162. Rozostupy riadkov v tabuľkách

TeXovské/LaTeXovské mechanizmy správy medzier medzi riadkami (tzv. „leading“) sa spoliehajú na TeXovský algoritmus výstavby odstavcov, ktorý porovnáva tvary za sebou idúcich riadkov a upravuje medzeru medzi nimi.

Tieto mechanizmy nemôžu pri vytváraní tabuľky fungovať rovnakým spôsobom, pretože tvorba odstavcov samotné riadky nevidí. Výsledkom je, že tabuľky sú občas vysádzané s riadkami nepríjemne blízko pri sebe (alebo občas aj veľmi ďaleko).

Tradičný typograf by upravil medzeru medzi riadkami tabuľky pomocou podpery („strut“; jedná sa o kovový oddelovač). Používateľ TeXu môže spraviť presne to isté: väčšina balíkov makier definuje príkaz `\strut`, ktorý udáva vhodnú medzeru vzhľadom na aktuálnu veľkosť textu. Umiestnenie príkazu `\strut` na koniec problémového riadku tabuľky je najjednoduchšie riešenie problému (ak funguje). Ďalšie uvedené riešenia sú špecifické pre LaTeX, ale niektoré sa dajú jednoducho preložiť na príkazy `Plain` TeXu.

Ak sa vo vašej tabuľke ukazuje systematický problém (t.j. každý riadok je posunutý o rovnaký úsek) použite príkaz `\extrarowheight` definovaný balíkom *array*:

```
\usepackage{array}% v preambule
...
\setlength{\extrarowheight}{length}
\begin{tabular}{....}
```

Na napravenie jediného riadku, ktorého zlé nastavenie príkaz `\strut` neopravuje, môžete postupovať vlastnou definíciou pomocou `\rule{0pt}{length}` — čo sa podobá definícii príkazu `\strut`. Balík *bigstrut* definuje podobný príkaz, ktorý môžete na tento účel použiť: `\bigstrut` sa sám odkryje nad aj pod aktuálnym riadkom, `\bigstrut[t]` iba nad riadkom a `\bigstrut[b]` iba pod riadkom.

Dostupné sú však aj obecné riešenia. Balík *tabls* na konci každého riadku tabuľky automaticky generuje podperu správnej veľkosti. Nevýhodou je pomalosť (kedže stojí v ceste všetkému vo vnútri tabuľky) a (nedostatočná) kompatibilita s inými balíkmi.

Balík *booktabs* obsahuje provokatívnu úvahu o tom, ako by mali byť navrhované tabuľky. Kedže problém rozostupov riadkov v tabuľkách sa najčastejšie objavuje pri kolíziách s čiarami, autorova myšlienka, že používatelia LaTeXu majú vo zvyku veľmi často linkovať svoje tabuľky, je zaujímavá. Balík na podporu autorovej schémy poskytuje linkovacie príkazy, ale stará sa aj o medziriadkové medzery. Najnovšie vydania *booktabs* sa pýšia kompatibilitou s balíkmi ako *longtable*.

Dokumentáciu *bigstrut* aj *tabls* môžete nájsť v podobe komentárov v samotných súboroch týchto balíkov.

`array.sty`: distribuované ako súčasť `macros/latex/required/tools`
`bigstrut.sty`: distribuované ako súčasť `macros/latex/contrib/multirow`
`booktabs.sty`: `macros/latex/contrib/booktabs`
`tabls.sty`: `macros/latex/contrib/misc/tabls.sty`

163. Tabuľky dlhšie než jedna strana

Tabuľky sú štandardne sádzané do svojich vlastných boxov, čo má za následok, že sa na hranici stránky nerozdelia. Bohužiaľ náš svet produkuje tabuľky dlhšie než jedna strana a my ich potrebujeme vysádzať.

Pre jednoduché tabuľky (ktorých tvar je veľmi pravidelný) môže byť najjednoduchším riešením prostredie `tabbing`, ktorého nastavenie je trochu zdĺhavé, ale nevyžaduje všetko na jednu stranu.

Balík *longtable* v prvom behu vybuduje celú tabuľku (po kúskoch), a potom použije informáciu, ktorú zapísal do `.aux` súboru, aby dostal správne umiestnenie (obvykle sa mu to podarí v dvoch priechodoch). Kedže balík má v čase konečného umiestňovania prehľad o celej tabuľke, tabuľka je vysádzaná jednotne po celej svojej dĺžke (stĺpce na jednotlivých stranách si navzájom zodpovedajú). O *longtable* je známe, že nespolupracuje s inými balíkmi, spolupracuje však s balíkom *colortbl*. Autor poskytol balík *ltxtable*, ktorý obsahuje väčšinu prostriedkov balíku *tabularx* (viď Otázku 161) aj pre dlhé tabuľky: majte sa na pozore pred zvláštnymi obmedzeniami jeho použitia — každá dlhá tabuľka by mala byť vo vlastnom súbore a mala by byť vložená `\LTXtable{width}{file}`. Kedže viacstranové tabuľky balíku *longtable* nemôžu existovať vo vnútri plávajúcich prostredí, poskytuje balík možnosti nadpisov v samotnom prostredí `longtable`.

Zdanlivou alternatívou k *ltxtable* je *ltablex*. Je však zastaralý a nie je úplne funkčný. Jeho najväčší problém je prísne obmedzená kapacita pamäte (*longtable* nie je tak obmedzený, za čo platí komplikovaným kódom). *ltablex* spracuje iba relatívne malé tabuľky, ale jeho používateľské rozhranie je oveľa jednoduchšie ako u *ltxtable*, takže ak pre vás obmedzenia nie sú problémom, môžete to skúsiť.

Balík *supertabular* začne a ukončí prostredie `tabular` prekaždú stranu tabuľky. Výsledkom je, že každá „strana tabuľky" sa skompiluje nezávisle a šírky navzájom si zodpovedajúcich stĺpcov sa môžu líšiť. Ak však príslušnosť stĺpcov nehrá úlohu alebo vaše stĺpce majú pevnú šírku, *supertabular* má vo vašom prípade veľkú výhodu v tom, že veci spracuje v jednom priechode.

Longtable aj *supertabular* dovoľujú pre tabuľku definovať nadpisy a poznámky pod čiarou. *longtable* odlišuje nadpisy a poznámky pri prvej a poslednej časti.

Balík *xtab* opravuje niektoré neblahosti *supertabular* a taktiež poskytuje prostriedok pre „posledný nadpis" (hoci toto zruší výhodu *supertabular* — spracovanie v jednom behu).

Balík *stabular* poskytuje jednoducho použiteľné rozšírenie prostredia `tabular`, ktoré umožňuje sadzbu tabuliek bežiacich cez koniec strany. Balík obsahuje aj rôzne užitočné rozširenia, ale neobsauje prostriedky nadpisov a poznámok pod čiarou ako väčšie balíky.

Dokumentáciu *ltablex* môžete nájsť v súbore balíku.

`longtable.sty`: distribuované ako súčasť `macros/latex/required/tools`

`ltablex.sty`: `macros/latex/contrib/ltablex/ltablex.sty`
`ltxtable.sty`: generované použitím `macros/latex/contrib/carlisle/ltxtable.tex`
`stabular.sty`: distribuované ako súčasť `macros/latex/contrib/sttools`
`supertabular.sty`: `macros/latex/contrib/supertabular`
`xtab.sty`: `macros/latex/contrib/xtab`

164. Ako zmeniť zarovnanie buniek tabuľky

Často je potrebné zmeniť zarovnanie `p` (odstavcovej; „paragraph“) bunky tabuľky, obvykle však nastávajú na konci konci riadku tabuľky problémy. S takouto `p` bunkou:

```
... & \centering blah ... \\
```

sa ľahko stretnete s chybami sťažujúcimi sa na „misplaced `\noalign`“ (zle umiestnený `\noalign`), „extra alignment tab“ (zarovnávací tabulátor navyše), prípadne na niečo podobné. Problém je, že príkaz `\\` znamená rozličné veci za rozličných okolností: prostredie `tabular` prepína jeho význam na hodnotu použiteľnú v tabuľke, `\centering`, `\raggedright` a `\raggedleft` ju menia na niečo nekompatibilné. Všimnite si, že problém vzniká iba v poslednej bunke riadku: keďže každá bunka je sádzaná do boxu, jej nastavenia sa strácajú u `&` (alebo `\\`), ktorý ho ukončuje.

Jednoduchým (starým) riešením je zachovanie významu `\\`:

```
\newcommand\PBS[1]{\let\temp=\\%
  #1%
  \let\\=\temp
}
```

čo použijete napríklad nasledovne:

```
... & \PBS\centering blah ... \\
```

Technika používajúca `\PBS` bola vyvinutá v dobe LaTeX 2.09, pretože skutočná hodnota `\\`, ktorú používalo prostredie `tabular`, bola dostupná iba ako interný príkaz. V súčasnosti je hodnota príkaz verejný a v princípe ju môžete použiť explicitne:

```
... & \centering blah ... \tabularnewline
```

Toto môže byť zakomponované do jednoduchého makra:

```
\newcommand{\RBS}{\let\\=\tabularnewline}
...
... & \centering\RBS blah ... \\
```

Použitie:

```
... & \centering\RBS blah ... \\
```

(zapamätajte si, že spätné lomítko zachovávate s `\PBS` *pred* príkazom, ktorý ju mení a obnovujete s `\RBS` *po* príkaze; `\RBS` je v skutočnosti mierne preferovaný, ale starý trik tu zostáva).

Fígle `\PBS` a `\RBS` nám dobre slúžia tiež v špecifikáciach formátu poľa („field format“) v preambule pri balíku *array*:

```
\begin{tabular}{...>{\centering\RBS}%
                   p{50mm}}
...
```

alebo

```
\begin{tabular}{...>{\PBS\centering}%
                   p{50mm}}
...
```

`array.sty`: distribuované ako súčast `macros/latex/required/tools`

165. Ako zmeniť celý riadok tabuľky

Každá bunka tabuľky je sádzaná do boxu, takže zmena štýlu fontu (prípadne niečoho iného) platí iba po koniec bunky. Ak máte tabuľku s mnohými bunkami alebo dlhú tabuľku, v ktorej potrebujete zvýrazniť množstvo riadkov, pridávanie štýlu fontu do každej bunky bude veľmi zdĺhavé.

S balíkom *array* môžete definovať modifikátory stĺpcov, ktoré zmenia štýl písma pre celý *stĺpec*. S trochou jemnosti môžete zmeniť modifikátory tak, aby upravovali riadky namiesto stĺpcov. Postupujeme nasledovne:

```
\usepackage{array}
\newcolumntype{$}{>{%
  \global\let\currentrowstyle\relax}%
}
\newcolumntype{^}{>{\currentrowstyle}}
\newcommand{\rowstyle}[1]{%
  \gdef\currentrowstyle{#1}%
  #1\ignorespaces
}
```

Teraz pred prvý špecifikátor stĺpcu vložíme „`$`“ a „`^`“ pred modifikátory nasledujúcich. Potom použijeme `\rowstyle` na začiatku každého riadku, ktorý chceme modifikovať:

```
\begin{tabular}{|$l|^l|^l|}   \hline
  \rowstyle{\bfseries}
  Nadpis & Veľký a & Tučný \\ \hline
  Text & text & text       \\
  Text & text & text       \\
  \rowstyle{\itshape}
  Text & text & v~kurzíve  \\
  Text & text & text       \\ \hline
\end{tabular}
```

Balík *array* funguje s rôznymi inými prostrediami typu `tabular` z iných balíkov (napríklad `longtable`), tento trik však bohužiaľ nefunguje vždy.
`array.sty`: distribuované ako súčasť `macros/latex/required/tools`

166. Spájanie buniek v stĺpci tabuľky

Nie je ťažké vymyslieť dizajn tabuľky, ktorý bude vyžadovať, aby sa určitá bunka rozprestierala na viacerých riadkoch. Príkladom môže byť tabuľka, v ktorej najľavejší stĺpec označuje zbytok tabuľky. Toto môžete spraviť (v jednoduchých prípadoch) použitím diagonálneho rozdelenia v rohových bunkách. Spomenutá technika vás však prísne obmedzuje v tom, čo môže byť použité ako obsah bunky.

Balík *multirow* vám umožňuje vytvárať viacriadkové bunky veľmi jednoducho. Pre najjednoduchšie použitie môžete napísať:

```
\begin{tabular}{|c|c|}
\hline
\multirow{4}*{Bežný text g}
      & Stĺpec g2a\\
      & Stĺpec g2b \\
      & Stĺpec g2c \\
      & Stĺpec g2d \\
\hline
\end{tabular}
```

a *multirow* umiestni „Bežný text g“ na vertikálny stred priestoru definovaného ostatnými riadkami. Všimnite si, že riadky neobsahujúce špecifikáciu „multi-row“ musia mať prázne bunky, v ktorých sa viacriadková bunka objaví.

„`*`“ môžete nahradiť špecifikáciou šírky stĺpcu. Parameter v tomto prípade môže obsahovať nepodmienené zlomy riadku:

```
\begin{tabular}{|c|c|}
\hline
\multirow{4}{25mm}{Bežný\\text g}
      & Stĺpec g2a\\
```

```
        & Stĺpec g2b \\
        & Stĺpec g2c \\
        & Stĺpec g2d \\
   \hline
   \end{tabular}
```
Podobný efekt (s možnosťou trochu väčšej sofistikovanosti) môžeme dosiahnuť vložením menšej tabuľky, ktorá zoradí text, do *-deklarovaného `\multirow`.

Príkaz `\multirow` môže byť použitý aj na písanie vertikálneho textu oboma smermi (s pomocou balíkov *graphics* alebo *graphicx*, ktoré poskytujú príkaz `\rotatebox`):
```
   \begin{tabular}{|l|l|}
   \hline
   \multirow{4}*{\rotatebox{90}{ahoj}}
        & Stĺpec g2a\\
        & Stĺpec g2b \\
        & Stĺpec g2c \\
        & Stĺpec g2d \\
   \hline
   \end{tabular}
```
Takto dostanete text bežiaci nahor; pre opačný smer použite uhol `-90`.

Multirow je nastavený tak, aby spolupracoval s balíkom *bigstrut* (ktorý je spomenutý v odpovedi na Otázku 162). Na vyjadrenie počtu riadkov vo viacriadkovom prostredí, ktoré boli otvorené pomocou `\bigstrut`, použijete voliteľný parameter príkazu `\multirow`.

Dokumentácia *multirow* a *bigstrut* sa dá nájsť v súboroch balíkov v podobe komentárov.
`bigstrut.sty`: distribuované ako súčasť `macros/latex/contrib/multirow`
`multirow.sty`: `macros/latex/contrib/multirow`

167. Plávajúce prostredia v prostredí multicolumn

Ak použijete
```
\begin{figure}
  ...
\end{figure}
```
v prostredí `multicols`, obrázok (figure) sa neobjaví. Ak namiesto toho použijete
```
\begin{figure*}
  ...
\end{figure*}
```
obrázok sa roztiahne na šírku stránky (ako prostredie `figure*` v štandardnej možnosti LaTeXu, `twocolumn`).

Pomocou umiestnenia „`[H]`" zavedeného balíkom *float* môžeme mať jednostĺpcové obrázky a tabuľky s popiskami. Možno sa však bude treba pohrať s umiestnením, pretože nebudú „plávať" a budú javiť iné zvláštne znaky chovania (ako nenápadné „pretekanie" mimo koniec stĺpca na konci prostredia `multicols`).
`float.sty`: `macros/latex/contrib/float`
`multicol.sty`: distribuované ako súčasť `macros/latex/required/tools`

168. Vertikálne rozloženie strán obsahujúcich floaty

LaTeX štandardne centruje floaty na plávajúcej strane (floatovej) vertikálne; nie sme sami, komu sa to nepáči. Ovládanie umiestnenia je bohužiaľ „zakopané" v interných príkazoch LaTeXu, takže pri zmene rozloženia je nutná opatrnosť.

Plávajúce strany používajú tri LaTeXovské dĺžky (t.j. TeXovské skipy) na definíciu svojho rozloženia:
`\@fptop` definuje vzdialenosť od začiatku strany k začiatku prvého floatu,
`\@fpsep` definuje odstup medzi floatmi,
`\@fpbot` definuje vzdialenosť od konca posledného floatu na strane ku koncu strany.

(Výstupná rutina v skutočnosti umiestňuje `\@fpsep` nad každý float, takže `\@fptop` obvykle obsahuje opravu.)

LaTeXovské štandardy sú:

```
\@fptop = (0pt + 1fil) − \@fpsep
\@fpsep = 8pt + 2fil
\@fpbot = 0pt + 1fil
```

takže medzery sa rozširujú, aby vyplnili priestor, ktorý nie je zabraný floatami, ale ak je na strane viac než jeden float, medzera medzi nimi sa rozšíri na dvojnásobok horného a spodného priestoru.

Najčastejšia potreba, aby floaty začínali na začiatku strany, sa dá jednoducho uspokojiť:

```
\makeatletter
\setlength{\@fptop}{-\@fpsep}
\makeatother
```

Pamätajte si, že toto je „globálne“ nastavenie (najlepšie vykonané v súbore triedy alebo prinajhoršom v preambule dokumentu). Zmena pre jediný float bude pravdepodobne (prinajlepšom) celkom komplikovaná.

169. Nečíslované oddíly v obsahu

Nejjednodušším způsobem, jak vytvořit v obsahu položku pro takové zvláštní části textu jako jsou například předmluvy, je užití čítače `secnumdepth`, který je popsán v dodatku C manuálu LaTeXu. Například:

```
\setcounter{secnumdepth}{-1}
\chapter{Předmluva}
```

Samozřejmě, že před opětovným použitím číslovaného oddílu je potřeba nastavit parametr `secnumdepth` zpět na jeho obvyklou hodnotu (ve standardních stylech 2).

Podobná nastavení jsou provedena automaticky v knižní třídě LaTeXu příkazy `\frontmatter` a `\backmatter`.

Následuje vysvětlení. Použije-li se `\chapter` bez hvězdičky:

1. vloží se něco do `.toc` souboru;
2. je-li `secnumdepth` ≥ 0, zvětší se čítač kapitoly o jedničku a vysází se jeho hodnota.
3. vysází se název kapitoly.

Ostatní příkazy na rozdělování textu (`\section`, `\subsection`,...) pracují obdobně, avšak testují jinou hodnotu.

Hodnota čítače `tocdepth` ovlivňuje, který nadpis bude vytištěn v obsahu. Běžně se má jeho hodnota nastavit v preambuli dokumentu, pak je pro celý dokument konstantní. Balík *tocvsec2* (`macros/latex/contrib/tocvsec2`) nabízí celkem vyhovující rozhraní umožňující měnit hodnotu čítače `secnumdepth` a/nebo `tocdepth` kdekoliv v těle dokumentu. To nabízí docela slušnou nezávislou kontrolu nad číslováním položek v obsahu.

Balík *abstract* (viz Otázku 216) zahrnuje volbu pro přidání abstraktu do obsahu, zatímco balík *tocbibind* (`macros/latex/contrib/tocbibind`) má volbu pro vložení samotného obsahu, bibliografie, rejstříku atd. do obsahu.

170. Obsahy a iné veci po kapitolách

Častý štýl „malého“ obsahu pre každú časť, kapitolu alebo dokonca sekciu je podporovaný balíkom *minitoc*. Balík podporuje tiež mini-zoznamy tabuliek a obrázkov; ale mini-bibliografie sú (čo si všíma aj dokumentácia) iným problémom — viď Otázku 113.

Základný princíp balíku spočíva v generovaní malého `.aux` súboru pre každú kapitolu a jeho spracovaní v kapitole. Jednoduché použitie môže byť:

```
\usepackage{minitoc}
...
\begin{document}
...
\tableofcontents
\dominitoc \listoffigures \dominilof ...
```

```
\chapter{blah blah}
\minitoc \mtcskip \minilof ...
```
hoci je možných veľa možností rozvíjania (napríklad `\minitoc` pre každú kapitolu nie je potrebné).

Babel minitoc nepozná, ale *minitoc* sa stará aj o dokumenty v iných jazykoch než v angličtine — dostupný je široký výber. Aktuálne verzie balíku *hyperref* však s `\minitoc` zaobchádzajú ako so „skutočnými" obsahmi.

Dokumentácia je dosť rozsiahla a celko dobre čitateľná: spracujte súbor distribúcie `minitoc.tex`.

`babel.sty`: `macros/latex/required/babel`
`hyperref.sty`: `macros/latex/contrib/hyperref`
`minitoc.sty`: `macros/latex/contrib/minitoc`

171. Viacnásobné indexy

Štandardné indexovacie schopnosti LaTeXu (tie, čo sú poskytované balíkom *makeidx*) umožňujú mať vo vašom dokumente iba jeden index. Dokonca aj neveľké dokumenty môžu byť vylepšené samostatnými indexami pre rozdielne témy.

Balík *multind* poskytuje jednoduché a priamočiare viacnásobné indexovanie. Označkujete každý príkaz `\makeindex`, `\index` a `\printindex` názvom súboru a indexovacie príkazy sú zapísané do (alebo čítané z) príslušného mena súboru s pripojenou zodpovedajúcou príponou (`.idx` alebo `.ind`). Príkaz `\printindex` je modifikovaný oproti štandardnej verzii z LaTeXu tak, aby nevytváral vlastný názov kapitoly alebo sekcie, vy rozhodujete, aké názvy (alebo dokonca úrovene sekcií) sa použijú, `\indexname` je úplne ignorované (viď Otázku 217).

Na vytvorenie „obecného" indexu a indexu „autorov" môžete napísať:
```
\usepackage{multind}
\makeindex{general}
\makeindex{authors}
...
\index{authors}{Robin Fairbairns}
...
\index{general}{FAQs}
...
\printindex{general}{Obecný index}
\printindex{authors}{Index autorov}
```
Pre dokončenie úlohy spustite LaTeX na vašom súbore dostatočne mnohokrát, aby návestia a iné veci boli stabilné, a potom spustite príkazy
```
makeindex general
makeindex authors
```
pred ďalším spustením LaTeXu. Všimnite si, že názvy indexových súborov, ktoré sa spracúvajú, nemusia mať žiadny vzťah k názvu LaTeXovského súboru, ktorý spracúvame. (K balíku sa nedodáva žiadna dokumentácia. Lepší popis, než vidíte hore, nedostanete.)

Balík *index* poskytuje vyčerpávajúcu sadu indexovacích prostriedkov, vrátane príkazu `\newindex`, ktorý dovoľuje definíciu nových štýlov indexu. `\newindex` si vezme „značku" (pre použitie v indexovacích príkazoch), náhrady za prípony súborov `.idx` a `.ind` a názov indexu. Môže tiež zmeniť položku, voči ktorej sa indexuje (môžete mať napríklad index umelcov referencovaných čislami obrázkov, na ktorých sú ukážky ich práce).

Pre použitie balíku *index* na vytvorenie indexu autorov spolu s „normálnym" indexom, začnite príkazmi v preambule:
```
\usepackage{index}
\makeindex
\newindex{aut}{adx}{and}{Index mien}
```
Príkazy nahrajú balík, definujú a „hlavný" index (pôvodný štýl) a následne definujú index autorov. V tele dokumentu môžeme nájsť príkazy ako:
```
\index[aut]{Robin Fairbairns}
...
```

```
\index{FAQs}
```
Príkazy uložia záznam do indexu autorov, potom záznam do hlavného indexu. Na konci dokumentu máme dva príkazy:
```
\printindex
\printindex[aut]
```
Tieto vytlačia hlavný index, a potom index autorov. Za predpokladu, že sa príkazy budú nachádzať v súbore myfile.tex, spustite po dostatočnom množstve behov LaTeXu, aby boli návestia stabilné, nasledujúce príkazy (tu sú ukázané shellovské príkazy z Unixu, ale princíp je rovnaký nezávisle na použitom systéme):
```
makeindex myfile
makeindex myfile.adx -o myfile.and
```
Potom znovu spustite LaTeX. Príkazy *makeindex* spracujú myfile.idx do myfile.ind (štandardná akcia), a potom myfile.adx do myfile.and. Tie dva súbory sú potrebné ako vstup pre dva príkazy \printindex v súbore myfile.tex.

Balík *splitidx* môže pracovať rovnakým spôsobom ako ostatné: nahrajte balík s voľbou split a deklarujte každý index príkazom \newindex:
```
\newindex[<názov_indexu>]{<skratka>}
```
a *splitidx* vygeneruje súbor \jobname.<skratka> pre položky indexu generované príkazmi ako \sindex[<skratka>]{<položka>}. Ako pri ostatných balíkoch, aj tu je táto metóda limitovaná celkovým počtom výstupných súborov TeXu. *splitindex* je však dodávaný s malým spustiteľným súborom *splitindex* (dostupný pre množstvo operačných systémov). Ak tento doplnkový program použijete (a nepoužijete voľbu split), nebudete limitovaní počtom indexov. Okrem tohto triku podporuje *splitidx* rovnaké veci ako *index*. Príklad použitia nájdete v dokumentácii.

Trieda memoir má vlastné funkcie pre viacnásobné indexy (ako aj voľby rozvrhnutia indexu, ktoré ostatné balíky delegujú na indexové štýly používané programom *makeindex*).

index.sty: macros/latex/contrib/index
makeidx.sty: súčasť distribúcie LaTeXu
memoir.cls: macros/latex/contrib/memoir
multind.sty: macros/latex209/contrib/misc/multind.sty
splitidx.sty a splitindex: macros/latex/contrib/splitindex

172. Odkazovanie na veci podľa mena

LaTeXovský mechanizmus návestí je navrhnutý pre neosobný svet akademických publikácií, v ktorých všetko má číslo. Ak si máme zaznamenať *názvy* vecí, ktorým sme priradili návestie, je potrebné rozšírenie. Sú dostupné dva balíky rozširujúce príkazy LaTeXu na tvorbu sekcií, aby poskytli možnosť odkazovania názvom sekcie.

Balík *titleref* je jednoduchým rozšírením poskytujúcim príkaz \titleref. Je to samostatný balík — nepoužívajte ho v dokumente, v ktorom potrebujete aj *hyperref*.

Balík *byname*, hoci pracuje so svojím „priateľom“ *smartref*, funguje (do určitej miery) s balíkom *hyperref*, ale odkazy, ktoré definuje, nie sú hyperlinky.

Trieda memoir zahŕňa funkcionalitu *titleref*, ale zrejme nefunguje s *byname* (i keď hľadaním v comp.text.tex na groups.google.com nájdete úpravu *byname* na vyriešenie problému).

Sada *hyperref* obsahuje balík *nameref*, ktorý funguje samostatne (t.j. bez *hyperref*: v tomto móde, samozrejme, nie sú jeho odkazy hyperlinkované). Ak nahráte samotný *hyperref*, *nameref* sa nahrá automaticky. Memoir vyžaduje *memhfixc* pri behu s *hyperref*. Avšak po uvedenej postupnosti príkazov môžu byť *nameref* príkazy použité v memoir dokumente:
```
\documentclass[...]{memoir}
...
\usepackage[...]{hyperref}
\usepackage{memhfixc}
```
Všetky tri balíky pre odkazy podľa mena definujú odkazovacie príkazy rovnakého názvu ako balík: \titleref, \byname a \nameref. Balík *nameref* tiež definuje príkaz \byshortnameref, ktorý používa voliteľný „krátky“ parameter nadpisu pre príkazy kapitol a sekcií.

byname.sty: distribuovaný s macros/latex/contrib/smartref

hyperref.sty: macros/latex/contrib/hyperref
memoir.cls: macros/latex/contrib/memoir
nameref.sty: distribuovaný s macros/latex/contrib/hyperref
smartref.sty: macros/latex/contrib/smartref
titleref.sty: macros/latex/contrib/misc/titleref.sty

173. Odkazovanie na návestia v iných dokumentoch

Pri tvorbe sady vzájomne súvisiacich dokumentov sa budete často chcieť odkazovať na návestia v druhých dokumentoch sady, ale samostatný LaTeX toto nepovoľuje.

Z toho dôvodu bol napísaný balík *xr*: ak napíšete

```
\usepackage{xr}
\externaldocument{volume1}
```

do vášho súčasného dokumentu nahrajú sa všetky odkazy z volume1.

Čo však v prípade, že majú oba dokumenty sekciu s návestím „introduction" (úvod; je to celkom pravdepodobné)? Balík poskytuje prostriedky na transformáciu všetkých importovaných návestí, takže ich nemusíte meniť v žiadnom dokumente. Napríklad:

```
\usepackage{xr}
\externaldocument[V1-]{volume1}
```

nahrá odkazy zo súboru volume1, ale pred každý vloží predponu V1-. Takže na úvod v dokumente „volume 1" by ste sa odkazovali:

```
\usepackage{xr}
\externaldocument[V1-]{volume1}
...
... úvod dokumentu volume1
                     (\ref{V1-introduction})...
```

Aby prostriedky *xr* fungovali s balíkom *hyperref*, potrebujete balík *xr-hyper*. Pre jednoduché krížové hyperodkazy (t.j. na lokálny PDF súbor, ktorý ste práve skompilovali) napíšte:

```
\usepackage{xr-hyper}
\usepackage{hyperref}
\externaldocument[V1-]{volume1}
...
... the \nameref{V1-introduction})...
```

a pomenovaný odkaz sa objaví ako aktívny link na kapitolu „úvod" súboru volume1.pdf.

Pre link na PDF dokument na webe, pre ktorý máte .aux súbor, napíšte:

```
\usepackage{xr-hyper}
\usepackage{hyperref}
\externaldocument[V1-]{volume1}%
    [http://mybook.com/volume1.pdf]
...
... the \nameref{V1-introduction})...
```

xr.sty: distribuované ako súčasť macros/latex/required/tools
xr-hyper.sty: distribuované s macros/latex/contrib/hyperref

174. Automatická veľkosť *minipage*

Prostredie minipage vyžaduje, aby ste špecifikovali šírku „stránky", ktorú chcete vytvoriť. Toto je občas nepohodlné: chceli by ste (ak je to možné) zabrať menej miesta, ale minipage vytvorí box presne podľa špecifikovanej šírky.

Balík *pbox* definuje \pbox, ktorého šírka je presne šírkou najdlhšieho obsiahnutého riadku (závislé od maximálnej širky, ktorú zadáte). Takže kým \parbox{2cm}{Hello\\world!} vytvotí box šírky presne 2cm, \pbox{2cm}{Hello\\world!} vytvorí box so šírkou 1.79cm (ak používate štandardný font cmr). Balík poskytuje aj \settominwidth[min]{length}{text} (čo vyzerá takmer ako štandardný príkaz \settowidth) a \widthofpbox funkčne analogické príkazu \widthof pre použitie s balíkom *calc*.

Balík *eqparbox* myšlienku balíku *pbox* rozširuje tak, že vám dovoľuje vysádzať sériu boxov s rovnakou (minimalizovanou) šírkou. (Príkaz neprijíma parameter limitujúci maximálnu šírku.) Dokumentácia balíku ukazuje nasledujúci príklad z humorného *životopisu*:

```
\noindent%
\eqparbox{place}%
    {\textbf{Widgets, Inc.}}
        \hfill
\eqparbox{title}%
    {\textbf{Senior Widget Designer}}
        \hfill
\eqparbox{dates}{\textbf{1/95--present}}

...

\noindent%
\eqparbox{place}%
    {\textbf{Thingamabobs, Ltd.}}
        \hfill
\eqparbox{title}%
    {\textbf{Lead Engineer}}
        \hfill
\eqparbox{dates}{\textbf{9/92--12/94}}
```

Kód zabezpečí, že všetky tri položky majú rovnakú šírku, takže riadky vyzerajú pravidelne. Príkaz `\eqboxwidth` vám dovoľuje použiť zmeranú šírku skupiny: dokumentácia ukazuje, ako môže byť príkaz použitý na vytvorenie rozumne vyzerajúcich stĺpcov miešajúcich riadky `c`, `r` alebo `l`, s ekvivalentnou položkou `p{...}` tým, že z riadkov s pevnou šírkou spravíte skupinu *eqparbox* a nakoniec použijete `\parbox` so šírkou, ktorá bola pre skupinu zmeraná.

Balík *varwidth* definuje prostredie `varwidth`, ktoré sádza obsah boxu tak, aby zodpovedal „užšej prirodzenej šírke“, ak nejakú nájde. (Poskytujete rovnaké parametre ako pri `minipage`.) *Varwidth* poskytuje svoj vlastný príkaz `\narrowragged`, ktorého úlohou je zúžiť riadky a dostať viac textu do posledného riadku odstavcu (teda vytvorenie riadkov s väčšou mierou zhody dĺžky riadkov, než je to typicky pri `\raggedright`).

Dokumentácia (v samotnom súbore balíku) vypisuje rozličné reštrikcie a veci, ktoré je ešte treba spraviť, ale balík sa ukázal ako užitočný pre rozličné úlohy.

`eqparbox.sty`: `macros/latex/contrib/eqparbox`
`pbox.sty`: `macros/latex/contrib/pbox`
`varwidth.sty`: `macros/latex/contrib/misc/varwidth.sty`

175. Poznámky pod čarou v tabulkách

Standardní LaTeXový příkaz `\footnote` neumí pracovat v prostředí tabulek. Toto prostředí totiž poznámkám neumožní „uniknout“ až na konec stránky. Pokud je tabulka plovoucí, nejlepší řešení je umístit tabulku do prostředí `minipage` a umístit pak poznámky na spodní okraj stránky nebo použít balík `macros/latex/contrib/misc/threeparttable.sty` od Donalda Arsenaua.

Balík *ctable* rozšiřuje model *threeparttable* a taktéž používá myšlenky balíku *booktabs*. Příkaz `\ctable` dělá všechnu práci: sázení tabulky, umístnění popisků a definování poznámek. „Tabulka“ může obsahovat diagramy a volitelný parametr `\ctable` udělá vytvořený float obrázkem místo tabulky.

Pokud není tabulka plovoucí, pak můžete použít jednu z možností:

1. Pomocí `\footnotemark` můžete vhodně umístit do textu značku poznámky před ukončením prostředí `tabular` vložte příkaz `\footnotetext`. Tento způsob je popsán v Lamportově knize, můžete však použít pouze jednu poznámku.
2. Vložte tabulku do prostředí `minipage`. To vám umožňuje použít všechny vlastnosti poznámek bez většího úsilí.

3. Použijte `macros/latex/contrib/misc/threeparttable.sty`. Balík je sice vytvořen pro plovoucí tabulky, ale můžete ho použít i v tomto případě.
4. Použijte `tabularx` nebo `longtable` z LaTeXové distribuce (`macros/latex/required/tools`). Klasické poznámky sice zvládá o poznání hůře než prostředí `tabular`, avšak povoluje poznámky pod čarou.
5. Stáhněte balík *footnote* (skrýtá se v archivu na `macros/latex/contrib/mdwtools`. Pak vložte vaši tabulku do prostředí `savenotes`. Alternativně můžete do preambule dokumentu napsat `\makesavenoteenv{tabular}` a tabulka zvládne poznámky pod čarou korektně.
6. Použijte *mdwtab* ze stejného adresáře. Ten umí nejen správně zvládat poznámky pod čarou, ale navíc ještě umí vaši tabulku zkrášlit. Ten však může být nekompatibilní s jinými balíky.

Dokumentace *threeparttable* je obsažena v souboru balíku samém, dokumentace *ctable* je distribuována jako PDF soubor.

176. Štýly popiskov

Zmeny štýlov popiskov môžete spraviť opätovnou definíciou príkazov, ktoré popisky vytvárajú. Takže napríklad `\fnum@figure` môže byť predefinovaný vo vašom vlastnom balíku alebo medzi `\makeatletter`–`\makeatother` (viď Otázku 243):

```
\renewcommand{\fnum@figure}%
  {\textbf{Fig.~\thefigure}}
```

čo spôsobí, že číslo bude vysádzané tučné. (Všimnite si, že v pôvodnej definícii bolo použité `\figurename` – viď Otázku 217.) Prepracovanejších zmien môžeme dosiahnuť úpravou príkazu `\caption`, kedže však na túto prácu neexistujú balíky, nie sú takéto zmeny (ktoré môžu byť dosť náročné) odporúčané bežných používateľom.

Balík *float* poskytuje určitú konkrolu na vzhľadom popiskov, aj keď je určený hlavne na vytváranie neštandardných floatov (plávajúcich tabuliek a obrázkov). Balíky *caption* a *ccaption* (všimnite si násobné '*c*') poskytujú rôzne formátovacie možnosti.

ccaption poskytuje „predĺžené" popisky a popisky, ktoré môžete umiestniť mimo prostredí float. Balík *capt-of* povoľuje taktiež popisky mimo prostredia float. Pri zaobchádzaní s vecami predpokladajúcimi postupnosť floatov (ako v predĺžených popiskoch), alebo prípadným miešaním neplávajúcich popiskov s plávajúcimi, je potrebné dbať na opatrnosť.

Trieda `memoir` zahŕňa možnosti balíku *ccaption*. Triedy `KOMA-script` tiež poskytujú široký rozsah príkazov na formátovanie popiskov.

Dokumentácia *caption* je dostupná po spracovaní súboru `manual.tex` vytvoreného pri rozbalovaní `caption.dtx`.

V minulosti odporúčaný balík *caption2* bol znovu prevzatý balíkom *caption*. *caption2* však zostáva dostupný pre staršie dokumenty.

`caption.sty`: `macros/latex/contrib/caption`
`capt-of.sty`: `macros/latex/contrib/misc/capt-of.sty`
`ccaption.sty`: `macros/latex/contrib/ccaption`
`float.sty`: `macros/latex/contrib/float`
balík `KOMA script`: `macros/latex/contrib/koma-script`
`memoir.cls`: `macros/latex/contrib/memoir`

177. Štýly nadpisov dokumentov

Balík *titling* poskutuje mnoho prostriedkov umožňujúcich manipuláciu so vzhľadom príkazu `\maketitle` (vrátane `\thanks` atď.). Balík definuje prostredie `titlingpage`, ktoré je konfigurovateľné. Poskytuje niečo medzi voľbou `titlepage` štandardných tried a prostredím `titlepage`.
`titling.sty`: `macros/latex/contrib/titling`

178. Vzhled nadpisů

Předpokládejme, že editor našeho oblíbeného časopisu vyžaduje, aby nadpisy oddílů byly centrované a vysázené kapitálkami a pododdíly byly zarovnány vpravo a vysázeny kurzívou. Nechce se nám však zaplétat do takového druhu programování jaký je popsán v *The LaTeX Companion* (viz Otázku 23; programování samotné je diskutováno v Otázce 243). Následující trik pravděpodobně editory uspokojí. Nově definované příkazy `\ssection` a `\ssubsection`:

```
\newcommand{\ssection}[1]{%
   \section[#1]{\centering\textsc{#1}}}
\newcommand{\ssubsection}[1]{%
   \subsection[#1]{\raggedright\textit{#1}}}
```

se pak použijí namísto obvyklých `\section` a `\subsection`. Není to však nejhezčí řešení. Čísla v nadpisech oddílů zůstávají vysázena tučně a hvězdičkovaná forma příkazů musí být předefinována zvlášť. Uvedené definice nebudou správně fungovat v případě, že se použije NFSS (viz Otázku 293) mimo LaTeX 2_ε (viz Otázku 294), jelikož v takovém případě se příkazy měnící typ písma chovají odlišně.

Balík `macros/latex/contrib/sectsty` nabízí jednoduše použitelnou sadu příkazů pro tuto práci, zatímco balík `macros/latex/contrib/titlesec` umožňuje pokročilejší řešení této problematiky (*titlesec* přichází s druhým balíkem – *titletoc* – který je používán pro úpravu formátu celého obsahu.

Záhlaví příkazů `\chapter` a `\part` jsou vytvářena jiným mechanismem, balík *sectsty* umí zacházet s oběma, zatímco *titlesec* umí upravit pouze `\chapter`. Balík `macros/latex/contrib/fncychap` nabízí širokou škálu předem definovaných designů nadpisů kapitol.

Balík *fncychap* poskytuje pěknou kolekci upravených nadpisů kapitol. Balík *anonchap* poskytuje jednoduché nástroje pro sazbu nadpisů kapitol „jako nadpisů kapitol" (tedy bez části „Kapitola"). Balík *tocbibind* poskytuje stejné příkazy, avšak za účelem dosažení jiného cíle. *fncychap* bohužel není sladěný s přítomností front a backmatter v dokumentech třídy `book`.

179. Prílohy

LaTeX poskytuje výnimočne jednoduchý mechanizmus príloh: príkaz `\appendix` prepína dokument zo stavu generovania sekcií (v triede `article`) alebo kapitol (v triedach `report` a `book`) do stavu produkovania príloh. Číslovanie sekcie alebo kapitoly je reštartované a reprezentácia počítadla sa prepne do abecedného módu. Takže:

```
\section{Moja inšpirácia}
...

\section{Rozvíjanie inšpirácie}
...

\appendix
\section{Ako ma kopla múza}
...
```

sa vysádza (v dokumente triedy `article`) nasledovne:

1 Moja inšpirácia

...

2 Rozvíjanie inšpirácie

...

A Ako ma kopla múza

...

čo je väčšinou dostačujúce. Všimnite si, že po tom, čo ste zapli sádzanie príloh, LaTeX vám neumožňuje cestu späť — po prílohe nemôže nasledovať normálna sekcia (`\section`) alebo kapitola (`\chapter`).

Balík *appendix* poskytuje viacero spôsobov ako túto jednoduchú situáciu spracovať. Priame použitie balíku vám dovoľuje mať oddelené nadpisy v tele dokumentu a obsahu. Dosiahnete to pomocou

```
\usepackage{appendix}
...
\appendix
```

```
\appendixpage
\addappheadtotoc
```
Príkaz `\appendixpage` pridáva nad prvú prílohu oddelený nadpis „Appendices" (prílohy) a príkaz `\addappheadtotoc` pridáva podobný nadpis do obsahu. Tieto jednoduché modifikácie pokrývajú potreby mnohých ľudí týkajúce sa príloh.

Balík tiež poskytuje prostredie `appendices`, ktoré poskytuje pestrejšie využitie Prostredie sa najlepšie ovláda voľbami balíku. Hore vedený príklad by sme dosiahli takto:
```
\usepackage[toc,page]{appendix}
...
\begin{appendices}
...
\end{appendices}
```
Skvelé je, že keď prostredie `appendices` skončí, môžete v sekciách a kapitolách pokračovať ako predtým — číslovanie nie je ovplyvnené.

Balík poskytuje aj ďalšiu možnosť sádzania príloh — vo forme podriadených úsekov v dokumente. Prostredie `subappendices` dovoľuje vkladať prílohy pre konkrétnu sekciu oddelene (použíte `\subsection` pre sekcie, alebo `\section` pre kapitoly). Môžete teda napísať:
```
\usepackage{appendix}
...
\section{Moja inšpirácia}
...
\begin{subappendices}
\subsection{Ako ma kopla múza}
...
\end{subappendices}

\section{Rozvíjanie inšpirácie}
...
```
Výsledkom bude niečo podobné:

1 Moja inšpirácia

...

1.A Ako ma kopla múza

...

2 Rozvíjanie inšpirácie

...

S balíkom sa dá robiť ešte veľa zaujímavých vecí, pre detaili sú užívatelia odkazovaný na dokumentáciu.

Trieda `memoir` obsahuje možnosti balíku *appendix*. Triedy `KOMA-script` poskytujú príkaz `\appendixprefix` na manipuláciu so vzhľadom príloh.

`appendix.sty`: `macros/latex/contrib/appendix`
balík `KOMA script`: `macros/latex/contrib/koma-script`
`memoir.cls`: `macros/latex/contrib/memoir`

180. Široké obrázky v dvojstĺpcových dokumentoch

Plávajúce obrázky a tabuľky obyčajne vychádzajú v rovnakej šírke ako stránka, ale v dvojstĺpcových dokumentoch sú obmedzené šírkou stĺpca. Toto ale občas nie je dostačujúce, takže existujú alternatívne verzie prostredí float — v dvojstĺpcových dokumentoch poskytuje `figure*` plávajúci obrázok šírky stránky (a `table*` tabuľku šírky stránky), ktorý sa postará o všetko potrebné.

Hviezdičkové plávajúce (float) prostredia sa môžu objaviť iba na vrchu stránky alebo na celej stránke — direktívy float rozloženia `h` alebo `b` sú jednoducho ignorované.

Rovnice šírky celej strany môžu byť bohužiaľ prispôsobené iba vnútru float prostredí. Mali by ste ich zahŕňať do prostredí `figure`, alebo používať balík *float*, prípadne *ccaption* na definíciu nového typu floatu.

`ccaption.sty`: `macros/latex/contrib/ccaption` float.sty: macros/latex/contrib/float

181. Odsazení odstavce po nadpisu

LaTeX používá stylů, jež neodsazují první odstavec za nadpisem. Je mnoho důvodů proč tomu tak je, ale ne každému to vyhovuje (např. to neodpovídá tradicím české typografie). Užijeme-li styl indentfirst.sty (macros/latex/required/tools), pak se tento mechanismus potlačí, takže první odstavec bude odsazen.

182. Poznámky v LaTeXových nadpisech

Umístění příkazu \footnote v argumentu příkazu \section není dostačující. Jelikož jsou argumenty příkazu \section používány v obsahu a potenciálně i v záhlaví stránky, není použití \protect\footnote také dobrý nápad.

Bohužel neexistuje rozumný způsob, jak zamezit zobrazování těchto poznámek v záhlaví při povolení zobrazování v obsahu. K jejich potlačení musíte udělat toto:

- Využijte toho, že povinné argumentu nejsou volány, pokud se zde vyskytují volitelné: \section[title]{title\footnote{title ftnt}}
- Použijte balík macros/latex/contrib/footmisc s volbou balíku stable – ten upraví poznámky pod čarou tak, že lehce a tiše zmizí, pokud jsou použité jako nepovinné argumenty.

183. Poznámky v popiskoch

Poznámky v popiskoch sú obzvlášť komplikované: predstavujú samostatný problém (navyše k problémom poznámok v nadpisoch a tabuľkách spomínaných v predchádzajúcich otázkach).

Takže okrem používania voliteľného parametru príkazu \caption (alebo podobného), aby sme sa vyhli migrácii poznámky do Zoznamu ... (List of ...), a vloženia objektu, ktorého popisky obsahujú poznámku, do minipage, musíme *naviac* čeliť tendencii príkazu \caption vytvárať text poznámky dvakrát. Pre tento posledný problém neexistuje čisté riešenie, ktorého by si bol autor vedomý. Ak týmto problémom trpíte, dobre vytvorený príkaz \caption v prostredí minipage vo vnútri floatu ako:

```
\begin{figure}
  \begin{minipage}{\textwidth}
    ...
    \caption[Popisky pre LOF]%
    {Skutočné popisky\footnote{blah}}
  \end{minipage}
\end{figure}
```

môže vytvoriť *dve* kópie tela poznámky „blah“. (Efekt sa objavuje pri popiskoch, ktoré sú dosť dlhé, aby na sadzbu vyžadovali dva riadky, a teda pri takom krátkom popisku by sa neobjavil.) Dokumentácia balíku *ccaption* popisuje skutočne strašný spôsob, ako problém obísť.
ccaption.sty: macros/latex/contrib/ccaption

184. Poznámky s rovnakým textom

Ak sa *rovnaká* poznámka objaví na rozličných miestach v dokumente, je často nevhodné opakovať celú poznámku znovu. Opakovaniu sa môžeme vyhnúť poloautomatickou cestou alebo jednoduchým označovaním poznámok, o ktorých vieme, že ich budeme opakovať, a následným odkazovaním na výsledok. Plne autmatické riešenie (také, čo detekuje a potláča) neexistuje.

Ak viete, že budete mať iba jednu opakovanú poznámku, riešenie je jednoduché: iba použite voliteľný parameter \footnotemark na zdôraznenie opakovania:

```
...\footnote{Opakujúca sa poznámka}
...
...\footnotemark[1]
```

... čo je veľmi jednoduché, keďže vieme, že bude existovať iba poznámka čislo 1. Podobná technika môže byť použitá, keď už sú poznámky stabilné, použijete číslo, ktoré LaTeX alokoval. Toto však môže byť únavné, keďže ľubovolná zmena sadzby by mohla zmeniť vzťahy medzi poznámkou a opakovaním: značkovanie je nevyhnuteľne lepšie.

Môžeme použiť jednoduché ručné značkovanie poznámok pomocou dedikovaného počítadla:

```
\newcounter{fnnumber}
...
...\footnote{Opakovaný text}%
\setcounter{fnnumber}{\thefootnote}%
...
...\footnotemark[\thefnnumber]
```

Je to ale trochu zdĺhavé. Na pomoc si môžeme privolať značkovacie mechanizmy LaTeXu, ale pred vyhodnotením `\ref` počas druhého priebehu LaTeXom sa objavujú nepekné chybové hlásenia:

```
...\footnote{Opakovaný text\label{fn:repeat}}
...
...\footnotemark[\ref{fn:repeat}]
```

Alternatívne môžete použiť príkaz `\footref`, ktorého výhodou je schopnosť fungovať, dokonca aj keď nie je značka poznámky vyjadrená ako číslo. Príkaz je definovaný v balíku *footmisc* a v triede `memoir` (prinajmenšom). `\footref` zredukuje predchádzajúci príklad na:

```
...\footnote{Opakovaný text\label{fn:repeat}}
...
...\footref{fn:repeat}
```

Toto je najčistejšie riešenie. Všimnite si, že príkaz `\label` *musí* byť vo vnútri parametru `\footnote`.

Balík *fixfoot* nám to trochu uľahčuje: pomocou príkazu `\DeclareFixedFoot` deklarujete poznámky (typicky v prambule dokumentu), ktoré budete opakovať, a potom použijete deklarovaný príkaz v tele dokumentu:

```
\DeclareFixedFootnote{\rep}{Opakovaný text}
...
...\rep{}
...\rep{}
```

Balík zaručuje, že opakovaný text sa objavuje na strane najviac raz: obvykle zrejme bude potrebný viac než jeden beh LaTeXu, aby ste sa opakovaní zbavili.

`fixfoot.sty`: `macros/latex/contrib/fixfoot`
`footmisc.sty`: `macros/latex/contrib/footmisc`
`memoir.cls`: `macros/latex/contrib/memoir`

185. Číslovanie poznámok „po stránkach"

Očividným riešením je resetovanie počítadla vždy, keď zmeníme stranu, pomocou interných mechanizmov LaTeXu (viď Otázku 276). Miesto zmeny stránky bohužial nie je predvídateľné (nie na konci tlačenej stránky, takže takéto spojenie funguje iba so šťastím).

Výsledkom je, že resetovanie poznámok je nevyhnuteľne dvojprechodový proces používajúci určitý druh návestí. Návestia sú tak či tak dôležité, kedže sa bežne vyskutuje požiadavka na značkovanie poznámok symbolmi (s veľmi malými množinami symbolov). Existujú tri balíky, ktoré toto dokážu.

Balík *footnpag* robí iba číslovanie poznámok po stránkach a nič iné.

Balík *perpage* poskytuje obecný mechanizmus resetovania počítadiel po jednotlivých stránkach, takže zjavne môže byť na túto úlohu použitý. Rozhranie je jednoduché: `\MakePerPage{footnote}` spraví, čo potrebujeme. Ak chcete nastaviť počítadlo na inú hodnotu než 1 (napríklad aby ste sa vyhli niečomu v LaTeXovskom zozname symbolov poznámok), použite: `\MakePerPage[2]{footnote}`.

Balík *footmisc* poskytuje rozličné spôsoby kontroly vzhľadu poznámok, medzi ktorými je aj voľba balíku `perpage`, ktorá upravuje číslovanie po stránkach.

Dokumentácia *footnpag* je dodávaná ako `.dvi` súbor `footnpag-user` distribúcie. Dokumentácia *perpage* je iba v súbore balíku (neobsahuje ale viac, než bolo uvedené hore. . .).

`footmisc.sty`: `macros/latex/contrib/footmisc`

footnpag.sty: macros/latex/contrib/footnpag
perpage.sty: macros/latex/contrib/misc/perpage.sty

186. Zmena okrajov v LaTeXu

Zmena veľkosti tela textu LaTeXovského je prekvapujúco ťažká úloha, najlepšia rada pre začiatočníkov je, aby to neskúšali. Medzi fundamentálnymi obmedzeniami TeXu, obmedzeniami spojenými s návrhom LaTeXu a dobrými dizajnovými a sádzačskými zvykmi existuje určitá spojitosť. Každá zmena musí byť teda dobre premyslená a musí byť zaistené, že funguje a zároveň je oku príjemná.

Lamportove varovanie v jeho sekcii „Customizing the Style" nesmie byť brané na ľahkú váhu. Jednopalcové okraje papieru formátu A4 sú prijateľné pre písacie stroje, ale nie pre sadzbu desaťbodovým (prípadne väčším) písmom, pretože sa takéto široké, husté riadky ťažko čítajú. Ideálne by na riadok nemalo byť viac ako 75 znakov (pre dvojstĺpcový text sa obmedzenie mení).

Ovládacie prvky LaTeXu dovoľujú zmeniť vzdialenosť od okajov stránky k ľavému a hornému okraju vášho textu, ako aj výšku a šírku textu. Zmena výšky a šírky vyžaduje viac zručnosti, než by ste čakali. Výška by si mala zachovať určitý vzťah k vzdialenostiam po sebe nasledujúcich účiarí (`\baselineskip`), šírka by mala byť obmedzená podľa už spomenutého pravidla.

Ovládacie prvky sú vyjadrené množinou parametrov stránky. V istom zmysle sú komplexné a pri predefinovávaní rozvrhnutia stránky si môžeme ľahko popliesť ich vzájomné vzťahy. Balík *layout* definuje príkaz `\layout` kresliaci diagram aktuálneho rozvrhnutia stránky s rozmermi (avšak bez ich závislostí). Toto FAQ odporúča používať balík na vytvorenie konzistentného nastavenia parametrov, o závislosti sa postarajú balíky.

Definitívnym nástrojom pre zmenu rozmerov a pozície tlačeného materiálu na stránke je balík *geometry*. Relatívne priamočiaro môže byť naprogramovaných množstvo nastavení rozvrhnutia. Dokumentácia v `.dtx` súbore (viď Otázku 44) je kvalitná a obsiahla.

Použítie balíku *vmargin* je trochu jednoduchšie. Balík obsahuje množinu rozmerov papierov (nadmnožina toho, čo poskytuje LaTeX 2_ε), opatrenia pre vlastné rozmery, nastavenia okrajov a opatrenia pre dvojstrannú tlač.

Ak to stále chcete spraviť sami, začnite sa zoznamovať s parametrami LaTeXu pre rozloženie stránky. Pozrite si napríklad sekciu C.5.3 LaTeX manuálu (str. 181–182) alebo príslušné sekcie v iných manuáloch LaTeXu (viď Otázku 23). Parametre `\oddsidemargin` a `\evensidemargin` vďačia za svoje názvy skutočnosti, že obvyle sú nepárne stránky na pravej strane dvojstranného rozloženia („recto") a párne na ľavej strane („verso"). Oba parametre odkazujú na *ľavý* okraj. Pravý okraj je potom vydedukovaný z veľkosti `\textwidth`. Začiatok je v DVI koordinátoch jeden palec od horného okraja a jeden palec od ľavého okraja. Kladné horizontálne hodnoty postupujú smerom doprava, kladné vertikálne smerom nadol. Takže pre okraje bližšie než jeden palec k ľavému a hornému kraju stránky môžete príslušné parametre (`\evensidemargin`, `\oddsidemargin`, `\topmargin`) nastaviť na záporné hodnoty.

Ďalším prekvapením je, že jednoduchou modifikáciou parametrov veľkosti textu nemôžete zmeniť šírku a výšku textu v dokumente. Parametre by mali byť menené iba v preambule dokumentu, teda pred výrazom `\begin{document}`. Pre zmenu v dokumente definujeme prostredie:

```
\newenvironment{changemargin}[2]{%
 \begin{list}{}{%
  \setlength{\topsep}{0pt}%
  \setlength{\leftmargin}{#1}%
  \setlength{\rightmargin}{#2}%
  \setlength{\listparindent}{\parindent}%
  \setlength{\itemindent}{\parindent}%
  \setlength{\parsep}{\parskip}%
 }%
\item[]}{\end{list}}
```

Toto prostredie akceptuje dva parametre a odsadí ľavý a pravý okraj podľa hodnôt príslušných parametrov. Záporné hodnoty spôsobia zúženie okrajov, takže `\begin{changemargin}{-1cm}{-1cm}` zúži ľavý a pravý okraj o 1cm.

Balík *chngpage* poskytuje na vytvorenie predchádzajúceho stavu hotové príkazy. Obsahuje opatrenia na zmenu posunov aplikovaných na váš text podľa toho, či ste na nepárnej alebo párnej stránke dvojstránkového dokumentu. Dokumentácia balíku (v súbore samotnom) navrhuje stratégiu zmeny rozmerov textu medzi stranami — ako bolo spomenuté vyššie, zmena rozmerov textu v tele stránky môže viesť k nepredvídateľným výsledkom.

`chngpage.sty`: `macros/latex/contrib/misc/chngpage.sty`

`geometry.sty`: `macros/latex/contrib/geometry`

`layout.sty`: distribuované ako súčasť `macros/latex/required/tools`

`vmargin.sty`: `macros/latex/contrib/vmargin`

187. Ako sa zbaviť čísel strán

Balík *nopageno* potlačí čísla strán v celom dokumente.

Pre potlačenie iba na jednej strane použite `\thispagestyle{empty}` niekde v texte strany. `\maketitle` a `\chapter` používajú `\thispagestyle` interne, takže to musíte zavolať po tom, čo ste zavolali `\maketitle` alebo `\chapter`.

Pre potlačenie číslovania postupnosti strán môžete použiť `\pagestyle{empty}` na začiatku postupnosti strán a obnovte pôvodný štýl stránky na konci (postupnosti). Bohužiaľ, stále musíte použiť `\thispagestyle` po každom príkaze `\maketitle` alebo `\chapter`.

Problémové príkazy triedy `memoir` (`\maketitle`, `\chapter` atď) vyvolajú vlastný štýl stránky (`title`, `chapter` atď.), ktorý môžete predefinovať použitím techník triede vlastných, aby boli ekvivaletné prostrediu „`empty`“. Triedy `KOMA-script` majú príkazy štýl stránky, ktorý má byť použitý, takže môžete napísať:

```
\renewcommand*{\titlepagestyle}{empty}
```

Alternatívou (vo všetkých triedach) je použitie celkom príjemného `\pagenumbering{gobble}`. Má to za následok, že každý pokus o tlač čísla strany nič nevyprodukuje, takže nie je problém zabrániť ľubovolnej časti LaTeXu v tlači čísla strany. Príkaz `\pagenumbering` však má vedľajší efekt — nastavuje (resetuje) číslo stránky na 1, čo nemusí byť príjemné.

Balík *scrpage2* oddeľuje reprezentáciu od resetovania, takže môžete napísať:

```
\renewcommand*{\pagemark}{}
```

aby ste dosiahli to isté ako pri triku s `gobble`, bez resetovania čísla stránky.

nopageno: `macros/latex/contrib/carlisle/nopageno.sty`

balík `KOMA script`: `macros/latex/contrib/koma-script`

`memoir.cls`: `macros/latex/contrib/memoir`

`scrpage2.sty`: distribuované ako súčasť `macros/latex/contrib/koma-script`

188. Zjištění šířky písmene, slova nebo sousloví

Slovo se vloží do boxu a poté se zjistí šířka tohoto boxu. Například:

```
\newdimen\stringwidth
\setbox0=\hbox{hi}
\stringwidth=\wd0
```

Je třeba však mít na paměti, že je-li v `\hbox`u sousloví, hodnota, kterou určíme, bude pouze přibližnou šířkou, kterou bude vysázené zaujímat ve skutečném textu, neboť mezislovní mezery se při sazbě odstavce mohou změnit.

V LaTeXu lze problém řešit takto:

```
\newlength{\gnat}
\settowidth{\gnat}{\textbf{small}}
```

Hodnota `\gnat` se v tomto případě nastaví na hodnotu šířky „small“ vysázeného tučně.

189. Aký je názov tohto súboru

Toto zistenie môžete potrebovať pre automatické generovanie záhlavia alebo päty stránky zaznamenávajúceho práve spracúvaný súbor. Nie je to jednoduché...

TeX si pamätá iba to, čo považuje za názov *práce* (*job*), v primitíve `\jobname`. Predstavuje názov súboru, ktorý bol prvý predaný TeXu, orezaný o meno adresáru a ľubovolnú príponu (ako napríklad `.tex`). Ak nebol predaný žiadny súbor (t.j. používate TEX interaktívne), `\jobname` má hodnotu `texput` (v tomto prípade názov daný `.log` súboru).

Toto je v prípade malých dokumentov udržiavaných v jednom súbore. Väčšina významných dokumentov bude však udržiavaná vo viacerých súboroch a TEX sa nesnaží držať si prehľad vstupných súborov práce. Prehľad si teda musí udržiavať užívateľ sám — jediným spôsobom je úprava vstupných príkazov, aby si pamätali detaily názvov súborov. Toto je práve v prípade `Plain` TEXu zložité, kedže syntax príkazu `\input` je „svojrázna“.

V prípade LATEXu majú vstupné príkazy pekne pravidelnú syntax. Stačí na ne teda použiť najjednoduchšie upravovacie techniky:

```
\def\ThisFile{\jobname}
\newcounter{FileStack}
\let\OrigInput\input
\renewcommand{\input}[1]{%
  \stepcounter{FileStack}
  \expandafter\let
    \csname NameStack\theFileStack\endcsname
    \ThisFile
  \def\ThisFile{#1}%
  \OrigInput{#1}%
  \expandafter\let\expandafter
    \ThisFile
    \csname NameStack\theFileStack\endcsname
  \addtocounter{FileStack}{-1}%
}
```

(Podobne aj pre `\include`.) Kód predpokladá, že budete vždy pre `\input` používať syntax LATEXu, t.j. vždy použijete parameter uzavretý do zložených zátvoriek.

Balík *FiNK* („File Name Keeper“) poskytuje v makre `\finkfile` bežný prostriedok na udržanie si prehľadu o názve aktuálneho súboru (spolu s jeho príponou). Sada zahŕňa súbor `fink.el` poskytujúci podporu pre *emacs* s AUC-TEXom.

`fink.sty`: `macros/latex/contrib/fink`

190. Všetky súbory použité v tomto dokumente

Keď zdielate dokument s niekým iným (možno v rámci spoločného vývoja), je vhodné, aby obaja korešpondenti mali rovnaký dokument, ako aj rovnakú sadu pomocných súborov. Váš korešpondent zjavne potrebuje rovnakú sadu súborov (ak napríklad používate balík *url*, druhá strana tiež musí mať *url*). Predpokladajme, že vy máte bezchybnú verziu balíku *skvelybalik*, ale na druhej strane majú stále nestabilnú verziu. Kým obe strany zistia, čo sa deje, môže byť takáto situácia veľmi mätúca.

Najjednoduchším riešením je LATEXovský príkaz `\listfiles`. Ten do súboru logu umiestňuje zoznam použitých súborov a ich verzií. Ak tento zoznam vypreparujete a pošlete ho so svojím súborom, môže byť použitý ako kontrolný zoznam v prípade problémov.

Majte na pamäti, že `\listfiles` zaregistruje iba veci vložené „štandardnými“ LATEXovskými mechanizmami (`\documentclass`, `\usepackage`, `\input`, `\include`, `\includegraphics` atď.). Ak však použijete primitívnu syntax TEXu:

```
\input mymacros
```

`mymacros.tex` *nebude* príkazom `\listfiles` zahrnutý do zoznamu, kedže ste obišli mechanizmus zaznamenávajúci jeho použitie.

Balík *snapshot* pomáha vlastníkovi LATEXovského dokumentu získať zoznam externých závislostí dokumentu (vo forme, ktorá môže byť začlenená na začiatok dokumentu). Balík je určený na tvorbu archívnych kópií dokumentov, ale má nasadenie aj pri situáciach výmeny dokumentov.

Systém *bundledoc* používa `\listfiles` na tvorbu archívu (napríklad `.tar.gz` alebo `.zip`) súborov vyžadovaných vašim dokumentom. Je dodávaný s konfiguračnými súbormi pre*teTeX* a *mikTeX*. Je užitočný, hlavne keď posielate prvú kópiu dokumentu.

`bundledoc`: `support/bundledoc`

snapshot.sty: macros/latex/contrib/snapshot

191. Označovanie zmien v dokumente

Často potrebujeme zreteľne vyznačiť, ako náš dokument zmenil. Najčastejší postup, „pruhy zmien" (change bars; tiež známe ako „pruhy revízií" — revision bars), však vyžaduje od programátora prekvapivo veľkú lstivosť (problém je, že TEX poriadne neposkytuje programátorovi žiadne informácie o aktuálnej pozícii, z ktorej by mohol byť vypočítaný domnelý začiatok alebo koniec pruhu; PDFTEX informácie *poskytuje*, ale nie sme si zatiaľ vedomí o žiadnom programátorovi, ktorý by to využil na napísanie changebar balíku založeného na PDFTEXu).

Najjednoduchší balík poskytujúci pruhy zmien je *backgrnd.tex* od Petra Schmitta. Bol napísaný ako aplikácia `Plain` TEX upravujúca výstupnú rutinu, ale zrejme funguje aspoň na jednoduchých LATEXovských dokumentoch. Múdri používatelia LATEXu budú po informácii o úprave ich výstupnej rutiny pomocou *backgrnd* ostražití a veľmi pozorne budú sledovať jeho chovanie (úprava LATEXovskej rutiny nie je niečo, na čo sa dá ľahko podujať. . .).

Najstarším riešením je balík *changebar*, ktorý používa príkazy `\special` poskytované ovládačom, ktorý používate. Z tohto dôvodu potrebujete balíku povedať, pre ktorý ovládač má generovať príkazy `\special` (rovnakým spôsobom ako pri balíku *graphics*); zoznam dostupných ovládačov je dosť obmedzený, zahŕňa však *dvips*. Balík je dodávaný so shellovským skriptom *chbar.sh* (pre použitie na Unixe), ktorý porovnáva dva dokumenty a generuje tretí, označený makrami *changebar* pre zvýraznenie zmien. Skvelý shareware editor *WinEDT* má makro generujúce makrá *changebar* (alebo iné) na zviditeľnenie zmien voči predchádzajúcej verzii vašeho súboru uloženého v *RCS* repozitári — viď `http://www.winedt.org/Macros/LaTeX/RCSdiff.php`.

Balík *vertbars* používa techniky balíku *lineno* (musí byť prítomný) a je teda najmenší z balíkov na značenie pruhov zmien, kedže všetko zložitejšie necháva inému balíku. Balík *framed* je ďalším z balíkov poskytujúcich pruhy, tentoraz ako vedľajší efekt poskytovanie iných funkcií: jeho prostredie `leftbar` je iba „chudobnejší" frame (pamätajte si však, že prostredie vytvára pre svoj obsah vlastný odstavec, takže je najlepšie používať ho pri konvencii označovania *celého* zmeneného odstavcu).

Trieda `memoir` dovoľuje okrajové redakčné poznámky, ktoré môžete použiť na ohraničenie oblastí zmeny textu.

backgrnd.tex: macros/generic/backgrnd.tex
changebar.sty: macros/latex/contrib/changebar
framed.sty: macros/latex/contrib/misc/framed.sty
lineno.sty: macros/latex/contrib/lineno
memoir.cls: macros/latex/contrib/memoir
vertbars.sty: macros/latex/contrib/misc/vertbars.sty

192. Podmienená kompilácia a komentáre

Aj keď LATEX (alebo ľubovolný iný balík odvodený z TEXu) nie je v skutočnosti kompilátor, ľudia často chcú, aby robil veci pre kompilátor typické. Častou požiadavkou je podmienená „kompilácia" a blokové komentáre. Na tento účel je dostupných viacero LATEXu špecifických prostriedkov.

Jednoduché príkazy `\newcommand{\csx {gobble}}[1]{}` a `\iffalse ... \fi` nie sú pre komentáre (ako obecné riešenie) dostačujúce, kedže vynechané veci sú TEXom aj tak preskúmané. Skúmanie zavádza obmedzenia na veci, ktoré môžete vynechať. Toto nemusí byť problémom *dnes*, ale v budúcnosti sa to môže vypomstiť. `\gobble` je naviac pre netriviálne parametre dosť neefektívny, kedže všetko, čo sa má vynechať, je pred vynechaním skopírované do zásobníku parametrov.

Ak sa vaša požiadavka týka dokumentu, z ktorého chýbajú celé kapitoly, zvážte LATEXovský systém `\include`/`\includeonly`. Ak vložíte vaše súbory pomocou `\include` (namiesto `\input` — viď Otázku 253), LATEX zapisuje informácie o tom, čo sa deje na konci každej kapitoly do `.aux` súboru. Pomocou `\includeonly` môžete LATEXu dodať vyčerpavajúci zoznam potrebných súborov. Súbory, ktoré nie sú vložené pomocou `\include`, sú celé preskočené, ale spracovanie dokumentu pokračuje, ako keby tam *boli*. Strany, poznámky a iné čísla nie sú ovplyvnené. Pamatajte si, že sekcie, ktoré chcete vložiť, si môžete vybrať interaktívne pomocou balíku *askinclude*.

Pomocou balíku *excludeonly* môžete dosiahnuť presný opak: dovoľuje vám z dokumentu vynechať zoznam súborov vložených pomocou `\include` použitím príkazu `\excludeonly`.

Ak si chcete vybrať konkrétne stránky dokumentu, použite balík *pagesel* Heika Oberdieka alebo balík *selectp*. Niečo podobné môžete spraviť s existujúcim PDF dokumentom (ktorý ste mohli skompilovať pomocou *pdflatex*u) použitím balíku *pdfpages*. Spravíte to s podobne vyzerajúcim dokumentom:

```
\documentclass{article}
\usepackage[final]{pdfpages}
\begin{document}
\includepdf[pages=30-40]{yoursource.pdf}
\end{document}
```

(Na vloženie celého dokumentu môžete napísať

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

vynechajúc začiatočnú a koncovú stranu dokumentu vo voliteľnom parametri.)

Ak chcete byť pri vkladaní alebo vynechávaní malých častí súboru, zvážte balíky *comment*, *version* alebo *optional*.

comment dovoľuje deklarovať oblasti dokumentu, ktoré majú byť vložené alebo vynechané; tieto deklarácie robíte v preambule súboru. Príkaz `\includeversion{version-name}` deklaruje prostredie `version-name`, ktorého obsah bude zahrnutý vo vašom dokumente, zatiaľ čo príkaz `\excludeversion{version-name}` definuje prostredie, ktorého obsah bude vynechaný. Balík používa celkom robustnú metódu vynechávania a dokáže si poradiť aj so zle formovaným textom (napríklad s nespárovanými zloženými zátvorkami alebo príkazmi `\if`).

version poskytuje podobné prostriedky ako `comment.sty` (t.j., príkazy `\includeversion` a `\excludeversion`); je oveľa jednoduchší, ale aj menej robustný (a hlavne si nevie poradiť s veľmi rozsiahlymi oblasťami vypúšťaného/vkladaného textu).

Dôležitým rozvojom *version*, je balík zavádzajúco pomenovaný *versions* (t.j. iba plurál pôvodneho názvu). *Versions* pridáva príkaz `\markversion{version-name}`, ktorý definuje prostredie pre tlač vloženého textu so zreteľnou značkou okolo neho.

Balík *optional* definuje príkaz `\opt`; jeho prvý parameter je „príznak vloženia“, druhým je vkladaný alebo vynechaný text. Vkladaný alebo vynechávaný text musí byť dobre utvorený (žiadne syntaktické chyby) a nemal by byť príliš veľký — ak je potrebný dlhý text, v parametri by mal byť `\input`. Dokumentácia (samotnom súbore balíku) vám prezradí, ako deklarovať, ktoré sekcie majú byť vložené: toto môžete spraviť v preambule dokumentu, ale dokumentácia navrhuje tiež spôsoby, ktorými to môžete spraviť na príkazovom riadku vyvolávajúcom LaTeX alebo interaktívne.

Konečne balík *verbatim* (ktorý by mal byť dostupný v každej distribúcii) definuje prostredie `comment`, ktoré umožňuje dedikovanému používateľovi editoru potlačiť časti LaTeXovského zdrojového súboru. Trieda `memoir` poskytuje rovnaké prostredie.

`askinclude.sty`: `macros/latex/contrib/misc/askinclude.sty`
`comment.sty`: `macros/latex/contrib/comment`
`excludeonly.sty`: `macros/latex/contrib/misc/excludeonly.sty`
`memoir.cls`: `macros/latex/contrib/memoir`
`optional.sty`: `macros/latex/contrib/misc/optional.sty`
`pagesel.sty`: distribuované s balíkmi Heika Oberdieka `macros/latex/contrib/oberdiek`
`pdfpages.sty`: `macros/latex/contrib/pdfpages`
`selectp.sty`: `macros/latex/contrib/misc/selectp.sty`
`verbatim.sty`: distribuované ako súčasť `macros/latex/required/tools`
`version.sty`: `macros/latex/contrib/misc/version.sty`
`versions.sty`: `macros/latex/contrib/versions/versions.sty`

193. Časti dokumentu z rozličných adresárov

Bežným spôsobom zostavovania veľkých dokumentov je ich rozdelenie na sadu menších súborov (napr. jeden na každú kapitolu) a uchovávanie všetkého, čo súvisí s jedným z týchto súborov, v podadresári.

TeX bohužiaľ nemá možnosť zmeny aktuálneho adresáru, takže všetky súbory, na ktoré sa odkazujete, musíte uviest relatívne voči adresáru hlavného súboru. Väčšina ľudí toto pokladá za neintuitívne.

Vhodným riešením môže byť použitie techniky rozšírenia cesty z Otázky 54. Ak však v dokumente používate viacero súborov rovnakého názvu (napr. `chapter1/fig1.eps` a `chapter2/fig1.eps`), pri vyvolaní `\input{sect1}` v hlavnom súbore nedávate TEXu žiadnu informáciu, na ktorý súbor sa odkazujete. Tento problém je jednoducho riešiteľný pri súboroch vytváraných ľuďmi (používajte rozličné názvy). Názvy automaticky vytvorených súborov sú často rovnaké a ich zmena je procedúra náchylná k vytvoreniu chýb.

Balík *import* vám v tomto prípade prichádza na pomoc. Definuje príkaz `\import`, ktorý akceptuje úplnú cestu a názov súboru v tomto adresári, zabezpečí pritom, aby veci fungovali správne. Takže napríklad súbor `/home/friend/results.tex` môže obsahovať

```
Graph: \includegraphics{picture}
\input{explanation}
```

`\import{/home/friend/}{results}` potom vloží oba súbory. Príkaz `\subimport` robí to isté pre podadresár (relatívna cesta radšej než absolútna), existujú zodpovedajúce príkazy `\includefrom` a `\subincludefrom`.

`import.sty`: `macros/latex/contrib/misc/import.sty`

194. Súbory Makefile pre LATEXovské dokumenty

Použitie *make* s LATEXom je komplikované. Potreba viacnásobného spustenia LATEXu (napr. pre ustálenie obsahu, návestí, pridanie bibliografie, indexu atď.) sa ťažko vyjadruje obvyklým druhom grafu závilostí, ktorý konštruujeme pre *make*.

Z tochto dôvodu bol jediným balíkom podobným *make* na CTANe (dlhú dobu) *latexmk*, čo je skript pre *Perl* analyzujúci závislosti vášho dokumentu, v prípade potreby spúšťajúci BIBTEX alebo *makeindex* (v prípade zmeny súborov, ktoré uvedené programy používajú ako vstup) atď. *Latexmk* je dobrým riešením, nedávno bol aktualizovaný a obsahuje množstvo skvelých vlastností dovoľujúcich mu pracovať ako veľmi jednoduchý WYSIWYG systém.

Systém *texinfo* (viď Otázku 16) je dodávaný s utilitou nazvanou *texi2dvi*, ktorá umožňuje konvertovanie súborov LATEXu alebo *texinfo* do DVI (alebo pomocou PDFTEXu do PDF).

Novším prírastkom je balík *latexmake* poskytujúci množinu pravidiel pre *make* podľa potreby vyvolávajúcich *texi2dvi*.

`latexmake`: `support/latexmake`

`latexmk`: `support/latexmk`

`texi2dvi`: distribuované ako súčasť `macros/texinfo/texinfo`

195. Koľko stránok je v mojom dokumente?

Jednoduché dokumenty (tie, čo začínajú stranou 1 a nemajú v číslovaní strán nijaké zlomy až po poslednú stranu) nepredstavujú pri hľadaní odpovede na túto otázku žiadny problém. Počet strán hlási balík *lastpage* v svojom návestí `LastPage`.

Pre komplikovanejšie dokumenty (napríklad knihy s inak čislovanou úvodnou časťou) tento jednoduchý prístup nestačí.

Balík *count1to* definuje návestie `TotalPages`. Toto je hodnota jeho kópie `\count1` (rezervovaný TEXovský počítací register) na konci dokumentu.

Balík *totpages* definuje návestie `TotPages`, ale tiež sprístupňuje svoj register na použitie ako LATEXovské počítadlo `TotPages`, na ktoré sa môžete odkazovať cez `\theTotPages`. `TotPages` je samozrejme asynchrónny rovnakým spôsobom ako čísla strán, ale „snapshoty" môžu byť vo výstupnej rutine bezpečne uskutočnené.

Trieda `memoir` definuje dve počítadlá (`lastpage` a `lastsheet`), ktoré sú nastavené (po prvom spracovaní dokumentu) na ekvivalent návestí `LastPage` a `TotalPages`).

Balíky *count1to* a *totpages* potrebujú podporu balíku *everyshi*.

`count1to.sty a everyshi.sty`: distribuované v `macros/latex/contrib/ms`

`lastpage.sty`: `macros/latex/contrib/lastpage`

memoir.cls: macros/latex/contrib/memoir
totpages.sty: macros/latex/contrib/totpages

196. Vloženie TEXovských súborov do LATEXu

Napriek tomu, že je LATEX pôvodne založený na Plain TEXu (viď Otázku 10), neobsahuje všetky príkazy Plain TEXu. Čo je však horšie, niektoré príkazy Plain TEXu sa v LATEXu objavujú s odlišnou sémantikou. Dôsledkom toho je, že pre sadzbu dokumentov Plain TEXu (alebo ich častí) LATEXom musia byť vykonané špeciálne opatrenia.

Skutočne spoľahlivé riešenia je preloženie príkazov Plain TEXu tak, aby produkovali ekvivalent pôvodnej sémantiky. Toto však v mnohých prípadoch nie je praktické a pre tieto prípady si často musíte pomôcť balíkom *plain*. Balík definuje prostredie plain, v ktorom môže byť spracovaný dokument Plain TEXu:

```
\begin{plain}
  \input{plain-doc}
\end{plain}
```

Je známe, že balík zlyháva napríklad pri dokumentoch používajúcich $\mathcal{AMS}$-TEX, určite by zlyhal aj pri požiadavke o načítanie Eplainu. Všetky tieto problémy sa dajú prekonať (aj keď to niekedy nie je ľahké), ale nám v mnohých prípadoch šetrí mnoho práce.
plain.sty: distribuované ako súčasť macros/latex/contrib/carlisle

197. Moje slová sa nerozdeľujú

Predpokladajme, že ste si vybrali správny TEXovský „jazyk" (viď Otázku 46), pravdepodobne nedosiahnete požadovaný výsledok sadzbou jedného jazyka a použitím rozdeľovacích pravidiel jazyka druhého. (Ak ste používateľ LATEXu, vyberte si správny jazyk pomocou *babel*. Takto môžete odhaliť potrebu ďalšej sady rozdeľovacích vzorov, viď Otázku 221).

Čo iné sa ešte môže pokaziť?

- Od TEXu verzie 3.0 boli medze povolenej vzdialenosti rozdelenia od začiatku alebo konca slova programovateľné (viď Otázku 230) a dotyčné hodnoty mohli byť z nejakého dôvodu v niektorých makrách, ktoré používate, nesprávne nastavené. TEX nebude rozdeľovať menej než `\lefthyphenmin` znakov od začiatku slova, ani menej než `\righthyphenmin` pred koncom slova, tým pádom nebude rozdeľovať slová kratšie než súčet týchto dvoch hodnôt. Napríklad pre angličtinu sú tieto hodnoty 2 a 3, takže TEX nebude rozdeľovať slová kratšie než 5 písmen, ak si o slove myslí, že je anglické.
- TEX nebude rozdeľovať slová, ktoré už boli rozdelené. Napríklad anglické priezvisko Smyth-Postlethwaite nebude rozdelené, čo môže spôsobovať problémy. Toto je správny sadzobný štýl pre angličtinu (nemusí byť správny pre iné jazyky), ak však naozaj rozdelenie potrebujete, môžete nahradiť spojovník v priezvisku príkazom `\hyph` definovaným
  ```
  \def\hyph{\penalty0\hskip0pt\relax}
  ```
  Tento spôsob ale obvykle neodporúčame. Balík *hyphenat* definuje sadu príkazov na vloženie rozdelenia na miesto rozličných interpunkčných znamienok.
- Slová môžu byť akcentované. Príčiny a riešenia sú uvedené v Otázke 233.
- Rozdelenie mohlo byť prehliadnuté. TEXovský algoritmus je dobrý, ale nie je neomylný a v niektorých jazykoch prehliadne vhodné rozdelenia. V takomto prípade musíte dať TEXu *explicitné* inštrukcie pre rozdelenie.

Príkaz `\hyphenation` vám dovoľuje zadať explicitné inštrukcie. Ak je slovo vôbec rozdelené (teda rozdeleniu nebráni nijaké z uvedených obmedzení), príkaz anuluje hocičo, čo prikazujú rozdeľovacie vzory. Príkaz akceptuje ako parametre jedno alebo viac rozdeľovaných slov — `\hyphenation{ana-lysis pot-able}` (anglické príklady); všimnite si, že (v tomto prípade pre slovo „analysis") môžete anulovať TEXom navrhované rozdelenie (ana-lysis je britské etimologické rozdelenie; niekomu sa zdá americké rozdelenie „nešťastné"...).
hyphenat.sty: macros/latex/contrib/hyphenat

198. Zastavenie všetkých rozdeľovaní

Táto akcia sa môže zdať nezvyčajnou (jednou z veľmi propagovaných vlastností TEXu je kvalita rozdeľovania), ale niekedy je potrebná. Skutočným problémom je, že kvalita TEXovského výstupu je štandardne do veľkej miery závislá na prítomnosti rozdeľovania.

TEX (zmätočne) ponúka štyri mechanizmy na potlačenie rozdeľovania (pred rozšíreniami TEXu verzie 3 to boli iba dve).

Prvou možnosťou je nastavenie pokút za rozdeľovanie `\hyphenpenalty` a `\exhyphenpenalty` na „nekonečnú" hodnotu (t.j. napríklad 10000). Toto znamená, že každý riadok, ktorý by obsahoval rozdelenie, bude pokutovaný dostatočne na to, aby k nemu nedošlo. Nevýhodou metódy je nutnosť opätovného vyhodnocovania odstavcov, a teda pomalosť celej metódy.

Ďalším spôsobom je výber jazyka, pre ktorý neexistujú rozdeľovacie vzory. Niektoré distribúcie vytvárajú jazyk `nohyphenation` a balík *hyphenat* túto techniku používa pre svoj príkaz `\nohyphens`, ktorý svoj parameter vysádza bez rozdelenia.

Tretia alternatíva je nastavenie `\left-` a/alebo `\righthyphenmin` na dostatočne veľkú hodnotu, aby sa zabránilo akémukoľvek rozdeľovaniu, keďže hodnota minima bude väčšia ako dĺžka najdlhšieho slova, ktoré je TEX ochotný rozdeliť (vhodnou hodnotou je 62).

Poslednou možnosťou z nášho zozname je potlačenie rozdeľovania celého textu použitím aktuálneho fontu príkazom

```
\hyphenchar\font=-1
```

Toto nie je práve praktický spôsob potlačenia rozdeľovania, keďže príkaz musí byť použitý pre každý font v dokumente, ale práve týmto spôsobom zabraňuje rozdeľovaniu LATEX samotný v `tt` a iných fontoch pevnej šírky.

Výber konkrétnej techniky záleží na tom, čo chcete robiť. Ak text, ktorého rozdelenie chcete potlačiť, nie je celý odstavec, vašou jedinou voľbou je použitie nerozdeľovaného jazyka: hodnota jazyku je zachovaná s textom (rovnakým spôsobom ako aktuálny font); na počítanie rozdeľovania sú použité hodnoty rozdeľovacích limitov a pokút aktívne na konci odstavca.

Na druhú stranu, pri písaní viacjazyčného dokumentu použitím balíku *babel* nerozdeľovaný jazyk alebo rozdeľovacie minimum na potlačenie rozdeľovania použiť *nemôžete*. Všetky tieto hodnoty sa menia pri prepnutí jazyka balíkom *babel*. V tomto prípade použite rozdeľovacie pokuty.

Ak jednoducho vypnete rozdeľovanie pre dlhší úsek textu, výstup bude mať nerovný okraj (s mnohými pretekajúcimi riadkami) a každý beh TEXu vás bude bombardovať hláškami o pretečení (overfull) a podtečení (underfull) riadkov. Aby ste sa tomuto vyhli, máte dve možnosti. Môžete použiť `\sloppy` (alebo jeho verziu v podobe prostredia `sloppypar`) a nechať TEX roztiahnuť riadky s podtečením a zalomiť riadky s pretečením pred tým, než nastane (a samozrejme roztiahnuť zbytok). Alternatívne môžete text vysádzať text zarovnaný vpravo a aspoň sa tak zbaviť riadkov s pretečením.

`hyphenat.sty`: `macros/latex/contrib/hyphenat`

199. Ako spraviť bold-tt alebo bold-sc

LATEX štandardne neposkytuje žiadne prostriedky spracovania tučných „teletype" alebo „small-caps" fontov. Existuje pre to praktický dôvod (Knuth nikdy takéto fonty nenavrhol), ale existujú aj typografické dôvody (polotučný font `cmtt` je v skutočnosti celkom tučný (v porovnaní s inými fontami pevnej šírky) a tučné fonty „small-caps" nie sú u mnohých profesionálnych typografov populárne).

Na CTANe existuje „extra" sada súborov METAFONTu, ktoré poskytujú tučné verzie `cmtt` a `cmcsc` („small-caps" font). V moderných distribúciach TEXu môžete začať tieto fonty využívať po umiestnení na vhodné miesto do *texmf* stromu (viď Otázku 48). Po opätovnom vybudovaní potrebných indexov súborov TEX (a priatelia) automaticky vybudujú všetky potrebné súbory fontov pri prvom odkaze na ne. Balík *bold-extra* vybuduje potrebne dátové štruktúry fontov, aby ste fonty mohli využívať z LATEXu.

Ak potrebujete použiť Type 1 fonty, nemôžete postupovať s fontami Knuthovho štýlu, keďže neexistujú Type 1 verzie *mf-extra* sady. Komerčné fonty pevnej šírky (vrátane štandardného balíku *Courier*) takmer vždy obsahujú tučnú verziu, takže tu problém nie je. PSNFSS (viď Otázku 94) typicky poskytuje „nepravé" „small-caps" fonty a nemá výčitky svedomia ani z ich poskytovania

v tučnej forme.
bold-extra.sty: macros/latex/contrib/misc/bold-extra.sty
bold tt a small caps fonty: fonts/cm/mf-extra/bold

200. Sazba tučných řeckých písmen

Jednoduchým řešením tohoto problému je příkaz \mathbf, který však nefunguje ve všech případech, protože malá řecká písmena chovají jinak než velká (z důvodu Knuthova esoterického kódování fontů). Nicméně za normálních okolností můžeme \mathbf použít pro řecké verzálky, ale AMS-LaTeXový balík amsmath.sty nepovoluje toto přepínání fontů a pak tedy musíte použít jednu z níže popsaných technik. V některých případech funguje plainTeXové řešení {\boldmath$\theta$}, ale \boldmath nemůže být použit v matematickém módu, pak to musíte vyřešit následovně:
$...\mbox{\boldmath$\theta$}...$,
to působí problémy v superskriptech a podobně. Tyto problémy mohou být přičítány použití balíku \boldmath.

- balík *bm*, který je součástí distribuce LaTeXových nástrojů (macros/latex/required/tools) definuje příkaz \bm, který lze použít kdekoliv v matematickém režimu.
- balík *amsbsy*, který je součástí AMS-LaTeXu (macros/latex/required/amslatex) definuje příkaz \boldsymbol, který pokrývá téměř všechny základní problémy. Všechna tato řešení můžete použít na všechny matematické symboly, nejen na řecká písmena.

201. Ako používať znak podčiarknutia

Znak podčiarknutia _ sa v TeXu obvykle používa na označenie dolných indexov v matematickom móde. Ak napíšete znak _ uprostred bežného textu, TeX sa bude sťažovať. Ak píšete dokument obsahujúci veľké množstvo týchto znakov, predstava písania _ (alebo ešte horšie \textunderscore) pre každý z nich vystraší väčšinu bežných ľudí.

Stredne pokročilý programátor makier môže vygenerovať rýchly „hack“ na povolenie sadzby _ vo význame znaku podčiarknutia. Kód je však trochu komplikovaný a hlavne existuje množstvo bodov, na ktorých sa dá ľahko pokaziť. Existuje však balík *underscore*, ktorý poskytuje obecné riešenie tejto požiadavky.

Existuje tu však jeden problém. Textové fonty OT1 obsahujú znak _ iba vo typewriter verzii kódovania (používanej fontami pevnej šírky ako cmtt). Takže sa musíte uistiť, že sa znaky podčiarknutia objavujú iba v texte sádzanom fontom písacieho stroja, alebo musíte používať bohatšie kódovanie, ako napríklad T1, ktoré obsahuje daný znak pre každý font.

Ak je požiadavkou iba občasné použitie znaku _, je možné použiť nasledujúcu konštrukciu:

```
\def\us{\char`\_}
...
\texttt{create\us process}
```

Táto konštrukcia nie je ani najmenej robustná (v bežnom hovorovom význame), ale je robustná pri expanzii (t.j. v LaTeXovskom význame slova), takže ju používajte opatrne (nemusíte sa však starať o nadpisy sekcii a podobné veci).
underscore.sty: macros/latex/contrib/misc/underscore.sty

202. Ako vysádzať znak ‘@’?

Pre dlhým časom robili niektoré balíky znak ‘@’ aktívnym, takže pre jeho sadzbu boli potrebné špeciálne opatrenia. Aj keď sú tieto balíky ešte stále dostupné, používa ich málo ľudí a tí, čo ich používajú, majú prístup k veľmi dobrej dokumentácii.

Bežným ľuďom (ako sú napríklad autori originálneho anglického FAQ) stačí jednoducho napísať znak ‘@’.

203. Porovnávanie „job name“

Token `\jobname` produkuje postupnosť znakov s kódom kategórie 12 („iné“) bez ohľadu na to, čo danými znakmi v skutočnosti je. Keďže je nevyhnutne potrebné porovnať makro s obsahom iného makra (použitím `\ifx`), je potrebné vytvoriť makro, ktorého expanzia vyzerá rovnako ako expanzia `\jobname`. Toto môžeme spraviť pomocou `\meaning`, ak „odrežeme“ prefix príkazu `\show`.

Celý príkaz vyzerá nasledovne:

```
\def\StripPrefix#1>{}
\def\jobis#1{FF\fi
  \def\predicate{#1}%
  \edef\predicate{\expandafter\StripPrefix
                  \meaning\predicate}%
  \edef\job{\jobname}%
  \ifx\job\predicate
}
```

A je použitý nasledovne:

```
\if\jobis{mainfile}%
  \message{YES}%
\else
  \message{NO}%
\fi
```

Zapamätajte si, že príkaz `\StripPrefix` nemusí byť definovaný, ak používate LaTeX — tu už interný príkaz (viď Otázku 243) `\strip@prefix`, ktorý môžete použiť, existuje.

204. Sadzba znaku Euro

Európska mena „Euro“ je reprezentovaná symbolom s mierne pochybným dizajnom, ale je to dôležitá mena a používatelia TEXu/LaTEXu ju potrebujú sádzať.

Európska komisia najskôr požadovala, aby symbol Euro bol vždy vysádzaný grotesk fontom, táto požiadavka však bola našťastie zrušená a teda je možné najlepšie využiť sadzobné úsilie, aby symbol vyzeral aspoň trochu solídne (typograficky).

Fonty v kódovaní TS1 poskytované ako súčasť distribúcie fontov EC obsahujú glyfy pre Euro. Fonty sa nazývajú Text Companion (TC) fonty a poskytujú rovnaký rozsah kresieb ako EC fonty samotné. Balík *textcomp* poskytuje na prístup k symbolu príkaz `\texteuro`, ktorý vyberie symbol zodpovedajúci okolitému textu. Dizajn symbolu v TC fontoch nie je obecne v láske... Napriek tomu, ak produkujete dokument pomocou Knuthovych Computer Modern fontov, používajte TC fontovú verziu symbolu.

Vstupné kódovanie *latin9* definované balíkom *inputenc* má znak Euro definovaný (pozícia 164, v iných ISO Latin znakových sadách obsadená „symbolom meny“). Kódovanie používa pre symbol príkaz `\texteuro`. V súčasnosti je príkaz dostupný iba z balíku *textcomp*. V kódovej stránke MicroSoftu je pozícia tiež, ale štandardizácia takýchto vecí prebieha dosť odlišným spôsobom a projektu LaTEX neboli poskytnuté detaily o zmene.

Obrysové fonty obsahujúce iba symboly Eura sú (zadarmo) dostupné z `ftp://ftp.adobe.com/pub/adobe/type/win/all/eurofont.exe` (súbor je zbalený ako samorozbalovací spustiteľný súbor pre *Windows*, ale na iných operačných systémoch môže byť dekódovaný ako archív formátu `.zip`). Sada *euro* obsahuje metriky, *dvips* „map“ súbory a makrá (pre Plain TEX a LaTEX) pre použitie týchto fontov v dokumentoch. Používatelia LaTEXu nájdu v sade dva balíky: *eurosans* poskytuje iba grotesk verziu (pre dodržanie zastaraného nariadenia o výhradne grotesk symboloch; balík poskytuje príkaz `\euro`), zatiaľ čo *europs* prispôsobí Euro symbol okolitému textu (poskytujúc príkaz `\EUR`). Pre použitie niektorého z balíkov s *latin9* kódovaním potrebujete definovať `\texteuro` ako alias pre Euro príkaz, ktorý definuje balík.

Adobe fonty sú pre prostredie bez Computer Modern fontov najlepšia voľba. Zjavne sú navrhnuté tak, aby sedeli s Adobe Times, Helvetica a Courier, ale sú zrejme použiteľné s širším rozsahom moderných fontov.

Balík *eurofont* poskytuje kompaktnú analýzu problému symbolu pre Euro vo svojej dokumentácii a poskytuje makrá na konfiguráciu použitého zdroja glyfov. Balík je však príliš veľký na každodenné použitie.

Sada *euro-ce* je celkom potešujúcim METAFONTovským návrhom poskytujúcim Euro symboly vo viacerých tvaroch. Súbor distribúcie `euro-ce.tex` poskytuje rady, pomocou ktorých môže fonty využiť používateľ Plain TEXu.

Symboly Euro sa dajú nájsť na množstve iných miest, ktoré tu pre úplnosť spomenieme.

Fonty *marvosym* obsahujú medzi množstvom dobrých vecí aj symbol Eura. Font na CTANe nie je Adobe *ATM* kompatibilný, ale kompatibilná verzia je zdarma k dispozícii na `http://www.YandY.com/download/marvosym.zip`. Font na CTANe je dodávaný so sadou makier na sadzbu všetkých symbolov, ktoré obsahuje.

Iné sady založené na METAFONTe obsahujúce Euro symboly môžete nájsť v balíku *china2e* (ktorého primárnym cieľom sú čínske dátumy a podobné veci) a fontoch *eurosym*.

`china2e bundle`: `macros/latex/contrib/china2e`
`EC fonts`: `fonts/ec`
`euro fonts`: `fonts/euro`
`euro-ce fonts`: `fonts/euro-ce`
`eurofont.sty`: `macros/latex/contrib/eurofont`
`eurosym fonts`: `fonts/eurosym`
`marvosym fonts`: `fonts/psfonts/marvosym`
`textcomp.sty`: súčasť distribúcie LATEXu.

205. Ako získať znaky copyright, trademark, ...

Obsiahly zoznam symbolov (viď Otázku 32) obsahuje príkazy pre symboly `\textcopyright`, `\textregistered` a `\texttrademark`, ktoré sú dostupné vo fontoch v TS1 kódovaní a ich použitie sa umožňuje balíkom *textcomp*.

V skutočnosti sú v štandardnom LATEXu povolené všetky tri príkazy, ale glyfy, ktoré dostanete, nie sú veľmi pekné. Konkrétne `\textregistered` sa správa zvláštne pri vložení do tučného textu (napríklad nadpis sekcie), pretože sa skladá z „small-caps" písmena, ktoré sa typicky pri požiadavke na vysádzanie v tučnom fonte degraduje na písmeno normálneho tvaru. Z glyfu sa teda stane 'r' v krúžku, ale správnym symbolom malo byť 'R' v krúžku.

Tomuto efektu sa použitím *textcomp* samozrejme vyhnete.

Ďalší problém nastáva, keď chcete `\textregistered` v pozícii horného indexu (aby symbol vyzeral podobne ako `\texttrademark`). Použitím horného indexu matematického módu za týmto účelom vyvolá množstvo nezmyselných chýb, takže *musíte* použiť

```
\textsuperscript{\textregistered}
```

206. Definice nových označení matematických funkcí v LATEXu

Použije se příkaz `\mathop` podobně jako v tomto příkladě:

```
\newcommand{\diag}{\mathop{\rm diag}}
```

Spodní a horní indexy funkce `\diag` budou umístěny obdobně jako u `\lim` (limita). K tomu, aby indexy byly vždy umístěny vpravo od označení funkce, se použije tato definice:

```
\newcommand{\diag}{\mathop{\mathrm{diag}}\nolimits}
```

AMS-LATEX (ve svém balíku *amsopn*) poskytuje příkaz `\DeclareMathOperator`, který udělá stejnou práci jako první definice nahoře. Na vytvoření našeho původního příkazu `\diag` napíšeme:

```
\DeclareMathOperator{\diag}{diag}
```

`\DeclareMathOperator*` deklaruje operátor tak, aby byly spodní a horní indexy umístněny v pozici `\limits` (viz Otázku 210).

207. Zátvorky a symboly premenlivej veľkosti

Jedným z mála do očí bijúcich nedostatkov v schopnostiach TEXu sádzať matematiku je nemožnosť sadzby oddeľovačov vo vnútri matematických výrazov. TEX poskytuje primitíva nazvané `\left` a `\right`, ktoré môžu byť použité na modifikáciu zátvoriek (akéhokoľvek druhu) okolo matematických výrazov, ako napríklad: `\left( <výraz> \right)` — veľkosť zátvoriek sa prispôsobí vertikálnemu rozmeru výrazu.

V mnohých prípadoch pri matematike budete potrebovať príkaz `\middle` na použitie vo výrazoch ako:

```
\left\{ x\in \mathbb{N} \middle| x \mbox{ párne} \right\}
```

na špecifikáciu množiny párnych prirodzených čísel Systém ε-TEX (viď 298) definuje presne takýto príkaz, ale používatelia Knuthovho pôvodného systému potrebujú pomoc. Balík Donalda Arseneaua *braket* poskytuje príkazy pre špecifikáciu množín (ako je uvedené vyššie) a pre Diracovove zátvorky (a symboly „bra“ a „ket“). Balík používa príkaz zabudovaný do ε-TEXu (ak zistí, že beží pod ε-TEXom).

`braket.sty`: `macros/latex/contrib/misc/braket.sty`

208. Prispôsobenie veľkostí metematických fontov

V `plain` TEXu musíte pri zavádzaní novej veľkosti fontu deklarovať, aké veľkosti fontov majú byť použité v matematike. Dá sa to spraviť deklaráciou `\textfont`, `\scriptfont` a `\scriptscriptfont` pre matematické rodiny fontov, ktoré používate. Všetky tieto veci sú popísané v kapitole 17 TEXbooku (Otázka 23) a iných knihách a tutoriáloch (Otázka 28), ktoré sa `plain` TEXom zaoberajú dostatočne detailne.

V LATEXu je samozrejme všetko automatizované: pre každú veľkosť (textového) fontu existuje schéma určujúca, aké veľkosti matematických fontov sa majú použiť. „Schéma“ najskôr skontroluje sadu „známych“ veľkostí textu, z ktorých pre každú sú dopredu deklarované veľkosti matematiky. Ak veľkosť textu nie je „známa“, sú veľkosti script- a scriptscriptfont vypočítané ako fixné násobky veľkostí textových fontov. (Použité koeficienty sú `\defaultscriptratio`=0.7, a `\defaultscriptscriptratio`=0.5.)

Vzorec pevného násobku môže produkovať nepríjemné výsledky (hlavne ak používate fonty, o ktorých si LATEX myslí, že sú dostupné iba vo fixnej sade veľkostí). Môžete taktiež chcieť úplne zmeniť myšlienky LATEXu, napríklad sadzbou matematiky citeľne menšej alebo väčšej, než je okolný text. Za týmto účelom môže byť použitý LATEXovský príkaz:

```
\DeclareMathSizes{<tfs>}{<ts>}{<ss>}{<sss>}
```

(tento príkaz používa aj LATEX na definíciu vlastnej sady veľkostí). Toto vytvorí (alebo znovuvytvorí) veľkosti matematických fontov pre veľkosť textového fontu `<tfs>` (ako `\textsize` sa použije `<ts>`, ako `\scriptsize` `<ss>` a ako `\scriptscriptsize` bude použité `<sss>`).

`\DeclareMathSizes` môže byť použitý iba v preambule dokumentu: dostupné je iba jedno priradenie pre každú veľkosť textového fontu na celý dokument.

209. Tri bodky

Tri bodky sú bežne požadované a LATEX ich natívne poskytuje v celkom slušnom rozsahu (`\dots`, `\cdots`, `\vdots` a `\ddots`). Použitím balíku *graphics* môžete zmeniť sklon príkazu `\ddots`:

```
$ ... \reflectbox{$\ddots$} ... $
```

Hoci postup funguje, nie je odpoúčaným spôsobom dosiahnutia žiadaného výsledku (viď nižšie). LATEXovský rozsah naviac nepostačuje požiadavkám každého. Tri balíky poskytujú rozšírenie.

Sada $\mathcal{AMS}$-LATEX poskytuje rozsah „sémanticky pomenovaných“ elíps na použitie v rôznych situáciach: `\dotsb` pre použitie medzi dvojicami binárnych operátorov, `\dotsc` medzi dvojicami čiarok a tak ďalej.

Balík *yhmath* definuje príkaz `\adots`, ktorý je podobný príkazu `\ddots` (iba so sklonom nadol namiesto sklonu nahor). *yhmath* obsahuje celkom zaujímavý font, ktorý rozširuje štandardný *cmex*, detaily nájdete v dokumentácii. Nevýhodou tohto nastavenia je (hoci `\adots` je iba makro), že sa balík pokúša nahrať vlastný font a v prípade, že font nemáte nainštalovaný, zapríčíňuje varovnú hlášku substitúcie „missing font“ (chýbajúci font).

Balík *mathdots* (okrem opravy chovania TEXovských/LATEXovských `\ddots` a `\vdots` pri zmene veľkosti fontu) poskytuje „inverzne diagonálne“ elipsy `\iddots` (fungujú podobne ako `\adots` balíku *yhmath*, len lepšie).

Dokumentácia *yhmath* je v distribúcii spracovaná (nemusíte teda balík inštalovať iba pre čítanie dokumentácie). Documentácia *mathdots* je na konci samotného súboru balíku.
`amslatex`: `macros/latex/required/amslatex`
`graphics.sty`: súčasť sady `macros/latex/required/graphics`
`mathdots.sty`: `macros/generic/mathdots.sty`
fonty `yhmath`: `fonts/yhmath`
makrá `yhmath`: `macros/latex/contrib/yhmath`

210. Pozície horných a dolných indexov operátorov

Najbežnejším ručne písaným štýlom pre výrazy je umiestnenie limitných výrazov operátorov ako `\sum` a `\int` fyzicky nad a pod operátor. V TEXu/LATEXu píšeme tieto výrazy pomocou horných a dolných indexov aplikovaných na operátor, ale nie vždy sa vo výstupe TEXu objavia ako v „ručne písanej" podobe.

Dôvod je ten, že keď sa výraz objaví v normálnom texte (je teda v štýle `\textstyle`), umiestnenie hraníc operátoru by mohlo viesť k nerovnému riadkovaniu (a teda k ťažko čitateľnému textu). Je preto obvyklé (v `\textstyle`) umiestňovať tieto hranice ako horné a dolné indexy premenných.

Toto však nie je obecne dostačujúce, takže je poskytované primitívum `\limits`:

```
$\sum\limits_{n=1}^{m} ...$
```

ktoré umiestni hranice priamo nad a pod symbol.

Naopak, môžete chcieť zmeniť usporiadanie hraníc v `\displaystyle`. Za týmto účelom existuje príslušný príkaz `\nolimits`:

```
\[\sum\nolimits_{n=1}^{m} ...\]
```

ktorý umiestni hranice ako v `\textstyle`.

(Zapamätajte si, že makro `\int` má obyvkle `\nolimits` zabudované do definície. V TEXbooku existuje príklad, ktorý ukazuje, ako zvláštne vyzerá `\int\limits` vysádzané.)

211. Text vo vnútri matematiky

Keď v TEXu/LATEXu píšeme matematiku, majú písmená, z ktorých tvoríme bežný text, špeciálny význam: všetky sa stanú jednopísmenovými názvami premenných. Písmená sú sádzané kurzívou, ale nie rovnakou kurzívou ako v bežnom texte: postupnosť písmen v matematike (napríklad „here") vyzerá zvlášte „zvlnená" oproti tomu istému slovu v texte vysádzanom kurzívou. Rozdiel je, že text v kurzíve je podrezaný (kerned), aby sa písmená k sebe dobre zmestili, ziatiaľ čo matamtika je sádzaná, aby vyzerala, ako keď navzájom násobíte h, e, r a písmeno e. Ďalšou zvláštnosťou matematiky v TEXu je ingorovanie medzier: prinajlepšom môžeme v tomto zvláštnom fonte písať slová.

Ak chceme mať dobre vyzerajúci text medzi matematikou, musíme zaviesť určité opatrenia. Ak používate LATEX, nasledujúca rada by vám mala pomôcť.

Najjednoduchšie je použiť `\mbox` alebo `\textrm`:

```
$e = mc^2 \mbox{here we go again}$
```

Problém je, že veľkosť textu zostane totožná s okolitým textom, takže

```
$z = a_{\mbox{other end}}$
```

vyzerá veľmi zle.

Ďalší jednoduchý postup využíva `\textrm` a vyzerá sľubnejšie:

```
$z = a_{\textrm{other end}}$
```

je určite v poriadku. Okolitý text však nemusí byť v našej antikve. Ak vám záleží na rovnakom texte, musíte si vybrať medzi `\textrm`, `\textsf` a tak ďalej.

(Matematická inštancia vašej antikvy (`\mathrm`) má správnu veľkosť, ale keďže je určená na použitie v matematike, medzery sú ignorované — používajte `\mathrm` iba pre názvy premenných v kolmej rímskej abecede.)

Tieto problémy môžete opraviť výbermi veľkosti písma v texte, napr.:

```
$z = a_{\mbox{\scriptsize other end}}$
```

Príklad bude fungovať, ak je okolitý text štandardnej dokumentovej veľkosti, inak dostanete nesprávnu veľkosť.

Tieto skratky sú (takmer) v poriadku pre „občasného“ matematika, ale skutoční matematici žiadajú postup, ktorý odbremeňuje sádzača od myšlienok vyžadovaných pri predchádzajúcich postupoch. $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX, ako obvykle, poskytuje všetko potrebné — príkaz `\text`. Príkaz v skutočnosti balík *amstext*, ale ten je nahratý „globálnym“ balíkom *amsmath*, takže kdokoľvek správne používa $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX, má príkaz poruke, takže aj vtipný matematici môžu napísať:

```
\usepackage{amsmath}
...
$z = a_{\text{other end}}$
```

a text bude rovnakej veľkosti a v rovnakom fonte ako text okolitý.

$\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX tiež prostredníctvom príkazu `\intertext` poskytuje opatrenia pre komentáre vložené do niektorej z jeho viacriadkových zobrazovacích štruktúr. Napríklad:

```
\begin{align}
  A_1&=N_0(\lambda;\Omega')-
  \phi(\lambda;\Omega'),\\
  A_2&=\phi(\lambda;\Omega')-
  \phi(\lambda;\Omega),\\
  \intertext{a konečne} A_3&=\mathcal{N}(\lambda;\omega).
\end{align}
```

umiestni text „a konečne“ na samostatný riadok pred posledný riadok zobrazenia. Príkaz môže byť použitý iba bezprostredne po príkaze `\\`.

Vyčerpávajúcu dokumentáciu $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeXu nájdete v súbore `amsldoc` (v distribúcii), ktorý je dostupný aj na webe: `ftp://ftp.ams.org/pub/tex/doc/amsmath/amsldoc.pdf`
`amsldoc.tex`,
`amsmath.sty` a
`amstext.sty` sú distribuované ako súčasť $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeXu
`\AMSLaTeX{}`: `macros/latex/required/amslatex`

212. Opätovné použitie rovnice

Pri opakovaní existujúcej rovnice nechcete iba rovnakú matematiku, chcete aj opätovne použiť pôvodné návestie. Balík *amsmath* nám prichádza na pomoc:

```
\usepackage{amsmath}
...
\begin{equation}
  a=b
  \label{eq1}
\end{equation}
...
Nezabudnite, že
\begin{equation}
  a=b
  \tag{\ref{eq1}}
\end{equation}
```

Druhá inštancia $a = b$ tu bude vysádzaná s kópiou návestia prvej inštancie vytvorenou príkazom `\tag`.

Vyčerpávajúcu dokumentáciu $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeXu nájdete v súbore `amsldoc` (v distribúcii), ktorý je dostupný aj na webe: `ftp://ftp.ams.org/pub/tex/doc/amsmath/amsldoc.pdf`
`amsldoc.tex` a
`amsmath.sty` sú distribuované ako súčasť $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeXu (`macros/latex/required/amslatex`)

213. Obrázky v TeXovém dokumentu

Součástí TeXových dokumentů mohou být obrázky. Protože pro obrázky existuje mnoho formátů, existuje i více způsobů, jak zařídit, aby se nám na daném místě dokumentu obrázek objevil. Obrázek se však nevkládá „přímo do zdrojového textu". Do zdrojového textu se umístí volání makra, které vytvoří místo pro obrázek. Skutečný import obrazových dat provede posléze TEXový ovladač.

Obvykle používaný formát obrázků je `eps` (zapouzdřený PostScript). Pro vložení takového obrázku lze použít balík `macros/latex209/contrib/epsfig`. Další informace o vkládání postscriptových obrázků a odkazy na starší balíky viz Otázku 103.

Vkládání bitmapových obrázků ve formátu `pcx` podporují ovladače v emTEXu (viz Otázku 43).

214. Obtékání obrázků

Někdy je třeba umístit obrázek dovnitř odstavce, na jednu stranu textového zrcadla – takovým obrázkům říkáme obtékané. Balík `macros/latex209/contrib/picinpar` definuje prostředí `window`, kterým lze obrázek umístit na levou nebo pravou stranu odstavce (nebo dokonce doprostřed).

Další podobné balíky můžete v případě zájmu najít na CTANu (`macros/latex/contrib/wrapfig` nebo `macros/latex209/contrib/window`).

215. Sazba názvů programů používaných spolu s TEXem

Donald Knuth definoval logo TEXu jako konkrétní ukázku toho, co TEX umí. Žel, mnozí si však myslí, že tím přivodil záplavu různých log, jako například AMS-TEX, PICTEX, BIBTEX atd. vytvořených pomocí velkého množství různých fontů, velikostí i základních linek — jistě vám to znepříjemní čtení. Lamport vymyslel logo LATEX, což pak přirozeně vedlo ke vzniku nynějšího loga LATEX 2_ε.

Není však nutně třeba uvádět tato loga v jejich komplikovaném tvaru všude tam, kde se příslušný název vyskytne, avšak pro ty, kdo trvají na jejich použití, existuje v souboru `info/biblio/texnames.sty` veliké množství předdefinovaných log. Fonty loga METAFONTu mohou být přidány k fontům, které LATEX 2_ε již zná (tak, že se pak může měnit jeho rozměr spolu s okolním textem) použitím balíku `macros/latex/contrib/mflogo`. Při použití Knuthova fontu pro MetaPost však můžou nastat problémy (můžete dostat META O T). Nezoufejte, autor samotný používá jenom „MetaPost".

Pro ty, kdo nepotřebují mít v dokumentu vysázená ta „správná" loga, stačí použít `AMS-\TeX{}` (AMS-TEX) namísto AMS-TEX, `Pic\TeX{}` (PicTEX) namísto PICTEX, `Bib\TeX{}` (BibTEX) namísto BIBTEX atp.

216. Jednosloupcový výtah v dvousloupcovém dokumentu

Občas je třeba dvousloupcový článek opatřit jednosloupcovým výtahem. Nejlepším způsobem je použít následující trik:

```
\documentclass[twocolumn]{article}
...
\begin{document}
... % \author, etc
\twocolumn[
  \begin{@twocolumnfalse}
    \maketitle
    \begin{abstract}
      ...
    \end{abstract}
  \end{@twocolumnfalse}
]
```

Takto však nebude pracovat správně, použijete-li `\thanks` v seznamu `\author`. Pokud potřebujete takto speciálně číslovaný seznam, docílíte toho následovně:

```
\title{Demonstration}
\author{Me, You\thanks{}}
\twocolumn[
  ... to samé, co výše ...
```

```
  ]
  {
    \renewcommand{\thefootnote}%
      {\fnsymbol{footnote}}
    \footnotetext[1]{Thanks for nothing}
  }
```

atd.

Alternativně můžete použít balík *abstract* (dostupný z `macros/latex/contrib/abstract`), který (kromě jiných usnadnění práce s abstrakty) poskytuje k použití příkaz `\saythanks` a prostředí `onecolabstract`, které odstraní potřebu „švindlovat" s `\thanks` a poznámkami. Můžete to použít následovně:

```
\twocolumn[
  \maketitle
  \begin{onecolabstract}
  ... text
  \end{onecolabstract}
]
\saythanks
```

217. Změna slov používaných LaTeXem

LaTeX používá několik symbolických jmen, které vytvářejí automatický text (nadpisy, tituly atd.). Např. část textu vysázená příkazem `\tableofcontents` je vždy volána příkazem `\contentsname`. Umožňuje však běžnému uživateli změnit jména používaná standardními třídami, což je užitečné hlavně při tvorbě neanglických dokumentů. Pokud třeba daný dokument píše český autor, chce, aby byl obsah nazván opravdu „Obsah" a ne „Table of Contents", v tom případě musí do preambule dokumentu vložit text

```
\renewcommand{\contentsname}{obsah}
```

Nicméně pro neanglického uživatele je přirozené používat *babel*, protože nabízí mnoho výhod a zpříjemňuje vytváření dokumentů v jejich přirozeném jazyce. Zvláště pokud v *babel*u vyberete jiný jazyk, tak vám zajistí, že LaTeXová symbolická jména budou vhodně překládána do vybraného jazyka podle vašeho pořadavku.

Bohužel výběr jmen v *babel*u nevyhoví vždy požadavkům uživatele, proto je třeba mechanismus pro náhradu standardních jmen. Kdykoliv *babel* vybere jiný jazyk, pak jsou všechna standardní jména přeložena právě do tohoto jazyka. *Babel* zvláště vybírá hlavní jazyk dokumentu, když je vykonán příkaz `\begin{document}`, který ihned zničí veškeré změny standardních jmen, která byla provedena v preambuli dokumentu používajícího *babel*. Proto *babel* definuje příkaz, který uživateli umožní změnit definice symbolických jmen na základní výrazy jeho mateřštiny:

`\addto\caption<language>`, kde `<language>` značí jazyk, který *babel* bude používat jako primární. Např.

```
\addto\captionczech{%
   \renewcommand{\contentsname}{Obsah}%
}
```

218. Výpisy zdrojových kódů v LaTeXu

Někteří neuroticky estetičtí programátoři považují „krásné" zdrojové kódy za něco k nezaplacení. Správně strukturované zdrojové kódy však mají své místo při psaní disertačních prací studenty informatických i jiných věd, od kterých se očekává, že v budoucnosti budou programovat. Běžně se používá doslovné citování kódů.

Doslovné vkládání kódů je však rozebíráno jinde (viz Otázku 129). Problém sazby specifikací algoritmů (otázka 219) je v tomto FAQ uveden také na jiném místě.

„Krásné" kódy jsou generovány pomocí pre-compileru, avšak balík `listings` toto převádí na LaTeX.

Dobrým pre-compilerem je systém `lgrind` (`nonfree/support/lgrind`), který obsahuje mnoho výhodných vylepšení, která se vám mohou hodit, a také má velice široký výběr jazyků.

Systém `tiny_c21` (`support/tiny_c21`) je o něco novější, uživatelé si zde mohou vygenerovat vlastní ovládací soubory pro daný jazyk, jehož kódy zpracovávají.

Systém `C++2LaTeX` (`support/C++2LaTeX-1_1`) přináší mnoho doporučení pro používání s jazyky C a C++.

Systém *highlight* (`support/highlight`) je obecný konvertor produkující formáty HTML, RTF a také TEX/LATEX. Tvrdí o sobě, že je vysoce přispůsobitelný, distribuce obsahuje definice mnoha jazyků. Dokumentace je poskytována jako `README` soubor v distribuci, který se sám nazýva manuálem.

listings: `macros/latex/contrib/listings`

219. Sadzba pseudokódu v LATEXu

Konsenzus týkajúci sa správnosti sadzby pseudokódu neexistuje. Máme teda na výber z množstva balíkov LATEXu určených na tvorbu esteticky príjemných výpisov pseudokódu.

Pseudokód sa od výpisu skutočného programu líši v tom, že nemá striktnú syntax ani sémantiku. Pseudokód má predstavovať čistú interpretáciu algoritmu, takže môže potrebovať použiť matematické zápisy, obrázky, tabuľky a iné vlastnosti LATEXu, ktoré sa v tradičných programovacích jazykoch neobjavujú. Sadzba výpisov programov je opísaná na inom mieste (viď Otázku 218).

Určite si môžete na sadzbu pseudokódu vytvoriť vlastné prostredie, napríklad pomocou prostredia `tabbing` alebo `list` — nie je to ťažké, ale môže to byť nudné. Oplatí sa teda vyskúšať nasledujúce balíky (všetky špecificky navrhnuté pre sadzbu pseudokódu).

Sada *algorithms* (obsahujúca balíky *algorithm* a *algorithmic*, oba potrebné pre bežné použitie) má jednoduché rozhranie a produkuje celkom pekný výstup. Poskytuje primitíva pre príkazy, ktoré môžu obsahovať ľubovolné príkazy LATEXu, komentáre a sadu iteratívnych a podmienených konštrukcií. Tieto primitíva môžu byť ľahko predefinované, aby na výstupe produkovali odlišný text. Podpora pre pridávanie nových primitív však neexistuje. Samotná sadzba pseudokódu sa koná v balíku *algorithmic*. Balík *algorithms* používa prostriedky balíku *float* na sekvenčné číslovanie algoritmov, umožnenie algoritmom „plávať“ (float) ako obrázky alebo tabuľky a na podporu vloženia zoznamu algoritmov na úvod dokumentu.

Balík *alg*, podobne ako *algorithms*, poskytuje plávajúce prostredie algoritmov spolu so všetkými ďalšími drobnosťami. *alg* ale môže svojim plávajúcim prostrediam vytvoriť nadpis v množstve (prirodzených) jazykov. *alg* naviac (narozdiel od *algorithms*), umožňuje jednoducho pridávať nové konštrukcie.

Balík *newalg* má tak trochu podobné rozhranie ako *algorithms*, ale jeho výstup má pripomínať celkom príjemnú sadzbu použitú v knihe „Introduction to Algorithms“ od Cormana, Leisersona, Rivesta a Steina. *newalg* bohužiaľ nepodporuje plávajúce prostredie ani prispôsobenie výstupu.

Poctivé použitie štýlu „Introduction to Algorithms“ dosiahneme Cormenovým vlastným balíkom *clrscode*: tento balík bol použitý v druhom vydaní knihy.

Balík *algorithm2e* existuje už veľmi dlho a je široko používaný a odporúčaný. Používa balík *float* na poskytnutie možnosti plávajúcich popisov algoritmov, stále však môžete použiť voľbu „`H`“ balíku *float*, ak chcete, aby sa algoritmus obajvil tam, „kde ho napíšete“.

Použitie balíku *program* sa trochu líši od použitia iných balíkov. Namiesto textového módu používa pri sadzbe programov matematický mód, zlomy riadkov sú dôležité. *program* neobsahuje plávajúce prostredie, ale čísluje algorithmy ako *alg* a *algorithms*. Prispôsobovanie a rozširovanie nie je podporované.

Žiadny z uvedených balíkov nie je dokonalý. Faktory, ktoré by mali ovplyvniť váš výber, sú preferovaný štýl výstupu, potreba rozšírenia alebo modifikácie kľúčových slov a potreba umiestniť algoritmy do plávajúceho prostredia ako obrázky a tabuľky .

Dostupnosť dokumentácie:

sady *algorithms* je poskytovaná v súbore `algorithms.ps` (tiež dostupný ako zdrojový súbor LaTeXu). Dokumentácia spomína balíky, ako by boli v skutočnosti jeden nazvaný *algorithms*.

balíku *program* (v stave, v akom je) sa nachádza v súbore `program.msg`.

balíku *clrscode* sa nachádza v súbore `clrscode.pdf` v distribúcii.

balíku *algorithm2e* môžete nájsť v distribúcii v súbore `algorithm2e.tex` (pre spracovanie potrebuje balík samotný).

`algorithm2e.sty`: `macros/latex/contrib/algorithm2e`

sada `algorithms`: `macros/latex/contrib/algorithms`

`alg.sty`: `macros/latex/contrib/alg`

`clrscode.sty`: `macros/latex/contrib/clrscode`

`float.sty`: `macros/latex/contrib/float`

`newalg.sty`: `macros/latex/contrib/newalg`

`program.sty`: `macros/latex/contrib/program`

220. Sazba symbolu hlavní hodnoty integrálu

Tento symbol („přeškrtnutý“ znak integrálu) není obsažen v žádném fontu, který by byl bežně dostupný uživatelům TEXu nebo LATEXu, může však být vytvořen pomocí následujícího makra:

```
\def\Xint#1{\mathchoice
   {\XXint\displaystyle\textstyle{#1}}%
   {\XXint\textstyle\scriptstyle{#1}}%
   {\XXint\scriptstyle\scriptscriptstyle{#1}}%
   {\XXint\scriptscriptstyle\scriptscriptstyle{#1}}%
   \!\int}
\def\XXint#1#2#3{{\setbox0=\hbox{$#1{#2#3}{\int}$}
     \vcenter{\hbox{$#2#3$}}\kern-.5\wd0}}
\def\ddashint{\Xint=}
\def\dashint{\Xint-}
```

Pro sazbu jednou přeškrtnutého znaku integrálu se použije příkaz `\dashint`, pro sazbu dvakrát přeškrtnutého znaku použijeme `\ddashint`.

221. Používání nových jazyků v babelu

Babel umí pracovat se širokou řadou jazyků, avšak občas se stane, že nový uživatel chce používat jazyk, který jeho instalace TEXu neumí používat. Např. požadavek

```
\usepackage[catalan]{babel}
```

vyvolá chybové hlášení

```
Package babel Warning: No hyphenation patterns
... were loaded fo the language ‘Catalan’.
... I will use the patterns loaded for
... \language=0 instead.
```

Problém je v tom, že TEX neví, jak dělit katalánská slova: musíte mu to říct dříve, než *babel* začne pracovat. V LATEXové instalaci musíte přidat do souboru `language.dat`, který je součástí babelovké instalace, řádek

```
catalan        cahyphen.tex
```

který, pokud odstraníte komentovací znaménko (`%`), řekne LATEXu, že má nahrát vzor pro katalánské dělení slov, pokud mu řeknete, aby vytvořil nový formát. Bohužel, v mnoha babelovských distribucích právě tento řádek není správný – musíte zkontrolovat jména souborů, obsahující vzory, které se chystáte použít. Jak můžete vidět, v autorově systému je očekáván soubor `cahyphen.tex`, ačkoliv jméno souboru, který je v systému, je `cahyph.tex`. V lepších distribucích je většina těchto souborů obsažena, chybějící soubory si můžete najít na CTANovském archívu. Pokud si stáhnete nový soubor, zajistěte jeho korektní instalaci (viz Otázku 52). Nakonec musíte obnovit formáty, které používáte (ve skutečnosti většina uživatelů používá *babel* v LATEXových dokumentech, takže zde bude dost formátů, které bude nutno upravit; nicméně uživatelé často opravují soubory, které s tímto nemají nic společného).

teTEX, fpTEX : pro obnovu všech formátu použijte toto: `fmtutil --all` Pokud si chcete být jisti, co děláte, (to není pro zbabělé), můžete si vybrat posloupnost formátů, které chcete upravovat, a pro každý z nich spustit `fmtutil --byfmt <formatname>`, kde `<formatname>`

je něco jako LaTeX nebo `fmtutil --byhyphen <hyphenfile>`, kde `<hyphenfile>` je soubor specifikující dělení slov do formátu – obvykle `language.dat`.

MikTeX V MikTeXovské distribuci do verze 2.0 udělejte: `Start` –> `Programs` –> `Miktex` –> `Maintenance` –> `Create all format files` nebo otevřete DOSovské okno a napište `initexmf --dump`. V MikTeXovských distribucích verze 2.0 a vyšší může být celá procedura provedena pomocí GUI. Nový jazyk vyberete takto: `Start` –> `Programs` –> `Miktex 2` –> `Miktex options` a vyberte `Languages`. Vyberte jazyk ze seznamu a stiskněte tlačítko `Apply` a svou volbu potvrďte stiskem tlačítka `OK`. Pak vyberte `general` a stiskněte tlačítko `Update Now`. Další možností je vyeditování souboru `language.dat` jak bylo popsáno výš a spusťte `initexmf --dump` jako u verzí nižších než 2.0.

POZOR: Je možné, že váš TeX skončí s chybovou hláškou týkající se nedostatku paměti při vytváření nových formátů. Většina TeXových implementací má na uchování dat týkajících se tohoto problému vyhrazeno pole fixní délky. Pokud shledáte, že vám přetekla paměť, může to být zásluha prohledávání seznamu dostupných jazyků ve vašem souboru `language.dat`, pak se tedy podívejte, který z jazyků může být odstraněn.

P. Makra pro jednotlivé typy dokumentů

222. Sazba článků pro časopisy

Vydavatelé časopisů mají širokou škálu požadavků na vzhled článků a zatímco mnoho vydavatelů přijímá elektronické předlohy v (La)TeXu, často nepředkládají makra doporučená pro uchování ve veřejných archívech. Přesto však je v CTANu dostupný značný počet maker. Hledáním podle vašeho jména časopisu v CTANovském katalogu (`http://www.tex.ac.uk/tex-archive/help/Catalogue/catalogue.html`) můžete objevit přesně to, co sháníte.

Nestane-li se tak, můžete kontaktovat případného vydavatele vašeho článku; mnoho vydavatelů má makra vystavena na svých webových stránkách nebo jinak zpřístupněná. Zkontrolujte, zda vydavatel nabízí makra vhodná pro prostředí, které chcete používat: někteří z nich nemají žádná makra pro současnou verzi LaTeXu, např. trvají na tom, že nejlepší je LaTeX 2.09. Někteří vydavatelé vás vyzvou, že jim to můžete poslat v jakékoliv podobě, že vůbec nezáleží na tom, jaká makra používáte, jiní vám pouze doporučí, že máte používat co nejméně rozšíření standardních balíků, aby bylo snadné převést váš článek do jejich podoby.

223. Výťah z viacerých článkov

Občas sa stane, že je treba vydať zborník konferencie, do ktorej boli zaslané príspevky v LaTeXu.

Najbližšie vždy použiteľnému riešeniu majú triedy Petera Wilsona `combine` a Federica Garciu `subfiles`.

`Combine` definuje prostriedky pre „`\import`“ celých dokumentov a umožňuje špecifikovať významné vlastnosti rozmiestnenia dokumentu, ako aj globálny obsah a iné. Prídavný balík *combinet* dovoľuje použitie `\title` a `\author` (atď.) `\import`ovaných dokumentov v globálnom obsahu.

`Subfiles` (trieda) sa používa v zložkových súboroch viacsúborového projektu, príslušný balík *subfiles* sa používa v hlavnom súbore. Pomocou opatrení môžu byž zložkové súbory vysádzané s iným formátom stránky, ...

„Surovší“ toolkit je poskytovaný balíkmi *includex* a *moredefs* od Matta Swifta (oba sú súčasťou balíku *frankenstein*) ponúka možnú cestu vpred.

Includex umožňuje použiť `\includedoc` na celé články — použijete `\include` na súbory kapitol (chapter) v obyčajnom výťahu. Nespraví to však všetku prácu za vás. Musíte analyzovať požívanie balíkov jednotlivými článkami a zaistiť, že zhodná množina bude uvedená v preambule hlavného projektu.

Úplne iným prístupom je použitie balíku *pdfpages* a vloženie článkov poskytnutých v PDF formáte do PDF dokumentu vytvoreného PDFLaTeXom. Balík definuje príkaz `\includepdf`, ktorý akceptuje parametre podobné parametrom príkazu `\includegraphics`. Uvedením kľúčových slov ako voliteľných parametrov príkazu môžete špecifikovať, ktoré stránky uvedeného súboru chcete

vložiť, ako aj rôzne detaily týkajúce sa rozvrhnutia vložených strán.
combine.cls: macros/latex/contrib/combine
combinet.sty: macros/latex/contrib/combine
frankenstein: macros/latex/contrib/frankenstein
pdfpages.sty: macros/latex/contrib/pdfpages
subfiles.cls atď.: macros/latex/contrib/subfiles

224. Životopis (Curriculum Vitae, Résumé)

Trieda pre životopisy vytvorená Andrejom Brodnikom je dostupná z macros/latex/contrib/vita. Pri tejto triede môže byť zvolený predmet (sú ponúkané voľby pre počítačových vedcov aj spevákov) i jazyk (môžete písať v češtine, slovenčine aj angličtine). Rôzne rozšírenia môžu byť vytvorené pomocou súborov pre voľbu triedy alebo makra definovaného na triedach pre vytvorenie nového typu položky.

Trieda Didiera Vernu curve je založená na modeli, v ktorom je CV (životopis) tvorené množinou *rubrík* (každá sa zaoberá dôležitou položkou, o ktorej chcete pojednávať, ako „vzdelanie“, „pracovné skúsenosti“ atď.). Dokumentácia je podporená niekoľkými vzorovými súbormi, je poskytnutý aj mód pre emacs.

Alternatívou používania oddelenej triedy je zavedenie balíku na jednu zo štandardných tried. Napríklad balík *currvita* od Axela Reicherta bol odporučený tímu FAQ. Jeho výstup isto vyzerá dobre.

Existuje aj LaTeX 2.09 balík *resume*, ktorý toho okrem rady, aby sme ho *nepoužívali*, veľa neobsahuje.
currvita.sty: macros/latex/contrib/currvita
curve.cls: macros/latex/contrib/curve
resume.sty: macros/latex209/contrib/resume

225. Listy a podobné dokumenty

LaTeX samotný poskytuje triedu letter, ktorá je všeobecne nenávidená. Súčasný autor už dávno vzdal všetky snahy. Ak ju napriek tomu chcete vyskúšať, ale vadí vám spôsob vertikálneho posunutia jednostranového listu, skúste nasledujúci hack:

```
\makeatletter
\let\@texttop\relax
\makeatother
```

umiestnite do preambuly svojho súboru.

„Urob si sám“ je obvyklou stratégiou. Knuth (pre použítie s jeho plain TeXom v knihe TeXbook) a Kopka s Dalym (v ich dokumente Guide to LaTeX) poskytujú spracované príklady.

Existujú však aj alternatívy — v skutočnosti je ich nechutne veľa. Nasledujúci zoznam poskytuje iba malý výber.

Najväčšiou a najobsiahlejšiou triedou je newlfm. Časť mena („lfm“) naznačuje, že trieda vie vytvárať listy (letters), faxy a memorandá. Dokumentácia je veľmi objemná a balík vyzerá byť flexibilný.

Trieda akletter od Axela Kielhorna je jedinou ďalšou triedou odporúčanou na zahrnutie v tomto FAQ, ktorej dokumentácia je dostupná v angličtine.

Trieda dinbrief, je taktiež odporúčaná, no jej dokumentácia je dostupná iba v nemčine.

Listové triedy sú zahrnuté aj v skvelých balíkoch KOMA-script (scrlttr2: dokumentácia v angličtine) a *ntgclass* (brief: dokumentácia iba v holandčine). Aj keď tieto balíky môžu byť kvalitné (kvalitu vzbudzujú už samotné balíky, v ktorých sa vyskytujú), neboli používateľmi odporúčané.
akletter.cls: macros/latex/contrib/akletter
brief.cls: distribuované ako súčasť macros/latex/contrib/ntgclass
dinbrief.cls: macros/latex/contrib/dinbrief
newlfm.cls: macros/latex/contrib/newlfm
scrlttr2.cls: distribuované ako súčasť macros/latex/contrib/koma-script

226. Vyvažovanie stĺpcov na konci dokumentu

Voľba štandardných tried *twocolumn* donúti LaTeX vysádzať text dokumentu do dvoch stĺpcov. Na poslednej stránke však majú stĺpce obvykle rôznu dĺžku — hovoríme, že takéto stĺpce sú *nevyvážené*. Mnohým ľuďom (väčšine) sa nevyvážené stĺpce nepáčia.

Najjednoduchším riešením je používanie balíku *multicol* namiesto voľby *twocolumn* (*multicol* štandardne stĺpce na poslednej strane vyvažuje). Používanie balíku *multicol* má svoju cenu: jeho špeciálna výstupná rutina znemožňuje používanie floatov vo vnútri stĺpcov, aj keď povoľuje floaty plnej šírky (napríklad prostredie `figure*`).

Výsledkom je stála potreba vyvažovanie stĺpcov na konci *twocolumn* dokumentu. Prácu môžeme samozrejme spraviť manuálne: `\pagebreak` vložený na správne miesto poslednej strany často vytvorí požadovaný efekt, nie je to však lákavé, a ak je posledná strana tvorená automaticky generovaným textom (napr. bibliografia alebo index), vloženie príkazu bude zložité.

Balík *flushend* poskytuje riešenie. Je to tak trochu nebezpečný kus makrokódu, ktorý upravuje jednu z najkomplikovanejších častí jadra LaTeXu bez nasadenia ochrán. Balík iba mení vlastnosti na konci dokumentu (jeho príkaz `\flushend` je štandardne povolený) a ďalší príkaz dovoľuje nastavenie finálneho vyváženia. Iné balíky zo sady poskytujú prostriedky na vkladanie materiálu plnej šírky do dvojstĺpcových dokumentov.

Balík *balance* upravuje aj výstupnú rutinu (trochu opatrnejšie ako *flushend*).

Používatelia by si mali byť vedomí, že balíky sú nachýlné nechať sa zmiasť v prítomnosti floatov: ak sa tento problém vyskytne, je pravdepodobné, že bude potrebné ručné nastavenie dokumentu. Práve tento problém viedol autora balíku *multicol* k potlačeniu jednostĺpcových (v zmysle šírky) floatov.

`balance.sty`: distribuovaný ako súčasť `macros/latex/contrib/preprint`
`flushend.sty`: distribuovaný ako súčasť `macros/latex/contrib/sttools`
`multicol.sty`: distribuovaný ako súčasť `macros/latex/required/tools`

227. Nadpis mojej sekcie je pre hlavičku stránky príliš dlhý

Príkazy LaTeXu na vytváranie sekcií štandardne sprístupňujú kapitoly a sekcie pre použitie v hlavičkách stránok (a pre podobné prípady). Hlavičky stránok majú začne obmedzený priestor a často sa stane, že nadpis je príliš veľký. Príkazy LaTeXu na vytváranie sekcií preto akceptujú voliteľný parameter:

```
\section[krátky nadpis]{úplný nadpis}
```

Ak je prítomný $<$krátky nadpis$>$, je použítý pre obsah aj nadpis stránky.

Používanie rovnakého textu v obsahu a záhlaví nemusí byť dostačujúce. Ak sú nadpisy vašich kapitol veľmi dlhé (ako viktoriánske romány), rozumný postup je mať skrátenú položku pre obsah a skutočne strohú položku v záhlaví.

Jedným z problémov je zvyklosť sádzať hlavičky vo verzálkach. Prečo ich teda nesádzať ako pre obyčajné čítanie? Bez modifikácie LaTeXu to je nemožné, ale balík *fancyhdr* poskytuje pre použitie v riadkoch hlavičky (a pätky) príkaz `\nouppercase` na potlačenie tendence LaTeX sádzať verzálky. Triedy v sade `KOMA-script` v prvom prípade nerobia verzálky.

Príkazy na vytváranie sekcií v skutočnosti požívajú príkazy „mark" na predávanie informácií hlavičkám, napríklad `\chapter` používa `\chaptermark`, `\section` používa `\sectionmark` atď. S touto znalosťou môžeme dosiahnuť trojvrstvovú štruktúru kapitol:

```
\chapter[stredná verzia]{rozsiahla verzia}
\chaptermark{stručná verzia}
```

Uvedené by malo uspokojiť každú potrebu.

Kapitoly to majú jednoduché, máloktorá kniha má hlavičku na začiatočnej stránke kapitoly. V prípade sekcií musíme vziať do úvahy vlastnosti `\*mark` príkazov: do hlavičky je umiestnená prvá značka (mark) na strane (pri zlyhaní tohoto postupu je to posledná značka z predchádzajúcej strany). Výsledkom je, že postup pre sekcie je únavnejší:

```
\section[[stredná verzia]{rozsiahla verzia%
          \sectionmark{stručná verzia}}
\sectionmark{stručná verzia}
```

Prvý príkaz `\sectionmark` sa postará o hlavičku stránky, do ktorej spadá príkaz `\section`, druhý sa stará o nasledujúce stránky. Všimnite si potrebu voliteľného parametru `\section`, aj keď „*stredná verzia*" je v skutočnosti rovnaký text ako „*rozsiahla verzia*".

Podobné opatrenie je potrebné aj pre kapitoly, ak trieda, ktorú používate, vkladá hlavičku stránky na začiatočnú stranu kapitoly.

Balík *titlesec* spravuje záhlavia úplne iným spôsobom. Používatelia tohto balíku by sa mali obrátiť na jeho dokumentáciu.

Trieda `memoir` sa vyhýba hlúpostiam poskytnutím ďalšieho nepovinného parametru pre príkazy kapitol a sekcií, napríklad:

```
\section[stredná verzia][stručná verzia]%
        {rozsiahla verzia}
```

Používatelia `memoir` si teda vždy môžu text hlavičky ušiť na mieru bez väčších problémov.

`fancyhdr.sty`: `macros/latex/contrib/fancyhdr`

balík `KOMA script`: `macros/latex/contrib/koma-script`

`memoir.cls`: `macros/latex/contrib/memoir`

`titlesec.sty`: `macros/latex/contrib/titlesec`

228. Číslovanie stránok n z m

Zistenie čísla poslednej stránky dokumentu z vnútra dokumentu je trochu záludné. Balík *lastpage* nám má v tomto smere uľahčiť život. Definuje návestie `LastPage`, ktoré predstavuje správne číslo (po dostatočne veľa spracovaniach LaTeXom). Trieda `memoir` tiež definuje návestie poslednej strany.

Dokumentácia balíku *fancyhdr* presne popisuje, ako môžete túto informáciu využiť na vytvorenie číslovania uvedeného v znení otázky.

dokumentácia `fancyhdr`: `macros/latex/contrib/fancyhdr`

`lastpage.sty`: `macros/latex/contrib/lastpage`

229. Číslovanie strán podľa kapitol

V minulosti bolo pre voľne viazané technické manuály obvyklé číslovať strany podľa kapitol (v určitých situáciach je to dobré, napr. pri korekciách kapitoly, pri ktorých pridáte pár strán, stačí distribuovať iba zmenenú kapitolu).

Problém vyzerá na prvý pohľad rovnaký ako v odpovedi na Otázku 265, základná technika je tiež dosť podobná.

Upratanie voľných koncov, zaistenie nastavenia čísla strany na správnu hodnotu na začiatku každej kapitoly atď. však predstavuje väčšiu výzvu. Toto je dôvod, pre ktorý bol napísaný balík *chappg*. Okrem zjavných vecí toho však robí viac.

Používatelia sa často pýtajú na čísla strán v sekcii, to ale nedáva zmysel. Číslovať strany môžete v objektoch dokumentu, ktoré vždy začínajú na nových stranách.

Dokumentácia *chappg* sa nachádza v súbore balíku.

`chappg.sty`: `macros/latex/contrib/misc/chappg.sty`

Q. Něco nefunguje tak, jak by mělo

230. Nezvykle rozdělená slova

Pravděpodobně je problém v pomíchaných verzích TeXu. Systém dělení se totiž s přechodem od verze 2.9 na 3.0 změnil. Používá-li se (`plain`) TeX verze 3.0 nebo pozdější, je třeba se ujistit, že soubor `plain.tex` je alespoň verze 3.0. Používá-li se LaTeX 2.09 (poslední verze byla uvedena 25.března 1992), bylo by dobré přejít na LaTeX 2_ε, jelikož zdrojové soubory LaTeXu 2.09 již nejsou veřejně k dispozici. Pokud nemůžete upgradovat na verzi LaTeX 2_ε, pak si můžete z CTANu stáhnout poslední verzi LaTeXu 2.09 vydanou 25. března 1992. Tím měl být váš problém vyřešen.

Používá-li se LaTeX 2_ε, je příčina chybného dělení pravděpodobně v souboru `hyphen.cfg`, který musí být vytvořen, používáme-li více jazyků najednou.

Pro toho, koho to zajímá, problém spočívá v tomto: V TEXu do verze 3.0 neumožňoval algoritmus dělení slov rozdělit slovo, pokud první část rozděleného slova neobsahovala alespoň dva znaky a jeho druhá část alespoň znaky tři. Počínaje verzí 3.0 lze řídit minimální velikost těchto částí pomocí celočíselných parametrů `\lefthyphenmin` a `\righthyphenmin`. V nových formátech `plain` a `lplain` jsou nastaveny na hodnoty 2 a 3. Mohou být samozřejmě nastaveny na libovolnou hodnotu, s tím, že překročí-li součet `\lefthyphenmin` + `\righthyphenmin` hodnotu 62, nebude docházet k žádnému dělení slov.

Jiný zdroj této zvláštnosti může pocházet z corkovsky kódovaných fontů vydaných roku 1995 (viz Otázku 47), které přišly s novým alternativním stylem dělení. V konfiguračním souboru LaTeXu 2_ε je možné u uvedených fontů povolit alternativní dělení, to může způsobit u některých slov zvláštní vedlejší efekt. Konfigurační soubory z LaTeXu 2_ε vydaného v prosinci 1995 však alternativní dělení nepoužívají.

231. cslatex ignoruje `\hyphenation` a `\righthyphenmin`

V českých a slovenských dokumentech `cslatex` tyto deklarace ignoruje, přestože v LaTeXu 2.09 to fungovalo. Příčina rozdílného chování tkví v tom, že styly `czech.sty` a `slovak.sty`, které `cslatex` používá, vkládají přepínací makro `\czech` resp. `\slovak` prostřednictvím `\AtBeginDocument`. Všechny změny nastavované v preambuli se tudíž týkají implicitního `\language 0`, jímž je americká angličtina. Toto chování má racionální důvod. Pokud při zavádění stylu pro češtinu (nadále budeme pro jednoduchost mluvit pouze o češtině, pro slovenštinu platí totéž) použijeme parametr `split`, aby se slova s rozdělovníkem dělila podle českých typografických pravidel, je znak minus aktivní a nebylo by tudíž možno zadávat záporná čísla a záporné rozměry. To by ovšem značně komplikovalo nastavování různých parametrů v preambuli dokumentu. Navíc by to způsoboovalo, že některé další styly by podle okolností někdy přestaly fungovat. Změny `\righthyphenmin` a `\lefthyphenmin` i definici výjimek dělení `\hyphenation` tedy musíme umístit až za `\begin{document}`.

Často je žádoucí, aby tyto změny byly specifikovány ve stylovém souboru. Pak ovšem musíme zajistit, aby se provedly až po nastavení češtiny. Budeme-li např. celý dokument sázet do úzkých sloupců, můžeme použít

```
\RequirePackage[split]{czech}
\AtBeginDocument{\righthyphenmin 2\relax}
\end{document}
```

Primitiv `\relax` chrání před případnou následující chybnou definicí, která by při své expanzi vytvořila číslici. Příkazem `\RequirePackage` si zajistíme, že se `czech.sty` načte dříve než náš styl a nastavení se skutečně provede pro češtinu.

Pokud vytváříme vícejazyčný dokument, je situace o něco složitější. Příkaz `\czech` totiž vždy nastaví standardní hodnoty. Pokud chceme, aby se `\righthyphenmin` po přepnutí češtiny vždy nastavil na hodnotu 2, můžeme toho dosáhnout například následujícími definicemi:

```
\RequirePackage[split]{czech}
\let\OriginalCzech\czech
\def\czech{\OriginalCzech \righthyphenmin 2\relax}
```

Specifikace výjimek dělení je poněkud jednodušší. Příkaz `\hyphenation` je vždy prováděn globálně. Abychom nepoškodili další makra případnými aktivními znaky, můžeme definice výjimek dělení uzavřít do skupiny například takto:

```
{\czech \hyphenation{po-po-ka-te-pe-tl}}
```

232. Podivné dělení slov v angličtině

Může se stát, že slavné automatické dělení slov TEXu nedělí v místech, která jsou doporučena ve slovníku. Dochází k tomu zřejmě proto, že TEX používá americkou angličtinu, jejíž pravidla pro dělení slov (tak jak jsou uvedena, například, ve Webster's Dictionary) jsou zcela odlišná od pravidel angličtiny britské (tak jak jsou uvedena, například, v Oxford Dictionaries). Na řešení tohoto problému se podílí především Sdružení uživatelů TEXu Spojeného Království (UK TEX

User community) (viz jejich časopis *Baskerville*, vydání 4.4), avšak naprosto uspokojivé řešení je náročné na čas.

233. Akcentovaná slova se nedělí

TEXovský algoritmus dělení slov neuvažuje jako jedno slovo takové, které má v sobě zahrnutý příkaz `\accent`. Jsou pro to dobré důvody, ale znamená to, že kvalitní sazba neanglických jazyků je obtížná.

Aby se TEX vyhnul těmto těžkostem, bylo přijaté corkovské kódování (viz Otázku 47), které obsahuje akcentované znaky jako samostatné znaky. Uživatelé LATEXu mohou dosáhnout tohoto řešení vložením příkazu

```
\usepackage[T1]{fontenc}
```

do preambule dokumentu. Jiná kódování (např. LY1 podporované Y&Y — viz Otázku 63) mohou být použita místo T1. Většina současných 8-bitových kódování by měla pracovat s většinou současných sad dělících šablon.

Snad v budoucnosti Omega (viz Otázku 297) poskytne celkem nové řešení.

234. Rozšíření kapacity TEXu

Občas se stane, že se objeví na terminálu zpráva: „memory capacity exceeded“ (překročení kapacity paměti). Za chybovou hláškou se pak v závorce objeví, co bylo vyčerpáno. Např.:

```
! TeX capacity exceeded, sorry
                    [main memory size=263001].
```

Většinu takovýchto chyb lze vyřešit *bez* jakéhokoli rozšiřování kapacity TEXu. Nejběžnějšími příčinami jsou nepárové závorky, velmi dlouhé řádky a špatně napsaná makra. Velmi dlouhé řádky (extra-long lines) vznikají často při chybném přenosu souborů mezi rozdílnými operačními systémy a zvláště tehdy, když znaky označující konec řádky nejsou převedeny vhodným způsobem (příznakem chyby zapříčiněné velmi dlouhými řádky je chybová zpráva o překročení velikosti bufferu `buf_size`).

Pokud je skutečně třeba rozšířit kapacitu TEXu, volba vhodné metody závisí na použité instalaci. Nejčistší formou jak rozšíření provést, je změnit parametry přímo v modulu 11 zdrojového WEB souboru. Pak musíte znovu překompilovat váš TEX a vygenerovat všechny formátové soubory.

Moderní implementace umožňují změnit velikost některé části paměti TEXu semi-dymanicky. Některé (třeba em-TEX) umožňují nastavit kritéria pro velikost přidělené paměti přímo z příkazové řádky, když program spouštíte. Poměrně často čte TEX při spouštění konfigurační soubor, v němž je specifikována velikost paměti. V systémech založených na *web2c* je tento soubor pojmenovaný `texmf.cnf`: podívejte se do dokumentace vaší implementace. Opět tento soubor musí být v mnoha případech po změně parametrů paměti znovu generován.

235. Plovoucí tabulky a obrázky LATEXu

Tabulky a obrázky (figures) mají sklon k tomu, aby nás udivovaly svým *odplouváním* z místa, kde měly být původně vysázeny. Děje se tak proto, aby bylo dosaženo dokonalého vzhledu dokumentu. Každý profesionální systém určený pro sazbu nechává „odplouvat“ tabulky a obrázky na taková místa, kde jejich umístění nevede k narušení typografických pravidel. Dokonce užijeme-li parametru `h` ve zdrojovém textu, abychom dosáhli umístění „zde“ (here), obrázek či tabulka nebude vytištěna „zde“ v případě, že by to tato pravidla narušilo. Pravidla samotná jsou poměrně jednoduchá a jsou popsána na str. 198, v kapitole C.9 manuálu LATEXu. V horším případě pravidla LATEXu mohou vést k nakupení plovoucích objektů takového rozsahu, jež vede k chybové zprávě „Too many unprocessed floats“ (Přespříliš nezpracovaných plovoucích objektů). To znamená, že omezený počet registrů, ve kterých LATEX uchovává plovoucí objekty, je naplněn. Následuje jednoduchý přehled toho, co je třeba udělat, abychom takové problémy vyřešili (v přehledu se hovoří o obrázcích, ale totéž platí i o tabulkách).

- Musí vaše obrázky být vůbec plovoucí? Když ne, zvažte volbu umístění `[H]` poskytovanou balíkem *float*: obrázky s takovýmto umístěním jsou poskládány tak, aby vypadaly, že jsou plovoucí, i když ve skutečnosti nejsou. Mějte se však na pozoru před „vytékajícími“ floaty: příkazy \caption jsou číslovány v pořadí, v jakém se objevují v dokumentu a [H] float může „převzít“ float, který ještě nebyl umístěn, takže čísla obrázků se popletou.

- Zvolili jsme správné parametry umístění obrázku? Předdefinované (default) parametry (tbp) většinou dostačují a můžete je také rozumně změnit (třeba přidat h). Cokoli uděláte, nikdy nezapomeňte h, poněvadž tím je řečeno: „jestliže to nemohu umístit zde (here), nemohu to umístit ani někde jinde“.
- Můžeme ochránit naše obrázky od odplutí nastavením parametru LaTeXu pro umístění plovoucích objektů? I zde lze říci, že přednastavené hodnoty jsou přiměřené, avšak mohou být v případě problémů přepsány. Blíže jsou popsány na str. 199 – 200, v kapitole C.9 manuálu LaTeXu.
- Najde se v našem dokumentu místo, kam by bylo možné přirozeně vložit příkaz \clearpage? Jestliže ano, pak je dobré tak učinit. Nahromaděné plovoucí objekty se příkazem \clearpage uvolní. (Pozn.: Příkaz \chapter používá \clearpage, a proto obrázky nemohou odplout za konec stávající kapitoly.)
- Skuste balík *placeins*: definuje příkaz \FloatBarrier, za který floaty nemůžou přejít. Volba balíku vám dovoluje deklarovat, že se floaty nemůžou dostat za příkaz \section, \FloatBarrier ale můžete umístit kamkoliv se rozhodnete.
- Jestliže jse obtěžováni floaty objevujícími se na začátku stránky (předtím, jak jsou specifikovány ve vašem textu), vyskoušejte balík *flafter*, který se tomuto problému vyhýbá tím, že trvá na objevení se floatů vždy až po jejich definici.
- Lze použít balíku *afterpage* pro LaTeX 2_ε (adresář macros/latex/required/tools). V dokumentaci k tomuto balíku je na jednom příkladě uvedena myšlenka vložení \clearpage *po* aktuální stránce (tam, kde se uvolněním nakupených plovoucích objektů nevytvoří nevzhledná mezera v textu), avšak je tam uvedeno i to, že tento balík je poněkud křehký. Tuhle možnost využívejte jenom jako poslední útočiště, když další dvě problém neřeší.
- Poslední možností, jak uvedené problémy obejít, je použití balíku *morefloats*, jenž jednoduše zvětší počet možných plovoucích objektů, se kterými může LaTeX pracovat najednou (z 18 na 36), což by mělo ve většině případů dostačovat.
- Jestliže nastane nutnost toho, že všechny obrázky mají být umístěny na konci dokumentu (například, při zasílání článku), nelze spoléhat na to, že to LaTeX učiní za nás. Lépe je použít balíku *endfloat*, který je k takovému účelu určen (najdeme ho na CTANu v macros/latex/contrib/endfloat).

morefloats: macros/latex/contrib/misc/morefloats.sty

236. Užití \pagestyle{empty} na první stránce v LaTeXu

Použije-li se \pagestyle{empty}, a přesto je první stránka označena číslem, pak je pravděpodobně užit i příkaz \maketitle. To není chyba, ale správná vlastnost LaTeXu! Standardní styly LaTeXu jsou psány tak, že u úvodní strany (strany obsahující příkazy \maketitle, \part, či \chapter) se použije styl strany (pagestyle) jiný než ve zbytku dokumentu. Uvedené příkazy spouští \thispagestyle{plain}. Tato vlastnost však není příliš vhodná v případě, že veškerý okolní text má nečíslované stránky (\pagestyle{empty}).

Změny této vlastnosti dosáhneme:

- Když vložíme \thispagestyle{empty} bezprostředně za příkaz \maketitle ; tj. mezi příkazy se nesmí vložit prázdný řádek.
- Použijeme-li *fancyhdr* nebo *scrpage2*, který umožňuje měnit styl úvodní stránky nezávisle na stylu stránek ostatních. Lze je nalézt v adresáři macros/latex/contrib/fancyhdr a jako součást balíku macros/latex/contrib/koma-script
- Když použijete buď třidu memoir (macros/latex/contrib/memoir) nebo nejakou třídu z KOMA-script, užívejte technik doporučených v otázce 187.

237. Ako vytvoriť orezové značky

Ak tlačíte niečo, čo bude nakoniec reprodukované v nezanedbateľných množstvách a zviazané, je obvyklé tlačiť na papier väčší než výsledný produkt a mimo tlačovú oblasť umiestniť orezové značky. Tieto orezové značky sú dostupné vydavateľstvu na nastavenie reprodukčných a rezacích strojov.

Môžete si ušetriť (značné) problémy programovania týchto značiek vlastnoručne použitím balíku *crop*, ktorý obsahuje nástroje na uspokojenie akéhokoľvek mysliteľného vydavateľstva. Používatelia triedy `memoir` balík nepotrebujú, keďže `memoir` má vlastné prostriedky na programovanie orezových značiek.

`crop.sty`: `macros/latex/contrib/crop`
`memoir.cls`: `macros/latex/contrib/memoir`

238. Podtržený text nejde rozdělit

Knuth nevytvořil žádnou podporu pro podtržený text: zastával názor, že podtržení není sázecí technika, ale pouze sloužilo ke zdůraznění části textu na psacích strojích. Odpovídající techniky v sazbě textů je přepnutí z antikvy do kurzívy (nebo naopak). Toto umí se svým argumentem provést příkaz `\emph`.

Přesto však typograficky negramotní lidé (např. ti, kteří používají dvojité řádkování (viz Otázku 146), nadále požadující psát podtrženě, mohou použít příkaz `\underline`, který používá matematické podtrhávání. Tato technika není zcela uspokojivá, nicméně text drží pohromadě a nechce se na konci řádku rozdělit. Řešením mohou být dále uvedené balíky. Balík *ulem*, který předefinuje příkaz `\emph` tak, že svůj argument podtrhne. Text se pak chová jako normální zvýrazněný text a láme se i na koncích řádků. (Balík je schopen i jiných efektů, čtěte dokumentaci uvedenou v souboru balíku samotném.) Balík *soul* definuje příkaz `\ul` (po kterém je balík částečně pojmenován) podtrhující text.

Dejte si pozor na standardní chování balíku *ulem*, které konvertuje příkaz `\emph` na podtrhovací; tomuto lze předejít nahráním balíku:

```
\usepackage[normalem]{ulem}
```

Dokumentace *ulem* je v balíku samotném.

`ulem.sty`: `macros/latex/contrib/misc/ulem.sty`
`soul.sty`: `macros/latex/contrib/soul`

239. Podivné chování `\rm`, `\bf`, atd.

Jestliže příkazy jako `\rm` a `\bf` v LaTeXu přestaly náhle fungovat tak, jak se od nich očekávalo, je to zřejmě proto, že administrátor systému nainstaloval verzi LaTeXu 2.09 spolu s NFSS (viz Otázku 293). V takovém případě je dobré si mu nahlas postěžovat a požádat jej, aby tuto verzi zaměnil za LaTeX 2_ε (viz Otázku 294), v němž příkazy `\rm`, `\bf` apod., jsou-li užity ve standardních třídách — `article`, `report` a `book` aj. — fungují stejně jako předtím. Než k záměně dojde, lze použít `oldlfont.sty`. Tento styl by měl být instalován zároveň s NFSS.

240. Odkazy na příkazy typu `\tenrm` ve starém LaTeXu

V LaTeXu 2.09 byla definována rozsáhlá množina příkazů, jež zpřístupňovaly vestavěné fonty. Například pro různé velikosti fontu `cmr` se v něm dalo použít příkazů: `\fivrm`, `\sixrm`, `\sevrm`, `\egtrm`, `\ninrm`, `\tenrm`, `\elvrm`, `\twlrm`, `\frtnrm`, `\svtnrm`, `\twtyrm` a `\twfvrm`.

Tyto příkazy nebyly nikde dokumentovány, avšak mnohé balíky jich užívaly k dosažení požadovaných efektů.

Protože však tyto příkazy nebyly veřejné, nestaly se součástí nového LaTeXu 2_ε. Chceme-li použít neupravené balíky původně určené pro LaTeX 2.09 v LaTeXu 2_ε, je potřeba použít zároveň balíku `rawfonts.sty`, který je součástí distribuce LaTeXu 2_ε.

241. Chybějící symboly

Nemůžeme-li se dostat k některým symbolům, jako například `\Box` and `\lhd`, pak zřejmě správce systému aktualizoval stávající verzi LaTeXu buď na NFSS (viz Otázku 293) či LaTeX 2_ε (viz Otázku 294). V případě, že máme NSFS, užijeme `oldlfont.sty`, stejně jako v Otázce 239. V LaTeXu 2_ε užijeme balík `latexsym`, jenž je součástí standardní distribuce LaTeXu 2_ε, anebo balík `amsfonts`, je-li instalován.

242. Chybné křížové odkazy v LaTeXu

Někdy, i když necháme zdrojový text proběhnout LaTeXem několikrát, dostáváme stále nesprávné křížové odkazy. Zapamatujme si, že příkaz `\label` musí následovat *po* příkazu `\caption`, případně musí být jeho součástí. Například,

```
\begin{figure}              \begin{figure}
\caption{Obrázek}   nebo    \caption{Obrázek%
\label{fig}                       \label{fig}}
\end{figure}                \end{figure}
```

Stejně efektivně můžete příkaz `\caption` ochránit před jeho sdruženým příkazem `\label` uzavřením popisku do vlastního prostředí. Lidé například často hledají pomoc potom, co

```
\begin{center}
  \caption{Obrázek}
\end{center}
\label{fig}
```

neoznačuje správně. Když skutečně potřebujete toto centrování (zasvěcení ho obyčejně odmítají), použijte:

```
\begin{center}
  \caption{Obrázek}
  \label{fig}
\end{center}
```

243. `\@` a `@` ve jménech maker

Častým zdrojem problémů v dokumentech psaných v LaTeXu je výskyt příkazu `\@`, či jiných příkazů obsahujících znak `@` (at). Nejobvyklejším chybovým hlášením je: „You can't use "`\spacefactor`" in vertical mode" („Nelze použít „`\spacefactor`" ve vertikálním módu"), ale mohou se objevit i jiná obdobná hlášení.

Podobné problémy jsou obvykle způsobeny vložením souboru třídy (class file) či některého stylového souboru (package file) do dokumentu LaTeXu 2_ε nevhodným způsobem, tj. bez použití příkazů `\documentclass` či `\usepackage`.

V LaTeXu se jména vnitřních maker definují obvykle tak, že obsahují znak `@`. Tímto způsobem se autoři stylů snaží vyhnout kolizi mezi názvy příkazů použitými uvnitř stylu a příkazů běžně užívanými v dokumentu. Aby však příkazy tohoto typu mohly vůbec fungovat, musí makra `\documentclass` a `\usepackage` měnit význam znaku `@`.

Problém s chybovým hlášením se jednoduše vyřeší tím, že se styly a třídy do dokumentu vkládají prostřednictvím příkazů `\usepackage` a `\documentclass`.

Je však možné namítnout: „V *The LaTeX Companion* se říká, že příkazy obsahující `@` lze použít!"

Ovšem. Například je tam dlouhý oddíl o příkazu `\@startsection` a o tom, jak jej používat k úpravě vzhledu nadpisů oddílů. Je o tom psáno na straně 15 *The LaTeX Companion*, ale je tam rovněž navrženo provádět obdobné změny v preambuli dokumentu (tj. před `\begin{document}`) mezi příkazy `\makeatletter` a `\makeatother`. Definice `\subsection` na straně 26 by mohla vypadat takto:

```
\makeatletter
\renewcommand{\subsection}{\@startsection
  {subsection}%                   % name
  ...
```

```
  {\normalfont\normalsize\itshape}}% style
\makeatother
```

244. Kde jsou fonty msx a msy

Fonty msx a msy byly navrženy Americkou matematickou společností (AMS) pro sazbu odborných článků v matematických časopisech již v prvopočátcích existence TEXu. Byly navrženy pro „starý“ METAFONT, který nebyl přenositelný a již se nepoužívá. Velmi dlouhou dobu byly tyto fonty dostupné pouze v rozlišení 300dpi, které se ovšem stěží hodí pro moderní tiskárny. AMS nyní předělala fonty pro soudobou verzi METAFONTu. Nové rodiny fontů byly pojmenovány msa a msb. Lze je získat v fonts/amsfonts/sources/symbols

Přesto nás fonty msx a msy nepřestávají sužovat. Jsou doposud lidé, kteří se nerozhodli k aktualizaci fontů. Avšak, i když nakonec každý bude používat nové fonty, zůstanou problémy se starými dokumenty, které se na msx a msy odkazují.

Máme-li zdrojový .tex soubor, který vyžaduje užití msx a msy, nejlepším způsobem, jak se zbavit problému se starými fonty, je taková úprava souboru, aby se používaly msa a msb namísto msx a msy (stačí změnit pouze jedno písmeno ve jménech fontů).

Máme-li dvi-soubor, který se odkazuje na msx a msy fonty, můžeme k přesměrování starých fontů na nové použít balík virtuálních fontů (viz Otázku 42) fonts/vf-files/msx2msa

245. Kde jsou fonty am

Příležitostně se stále objevuje požadavek na soubor písem am; am je zkratka „Almost [Computer] Modern“ (téměř [Computer] Modern). Tyto fonty jsou předchůdci fontů Computer Modern, které všichni známe a máme (nebo nemáme) rádi[7]. Není toho mnoho, co by se s těmito fonty dalo ještě dělat. Jsou, jak ostatně jejich název napovídá, téměř (ale ne úplně) shodné se souborem fontů cm. Máme-li před sebou dokument, jenž požaduje am fonty, jediná rozumná věc, kterou můžeme udělat, je nahradit v dokumentu am za cm. Dvi soubory, které by požadovaly am fonty, se vyskytují tak zřídka, že se nikdo neujal náročného úkolu vytvoření transformace pomocí virtuálních fontů. Můžeme však využít toho, že většina ovladačů umožňuje použít konfigurační soubor, v němž lze zadat substituci fontů. Zadáme-li, že veškeré fonty am mají být nahrazeny odpovídajícími fonty cm, výstup by měl být téměř správný.

246. „Příliš dlouhý řetězec“ v BIBTEXu

Upozornění BIBTEXu „Warning—you've exceeded 1000, the global-string-size, for entry foo“ (Celková velikost řetězce v položce foo je větší než 1000) není toho druhu, jehož se lze vyvarovat jednoduchým pozměněním stylu BIBTEXu. Ke zvětšení přípustné délky řetězců je třeba znovu překompilovat BIBTEX, což není praktické a příliš žádoucí. Proto je třeba problém řešit změnou databáze bibliografických citací.

Problém je obvykle v tom, že databáze obsahuje velmi rozsáhlý abstrakt či anotaci. Jedinou možností nápravy je vynětí tohoto záznamu z databáze tak, aby nebyl překročen limit BIBTEXu. Avšak většinou je potřeba záznam v databázi nějakým způsobem zachovat, jelikož bude užit v sazbě. V takovém případě se záznam vloží do zvláštního souboru:

```
@article{long.boring,
  author =    "Fred Verbose",
  ...
  abstract =  "{\input{abstracts/long.tex}}"
}
```

Tímto způsobem se zařídí, že BIBTEX pracuje pouze se jmény souborů, ale v případě potřeby TEX zařídí vložení celého vlastního textu do dokumentu.

[7] Fonty získaly přívlastek „Almost“ (téměř) po tom, co jejich první implementace v METAFONTu79 nedopadla tak úplně dobře. Knuthův původní záměr byl, že tyto fonty budou konečným řešením.

247. MikTeX nevytváří LaTeXový formát

Práce na MikTeXu 1.20e přestaly 1. prosince 2000. Od této verze již nebyl aktivně vyvíjen. Vydání LaTeXu z 1. června 2000 již nebylo do této distribuce integrováno. Skripty v MikTeXu neumějí vytvořit formátový soubor novějších verzí LaTeXu. Řešení tohoto problému je přístupné na adrese `http://www.miktex.org/1.20/important.html`; zde je link na náhradu souboru `latex.zip` a návod, jak jej nainstalovat. Tento problém byl již dán do pořádku ve verzi MikTeXu vydané 23. ledna 2001.

248. Začátek řádku není rovný

Tato odpověď se dotýká dvou tříd problémů: chyby formy

```
! Missing number, treated as zero.
<to be read again>
                     g
<*> [grump]
```

a toho, že na začátku řádku byla jedna hvězdička, která se záhadně neobjevila na výstupu. Oba problémy vznikly, protože „`\\`“ bere pouze nepovinné argumenty. Příkaz „`\\*`“ znamená „zlom řádek zde a zabraň stránce, aby se zde zlomila“. Příkaz „`\\[<dimen>]`“ znamená „zde zlom řádek a pak přidej vertikální mezeru o `<dimen>` řádcích“. Tak proč si tedy „`\\`“ plete při použití těchto parametrů začátek řádku? Čeká na první nebílý znak a v testu, který používá, ignoruje konec řádku ve vašem vstupním souboru. Řešením je uzavření tohoto textu na začátku řádku do složených závorek:

```
{\ttfamily
  /* C-language comment\\
  {[grump]} I don't like this format\\
  {*}/
}
```

Výše zmíněný text pochází z aktuálního příspěvku do konference `comp.text.tex`; tento případ může být také vyřešen pomocí prostředí `verbatim`.

Problém se objevuje také v matematickém módu, v polých atd. V tomhle případě není závorkování věcí ve velkém dobrý nápad; může být použité TeXovské primitivum `\relax` (které nedělá nic kromě blokování hledání tohoto druhu). Z jiného příkladu z `comp.text.tex`:

```
\begin{eqnarry}
  [a] &=& b \\
  \relax[a] &=& b
\end{eqnarry}
```

co je použití, které toto FAQ nedoporučuje: prostudujte si odpověď na Otázku 285.

249. Balík hlási „command already defined“

Nahráte pár balíkov a druhý z nich hlási, že jeden z príkazov, ktoré definuje, je už prítomný. Napríklad oba balíky *txfonts* a *amsmath* definujú príkaz `\iint` (a `\iiint` atď.); takže

```
...
\usepackage{txfonts}
\usepackage{amsmath}
```

vytvorí reťazec chybových hlásení formy:

```
! LaTeX Error: Command \iint already
   defined.  Or name \end... illegal,
   see p.192 of the manual.
```

Obecne platí, že veci, ktoré *amsmath* definuje, definuje dobre; existuje však dobrý prípad pre použitie *txfonts* verzie príkazu `\iint` — asociované `tx` fonty majú dvojitý symbol integrálu, ktorý nemusí byť robený spôsobom ako pri *amsmath*. V prípade, že nahrávate viaceré balíky symbolov, z ktorých každý definuje rovnaký symbol, pravdepodobne budete tento problém silno pociťovať (`\euro` je bežnou obeťou).

Existujú podobné prípady, keď jeden balík redefinuje príkaz iného balíku, ale neobjavia sa žiadne chyby, pretože redefinujúci balík nepoužíva `\newcommand`. V takomto prípade si zmenu často všimnete iba z dôvodu, že predpokladáte definíciu prvým balíkom. Balíky *amsmath–txfonts* sú práve takouto dvojicou; *txfonts* nevyvoláva chyby.

S problémom sa môžete vysporiadať uložením a obnovením príkazu. Macro programátori sa o toto môžu postarať sami; pre ostatných z nás existuje balík *savesym*. Postupnosť:

```
\usepackage{savesym}
\usepackage{amsmath}
\savesymbol{iint}
\usepackage{txfonts}
\restoresymbol{AMS}{iint}
```

spraví, čo chceme; zachovajúc *amsmath* verziu príkazu ako `\AMSiint` a sprístupňujúc *txfonts* verziu.

Dokumentácia *savesym* veľa neobsahuje: jedinými príkazmi sú `\savesymbol` a `\restoresymbol`, ktoré boli spomenuté vyššie.

`amsmath.sty`: Súčasť `macros/latex/required/amslatex`
`savesym.sty`: `macros/latex/contrib/savesym/savesym.sty`
`txfonts.sty`: Súčasť `fonts/txfonts`

250. Prečo nefunguje `\linespread`?

Príkaz `\linespread{factor}` má vynásobiť aktuálny `\baselineskip` číslom `<factor>`; podľa všetkého to však nerobí.

Príkaz je v skutočnosti ekvivalentný `\renewcommand{\baselinestretch }{factor}`: napísané týmto spôsobom nás už tak neprekvapí, že efekt nie je okamžitý. Činiteľ `\baselinestretch` je použitý, keď je vybraný font; iba zmena `\baselinestretch` nezmení font viac ako príkaz `\fontsize{size}{baselineskip}` — za každý z príkazov musíte doplniť `\selectfont`. Takže:

```
\fontsize{10}{12}%
\selectfont
```

alebo:

```
\linespread{1.2}%
\selectfont
```

Balíky ako *setspace*, ktorého prácou je správa účiaria, si samozrejme s týmto poradia — viď Otázku 146. Ak sa chcete vyhnúť balíku *setspace*, dajte si pozor na správanie zmien rozostupu riadkov v odstavci: prečítajte si Otázku 251.

`setspace.sty`: `macros/latex/contrib/setspace/setspace.sty`

251. Iba jeden `\baselineskip` na odstavec

`\baselineskip` nie je (ako by ste dúfali) vlastnosť riadku, ale odstavcu. Výsledkom je, že v `10pt` (nominálne) dokumente (so štandardnou `\baselineskip` hodnotou `12pt`) bude jediný väčší znak ako:

```
{\Huge A}
```

bude do odstavca natlačený: TeX sa postará o to, že si s horným riadkom nebudú vadiť, ale nedá mu „priestor na dýchanie", ako to robí pri texte štandardnej veľkosti; jeho veľkost (`24.88pt`) je teda zobraná do úvahy, ale jeho `\baselineskip` (`30pt`) nie je. Podobne

```
Text odstavca ...
{\footnotesize Rozšírená vsuvka ...
   ... do odstavca.}
      ... odstavec pokračuje ...
```

bude vyzerať zle, kedže `8pt` vsuvka skončí vysádzaná na `12pt` `\baselineskip` odstavca, namiesto preferovaných `8.5pt`.

Ako sa teda s týmito problémami vysporiadať? Sekcie (mierne) väčších rozmerov sú typicky opravené pomocou *podpery* (*strut*): toto slovo pochádza z klasickej typografie využívajúcej premiestňovateľné kovové časti a označuje oddeľovač boxov (obsahujúcich kovové znaky). Vždy, keď zmeníte veľkosť fontu, LaTeX redefinuje príkaz `\strut` aby poskytoval ekvivalent kovovej podpery zvolenej veľkosti. Takže pre horný príklad napíšeme

```
Text odstavca ...
   {\Huge A\strut}
   ... odstavec pokračuje ...
```

Rozšírené vloženia (či už väčšieho alebo menšieho textu) však budú vždy pôsobiť problémy; môžete použit podperu pre väčší text, používanie pre každý riadok ale bude únavné a neexistuje nič, čo by sme mohli nazvať „zápornou podperou", ktorá by riadky pritiahla k sebe pre menší text.

Jediným uspokojujúcim spôsobom na zaobchádzanie s rozšírením vložením pri rozličnej veľkosti je vysádzanie do oddeleného odstavca. Postačujúcou cestou, ako toto dosiahnuť, je prostredie `quote`, ktoré vysádza svoj text mierne odsadený voči zvyšku odstavca:

```
Text odstavca ...
\begin{quote}
  \footnotesize This is an inset account
  of something relevant to the enclosing
  paragraph...
\end{quote}
... odstavec pokračuje ...
```

252. Čo je s `\bf`, `\it`, ...?

Príkazy výberu fontu systému LaTeX 2.09 boli `\rm`, `\sf`, `\tt`, `\it`, `\sl`, `\em` a `\bf`; boli modálnymi príkazmi, takže ste ich mohli použiť nasledovne:

```
{\bf Fred} was {\it here\/}.
```

Zmena fontu bola vložená do skupiny, aby bol obmedzený jej efekt; všimnite si príkaz úpravy kurzívy `\/`, ktorý bol potrebný na konci sekcie v kurzíve.

Pri vydaní LaTeX 2_ε v lete 1994 boli tieto jednoduché príkazy zastaralé, ale keďže ich používanie je hlboko zaryté v mozgoch používateľov LaTeXu, príkazy samotné zostali v LaTeXu *s LaTeX 2.09 sémantikou*. Táto sémantika bola čiastočným dôvodom ich označenia za zastaralé: každý z `\`*`xx`* príkazov potlačuje akékoľvek iné nastavenie fontu, zachovávajúc iba veľkosť. Takže napríklad

```
{\bf\it Here we are again\/}
```

ignoruje `\bf` a vytvára text v kurzíve, strednej hrúbky (a korekcia kurzívy má skutočný efekt), zatiaľ čo

```
{\it\bf happy as can be\/}
```

ignoruje `\it` a vytvára rovný tučný text (a korekcia kurzívy nemá čo robiť). To isté platí, ak zmiešate LaTeX 2_ε výbery fontu s príkazmi strého štýlu:

```
\textbf{\tt all good friends}
```

ignoruje `\textbf`, ktorý obsahuje text a vytvára text písacieho stroja strednej hrúbky.

Prečo sú teda tieto príkazy zastaralé? — je to pre zmätok ako v poslednom príklade. Alternatívne (LaTeX 2_ε) príkazy sú diskutované v zbytku tejto odpovede.

Fontové príkazy systému LaTeX 2_ε sú v dvoch formách: modálne príkazy a príkazy textových blokov. Štandardná sada modálnych príkazov poskytuje hrúbky `\mdseries` a `\bfseries`, sklon `\upshape`, `\itshape`, `\scshape` a `\slshape` a rodiny `\rmfamily`, `\sffamily` a `\ttfamily`. Výber fontu požaduje rodinu, sklon a sériu (ako aj, samozrejme, veľkosť). Pár príkladov:

```
{\bfseries\ttfamily
    and jolly good company!}
```

vytvára tučný text písacieho stroja (všimnite si však neprítomnosť tučného strojopisného fontu (viď Otázku 199) v štandardných fontoch Computer Modern), alebo

```
{\slshape\sffamily
    Never mind the weather\/}
```

(všimnite si korekciu kurzívy potrebnú tiež pri sklonených fontoch).

Príkazy textových blokov systému LaTeX 2_ε vezmú prvé dve písmená modálnych príkazov a vytvoria z nich príkaz `\text`*`xx`*. Takto sa `\bfseries` stane `\textbf{text}`, `\itshape` sa stane `\textit{text}` a `\ttfamily` sa stane `\texttt{text}`. Blokové príkazy môžu byť vnárané ako:

```
\textit{\textbf{Never mind the rain}}
```

na vytvorenie tučného textu v kurzíve (zapamätajte si, že blokové príkazy poskytujú korekciu kurzívy tam, kde je to potrebné), môžu byť tiež vzájomne vnárané s LaTeX 2_ε modálnymi príkazmi:

```
\texttt{\bfseries
    So long as we're together}
```

pre tučný text písacieho stroja, alebo

```
{\slshape\textbf{%
    Whoops!  she goes again}\/}
```

pre tučnú naklonenú inštanciu aktuálnej rodiny (všimnite si znova aplikovanú korekciu kurzívy na konci skupiny modálneho príkazu).

Nové príkazy (ako je uvedené vyššie) potlačujú príkazy rovnakého typu. Toto takmer vo všetkých prípadoch vylučuje smiešne nápady ako „rovný sklonený“ font alebo „teletype roman“ fonty. Existuje však pár bezprostredných zvláštností. Prvou je konflikt medzi `\itshape` (alebo `\slshape`) a `\scshape`: kým mnohí tvrdia, že „small-caps“ font v kurzíve je typograficky nevhodné, takéto fonty existujú. Balík *smallcap* Daniela Taupina povoľuje použitie inštancií v EC fontoch (viď Otázku 47) a podobné techniky by mohli byť prenesené na iné sady fontov. Druhou zvláštnosťou je konflikt medzi `\upshape` a `\itshape`: Knuth vlastne poskytuje font rovnej kurzívy, ktorý LaTeX používa pre symbol „£“ v štandardnej sade fontov. Kombinácia je dostatočne zvláštna na to, že zatiaľ čo existuje definovaný sklon fontu, neexistuje žiaden štandardný príkaz LaTeXu; pre použitie tochto sklonu potrebuje (excentrický) používateľ najjednoduchšie LaTeXovské príkazy výberu fontu:

```
{\fontshape{ui}\selectfont
    Tra la la, di dee}
```

`smallcap.sty`: `macros/latex/contrib/smallcap`

R. Proč to dělá tohle?

253. Co se děje při použití příkazu `\include`?

Původní verze LaTeXu poskytovala příkaz `\include`, který pouze řešil problém dlouhých dokumentů: na relativně pomalých počítačích se používal příkaz `\includeonly`. S obrovským nárůstem rychlosti počítačů příkaz `\includeonly` ztratil na ceně (i když stále má svůj význam ve velmi rozsáhlých projektech). Proto je tento zjednodušený příkaz zahrnut i v současných verzích LaTeXu a plete lidi, kteří toto nepochopili.

V případě, že použijete `\includeonly`, tak `\include` vytváří zvláštní `.aux` soubory pro každý vložený soubor a vytváří checkpointy na důležité parametry (jako číslo stránky, obrázku, tabulky nebo poznámky pod čarou). Jako přímý důsledek musí vymazat stránku před i po příkazu `\include`. Tento mechanismus nepracuje, pokud se příkaz `\include` objeví v souboru, který byl sám vložen do jiného. Toto LaTeX ohlásí jako chybu. Nyní tedy můžeme odpovědět na dvě nejzákladnější otázky:

- Proč LaTeX vsouvá na začátek a konec stránky příkaz `\include`?
  Odpověď: Protože musí. Pokud se vám to nelíbí, nahraďte příkaz `\include` příkazem `\input` – nebudete moct víckrát použít příkaz `\includeonly`, ale pravděpodobně to stejně nebudete potřebovat, takže žádné obavy.
- Proč nemůžu vnořovat vložené soubory? Ve verzi LaTeXu 2.09 to vždy fungovalo.
  Odpověď: Ve skutečnosti to nefungovalo (ani ve verzi 2.09), chyba však nebyla diagnostikována. Když jste ale byl v té době spokojený s chováním pod LaTeXem verze 2.09, nahraďte příkaz `\include` příkazem `\input` (s přidaným příkazem `\clearpage`).

254. Proč jsou ignorovány parametry odstavce

TeX při uspořádávání textu nepracuje tak, že by bral slovo za slovem, či řádku za řádkou. Nejmenší jednotkou, kterou TeX formátuje, jsou celé odstavce. Odstavec je načten celý do paměti a není dále zpracováván, dokud není načtena značka konce odstavce. Právě v ten okamžik se uplatní parametry odstavce. Chyby použití parametrů formátování odstavce často vznikají právě proto, že se zapomíná na to, v jaké posloupnosti se odstavec zpracovává.

Předpokládejme následující větu v LaTeXu:

```
{\raggedright % zarovnávání  textu  vlevo
Tento text by měl být na výstupu zarovnán
pouze vlevo.  Chceme,  aby  se  tak stalo
pouze  v~tomto  odstavci,  a~ proto  zde
ukončíme skupinu.}

Další text je zpracován normálně...
```

TeX otevře skupinu a nastaví parametry formátování odstavce tak, aby byl text uvnitř této skupiny zarovnáván pouze vlevo, poté uloží dvě věty textu, uzavře skupinu a nastaví původní parametry odstavce. Poté načte prázdný řádek, který je zpracován stejně jako příkaz `\par`; vysází dvě věty textu. Jelikož však skupina byla ukončena před načtením konce odstavce, nastavení parametrů uvnitř skupiny ztratilo význam a odstavec bude vysázen s běžnými parametry.

Aby nastavené parametry zůstaly v platnosti po celou dobu zpracovávání odstavce, je třeba ukončit odstavec uvnitř skupiny. Nahradíme-li poslední tři řádky v předchozím příkladě za:

```
ukončíme skupinu.\par}
Další text je zpracován normálně...
```

ukončí se odstavec ve chvíli, kdy jsou parametry odstavce nastavené v uzavřené skupině stále v platnosti.

Další alternativa spočívá v tom, že si můžete definovat nové prostředí, které udělá příslušnou práci za vás. Pro příklad, který byl zmiňován výše, je již v LaTeXu definováno takovéto prostředí:

```
\begin{flushleft}
  Tento text by měl být...
\end{flushleft}
```

255. Proč se v LaTeXu užívá ochran (protection)

LaTeX si ukládá některá data, jež bude zpracovávat až později. Těmito daty jsou zejména argumenty některých příkazů, takzvané pohyblivé argumenty. Pohyblivé, protože se s daty nějakým způsobem manipuluje. Jedná se o argumenty těch příkazů, které zapisují do obsahu, seznamu tabulek atd., tj. data, která jsou zapisována do pomocného souboru, z něhož jsou později opět čtena. Jinými daty jsou ta, která se mohou objevit v záhlavích. Nejvýznamnějšími příkazy tohoto typu jsou popisy obrázků a tabulek (captions) a veškeré nadpisy. Úplný seznam lze najít v Lamportově manuálu (viz Otázku 23).

Co se za tím vším skutečně skrývá? Příkazy, jež se použijí v pohyblivých argumentech, jsou v průběhu ukládání plně expandovány. Někdy je výsledkem takového rozvoje špatný TeXovský kód, což se projeví až při jeho následném čtení. Příkazem `\protect\cmd` je LaTeXu řečeno, aby uložil `\cmd` bez expanze, jako `\cmd`.

Co je to „křehký příkaz“ (fragile command)? To je příkaz, který je během ukládání rozvinut do chybného TeXovského kódu.

Co je to „robustní příkaz“ (robust command)? To je příkaz, který je během ukládání rozvinut do správného TeXovského kódu.

Nikdo se (samozřejmě) z takto nepřehledné situace neraduje. Skupina projektu LaTeX3 při práci na LaTeXu 2$_\varepsilon$ odstranila potřebu některých ochran, avšak techniky, které jsou jim dostupné v současném LaTeXu, činí věc poměrně složitou. Dlouhodobým cílem této skupiny zůstává odstranění všech ochran.

256. Proč `\verb` nefunguje uvnitř ...

Funkce příkazu LaTeXu pro sazbu textu v tom tvaru, v němž je uveden ve zdrojovém souboru (`verbatim`), je založena na využití změny kategorie (category codes) jednotlivých znaků. Knuth v této souvislosti říká: „Je potřeba věnovat jistou péči tomu, aby vše proběhlo ve správném sledu…". Kategorie znaku se od okamžiku, kdy je mu přiřazena, nemění. Proto `\verb` předpokládá, že je prvním příkazem, který se dívá na svůj textový parametr. Není-li tomu tak, TEX již přiřadil kategorie jednotlivým znakům a `\verb` již nemá šanci kategorii měnit. Například:

```
\verb+\error+
```

bude fungovat (vysází se „`\error`"), ale

```
\newcommand{\unbrace}[1]{#1}
\unbrace{\verb+\error+}
```

fungovat nebude (pokusí se spustit příkaz `\error`). Další chyby, se kterými se můžete setkat jsou „`\verb` ended by end of line" nebo „`\verb` illegal in command argument".

Proto se také v manuálu LATEXu tolik naléhá na to, aby se příkaz verbatim neobjevil v argumentu žádného jiného příkazu. Tyto příkazy jsou nejen křehké (fragile), ale dokonce zcela nepoužitelné jako parametr jiných příkazů, bez ohledu na ochranu pomocí `\protect` (viz Otázku 255).

Jako první by jste si měli položit otázku, jestli je `\verb` skutečně nezbytné.

- Když `\texttt{\emph {váš text}}` produkuje stejné výsledky jako `\verb+`*`váš text`*`+`, pak `\verb` není zapotřebí.
- Když používáte `\verb` k vysázení URL, emailové adresy nebo něčeho podobného, pomůže příkaz `\url` z balíku *url* (`macros/latex/contrib/misc/url.sty`): netrpí problémy `\verb`.
- Když vkládáte `\verb` do parametru boxovacího příkazu (jako je `\fbox`), zvažte použití prostředí `lrbox`:

  ```
  \newsavebox{\mybox}
  ...
  \begin{lrbox}{\mybox}
    \verb!VerbatimStuff!
  \end{lrbox}
  \fbox{\usebox{\mybox}}
  ```

Jinak jsou zde tři částečná řešení tohoto problému:

- Balík *fancyvrb* (`macros/latex/contrib/fancyvrb`) definuje příkaz `\VerbatimFootnotes`, který předefinovává příkaz `\footnotetext` (a odtud i `\footnote`) tak, že můžete vkládat příkaz `\verb` jako jeho argument. Tento přístup by mohl být v podstatě rozšířen i na argumenty ostatních příkazů, ale, bohužel, může kolidovat s jinými balíky: například `\VerbatimFotenotes` ovlivňuje volbu `para` z balíku `footmisc`.
- Balík `fancyvrb` definuje příkaz `\SaveVerb` s odpovídajícím příkazem `\UseVerb`, který vám umožňuje uložit a pak znovu použít obsah jeho argumentů. Detaily týkající se použití tohoto extrémně silného násroje hledejte v dokumentaci k tomuto balíku.
  Poněkud jednodušší je balík `verbdef` (`macros/latex/contrib/misc/verbdef.sty`), který definuje robustní příkaz, který expanduje do zadaného argumentu.
- Pokud vám dělá problémy jediný znak (v jeho nepřítomnosti by jste klidně mohly použít `\texttt`), zvažte použití `\string`. `\texttt{my\string_name}` vysází to samé jako `\verb+my_name+` a bude fungovat v parametru příkazu. Nebude ale fungovat v posuvném parametru, žádná dávka ochran (`\protect` — viz Otázku 255) jej k tomu v tomhle případě nedonutí.

257. Žiaden riadok na ukončenie

Chyba

```
! LaTeX Error: There's no line here to end.

See the LaTeX manual or LaTeX Companion
...                             for explanation.
```

je reakciou LATEXu na príkaz `\\` tam, kde to nečaká. Najčastejším prípadom je, keď chcete mať návestie položky zoznamu na samostatnom riadku:

```
\begin{description}
\item[Veľmi dlhý popisok] \\
  Text...
```

```
\end{description}
```

\\ je vlastne v tomto prípade dosť zlý príkaz (aj keby fungoval), pretože by donútil odstavec tvorený textom položky, aby ukončil riadok, na ktorom nie je nič okrem návestia. Toto by viedlo k varovnej hláške „`Underfull \hbox`“ (obvykle s „nekonečnou“ nevhodnosťou hodnoty 10000); táto správa síce okrem spomalenia behu LaTeXu neškodí, ale každá správa, ktorá neprináša žiadnu informáciu, zbytočne odvracia pozornosť uzívateľa.

Správnym riešením je napísanie nového druhu prostredia `description`, ktoré robí to, čo chcete. (The *LaTeX Companion* — viď Otázku 23 — poskytuje poskytuje v týchto veciach celkom široký výber.)

Jednoduchým a rýchlym riešením, ktoré sa vyhne varovaniam, je napísať:

```
\begin{description}
\item[Veľmi dlhý popisok] \hspace*{\fill} \\
  Text...
\end{description}
```

čo vyplní podtečený (under-full) riadok predtým, než si vynúti jeho uzavretie. Balík *expdlist* poskytuje rovnakú funkčnosť svojím príkazom `\breaklabel`, *mdwlist* ju poskytuje príkazom `\desclabelstyle`.

Ďalším častým prípadom správy je používanie prostredia `center` (alebo `flushleft`, prípadne `flushright`) a rozhodnutie, že potrebujete zvlášť oddeliť riadky prostredia:

```
\begin{center}
  Prvý riadok (nadpisu)\\
  \\
  telo centrovaného textu...
\end{center}
```

Riešenie je prosté: použite príkaz \\ spôsobom, akým sa používať má, aby poskytol viac priestoru než jeden riadkový zlom. \\ akceptuje voliteľný parameter špecifikujúci, koľko priestoru naviac pridať; požadovaný efekt v hore uvedenom príklade docielite takto:

```
\begin{center}
  Prvý riadok (nadpis)\\[\baselineskip]
  telo centrovaného textu...
\end{center}
```

`expdlist.sty`: `macros/latex/contrib/expdlist`
`mdwlist.sty`: distribuované ako súčasť `macros/latex/contrib/mdwtools`

258. Chyby související se změnou velikosti písmen

TeX poskytuje dva primitivní příkazy umožňující konverzi textu do malých písmen, `\lowercase`, a velkých písmen, `\uppercase`. Tyto příkazy se nepoužívají příliš často, ale dokáží nadělat v dokumentu zmatek.

Oba příkazy nerozvíjejí text, který je jejich parametrem — výsledkem příkazu `\uppercase{abc}` je „`ABC`“, avšak výsledek příkazu `\uppercase{\abc}` je `\abc` ať už `\abc` znamená cokoli. Příkazy jednoduše interpretují tabulku ekvivalentů mezi velkými a malými formami znaků. Nemají například smysl pro matematiku, a tak

```
\uppercase{About $y=f(x)$}
```

dává

```
ABOUT $Y=F(X)$
```

což zřejmě není to, co jsme původně zamýšleli.

`\uppercase` a `\lowercase` navíc špatně pracují s neamerickými znaky, např. `\uppercase{\csx {ae}}` je to samé jako `\ae`.

LaTeX poskytuje příkazy `\MakeUppercase` a `\MakeLowercase`, které opravují posledně zmíněný problém. Tyto příkazy jsou používány ve standardních třídach k vytvoření verzálkových hlaviček kapitol a sekcí.

`\MakeUppercase` a `\MakeLowercase` bohužel neřeší další problémy `\uppercase`, takže například titulek sekce obsahující `\begin{tabular}` … `\end{tabular}` vytvoří hlavičku obsahující `\begin{TABULAR}`, což je však prostředí, které vůbec neexistuje. Nejjednodušším řešením je použití uživatelem definovaného, např.:

```
\newcommand{\mytable}{\begin{tabular}...
  \end{tabular}}
\section{Titulek sekce \protect\mytable{}
  s~tabulkou}
```

Všimněte si, že `\mytable` musí být chráněno, jinak bude expandováno a přepsáno do verzálek; stejný výsledek dosáhnete deklarováním s `\DeclareRobustCommand` (`\protect` již nebude potřeba).

Balík *textcase* Davida Carlisla adresuje množství z těchto problémů transparentním způsobem. Definuje příkazy `\MakeTextUppercase` a `\MakeTextLowercase`, které mění velikost písmen, se skvělými vlastnostmi standardních LaTeXovských `\Make*` příkazů, ale bez uvedených problémů. Balík nahrajete pomocí `\usepackage[overload]{textcase}` a on redefinuje LaTeXovské příkazy (*ne* TeXovské primitívní příkazy! `\uppercase` a `\lowercase`) tak, aby nadpisy sekcí a podobné věci neprodukovaly špatné nadpisy stránek.
`textcase.sty`: distribuováno jako součást `macros/latex/contrib/carlisle`

259. Proč je znak # v makrech uveden dvakrát

Při psaní makra je třeba mít na paměti, že `##` zastupuje `#` obdobně jako `#1` zastupuje cokoli, co je prvním argumentem makra.

Definujeme-li a poté použijeme makro:

```
\def\a#1{+++#1+++#1+++#1+++}  \a{b}
```

makro po expanzi dává „+++b+++b+++b+++", což jsme očekávali. Avšak změníme-li nyní část makra:

```
\def\a#1{+++#1+++\def\x #1{xxx#1}}
```

`\a{b}` se rozvine na „+++b+++`\def\x b{xxxb}`". Definuje se zde makro `\x`, které je *ohraničené* znakem `b` a které nemá žádný argument. Může se to zdát divné, ačkoli jde jen o zvláštní případ předchozího příkladu. Chceme-li, aby v `\a` bylo definováno makro `\x` s jedním argumentem, musíme upravit předchozí makro takto:

```
\def\a#1{+++#1+++\def\x ##1{xxx##1}}
```

`\a{b}` se nyní rozvine do tvaru „+++b+++`\def\x #1{xxx#1}`", neboť `#1` bude nahrazeno znakem „b" a `##` bude nahrazeno `#`.

Pro definici vnořenou uvnitř definice se použije `####1`, jelikož v každé úrovni je `##` nahrazeno `#`. V další úrovni je nutné použít osm `#`, atd.

260. Medzery v makrách

Je jednoduché napísať makrá produkujúce medzeru v sádzanom výstupe tam, kde to nie je želané ani očakávané. Medzery vytvorené makrami sú zradné, pretože sa nezdružujú s medzerami okolo makra (na rozdiel od následných medzier, ktoré píšete), takže váš výstup môže mať jednu nafúknutú medzeru zloženú z jednej alebo viacerých medzier, ktoré neboli združené. Výstup môže mať medzeru aj tam, kde ju vôbec nikto nečaká.

Medzery vznikajú (vo vnútri makra ako aj inde), znakmi medzery, tabulátoru alebo konca riadku. Pri písaní makier by ste si mali pamätať dve jednoduché pravidlá: pravidlá ignorovania prázdneho miesta (medzier) pri písaní makier sú rovnaké ako pri písaní obyčajného textu a pravidlá ignorovania medzier *neplatia* na medzery vzniknuté počas expanzie makra.

Medzery sú ignorované vo vertikálnom móde (medzi odstavcami), na začiatku riadku a po názve príkazu. Kedže postupnosti medzier splývajú do jednej medzery, vyzerá to, že medzery sú ignorované aj vtedy, ak nasledujú inú medzeru. Medzery môžu mať po niektorých druhoch nezátvorkovaných parametrov (napr. priradenie premenným *count* a *dimen* v Plain TeXu) a po určitých kontrolných slovách (napr. v `\hbox to`) syntaktický význam, takže znova máme prípady, kde to „vyzerá", ako by medzery boli ignorované keď pracujú potichu samy pre seba.

Všimnite si nasledujúce makro celkom verne adaptované z makra, ktoré sa objavilo na `comp.text.tex`:

```
\newcommand{\stline}[1]
  { \bigskip \makebox[2cm]{ \textbf{#1} } }
```

(originál bol na jeden riadok).
Definícia makra obsahuje päť medzier:

- po symbole `{` otvárajúcom telo makra; táto medzera bude ignorovaná nie z dôvodu, že sa makro objavuje na začiatku riadku, ale pretože makro bolo navrhnuté na operovanie medzi odstavcami
- za `\bigskip`; táto medzera bude ignorovaná (počas definície makra), pretože nasleduje názov príkazu
- za symbolom `{` povinného parametru `\makebox`; hoci sa táto medzera nevyhnutne objaví na začaitku výstupného riadku, *nebude* ignorovaná
- za symbolom `}` uzatvárajúcim parameter `\textbf`; táto medzera nebude ignorovaná, ale môže byť prehliadnutá, ak je parameter do povolených `2cm`
- za symbolom `}` uzatvárajúcim povinný parameter `\makebox`; táto medzera nebude ignorovaná

Pôvodný autor makra sa obával, že začiatky riadkov s týmto makrom nebudú pri ľavom okraji a text za makrom nebude vždy správne zarovnaný. Tieto problémy vznikli z medzery na začaitku povinného parametru `\makebox` a medzery bezprostredne za tým istým parametrom. Autor napísal svoje makro týmto spôsobom, na zdôraznenie významu rozličných častí. Význam sa bohužiaľ srtatil v problémoch spôsobených makrom.

Hlavná technika potlačenia medzier je použitie znakov `%`: všetko za `%` je ignorované, dokonca aj samotný koniec riadku (takže ani koniec riadku neprispeje nechcenou medzerou). Druhou technikou je zabezpečenie toho, že koncu riadku predchádza názov príkazu (kedže sa koniec riadku správa ako medzra, bude po názve príkazu ignorovaný). Hore uvedený príkaz by sme teda napísali (ako skúsený programátor s rovnakým dôrazom na štruktúru):

```
\newcommand{\stline}[1]{%
  \bigskip
  \makebox[2cm]{%
    \textbf{#1}\relax
  }%
}
```

Uistili sme sa, že každá medzera v upravenej definícii je ignorovaná, takže sa žiadna neobjaví vo výstupe. Upravená definícia sa explicitne vysporiada s každým koncom riadku (hoci, ako sme uviedli vyššie, medzera na konci prvého riadku makra by bola pri použití makra ignorovaná). Toto je v skutočnosti najlepšia technika — je jednoduchšie slepo potlačiť medzery, než v každom bode analyzovať, či to vlastne potrebujete. Na potlačenie medzier boli použité tri techniky:

- umiestnenie znaku `%` na koniec riadku (na riadkoch 1, 3 a 5),
- „prirodzené“ ukončenie riadku riadiacou sekvenciou (riadok 2) a
- ukončenie riadku „umelou“ riadiacou sekvenciou (riadok 4); riadiaca sekvencia v tomto prípade (`\relax`) nemá za mnohých okolností žiadny účinok (aku tu), ale toto použitie je zastaralé — znak `%` by bol lepší.

Dávajte si pozor na (obvyklé) pokušenie umiestniť medzeru *pred* znak `%`: ak to spravíte, môžete úplne vynechať aj `%`.

V „skutočnom živote“ sú medzery objavujúce sa v makrách samozrejme záhadnejšie ako v našom príklade. najbežnejšie medzery vznikajú z nechránených koncov riadkov a toto je chyba, ktorá sa občas vyskytne aj v makrách písaných skúsenými programátormi.

261. Definícia aktívnych znakov

Jednotlivé znaky sa môžu správať ako makrá (definované príkazy) a Plain TEX aj LATEX definujú znak „~“ ako nezlomiteľnú medzeru. Znak sa spraví definovateľným, alebo „aktívnym“, nastavením jeho *kódu kategórie* (catcode) na `\active` (13): `\catcode‘_=\active`.

Ľubovolný znak v princípe môže byť týmto spôsobom aktivovaný a definovaný ako makro (`\def_{\_}` – odpoveď na Otázku 201), ale musíte sa mať na pozore: kým ľudia čakajú aktívnu vlnku, iné aktívne znaky môžu byť nečakané a môžu zle interagovať s inými makrami. Definovaním aktívneho znaku naviac znemožníte použitie znaku na iné účely a existuje iba málo voľných znakov, ktoré by mohli byť týmto spôsobom zmenené.

Pre definovanie znaku „`z`“ ako príkazu je potrebné napísať niečo ako:

```
\catcode`\z=\active
\def z{Ahoj}%
```

a každé následné „`z`“ v texte sa stane pozdravom. Toto by bolo pre väčšinu dokumentov zlým nápadom, ale mohlo by to mať špeciálne využitia. (V `\def z`, už „`z`“ nie je interpretované ako písmeno; medzera teda nie je potrebná — `\defz` postačuje; rozhodli sme sa ponechať medzeru aspoň pre malé zprehľadnenie.) Niektoré balíky LaTeX obsahujú takéto definície. Napríklad balík *shortvrb* a jeho príkaz `\MakeShortVerb`.

TeX Používa kódy kategórií na interpretáciu znakov pri tom, ako sú čítané zo vstupu. *Zmena hodnoty catcode neovplyvní znaky, čo boli už prečítané.* Z tochto dôvodu je vhodné, ak majú znaky pevný kód kategórie v celom dokumente. Ak je hodnota catcode zmenená za určitým účelom (toto robí príkaz `\verb`), potom upravené znaky nebudú pri výskyte v parametri iného príkazu správne interpretované (ako napríklad v Otázke 256). Exemplárny prípad je balík *doc*, ktorý spracúva .dtx súbory používajúc balík *shortvrb* pre definíciu `|...|` ako skratky pre `\verb|...|`. `|` je ale používané tiež v preambuliach tabuľkových prostredí, takže tabuľky v `.dtx` súboroch môžu mať vertikálne oddelenie riadkov medzi stĺpcami iba pomocou určitých špeciálnych opatrení.

Ďalším dôsledkom je, že priradenia catcode uskutočnené v makrách často nefungujú, ako by sa čakalo (viď Otázku 262). Napríklad definícia

```
\def\mistake{%
\catcode`_=\active
\def_{\textunderscore\-}%
}
```

nefunguje, pretože sa pokúša definovať bežný znak `_`: Keď je makro použité, zmena kategórie neplatí pre znak podčiarknutia v definícii makra. Namiesto toho môžete použiť:

```
\begingroup
\catcode`_=\active
\gdef\works{%   všimnite si globálne \gdef
  \catcode`_=\active
  \def_{\textunderscore\-}%
}
\endgroup
```

Alternatívny spôsob vytvorenia takejto izolovanej definície záleží na zvláštnych vlastnostiach `\lowercase`, ktorý zmení znaky bez zmeny ich hodnoty catcode. Kedže vždy existuje *jeden* aktívny znak („`~`“), môžeme oklamať `\lowercase`, aby zaplátalo definíciu bez toho, aby sme explicitne zmenili catcode:

```
\begingroup
  \lccode`\~=`\_
  \lowercase{\endgroup
    \def~{\textunderscore\-}%
  }%
```

Tieto dve definície majú rovnaký celkový efekt (znak je definovaný ako príkaz, ale znak nezostáva aktívny) až na to, že prvá definuje príkaz `\global`.

Pre použitie aktívnych znakov v matematickom móde je lepšie nechať znaku bežný catcode a prideliť mu špeciálny aktívny *matematický kód*:

```
\begingroup
  \lccode`~=`x
  \lowercase{\endgroup
    \def~{\times}%
  }%
\mathcode`x="8000
```

Špeciálny znak nemusí byť predefinovaný kedykoľvek sa stane aktívnym — definícia príkazu pretrvá aj po návratu catcode na pôvodnú hodnotu; definícia sa stane znova dostupnou, keď sa znak opäť stane aktívnym.

`doc.sty`: distribuované ako súčasť zdrojových súborov `macros/latex/base`
`shortvrb.sty`: distribuované ako súčasť `macros/latex/required/tools`

262. Aktívne znaky v parametroch príkazov

Občas je celkom príjemné spraviť jeden alebo dva znaky aktívnymi v parametri príkazu, aby sme uľahčili autorom písanie parametrov.

Aktívne znaky *môžu* byť v takýchto situáciach bezpečne použité, je však treba opatrnosti.

Pri zvažovaní tejto odpovede vyvstal príklad: ašpirujúci programátor makier na `comp.text.tex` žiadal o pomoc pri snahe donútiť `#` a `b` produkovať hudobné symboly krížiku a béčka v makre pre špecifikáciu akordov.

Prvým problémom je, že `#` aj `b` majú dôležité využitie na inom mieste v TEXu (mierne povedané!), takže znaky sa môžu stať aktívnymi iba počas vykonávania príkazu.

Použitím techník diskutovaných v Otázke 261 môžeme definovať:

```
\begingroup
  \catcode`\#=\active
  \gdef#{$\sharp$}
\endgroup
```

a:

```
\begingroup
  \lccode`\~=`\b
  \lowercase{\endgroup
    \def~{$\flat$}%
  }
```

Druhým problémom je časovanie: príkaz musí spraviť každý zo znakov aktívnym *predtým*, než sú načítané jeho parametre. Toto znamená, že príkaz samotný nemôže mať parametre, ale musí byť rozdelený na dve časti. Píšeme teda:

```
\def\chord{%
  \begingroup
    \catcode`\#=\active
    \catcode`\b=\active
    \Xchord
}
\def\Xchord#1{%
    \chordfont#1%
  \endgroup
}
```

a príkaz môžeme použiť ako `\chord{F\#}` alebo `\chord{Bb minor}`.

Dôležité sú dve vlastnosti:

- `\begingroup` v `\chord` otvára skupinu, ktorá je uzavretá pomocou `\endgroup` v `\Xchord`; táto skupina obmedzuje zmenu kódov kategórie, čo je dôvodom existencie celého cvičenia.
- Hoci `#` je aktívny počas vykonávania `\Xchord`, *nie je* aktívny, keď je práve definovaný, takže použitie `#1` nepotrebuje špeciálnu pozornosť.

Zapamätajte si, že technika použitá v makrách ako `\chord` je analogická technike použitej v príkazoch ako `\verb`; a rovnako ako `\verb` (viď Otázku 256), `\chord` nebude fungovať vo vnútri parametru iného príkazu (chybové správy, ak sa vôbec objavia, budú pravdepodobne dosť zvláštne).

263. Definícia makra z parametru

Je bežné chcieť príkaz na vytvorenie iného príkazu, často je požadované, aby bol názov nového príkazu odvodený z parametru. LATEX to robí celý čas, napríklad `\newenvironment` vytvára start- a end-environment príkazy, ktorých názvy sú odvodené z názvu environment príkazu.

Zjavný prístup:

```
\def\relay#1#2{\def\#1{#2}}
```

nefunguje (TEX ho interpretuje ako divnú redefiníciu `\#`). Trik spočíva v použití `\csname`, čo je TEXovské primitívum na generovanie názvov príkazov z náhodného textu, spolu s `\expandafter`. Uvedená definícia by mala vyzerať:

```
\def\relay#1#2{%
```

```
      \expandafter\def\csname #1\endcsname{#2}%
    }
```
S touto definíciou je `\relay{blah}{bleah}` ekvivalentné `\def\blah{bleah}`.

Všimnite si, že definícia `\relay` vynecháva zátvorky okolo názvu príkazu v spúšťanom `\newcommand`. Toto je z dôvodu, že nie sú potrebné (to sú v skutočnosti zriedvo) a za týchto okolností robia kód makra mirne nudnejším.

Vytvorený názov nemusí byť (samozrejme) *iba* parameter:
```
\def\newrace#1#2#3{\expandafter\def
    \csname start#1\endcsname{%
      #2%
  }%
  \expandafter\def
    \csname finish#1\endcsname{%
      #3%
  }%
}
```
S príkazmi
```
\def\start#1{\csname start#1\endcsname}
\def\finish#1{\csname finish#1\endcsname}
```
by sa tieto mohli správať trochu ako LaTeXovské prostredia (environments).

264. Zistenie, že je niečo prázdne

Predpokladajme, že potrebujete vediet, či je parameter vášho príkazu prázdny, teda rozoznať medzi `\cmd{}` a `\cmd{blah}`. Toto je celkom jednoduché:
```
\def\cmd#1{%
  \def\tempa{}%
  \def\tempb{#1}%
  \ifx\tempa\tempb
    <prázdny>
  \else
    <neprázdny>
  \fi
}
```
Prípad, keď chcete ignorovať parameter pozostávajúci iba z medzier (nemusí byť teda úplne prázdny), je komplikovanejší. Tento prípad je vyriešený vo fragmente kódu *ifmtarg* definujúcom príkazy `\@ifmtarg` a `\@ifnotmtarg`, ktoré rozoznajú (v opačných smeroch) medzi druhým a tretím parametrom. Kód balíku sa objavuje aj v LaTeXovskej triede `memoir`.

Ifmtarg tvorí vyzývavé čítanie, existuje aj diskusia tochto problému v čísle dva série článkov „Around the bend" Mika Downesa.

`Around the bend`: séria článkov — `info/aro-bend`
`ifmtarg.sty`: `macros/latex/contrib/misc/ifmtarg.sty`
`memoir.cls`: `macros/latex/contrib/memoir`

265. Číslovanie rovníc, obrázkov a tabuliek

Mnoho LaTeXovských tried (vrátane štandardnej triedy `book`) čísluje veci po kapitolách, takže obrázky v kapitole 1 sú číslované 1.1, 1.2 a tak ďalej. Toto niekedy nie je pre požiadavky používateľa vhodné.

Namiesto prepísanie celej triedy je možné použiť balíky *removefr* a *remreset*. Oba definujú príkaz `\@removefromreset`, po vložení balíku teda môžete napísať niečo ako:
```
\makeatletter
\@removefromreset{figure}{chapter}
```
a automatické prečíslovávanie sa zastaví. Potom musíte znovu definovať spôsob, akým sú čísla obrázkov (v tomto prípade) tlačené:

```
  \renewcommand{\thefigure}{\@arabic\c@figure}
  \makeatother
```

(nezabudnite pre každé počítadlo, s ktorým chcete manipulovať, uskutočniť všetky kroky uvedené v `\makeatletter` ... `\makeatother`).

Technika môže byť použitá aj na zmenu pri resetovaní počítadla vo viacúrovňovej štruktúre. Predpokladajme, že vaša trieda čísluje stránky ako *<kapitola>.<sekcia>.<obrázok>* a vy ich chcete číslovať po kapitolách. Skúste:

```
    \@removefromreset{figure}{section}
    \@addtoreset{figure}{chapter}
    \renewcommand{\thefigure}%
                 {\thechapter.\@arabic\c@figure}
```

(príkaz `\@addtoreset` je súčasť LaTeXu samotného).

Balík *chngcntr* poskytuje jednoduché prostriedky sprístupnenia dvoch diskutovaných druhov zmien definujúc príkazy `\counterwithin` a `\counterwithout`. Trieda `memoir` tieto funkcie poskytuje tiež.

`chngcntr.sty`: `macros/latex/contrib/misc/chngcntr.sty`
`memoir.cls`: `macros/latex/contrib/memoir`
`removefr.tex`: `macros/latex/contrib/fragments/removefr.tex` (toto je konštruované ako „fragment" pre použitie v iných balíkoch: nahrajte pomocou `\input{removefr}`)
`remreset.sty`: distribuované ako súčasť `macros/latex/contrib/carlisle`

266. `\edef` nefunguje s `\protect`

Robustné príkazy LaTeXu sú buď „prirodzene robustné" — nikdy nepotrebujú `\protect`, alebo „self-protected" (chránia sa samy) — majú `\protect` nejakým spôsobom zabudované do svojej definície. „Self-protected" príkazy sú robustné iba v kontexte, kde je mechanizmus `\protect` správne spracovaný. Kedže `\edef` je TeXovské primitívum a nie príkaz LaTeXu, telo definície `\edef` nespracúva `\protect` správne.

Tento problém je riešený interným príkazom LaTeXu `\protected@edef`, ktorý robí prácu `\edef`, zatiaľ čo udržuje mechanizmus `\protect` funkčný. Existuje zodpovedajúci príkaz `\protected@xdef`, ktorý robí to, čo `\xdef`.

Kedže sú tieto príkazy interné, musia byť samozrejme pozorne sledované (viď Otázku 243).

267. Voliteľné parametre ako `\section`

Voliteľné parametre v makrách definovaných pomocou `\newcommand` nepracujú úplne rovnakým spôsobom ako voliteľný parameter `\section`. Štandardná hodnota voliteľného parametru `\section` je hodnota povinného parametru, ale `\newcommand` vyžaduje, aby ste štandardné hodnoty vedeli dopredu.

Potrebný trik je použitie makra v štandardnom parametri:

```
\newcommand\thing[2][\DefaultOpt]{%
  \def\DefaultOpt{#2}%
  ...%
}
```

268. Tvorenie návestí z počítadiel

Predpokladajme, že máme LaTeXovské počítadlo, ktoré sme definovali pomocou `\newcounter{foo}`. Hodnotu počítadla môžeme zvýšiť pomocou `\addtocounter{foo}{1}`, ale je to ťažkopádne na operáciu, ktorá nastáva tak často, takže existuje príkaz `\stepcounter{foo}`, ktorý vykonáva tento špeciálny prípad zväčšenia o jednotku.

Existuje interná premenná LaTeXu „aktuálne návestie", ktorá si pamätá poslednú vec s možnosťou návestia, ktorú LaTeX spracoval. Mohli by ste (ak by ste na tom trvali) tú hodnotu nastaviť relevantným príkazom TeXu (zabezpečiac potrebné opatrenia, aby interný príkaz fungoval) — ale nie je to potrebné. Ak namiesto hore uvedených metód použijete `\refstepcounter{foo}`, interná premenná bude nastavená na novú hodnotu a (kým nepríde niečo iné) `\label` bude odkazovať na počítadlo.

269. Zistenie, či ste na párnej alebo nepárnej strane

Otázka 279 diskutuje problém, ako donútiť príkazy `\marginpar`, aby svoj výstup umiestnili na správny okraj dvojstranových dokumentov. Toto je obecný problém znalosti, kde leží určitý kus textu: výstupná rutina je asynchrónna a TeX/LaTeX obvykle spracuje dobrý kus ďalšej strany, než sa vôbec rozhodne akúkoľvek stranu vyprodukovať. Dôsledkom je, že počítadlo `page` (interne známe v LaTeXu ako `\c@page`) je obvykle spoľahlivé, iba ak ste vo výstupnej rutine.

Riešením je použitie nejakej verzie `\label` mechanizmu na zistenie, na ktorom konci stránky ste. Hodnota počítadla strán objavujúca sa v príkaze `\pageref` bola vložená do behu výstupnej rutiny a je teda bezpečná.

`\pageref` samotný však bezpečný nie je: môžete dúfať, že

```
\ifthenelse{\isodd{\pageref{foo}}}{odd}{even}
```

spraví potrebné, ale o balíkoch *babel* aj *hyperref* sa vie, že zasahujú do výstupu `\pageref`; buďte opatrní!

Balík *chngpage* musí túto funkcionalitu poskytovať pre vlastné využitie, a preto poskytuje príkaz `\checkoddpage`. Tento nastavuje návestie na privátne použitie a časť návestia s odkazom na stranu je potom preskúmaná (*hyperref* bezpečným spôsobom) na nastavenie platnosti `\ifcpoddpage`, ak bol príkaz zavolaný na nepárnej strane. Trieda `memoir` obsahuje rovnaký príkaz nastavujúci platnosť `\ifoddpage`. `\label` samozrejme prispieva k LaTeXovským chybovým hláškam „Rerun to get cross-references right“ ...

`chngpage.sty`: `macros/latex/contrib/misc/chngpage.sty`
`memoir.cls`: `macros/latex/contrib/memoir`

270. Ako zmeniť formát návestí

Štandardne pri vytvorení návestia toto preberie vzhľad označeného počítadla: konkrétne je nastavené na `\the<counter>` — čo by bolo použité, ak by ste chceli vysádzať počítadlo do vášho textu. Toto nie je vždy to, čo potrebujete: ak napríklad máte vnorené číslované zoznamy s vonkajškom číslovaným a vnútrom označeným písmenami, očakávalo by sa, že sa chcete na položky vnútorného zoznamu odkazovať ako „2(c)“. (Zapamätajte si, že môžete zmeniť štruktúru položiek zoznamov — viď Otázku 157.) Zmena je samozrejme možná explicitným označením rodiča a použitím tohto návestia na skonštruovanie vysádzaného výsledku — niečo ako

```
\ref{parent-item}(\ref{child-item})
```

by bolo nudné a náchylné k chybám. Bolo by naviac nevhodné, kedže by ste konštruovali vizuálnu reprezentáciu, ktorá je neflexibilná (nemohli by ste vôbec zmeniť všetky odkazy na prvky zoznamu jedným ťahom).

LaTeX má v skutočnosti príkaz formátujúci návestia zabudovaný do každej definície návestia; štandardne je neplatný, ale je užívateľovi dostupný na programovanie. Pre ľubovolné návestie *<counter>* existuje interný príkaz LaTeXu `\p@<counter>`; napríklad definícia návestia na vnútornej položke zoznamu je pravdepodobne uskutočnená príkazom `\p@enumii{\csx {theenumii}}`. Interné fungovanie tu bohužiaľ nie je úplne správne, musíte teda upraviť príkaz `\refstepcounter`:

```
\renewcommand*\refstepcounter[1]{%
  \stepcounter{#1}%
  \protected@edef\@currentlabel{%
    \csname p@#1\expandafter\endcsname
      \csname the#1\endcsname
  }%
}
```

So zmenou na správnom mieste teraz môžete napríklad zmeniť návestia na všetkých vnútorných zoznamoch pridaním nasledujúceho kódu do preambuly:

```
\makeatletter
\renewcommand{\p@enumii}[1]{\theenumi(#1)}
```

```
\makeatother
```

Toto zabezpečí, že návestia číslovaných zoznamov druhej úrovne vyzerajú ako „1(a)“ (a tak ďalej). Analogická zmena funguje pre ľubovolné počítadlo používané v príkaze `\label`.

Balík *fncylab* v skutočnosti spraví všetko uvedené (vrátane úpravy LaTeXu samotného). S týmto balíkom je uvedený kód (celkom efektívne) vykonaný príkazom:

```
\labelformat{enumii}{\theenumii(#1)}
```

Uvedený príklad, ktorý môžeme uskutočniť viacerými rozličnými spôsobmi, bol označený zastaralým objavením sa balíku *enumitem*, ktorý je diskutovaný v odpovedi o dekorovaní číslovaných zoznamov (Otázka 157).

`enumitem.sty`: distribuované ako súčasť `macros/latex/contrib/bezos`

`fncylab.sty`: `macros/latex/contrib/misc/fncylab.sty`

271. Ako prekročiť limit 9 parametrov

Ak sa nad tým zamyslíte, zistíte, že Knuthova syntax definície príkazov:

```
\def\blah#1#2 ... #9{<macro body>}
```

je limitovaná na 9 parametrov. Neexistuje priamy spôsob ako toto obísť. Ako by ste vyjadrili desiaty parameter a zaistili, že syntax nepohltí iné správne použitie?

Ak skutočne potrebujete viac než 9 parametrov, správny postup je:

```
\def\blah#1#2 ... #9{%
  \def\ArgI{{#1}}%
  \def\ArgII{{#2}}%
  ...
  \def\ArgIX{{#9}}%
  \BlahRelay
}
\def\BlahRelay#1#2#3{%
  % parametre 1-9 sú teraz
  %   \ArgI-\ArgIX
  % parametre 10-12 sú
  %   #1-#3
  <macro body>%
}
```

Táto technika je ľahko rozšíriteľná TeXovskými virtuózmi, ale zle sa odporúča.

Používatelia LaTeXu sa majú o niečo lepšie, kedže im stačí iba zadať počet parametrov v príkaze `\newcommand`, ktorý definuje každú časť prevodného mechanizmu: Knuthove reštrikcie platia pre `\newcommand` ako aj pre `\def`. Používatelia LaTeXu však tiež majú cestu von z takejto barbarskej syntaxe príkazov – balík *keyval*. S *keyval* a trochou programovania môžete písať celkom sofistikované príkazy, ktorých vyvolanie môže vyzerať takto:

```
\flowerinstance{species=Primula veris,
  family=Primulaceae,
  location=Coldham's Common,
  locationtype=Common grazing land,
  date=1995/04/24,
  numplants=50,
  soiltype=alkaline
}
```

Výhoda takejto „ukecanosti“ je automatická zrozumiteľnosť. Sádzač si nemusí pamätať, že parameter 12 je `soiltype` atď. Príkazy môžu byť kopírované z poznámok k políčkam rýchlo a presne.

`keyval.sty`: distribuované ako súčasť `macros/latex/required/graphics`

272. Príkaz s dvomi voliteľnými parametrami

Ak ste si už prečítali Otázku 271, môžete zrejme riešenie problému uhádnuť: predávanie príkazov.

LaTeX povoľuje príkazy s jediným voliteľným parametrom takto:

```
\newcommand{\blah}[1][Default]{...}
```

Správne môžete buď s prítomným voliteľným parametrom (`\blah[nonDefault]`) alebo bez neho (`\blah`). V druhom prípade bude mať `\blah` parameter `Default`.

Pre definovanie príkazu s dvoma voliteľnými parametrami použijeme techniku predávania nasledovne:

```
\newcommand{\blah}[1][Default1]{%
  \def\ArgI{{#1}}%
  \BlahRelay
}
\newcommand\BlahRelay[1][Default2]{%
  % prvý voliteľný parameter je teraz
  %   v \ArgI
  % druhý je v #1
  ...%
}
```

`\BlahRelay` môže mať samozrejme toľko povinných parametrov, koľko je povolených po zabratí jedného „miesta" vlastným voliteľným parametrom, teda 8.

Varianty `\newcommand` (a priateľov) s názvami ako `\newcommandtwoopt` sú dostupné v balíku *twoopt*. Je však pravdepodobne lepšie, ak sa môžete písať príkazy sami, aby ste videli, prečo nie sú z programátorského hľadiska dobrým nápadom.

Príkaz s dvomi voliteľnými parametrami sa približuje hranici toho, čo je ešte rozumné: techniku môžete zjavne rozšíriť tak, aby poskytovala toľko volteľných parametrov, koľko si viete predstaviť. Pozrite si však poznámky k použitiu balíku *keyval* (znova v Otázke 271), ktoré poskytujú alternatívny spôsob postupu.

Alternatívnym prístupom je poskytovaný programom *newcommand* od Scotta Pakina, ktorý vezme názov príkazu a definíciu množiny parametrov príkazu (v celkom zrozumiteľnom jazyku) a ako výstup poskytne TEXovské/LATEXovské makrá umožňujúce definíciu príkazu. Príkaz vyžaduje, aby bol na vašom systéme nainštalovaný *python*.

Distribúcia *twoopt* obsahuje dokumentačný súbor `twoopt.pdf`. Podobne distribúcia *newcommand* obsahuje súbor `newcommand.pdf`.

`newcommand.pdf`: `support/newcommand`
`twoopt.sty`: distribuované ako súčasť `macros/latex/contrib/oberdiek`

273. Úprava prezentácie čísel sekcií

Obecné problémy úpravy vzhľadu nadpisov sekcií sú dosť komplexné a sú pokryté odpoveďou na Otázku 178.

Ľudia však často chcú zmeniť iba vzhľad čísla sekcie v nadpise a niektorým z nich nevadí písanie pár makier. Táto odpoveď je pre nich.

Spôsob sadzby čísla sekcie určuje príkaz `\@seccntformat`, ktorý dostane „názov" (section, subsection, ...) nadpisu ako parameter. Obvykle dá na výstup iba číslo sekcie, a potom `\quad` medzeru. Predpokladajme, že chcete dať za každé číslo sekcie (subsekcie, subsubsekcie, ...) bodku. Triviálna zmena môže byť implementovanáa jednoduchou modifikáciou príkazu:

```
\renewcommand*{\@seccntformat}[1]{%
  \csname the#1\endcsname.\quad
}
```

Veľa ľudí (z určitého dôvodu) chce bodku iba za číslo sekcie. Pre dosiahnutie tochto cielu musíme zmeniť chovanie `\@seccntformat` podľa jeho parametru. Nasledujúca technika je tak trochu plýtvaním, ale stále je dostačujúco efektívna:

```
\let\@@seccntformat\@seccntformat
\renewcommand*{\@seccntformat}[1]{%
  \expandafter\let\expandafter\@tempa
    \csname @seccntformat@#1\endcsname
  \ifx\@tempa\relax
    \expandafter\@@seccntformat
```

```
    \else
      \expandafter\@tempa
    \fi
      {#1}%
  }
```
Kód sa pozrie či bol definovaný príkaz druhej úrovne. Ak áno, použije ho, inak použije pôvodný. Príkaz druhej úrovne na zavedenie bodiek (iba) za čísla sekcií má rovnakú definíciu ako pôvodná verzia „všetky úrovne zhodne“:
```
  \newcommand*{\@seccntformat@section}[1]{%
    \csname the#1\endcsname.\quad
  }
```
Všimnite si, že všetky definície príkazov v tejto odpovedi sa týkajú interných príkazov LaTeXu (viď Otázku 243), takže uvedený kód by mal byť radšej v súbore balíku.

Triedy `koma-script` majú na zmenu prezentácie čísel sekcií odlišné príkazy: `\partformat`, `\chapterformat` a `\othersectionlevelsformat`, ale inak sú ich možnosti podobné čistému LaTeXu.

`sada KOMA script:` `macros/latex/contrib/koma-script`

274. Za moje prostredie je pridaná medzera

Napísali ste si vlastné prostredie `env`, ktoré aj funguje, až na to, že na začiatku prvého riadku textu za `\end{env}` je vysádzaná medzera. Toto sa s podobnými prostrediami dodávanými LaTeXom nestáva.

Mohli by ste vydať obmedzenie, že vaši užívatelia musia vždy umiestniť znak „`%`“ za prostredie, prostredia LaTeXu to nepotrebujú.

Tajomstvom prostredí LaTeXu je interný príznak spôsobujúci ignorovanie nechcených medzier. Našťastie internú formu používať nemusíte: od roku 1996 obsahuje LaTeX používateľský príkaz `\ignorespacesafterend`, ktorý interný príznak nastavuje.

275. Definície LaTeXovských príkazov

Existuje viacero dôvodov, prečo by ste mohli chcieť vedieť definície príkazov LaTeXu: od najjednoduchšej obyčajnej zvedavosti, až po potrebu opraviť niečo, aby to fungovalo tak, ako chcete vy. Nič z uvedeného nie je *čistým* motívom, ale vedomosť a skúsenosť zriedkakedy prichádzajú prostredníctvom najčistejších motívov.

Najjednoduchšou odpoveďou je skúsiť `\show` v LaTeXovskom behu berúcom príkazy z terminálu:
```
  *\show\protected@edef
  > \protected@edef=macro:
  ->\let \@@protect \protect
    \let \protect \@unexpandable@protect
    \afterassignment \restore@protect \edef .
```
(Výstup je preusporiadaný z dosť mätúcej verzie, ktorú TeX produkuje.) Teraz sa môžeme zamyslieť nad `\@unexpandable@protect`:
```
  *\show\@unexpandable@protect
  > \@unexpandable@protect=macro:
  ->\noexpand \protect \noexpand .
```
a začíname pozorovať, ako funguje jedna časť mechanizmu `\protect` (môžeme asi celkom bezpečne uhádnuť, čo robí `\restore@protect`).

Mnoho príkazov jadra je deklarovaných robustne:
```
  *\show\texttt
  > \texttt=macro:
  ->\protect \texttt  .
```
takže `\show` veľmi nepomôže. Definujte príkaz `\pshow` podľa nasledujúcej ukážky a použite ten:
```
  *\def\pshow#1{{\let\protect\show #1}}
  *\pshow\texttt
```

```
> \texttt =\long macro:
#1->\ifmmode \nfss@text {\ttfamily #1}%
    \else \hmode@bgroup \text@command {#1}%
          \ttfamily \check@icl #1\check@icr
    \expandafter \egroup \fi .
```

Všimnite si, že názov chráneného príkazu je „základný“ príkaz s pripojenou medzerou. Toto je trochu obtiažne viditeľné na pár miestach vyššie. (Výstup bol znova upravený.)

Ak máte flexibilný textový editor, rovnaké zistenie môžete komfortnejšie uskutočniť prezrením súboru `latex.ltx` (ktorý sa dá zvyčajne v TDS systéme nájsť v adresári `tex/latex/base`).

Súbor `latex.ltx` je vlastne výstupom *docstrip* procesu na veľkom množstve `.dtx` súborov (viď Otázku 44), môžete sa teda odkazovať na ne. Distribúcia LaTeXu obsahuje súbor `source2e.tex`, väčšina systémov ju zachováva v `tex/latex/base`. `Source2e.tex` môže byť spracovaný, aby poskytol kompletný výpis jadra LaTeXu (proces nie je úplne priamočiary, ale súbor produkuje správy radiace, čo treba spraviť). Výsledkom je obrovský dokument s indexom čísiel riadkov riadiacich sekvencií celého jadra a oddelený index zmien zaznamenaných v každom súbore odvtedy, čo ho prevzal LaTeX tím.

Vytlačené jadro je dobrá vec, ale ťažko sa s ním manipuluje a často iba sedí na polici zriedkavo využívaný. Jedným problémom je rozdielna dokumentácia: obsiahnuté sú moduly od veľmi dobre zdokumentovaných, cez moduly obsahujúce iba automatickú dokumentáciu, až po moduly bez akejkoľvek užitočnej dokumentácie.

Každý `.dtx` súbor jaderného modulu bude v LaTeXu spracovaný oddelene, takže nemusíte pracovať s celým súborom `source2e`. Ľahko môžete zistiť, ktorý modul definuje makro, ktoré vás zaujíma: použite svoj „flexibilný textový editor“ aby ste zistili definíciu v `latex.ltx`, potom z toho bodu hľadajte spätne riadok začínajúci `%%% From File:` — tento riadok vám povie, ktorý `.dtx` súbor obsahuje definíciu, ktorá vás zaujíma. Pre `\protected@edef` týmto postupom nájdeme:

```
%%% From File: ltdefns.dtx
```

Ak sa na súbor pozrieme, `ltdefns.dtx` obsahuje obsiahlu rozpravu o metódach spracovania ochrán (`\protect`) a obsahuje tiež určitú automaticky skonvertovanú LaTeX 2.09 dokumentáciu.

Jadro samozrejme nie je celý LaTeX: váš príkaz môže byť definovaný v jednom zo súborov LaTeXovských tried alebo balíkov. Definíciu `\thebibliography` napríklad nájdeme v triede `article`, ale súbor `article.dtx` neexistuje. Niektoré takéto súbory sú generované z častí jadra, niektoré z iných súborov distribúcie. Zdroj zistíte pri pohľade na začiatok súboru: v `article.cls` nájdeme:

```
%% This is file ‘article.cls’,
%% generated with the docstrip utility.
%%
%% The original source files were:
%%
%% classes.dtx  (with options: ‘article’)
```

takže potrebujeme formátovať súbor `classes.dtx`, aby sme videli definíciu v kontexte.

Všetky tieto .dtx súbory sú na CTANe ako súčasť hlavnej distribúcie LaTeXu.

`\LaTeX{} distribúcia`: `macros/latex/base`

276. „Master“ a „slave“ počítadlá

Je bežné mať veci číslované po kapitolách (napríklad v štandardných triedach `book` a `report` sú takto číslované obrázky, tabuľky a poznámky pod čiarou). Resetovanie sa uskutočňuje automaticky, keď sa zvýši hlavné („master“) počítadlo (keď je vykonaný príkaz `\chapter` začínajúci kapitolu $<n>$, počítadlo `chapter` sa zvýši a všetky závislé počítadlá sú nastavené na nulu).

Ako by ste to spravili sami? Môžete chcieť číslovať napríklad algoritmy po sekciách. Ak tieto veci definujete ručne, vzťah deklarujete pri definícii počítadla:

```
\newcounter{new-name}[master]
```

Uvedený kód nám hovorí, že pri každom zvýšení počítadle <*master*> bude resetované počítadlo <*new-name*>.

Čo ale v prípade, že máte nespolupracujúci balík definujúci veci za vás, ale tento neposkytuje programátorovi rozhranie na prinútenie počítadiel správať sa tak, ako chce?

Príkaz `\newcounter` používa interný príkaz LaTeXu a vy ho môžete využiť tiež:

```
\@addtoreset{new-name}{master}
```

(pamätajte si však, že to musí byť medzi `\makeatletter` a `\makeatother` alebo vo vašom vlastnom balíku).

Balík *chngcntr* zapuzdruje príkaz `\@addtoreset` do príkazu `\counterwithin`. Takže:

```
\counterwithin*{corollary}{theorem}
```

spraví z počítadla corollary podriadené počítadlo (slave) počítadla theorem. Príkaz bez hviezdy:

```
\counterwithin{corollary}{theorem}
```

spraví to isté a zároveň redefinuje `\thecorollary` ako *<číslo teorémy>.<číslo dôsledku>*, čo je dobrou schémou, ak sa niekedy chcete odkazovať na dôsledky — môže existovať veľa „dôsledkov 1" v každom dokumente, takže je dobré zviazať jeho číslo s počítadlom theorem, ku ktorému patrí. Toto platí asi pri každom počítadle vo vnútri iného. Ak nepoužívate *chngcntr*, pre potrebné techniky si pozrite Otázku 155.

Všimnite si, že postup nefunguje, ak je nadriadené (master) počítadlo `page`, číslo aktuálnej strany. Počítadlo `page` je zväčšované hlboko vo vnútri výstupnej rutiny, ktorá je volaná chvíľku po tom, čo sa začal objavovať text pre novú stranu: na vysporiadanie sa s týmto sú potrebné špeciálne postupy. S jedným špeciálnym prípadom sme sa vysporiadali na inom mieste: Otázka 185. Jeden z opísaných postupov v spomenutej otázke, použitie balíku *perpage*, môže byť aplikovaný na akékoľvek počítadlo. Príkaz:

```
\MakePerPage{counter}
```

prinúti *<counter>* resetovať sa na každej strane. Balík používa mechanizmus podobný návestiam a môže vyžadovať viac než jeden beh LaTeXu na stabilizovanie hodnôt počítadiel — LaTeX vygeneruje obvyklé varovania o zmene návestí.

`chngcntr.sty`: `macros/latex/contrib/misc/chngcntr.sty`
`perpage.sty`: `macros/latex/contrib/misc/perpage.sty`

277. Ovládanie vdov a sirôt

Vdovy (posledné riadky odstavcov na začiatku strany) a siroty (prvé riadky odstavcov na konci strany) prerušujú čitateľa a obecne sú považované za zlú formu; TeX/LaTeX vykonáva určité opatrenia, aby sa im vyhol, úplne automatická prevencia je často nemožná. Ak sádzate vlastný text, zvážte miernu zmenu slov tak, aby zlom dopadol inak. Rutina pre vytváranie stránky pri jej formovaní berie na vedomie `\widowpenalty` a `\clubpenalty` (vzťahuje sa k sirotám!). Tieto pokuty (penalties) sú obvykle nastavené na strednú hodnotu `150`; toto mierne odradzuje od zlých zlomov. Hodnoty môžete zvýšiť výrazom (napríklad) `\widowpenalty=500`; vertikálne zoznamy (z ktorých sú zhotovené stránky) sa typicky dajú veľmi slabo roztiahnuť alebo stlačiť, takže rutina musí zvážiť efekt roztiahnutia neroztiahnuteľného a pokuty, pokuta málokedy vyhraje. Tomuto rozporu sa dá predísť povolením rutine skrátiť stránku použitím direktívy `\raggedbottom`; mnoho vydavateľov však trvá na štandardnom `\flushbottom`; je málokedy akceptovateľné priviesť do vertikálneho zoznamu roztiahnuteľnosť, okrem bodov (ako nadpisy sekcií), kde to dizajn dokumentu explicitne povoľuje.

Keď ste už vyčerpali automatické opatrenia a máte konečný návrh, ktorý chcete „vypilovať", musíte použiť ručné opatrenia. Zbaviť sa siroty je jednoduché: pred odstavec uveďte `\clearpage` a odstavec nemôže začať na zlom mieste.

Zbaviť sa vdovy je komplikovanejšie. Ak je odstavec dlhý, mohlo by sa dať vysádzať ho „tesne": vložte `\looseness=-1` bezprostredne za posledné slovo odstavca. Ak to nefunguje, skúste zmeniť veľkosť stránky: `\enlargethispage\baselineskip` by mohlo pomôcť a dostať celý odstavec na jednu stranu. Zmenšením veľkosti stránky pomocou

```
\enlargethispage{-\baselineskip}
```

môže vytvoriť (viac-menej) akceptovateľnú „dvojriadkovú vdovu". (Všimnite si: `\looseness=1` zvyšujúc dĺžku strany o jednotku málokedy funguje — dotyčný odstavec má zvyčajne jednoslovný posledný riadok, ktorý nevyzerá oveľa lepšie ako priamo vdova.)

278. Proč LaTeX dělí poznámky pod čarou na více stránek?

LaTeX dělí poznámky pod čarou na více stránek, pokud si myslí, že to je to nejlepší, co může udělat. Typicky se tak stane, pokud poznámka pod čarou vyjde na samotný konec stránky, a tím ji může přeplnit. LaTeX by se mohl pokusit problém vyřešit ochuzením stránky o poznámku a taky řádek obsahující značku poznámky, ale jeho priority mu poradí, že rozdělení rozdélení poznámky je preferováno.

Jako vždy je nejlepším řešením problému změnit váš text tak, aby se v něm tento problém nevyskytl. Uvažujte, zda by se poznámka ve vašem textu mohla objevit na předchozí či další stránce. Pokud to není možné, můžete docílit toho, že LaTeX začne jinak „pohlížet" na tyto priority: ty jsou ovládané příkazem `\interfootnotelinepenalty` — čím má větší parametr, tím méně LaTeX rozděluje poznámky. Nastavením `\interfootnotelinepenalty=10000` (základní hodnota je 100) zcela zabráníte zabráníte rozdělování poznámek pod čarou. To ale vyvolá hlášku „Underfull \vbox" pokud nezadáte `\raggedbottom`.

Alternativní technika spočívá v malém podvodu na LaTeX, kdy pomocí parametrů příkazu `\enlargethispage` změníte velikost aktuální stránky (např. mu můžete dát na vstup `\enlargethispage{\baselineskip }` pro přidání jednoho řádku na tuto stránku, můžete však použít jakoukoliv běžnou TeXovou jednotku délky, třeba 15mm nebo -20pt). Zmenšení velikosti běžné stránky si může vynutit useknutí textu a jeho přesunutí na další stránku; zvětšení stránky může způsobit, že se poznámky spojí do jednoho celku. To může být nevyhnutelné, pokud chcete změnit velikost více než jedné stránky.

`fnbreak.sty`: `macros/latex/contrib/fnbreak`

279. Jak dostat `\marginpar` na správnou stranu?

V ideálním světé by okrajové poznámky byly na analogických místech na každé stránce: na sudých stránkách v levém okraji, na lichých v pravém. Hned vidíme, že poznámka vlevo musí být vysázena jinak než poznámka napravo. LaTeXovský příkaz `\marginpar` proto v oboustraných (`twoside`) dokumentech akceptuje dva parametry:

```
\marginpar[left text]{right text}
```

LaTeX používá „zjevný" test, aby dostal `\marginpar` na správný okraj, ale úskalí je v tom, že vytváří stránky asynchronně. Když je `\marginpar` spracován, zatímco je budována stránka n, ale není použit až do stránky $n+1$, pak se `\marginpar` objeví na špatném okraji stránky. Toto je instance obecnějšího problému: viz Otázku 269.

Řešením tohoto problému by mohlo být zapamatování si, na kterou stranu stránky má být použit `\marginpar`. Toto umí balík `macros/latex/contrib/mparhack`, který využívá značky uložené v `.aux` souboru.

280. Kam zmizli moje písmená?

Napísali a spracovali ste zjavne zmysluplný text, ale výsledok neobsahuje ani stopu po niektorých písmenách, ktoré ste napísali. Pravdepodobným dôvodom je, že vybraný font neobsahuje reprezentáciu dotyčných znakov.

Ak napríklad napíšem „that will be £44.00" do obyčajného TeXovského/LaTeXovského dokumentu alebo si vyberiem font `rsfs10` (obsahujúci iba veľké písmená) a napíšete takmer čokoľvek, znak libry (£) alebo akékoľvek malé písmená alebo číslice budú vo výstupe chýbať. Chybové hlásenie neexistuje, musíte si prečítať log súbor, kde nájdete záhadné malé správy ako

```
Missing character:
           There is no ^^a3 in font cmr10!
Missing character:
             There is no 3 in font rsfs10!
```

(prvá demonštruje neochotu môjho TeXu vysporiadať sa so znakmi s osembitovou znakovou sadou, zatiaľ čo príklad `rsfs10` ukazuje, že TeX zaznamenáva dotyčný chybný znak, ak si myslí, že je to možné).

Trochu lepšie pochopiteľné sú diagnostiky, ktoré dostanete z *dvips*, keď používate OT1 a T1 verzie fontov dodávaných v štandardnom kódovaní Adobe:

```
dvips: Warning: missing glyph 'Delta'
```

Proces generujúci metriky pre používanie fontu generuje inštrukciu pre *dvips*, aby produkovalo tieto diagnostiky, aby ich neprítomnosť v tlačenom výstupe bola menej prekvapujúca, než by mohla byť. Dosť glyfov poskytnutých v Knuthovych textových kódovaniach a v Corkovskom kódovaní nie je v Adobe fontoch dostupných. V týchto prípadoch *existuje* vysádzaný symbol týchto znakov: *dvips* vytvára čierny obdĺžnik veľkosti fontu.

281. Príkazy požierajú nasledujúce medzery

Ľudia sú stále prekvapení, že jednoduché príkazy požierajú medzeru, ktorá za nimi nasleduje: je to jednoducho tak. Efekt vzniká pre spôsob práce TEXu, Lamport popisuje riešenie (umiestnite jedny zložené zátvorky za vyvolanie príkazu) v popise syntaxe LATEXu. Takže požiadavka je vlastne súčasťou definície LATEXu.

Anglické FAQ je napríklad napísané s definíciami *vyžadujúcimi* písanie `\fred{\relax }` pre takmer všetky vyvolania makier bez ohľadu na to, či je nasledujúca medzera potrebná: anglické FAQ je však písané veľmi oddanými (a podľa niekoho zvláštnymi) ľuďmi. Mnoho používateľov si myslí, že písanie tých všetkých zložených zátvoriek sa veľmi rýchlo stane veľmi únavným, a radšej by ich vôbec nepísali.

Alternatívnou štruktúrou, ktorá neporušuje návrh LATEXu, je napísať `\fred\` — príkaz `\` je „samoukončovací" (ako `\\`) zložené zátvorky za neho už písať nemusíte. Znížite teda počet extra znakov na jeden.

Ak je aj ten jeden znak priveľa, balík *xspace* definuje príkaz `\xspace`, ktorý háda, či by za ním mala byť medzera, a ak si myslí, že mala, vloží ju. Takže „`fred\xspace jim`" vytvorí „fred jim", zatiaľ čo „`fred\xspace. jim`" vytvorí „fred. jim". Jeho použitie by samozrejme bolo úplne nezmyselné, `\xspace` ale môžete zabudovať do vlastných makier:

```
\usepackage{xspace}
...
\newcommand{\restenergy}%
          {\ensuremath{mc^2}\xspace}
...
and we find \restenergy available to us...
```

Príkaz `\xspace` musí byť poslednou vecou v definícii vášho makra (ako v príklade); nie je to úplne bezchybné, ale poradí si s väčšinou situácii v texte.

Balík *xspace* vám nič neušetrí, ak modifikované makro použijete raz alebo dvakrát v dokumente. V každom prípade buďte pri použití `\xspace` opatrní — zmení vašu vstupnú syntax, čo môže byť mätúce, hlavne pre spolupracujúceho autora (predovšetkým ak vytvoríte nejaké príkazy, ktoré to používajú, a nejaké, ktoré nie). Žiadny príkaz zabudovaný do LATEXu alebo akejkoľvek triedy, prípadne balíku, samozrejme `\xspace` používať nebude.

`xspace.sty`: distribuované ako súčasť `macros/latex/required/tools`

282. Matematické symboly sa nezvetšujú

„Veľké" matematické symboly štandardne zostávajú rovnakej veľkosti nezávisle na veľkosť fontu textu dokumentu. Existuje pre to dobrý dôvod: fonty `cmex` nie sú navrhnuté na zmenu veľkosti, takže TEXovský algoritmus umiestňovania matematiky nepracuje tak dobre, ako by mohol, ak by boli fonty zväčšovateľné.

Toto správanie však mätie očakávania používateľov a môže viesť k trochu zvláštne vyzerajúcim dokumentom. Ak napriek varovanie chcete, aby sa fonty dali zväčšovať, použite balík *exscale* — stačí ho iba nahrať.

`exscale.sty`: súčasť distribúcie LATEXu.

283. Prečo používať *fontenc* namiesto *t1enc*?

V raných časoch LaTeX 2ε bol jediním spôsobom používania T1 kódovania balík *t1enc*. S vydaním z leta 1994 sa objavil balík *fontenc* a poskytol vyčerpávajúcu podporu použitia kódovania.

Napriek tomu tu balík *t1enc* zostáva (ako súčasť kódu kompatibility LaTeX 2.09), ale robí veľmi málo: iba vyberie kódovanie fontov T1 a na používateľovi nechá záležitosti generovania potrebných kódov znakov.

Generovanie takýchto kódov znakov by mohla byť jednoduchá záležitosť, keby T1 kódovanie zodpovedalo ľubovolnému široko podporovanému kódovaciemu štandardu, pretože v takomto prípade by ste mohli očakávať generovanie kódov znakov klávesnicou. T1 kódovanie je však zmesou rôznych štandardných kódovaní a zahŕňa kódové miesta v oblastiach tabuľky, ktoré štandardné kódovanie špecificky vylučujú, takže žiadne T1 klávesnice neboli (a nikdy nebudú) vyrobené.

Balík *fontenc* naproti tomu generuje kódové miesta T1 z bežných LaTeX príkazov (generuje napríklad kódové miesto znaku `é` z príkazu `\'e`). Pokiaľ teda nemáte programovo generovaný T1 vstup, použite `\usepackage[T1]{fontenc}` namiesto `\usepackage{t1enc}`.

284. Prečo sa trápiť s *inputenc* a *fontenc*?

Štandardné vstupné kódovanie pre západnú Európu (čakajúc príchod Unicode) je ISO 8859–1 (bežne známe podtitulom štandardu „Latin-1“). Latin-1 je v pokrytí kódových miest pozoruhodne blízko TeXovskému T1 kódovaniu.

Prečo by sme sa mali za týchto okolností trápiť s *inputenc* a *fontenc*? Kedže sa v podstate kopírujú, mohli by sme ich odstrániť a použiť *t1enc* (napriek jeho nedostatkom — viď Otázku 283).

Nerobí sa to ale pre množstvo drobných dôvodov:

Zmätenie Spokojne ste v tomto móde písali a z nejakého dôvodu prepnete na písanie v nemčine: efekt použitia „ß“ je tak trochu prekvapivý, kedže T1 a Latin-1 s kódovým miestom zaobchádzajú inak.

Kompatibilita Zistíte, že potrebujete pracovať s kolegom vo východnej Európe: ich klávesnice bude zrejme nastavená tak, aby produkovala Latin-2, takže jednoduché mapovanie nefunguje.

Tradičný LaTeX Napíšete niečo ako `\'e` namiesto `é`; iba *fontenc* má prostriedky pre konverziu tejto LaTeXovskej sekvencie do T1 znaku, takže primitívum `\accent` prekĺzne do výstupu a rozdeľovanie je ohrozené.

Kombinácia *inputenc–fontenc* vyzerá pomalá a ťažkopádna, je však bezpečná.

285. Prečo nepoužiť *eqnarray*?

Prostredie `eqnarray` je pre príležitostného používateľa matematiky v LaTeXovských dokumentoch atraktívne: vyzerá, že povoľuje zarovnané systémy rovníc. Skutočne, tieto veci prostredie poskytuje, ale v rozostupoch znakov urobí pekný neporiadok. V systéme:

```
\begin{eqnarray}
  a & = & b + c \\
  x & = & y - z
\end{eqnarray}
```

nie sú rozostupy okolo znakov „=“ tie definované v metrike fontu, z ktorého pochádza glyf — sú to `\arraycolsep`, čo môže byť nastavené na veľmi divnú hodnotu z dôvodov spojených so skutočnými poliami na inom mieste dokumentu.

Používateľa oveľa lepšie obslúži sada AMS-LaTeX, ktorá poskytuje prostredie `align` navrhnuté rešpektujúc potreby matematikov (oproti pohodliu LaTeXovských programátorov). Pre tento jednoduchý dôvod (align je schopné oveľa lepších vecí) použite:

```
\begin{align}
  a & = b + c \\
  x & = y - z
\end{align}
```

`AMSLaTeX`: `macros/latex/required/amslatex`

286. Prečo používať `\[...\]` namiesto `$$...$$`?

LaTeX definuje príkazy „inline“ a „display“ matematiky, zjavne analogické k tým, čo sú odvodené z TeXovských príkazov na zátvorkovanie matematických sekvencií so znakmi doláru (alebo dvojicami znakov doláru).

Ako sa ukázalo, LaTeXovské inline zoskupovanie matematiky `\( ... \)` má presne rovnaký účinok ako TeXovská primitívna verzia `$ ... $`. (Jediný rozdiel je, že LaTeXovská verzia kontroluje, či ste nepoužili `\(` a `\)` zle.)

Za týchto okolností sa často nájdu používatelia LaTeXu majúci nejaké skúsenosti s používaním Plain TeXu, ktorí jednoducho predpokladajú, že LaTeXovské zoskupovanie display matematiky `\[ ... \]` môže byť vymenené TeXovským primitívnou display matematikou `$$ ... $$`.

Bohužiaľ sa mýlia: ak LaTeXovský kód bude upravovať display matematiku, môže tak učiniť iba úpravou `\[` a `\]`. Najzjavnejším spôsobom, ako sa to prejaví, je, že voľba triedy `fleqn` jednoducho nefunguje pre rovnice napísané pomocou `$$ ... $$`, či už používate iba štandardné triedy, alebo balík *amsmath*.

Existujú aj zákernejšie efekty (hlavne s balíkom *amsmath*), jednoduché pravidlo je `\[ ... \]`, kedykoľvek je v LaTeXu potrebná neozdobená display matematika.

287. Proč nemůžu nahrát PICTeX

PICTeX je poměrně náročný na systémové zdroje. Naštěstí většina moderních TeXových distribucí nabízí poměrně dost prostoru a moderní počítače již jsou oproti svým předchůdcům o mnoho rychlejší, takže uživatele tato skutečnost nemusí znervózňovat. Nicméně PICTeX má jednu nešťastnou tendenci – rád zaplňuje pole s pevnou délkou alokovaná TeXem – zvláště 256 „rozměrových“ registrů. To je problém zvláště, pokud v LaTeXu používáte *pictex* a další balíky, které potřebují stejné systémové zdroje. Pokud k tomu dojde, pak vám LaTeX vypíše chybovou hlášku

```
! No room for new \dimen.
```

S touto chybou nelze nic udělat: nemůžete totiž zvýšit počet dostupných rozměrových registrů, aniž by to udělal sám TeX (toto umí automaticky ε-TeX a Omega – viz Otázku 297 a Otázku 298 a VTeX od firmy MicroPress Inc. – viz Otázku 63). Je celkem praktické (u většiny moderních distribucí) použít rozšířenou sadu registrů ε-TeXu: použijte balík *etex* (který se dodává s distribucemi ε-TeXu) a alokační mechanismus je pozměněn, aby se vyspořádal s většími sadami registrů: PICTeX se teď nahraje.

Když nemůžete použít ε-TeX, musíte změnit PICTeX; bohužel jeho autor není již delší dobu v TeXovém světě aktivní, proto se musíte uchýlit k „patchování“. Dostupná jsou dvě řešení:

- CONTeXTové moduly `m-pictex.tex` (pro Plain-TeX a jeho varianty) a `m-pictex.sty` (pro LaTeX; oba soubory jsou dostupné v souboru `macros/context/cont-tmf.zip`). Soubor `m-pictex.tex` nabízí důmyslné řešení založené na úpravě kódu příkazu `\newdimen`.
- Alternativně `pictexwd` Andrease Schella (`graphics/pictex/addon`) nahrazuje PICTeX verzí, které používá o 33 „rozměrových“ registrů méně. Takže místo souborů `pictex` používejte `pictexwd`.

A jak může někdo použít PICTeX, když je těžké sehnat manuál (viz Otázku 33)? Naštěstí pro uživatele MS-DOSu a Windows, může být *MathsPic* (`graphics/pictex/mathspic`) použit pro přeložení z jiného jazyka do instrukcí PICTeXu a manuál k MathsPicu je přímo součástí distribuce. *MathsPic* je napsán v Basicu; Perlová verze by měla být dostupná během roku 2001.

S. Chyby v TeXu

288. Jak na chyby v TeXu?

Poněvadž TeX je makroprocesor, je často poměrně obtížné porozumět jeho chybovým hláškám; to je bohužel vlastnost makroprocesorů. Knuth se snažil vnést jasno do tohoto problému ve svém TeXbooku tím, že uživatelům naznačil, aby získlali detektivní schopnosti moderního Sherlocka Holmese; ačkoliv má tento přístup jakýsi romantický nádech, není pro aktivní uživatele (La)TeXu příliš užitečný. Tato odpověď (z části odvozená z článku Sebastiana Rathze z *TUGboatu* **16**(4)) nabízí obecnou radu, jak reagovat na chyby pomocí chybových zpráv a jiné odpovědi v této sekci pojednávají o jiných bežných (ale matoucích) chybách, se kterými se můžete v TEXu setkat. V Sebastianově článku se objevuje obsáhlý seznam rad, v němž mimo jiné stojí:

- Podívejte se podrobně na chybu; některé chyby se nejprve mohou zdát záhadné, ale často obsahují přímé vodítko k vyřešení problému. Pro další detaily je dobré si pročíst následující otázku.
- Přečtěte si soubor `.log`; obsahuje podrobnější informace o věcech, kterým třeba nerozumíte, často nejsou předloženy jako chyby.
- Buďte si vědomi množství souvislostí, které TEX obnáší. Chybové hlášky vám dávají kousky TEXového kódu (nebo samotného dokumentu), aby vám ukázal, kde se právě stala chyba, je možné zkontrolovat, kolik těchto chyb se ve vašem dokumentu objevuje. Nyní LATEX dává TEXu pokyn, aby vám vypsal pouze jeden řádek z kontextu, avšak pokud chcete, můžete mu říct, aby vypsal více řádků. To provedete umístěním příkazu `\setcounter{errortextlines}{999}` v preambuli vašeho dokumentu.
- Jako poslední východisko může být použitelným nástrojem trasování; čtení celého dokumentu vás může velice rychle zavést ke zdroji vašeho problému. Musíte si přečíst některé pasáže z TEXbooku (viz Otázku 23), abyste tomuto problému zcela porozuměli. Příkaz `\tracingall` nastavuje maximální trasování; nastavuje také výstup přicházející na interaktivní terminál. Běžně je nejjednodušší pročíst trasování použitím vašeho oblíbeného textového editoru k prohlížení `.log` souborů.

289. Struktura chybových hlášek v TEXu

TEXové chybové hlášky připomínají dobu, kdy byl TEX vymyšlen (70. léta minulého století): k uživatelům nejsou příliš přátelské, ačkoliv obsahují veškeré informace, které může TEX nabídnout, obvykle jsou vyloženy přehledně a stručně. Všechny chybové hlášky mají v TeXu stejnou strukturu:

- chybová hláška
- kontext
- stručný návod na řešení problému

Chybová hláška uvede stav TEXu, který způsobil daný problém. Bohužel, v soustavě balíků maker jako je LATEX však bývá odhalení konkrétního problému v makrech vyšší úrovně poměrně obtížné. Mnohé chyby, které se v LATEXu objevují, se tváří jako generické chyby s chybným textem vloženým LATEXem nebo jeho balíky.

Kontext chyby je stylizovaná reprezentace toho, co TEX udělal, když detekoval chybu. Jak bylo poznamenáno v Otázce 288, makrobalíček může TEXu říct, jak dlouhý kontext má zobrazit. Každá řádka kontextu je rozdělena v bodě chyby; pokud chyba ve skutečnosti nastane v makru volaném z aktuální řádky, zlom nastane v bodě volání. Pokud má makro parametry, tak se řádka zlomí až za posledním z nich. Např.

```
\blah and so on
```

vyvolá chybu

```
! Undefined control sequence.
1.4 \blah
          and so on
```

zatímco

```
\newcommand{\blah}[1]{\bleah #1}
\blah{to you}
```

má za důsledek hlášku

```
! Undefined control sequence.
\blah #1->\bleah
                 #1
1.5 \blah{\bleah}, folks
```

Pokud chyba nastane v argumentu, pak se nám mohou objevit věci jako např. po zadání

```
\newcommand{\blah}[1]{#1 to you}
\blah{\bleah}, folks
```

si můžeme přečíst

```
! Undefined control sequence.
<argument> \bleah

1.5 \blah{\bleah}
                   , folks
```

Příkazový řádek akceptuje jednoznakové příkazy, jejichž seznam si můžete nechat vypsat zadáním znaku `?`. Jedním z bezprostředně nejdůležitějších příkazů je říkaz `h`, který vám dá na výstup přesnější zprávu občas doplněnou pokynem, jak chybu odstranit, případně jak se jí vyhnout. Pokud stisknete `enter`, nebo jinou klávesu, kterou váš systém používá k ukončení řádky, TeX se pokusí pokračovat (často s poněkud menším úspěchem.)

290. Pravá složená závorka navíc

Viděli jste ve vašem LaTeXovém zdroji znak „`}`“ navíc a přesto LaTeX, v tomto místě nehlásil chybu? Toto je způsob, jak říci TeXu, že jste vložili křehký příkaz s pohyblivým argumentem (viz Otázku 255). Příkladem takovéhoto křehkého příkazu je třeba `\footnote`; pokud ho vložíte jako argument příkazu `\section`, pak vám sekvence

```
\section{Mumble\footnote{I couldn't think of anything better}}
```

dá na výstup:

```
! Argument of \@sect has an extra }.
```

Obvykle se tato situace řeší použitím robustního příkazu namísto toho, který používáte nebo přeměnit váš příkaz na robustní pomocí prefixu `\protect`, který ve výše zmíněném případě může vypadat takto:

```
\section{Mumble\protect\footnote{I couldn't think of anythimg better}}.
```

Na závěr ještě jedna poznámka: `\protect` vždy nemusí být řešením, Otázka 175 popisuje jednotlivé případy.

291. Není místo pro přidání nových věcí

Technologie, která byla Knuthovi dostupná v době, kdy vytvářel TeX, příliš neumožňovala spravovat dynamické paměti; většinou jsou místa v paměti v TeXu alokována jako pole pevné délky. Mnoho těchto fixovaných polí je v moderních implementacích TeXu rozšiřitelných, avšak velikost polí těchto registrů je obecně omezena počtem 256; většinou nemůže být změněna, pokud stále budete chtít zavolat výsledky TeXu (viz Otázku 6). Pokud naplníte jeden z těchto registrů, můžete dostat chybovou hlášku

```
! No room for a new \<thing>.
```

Zde se místo `\<thing>` může objevit `\count` (objekt, který tvoří základ příkazu `\newcounter`), `\skip` (objekt, který tvoří základ příkazu `\newlength`), `\box` (objekt, který tvoří základ příkazu `\newsavebox`), `\dimen`, `\muskip`, `\toks`, `\read`, `\write` nebo `\language` (všechny typy balíků, které se zde používají, jsou v LaTeXu skryté; limit počtu objektů `\read` a `\write` je omezen číslem 16).

Není zde žádná možnost, jak tuto chybu přímo opravit, nemůžete totiž přímo zvětšit počet dostupných registrů, aniž by to učinil přímo LaTeX. (Jistě, Omega a ε-TeX – viz Otázku 297 respektive Otázku 298 – to umí, stejně jako VTeX firmy MicroPress Inc – viz Otázku 63). Nejobecnější způsob, jak se setkat s jednou z těchto chyb, je použít nefunkční makro nebo použít dobré makro na špatném místě (případy jsou diskutovány v Otázce 292). Nicméně občas potřebujete více, než vám TeX může nabídnout, pokud se tak stane, musíte použít jiný způsob vytvoření objektů. Příkladem je složitější nahrávání PICTeXu v LaTeXu (viz Otázku 287).

292. `epsf` přestává po jisté době spolupracovat

Zdá se, že některé kopie dokumentace `epsf.tex` naznačují, že příkaz `\input epsf` je potřebný pro každý vložený symbol. Pokud vezmete toto doporučení příliš doslovně, pak se vám za chvíli objeví chyba

```
! No room for a new \read.
```
Je to protože pokaždé, když se `epsf.tex` nahrává, je alokován nový handle pro soubor určený pouze pro čtení, který kontroluje tento soubor a pak zde není dost místa pro další věci (viz Otázku 291). Řešení je jednoduché – toto je totiž případ nesprávného použití maker; každý totiž potřebuje číst `epsf.tex` pouze jednou, změňte tedy
```
...
\input epsf
\epsffile{...}
...
\input epsf
\epsffile{...}
atd.
```
na
```
\input epsf
```
kdesi na začátku dokumentu a pak doplňte váš příkaz `\epsffile` přizpůsobením velikosti pomocí příkazu `\epsfxsize`.

T. Současný vývoj

293. Makro NFSS (The New Font Selection Scheme)

NFSS bylo rozšířením LaTeXu napsané Frankem Mittelbachem a Rainerem Schöpfem. Je popsáno v TUGboatu, 1989, **10**(2). V češtině je filosofie makra popsána na str. 66–70 knihy Petra Olšáka *Typografický systém TeX*. V tradiční sazbě jsou písma popsána čtyřmi parametry: *rodina písma* (family) (např. computer modern), *duktus* (series) (tj. světlost či temnost písma), *varianta* (shape) (např. kurzíva) a *stupeň* (size) (např. 10pt). NFSS zavádí mechanismus, který umožňuje uživateli nezávisle měnit libovolný z těchto parametrů. NFSS umožňuje poměrně snadno užívat nestandardní fonty (např. PostScriptové fonty) v LaTeXu a jednoduše měnit matematické fonty. Makro umožňuje rovněž dynamické načítání fontů (tj. nikoliv v době tvorby formátu, ale až při formátování dokumentu).

Jelikož verze LaTeXu 2.09 již není udržována a vylepšována, název NFSS se stal zavádějícím, protože nyní vlastně neexistuje žádný „starý" způsob výběru fontu, se kterým by tento nový mohl kontrastovat, neboť LaTeX (tj. LaTeX 2_ε) v sobě NFSS zahrnuje.

294. LaTeX 2_ε (nový standard LaTeXu)

LaTeX 2_ε je nová verze balíku maker LaTeX, která je připravena a rozvíjena tvůrčí skupinou projektu LaTeX3. Testovací fáze LaTeXu 2_ε začala v červnu roku 1994 a je nyní standardem LaTeXu. LaTeX 2.09 není již více podporován.

LaTeX 2_ε je zpětně kompatibilní s LaTeXem 2.09, ale má navíc některé nové vlastnosti:

- NFSS (viz Otázku 293) se stává standardem.
- SliTeX je nyní do něj zahrnut, takže není potřeba používat zvláštního formátu.
- Lepší ovládání plovoucích objektů (prostředí `figure` a `table`).
- Obsahuje dokumentované rozhraní pro autory nových stylů a tříd (avšak zatím chybí obdobné rozhraní pro návrháře vzhledu dokumentu)
- Rozšířené příkazy pro práci s boxy, např. možnost volby výšky minipage.
- Křížový odkaz `\ref` může být použit v `\caption` bez ochrany příkazem `\protect`.
- `\newcommand` umožňuje definovat příkazy s jedním volitelným argumentem.
- Standardní balík pro vkládání grafiky a užití barev.

LaTeX 2_ε je průběžně udržován a vyvíjen, proto lze sepsat chyby a těžkosti při jeho užívání vyvoláním příkazu „`latex latexbug`" a vygenerovanou zprávu zaslat na elektronickou adresu `latex-bugs@uni-mainz.de`.

295. Projekt LaTeX3

Tvůrčí skupina projektu LaTeX3 (viz `http://www.latex-project.org/latex3.html`) je malá skupina dobrovolníků, jejichž cílem je vytvořit nový systém pro sazbu dokumentů založený na zásadách, jež jsou obsaženy v současném LaTeXu Leslieho Lamporta. Tento nový systém zůstane volně šiřitelný a bude na všech úrovních podrobně zdokumentován.

Skupina projektu LaTeX3 již uvedla svůj první produkt, LaTeX 2_ε (viz Otázku 294). Balík maker vychází z původního Lamportova kódu a je upraven tak, aby ho bylo možno lépe udržovat a rozšiřovat.

Prvním výrobkem týmu projektu LaTeX 3 byl LaTeX 2_ε, který byl vydán v roce 1994 (dnes se mu říká prostě LaTeX, protože je to jediná běžně dostupná verze).

LaTeX 2_ε měl spojit výhody různých verzí LaTeXu do jednoho programu, zároveň ale autoři nechtěli změnit nic, co nebylo nutné. To umožnilo týmu zachovat jednu verzi LaTeXu souběžně s vývojem LaTeXu 3.

Některé starší články diskutující, kterými směry by se měl LaTeX 3 vyvíjet, jsou k nalezení v adresáři `info/ltx3pub`. Další články podobného zaměření můžete nalézt na webové stránce tohoto projektu na URL `http://latex-project.org/articles.html`. Na tomto webu také najdete některé experimentální kódy tohoto projektu. Prostřednictvím mailing listu `latex-l` se můžete zúčastnit diskuse o budoucnosti TeXu. Stačí se přihlásit prostřednictvím E-mailové zprávy obsahující „subscribe latex-l <vaše jméno>“ na adresu `listserv@urz.Uni.Heidelberg.de`.

296. Užití TeXu k sazbě z SGML souborů

Jonathan Fine (`J.Fine@pmms.cam.ac.uk`) vytváří software, který bude umožňovat TeXu sázet přímo ze souborů SGML. Výsledkem práce by měl být balík maker, jenž, oproti dříve zmíněným konverzním programům (viz Otázku 78), interpretuje přímo zdrojový SGML soubor.

297. Projekt Omega

Omega (Ω) je program vycházející z TeXu, který pracuje s „širokou“ vnitřní reprezentací znaků (podporuje plný Unicode verze 3). To umožňuje pracovat s většinou písem světa bez komplikací s jejich kódováním. Omega má také velmi mocný koncept vstupních a výstupních filtrů, které uživateli umožňují pracovat s existujícími překódovávacími tabulkami, atd.

Dostupná je také emailová diskusní skupina: pro přihlašení pošlete spávu „`subscribe`“ na `omega-request@omega.cse.unsw.edu.au`

Omega byla poprvé vydána v listopadu 1996 půdovními tvůrci projektu (John Plaice a Yannis Haralambous), poslední verze je udržována na CTANu. Omega je v současnosti open source projektem, detaily *cvs* repositáře a jiné informace jsou na webovské stránce projektu (`http://omega.cse.unsw.edu.au`).

Implementace Omegy je dostupná jako část distribucí teTeX, mikTeX, fpTeX a CMacTeX (viz Otázku 61). Je také distribuována jako součást TeX Live CD-ROM (viz Otázku 60). Je též dostupná na CTANu v adresáři `systems/omega`.

Domovská stránka projektu Omega je `http://omega.cse.unsw.edu.au:8080`

298. Projekt $\mathcal{NTS}$

Projekt $\mathcal{NTS}$ (New Typesetting System) poprvé spatřil světlo světa na Hamburgském setkání DANTE v roce 1992 jako odpověď na úsilí vytvořit něco lepšího, než je TeX. Cílem projektu není jen pouhé rozšíření TeXu, neboť, za prvé, vývoj TeXu byl Knuthem zastaven (viz Otázku 18), a za druhé, i kdyby bylo dovoleno dále vyvíjet program TeX, někteří členové týmu $\mathcal{NTS}$ cítí, že se program ve své současné formě k dalšímu vývoji prostě nehodí. Ačkoli všichni účastníci projektu jsou úzce spojeni a svázáni s TeXem, docházejí k závěru, že konečný produkt bude mít s TeXem, kromě filosofie, jen velmi málo společného.

Nejprve, navzdory výhradám vyjádřeným při zahajovacím setkání, se skupina zaměřuje na rozšíření TeXu. Úpravy a rozšíření se implementují do TeXu za pomoci standardního změnového souboru (change-file). Tato rozšíření spolu s opravdovým TEXem, vytvoří systém nazvaný ε-TEX, který by měl být 100% kompatibilní s TEXem. Nadto bude možné vytvořit formát, jenž bude TEXem *bez* veškerých úprav a rozšíření.

Nejnovější základní zdroj ε-TEXu (t.j. change-file) je dostupný na CTANu. Implementace ε-TEXu jsou také distribuovány na TEX Live CD-ROMu (viz Otázku 60) a s většinou moderních volně dostupných distribucí TEXu.

Projekt vytvořil β-verzi TEXu napsanou (od začátka) v Javě. Jelikož to *není* TEX (v drobnostech je mírně nekompatibilní), je program znám jako $\mathcal{NTS}$. Jak by se dalo čekat, první reimplementace běží celkem pomalu, ale jeho operace byla veřejně demonstrována a β-verze je dostupna na CTANu.
`e-TeX`: `systems/e-tex`
`NTS`: `systems/nts`

299. Projekt PDFTEX

PDFTEX (formálně známý jako TEX2PDF) vznikl z projektu Hán Thé Thánha, postgraduálního studenta Fakulty informatiky Masarykovy univerzity v Brně. Základní idea je velice jednoduchá: vytvořit verzi TEXu, která by na výstup dala `PDF` soubor místo souboru `DVI`. PDFTEX implementuje velice málo nových primitiv pro převedení do PDF výstupu a pro kontrolu rysů PDF dokumentu. Hán Thé Thán vytvořil PDFTEX v rámci svého postgaduálního výzkumu v oblasti počítačové sazby. Poslední verze obsahuje i podporu pro sazbu neotřelých typografických technik.

Jelikož Han už dokončil své studium, měl málo času pro práci na PDFTEXu, podpora je poskytována tímem expertů (většinou z Evropy).

Poslední verze PDFTEXu je dostupná v `systems/pdftex` a jeho různé implementace jsou dodávány jako součast teTEXu, mikTEXu, fpTEXu a CMacTEXu (viz Otázku 60). Verze vytvořená pro OzTEX je také dostupná v `nonfree/systems/mac/pdftex/pdftex_for_oztex.sit.bin`.

Více se o PDFTEXu můžete dozvědět prostřednictvím mailing-listu. Pro přihlášení stačí pouze poslat zprávu obsahující text „subscribe pdftex“ na adresu `majordomo@tug.org` (budete požádáni, abyste vaši registraci potvrdili).

300. Budúce webové technológie a TEX

Otázka 79 adresuje problémy konverzie existujúcich TEXovských dokumentov pre prehliadanie na webe ako HTML. Všetky prítomné techniky sú tak trochu chybné: odpoveď vysvetlí prečo.

Mnohé veci sa však menia s lepšou dostupnosťou fontov, „prefíkaným“ HTML programovaním a podporou pre nové webové štandardy.

Technológie fontov Priama reprezentácia matematiky v browseroch bola doposiaľ obmedzovaná limitovaným rozsahom symbolov vo fontoch, na ktorých dostupnosť sa môžete spoľahnúť. V budúcnosti môžeme očakávať širokú dostupnosť Unicode fontov s lepším pokrytím symbolov.

XML Jadrom rozsahu nových štandardov je XML poskytujúce možnosti pre lepšie štruktúrované značkovanie, obmedzená sa podpora už objavila v niektorých browseroch.
Konverzia TEXovského/LATEXovského zdrojového súboru do XML je už dostupná (aspoň prostredníctvom TEX4ht), práce v tejto oblasti pokračujú. Alternatívna technika, vytváranie XML (teda produkovanie webu priateľských dokumentov) a použitie TEXu/LATEXu na sadzbu, je tiež celkom pokročilá. Jedna užitočná technika je *transformácia* XML do LATEXu pomocou XSLT a následne jednoduché použitie LATEXu. Ako alternatívu môžete vyskúšať sadzbu priamo z XML (viď Otázku 89).

Priama reprezentácia matematiky MathML je štandard pre reprezentáciu matematiky na webe. Jeho pôvodná verzia je znateľne obmedzená, ale snahy o jej obohatenie sú na ceste. Podpora browserov pre MathML (napríklad v programoch *amaya*, „open-source“ verzia netscape browseru *mozilla* a v špeciálne rozšírených verziách *Internet Explorer*u) sa postupne stáva dostupnou. Existujú podklady, že používaelia TEXu začínajú tieto browsery používať. Práca na vytváranie MathML v projektoch TEX4ht a *TtH* je celkom pokročilá.

Grafika SVG je štandard pre prezentáciu grafiky na webe. Hoci prirodzeným použitím je kon-

verzia existujúcich obrázkov, reprezentácia rovníc je tiež možná, (namiesto bitových máp používaných v minulosti a zatiaľ, čo čakáme na široké nasadenie MathML).
Zásuvné moduly pre browsery, ktoré spracúvajú SVG, sú už dostupné (jeden ponúka napríklad Adobe).

301. Projekt TEXtrace

TEXtrace je sada unixovských skriptov, ktoré používajú freeware balík na trasovanie okrajov, aby generovali Type 1 obrysové fonty z METAFONTovského bitmapového výstupu. Je nepravdepodobné, že výsledok bude dosahovať kvalít komerčne vytvoreného Type 1 fontu, ale stále zostávajú fonty, ktoré mnohí ľudia pokladajú za užitočné, ale vôbec nelákajú platených odborníkov.

Projekt začal Péter Szabó a jeho aktuálny stav je dostupný prostredníctvom stránky projektu na SourceForge. Fonty na CTANe hodné vašej pozornosti generované pomocou TEXtrace (okrem vlastných EC/TC fontových sád autora) sú sada CM-Super Vladimira Volovicha pokrývajúca fontové sady EC, TC, a Cyrilic LH, a sada MusixTEX fontov Takanoriho Uchiyamu.

Ďalším systémom inšpirovaným TEXtrace je *mftrace*: je to malý program v *Python*e vykonávajúci rovnakú prácu; oba systémy sú čoraz viac používané na poskytovanie public domain Type 1 fontov.

`CM-Super fonty`: `fonts/ps-type1/cm-super`
`mftrace`: `http://www.cs.uu.nl/~hanwen/mftrace/`
`\TeX{}trace`: `http://sourceforge.net/projects/textrace/`
`Type 1 verzie EC a TC fontov`: `fonts/ps-type1/ec`
`musixtex fonty`: `fonts/musixtex/ps-type1/musixps-unix.tar.gz`

302. TEXovské prostredie prípravy dokumentov

Otázka 19 popisuje dôvody (alebo ospravedlnenia) veľkého rozdielu užívateľských rozhraní medzi „typickými“ TEXovkými prostrediami a komerčnými textovými procesormi.

V súčasnosti konečne existuje množstvo nástrojov pokúšajúcich sa túto „dieru“ premostiť alebo zmenšiť. Jedna časť sa jednoducho zameriava na poskytovanie používateľovi čitateľných zdrojových dokumentov. Na druhej strane máme *TeXmacs* (`http://www.texmacs.org/`), dokumentový procesor používajúci algoritmy a fonty TEXu na zobrazovanie editoru aj tlač. *TeXmacs* nepoužíva jazyk TEX samotný (hoci medzi inými formátmi je podporovaný export a import LATEXu). Trochu bližšie k LATEX je LyX (`http://www.lyx.org/`), ktorý má vlastné zobrazovanie editoru a aj formáty súborov, ale tlačený výstup robí exportom do LATEXu. Zobrazenie editoru vyzerá ako tlačený výstup, ale máte možnosť vkladať vlastný LATEXovský kód. Ak používate konštrukcie, ktorým LyX nerozumie, zobrazí ich jednoducho ako zdrojový text označený červenou farbou, ale exportuje ich správne.

Kedže na napísanie editoru od začiatku tak, aby bol skutočne dobrým editorom (ako aj obsluhou TEXu), nie je náhodou, že mnohé prístupy boli implementované pomocou rozšíriteľného editoru *emacs*. Jednou zo skrášľujúcich vecí je zvýrazňovanie syntaxe: označovanie TEXovských tokenov, komentárov a iných vecí špeciálnymi farbami. Mnoho voľne dostupných editorov (vrátane *emacs*u) vedie TEX týmto spôsobom „obslúžiť“. Pod Windows je jedným z populárnejších editorov s takouto podporou shareware produkt *winedt* (`http://www.winedt.com/`). Celkom dobrou freewarovou alternatívou na platforme Windows môže byť editor *PSPad* (`http://www.pspad.com/`) podporujúci okrem TEXu množstvo iných jazykov. Medzi skrášľujúce nástroje môžeme zaradiť aj *emacs*ovský balík *x-symbol* (`http://x-symbol.sourceforge.net`), ktorý WYSIWYG časť svojej práce uskutočňuje nahrádzaním jednotlivých TEXovských tokenov a akcentovaných sekvencií písmen za príslušne vyzerajúce znaky na obrazovke.

Iný druh nástrojov sa zameriava na aktualizáciu a skorší prístup k náhľadom sádzaného dokumentu. Nedávnym prídavkom v mnohých prehliadačoch, editoroch a TEXovských spustiteľných súboroch sú tzv. „source specials“ pre krížovú navigáciu. Keď TEX zostavuje dokument, na požiadanie vloží do vysádzaného výstupu pre každý riadok vstupu špeciálne značky. Značky sú interpretované DVI prehliadačom, ktorý môže byť nastavený, aby udržoval súvislosť medzi zobrazovanou stranou a príslušnou pozíciou vstupného súboru v editore, alebo nechať kurzor editoru skočiť na príslušný riadok podľa kliknutia do okna náhľadu.

*emacs*ovský balík kombinujúci tento druh presunov v editore spolu s automatickou rýchlou rekompiláciou je *WhizzyTeX* (`http://pauillac.inria.fr/whizzytex/`), ktorý sa najlepšie používa s prehliadačom od rovnakého autora. Jenoduchší balík nesúci sa v podobnom duchu je *InstantPreview* (`http://www.activetex.org/`) využívajúci neustále bežiacu kópiu TEXu (bežiaceho pod názvom `TeX daemon`) aby dosiahol rýchleho spracovania.

Ďalší *emacs*ovský balík nazývajúci sa *preview-latex* (`http://preview-latex.sourceforge.net`) sa pokúša vyriešiť problém vizuálnej korelácie medzi zdrojom a náhľadom priamejším spôsobom: používa LATEXovský balík na nasekanie zdrojového dokumentu na zaujímavé fragmenty (obrázky, text, matematika, ...), ktoré nechá prebehnúť LATEXom a vymení zdrojový text týchto fragmentov za príslušné vykreslené výstupné obrázky. Kedže nepozná štruktúru obrázkov na aktuálnej pozícii kurzoru, pri editovaní je zobrazený zdrojový text (namiesto náhľadu). Tento prístup je viacmenej hybridom prístupov skrášľovania zdrojového textu a rýchleho náhľadu (kedže pracuje v zdrojovom bufferi, ale používa náhľady vykreslené LATEXom).

Ambicióznejším kandidátom je TEXlite. Tento systém je dostupný iba po požiadaní jeho autora; nepretržite aktualizuje obrazovku pomocou špeciálnej verzie TEXu zhromažďujúcej svoj stav do komprimovaného formátu na každej strane a používajúceho „háky“ (hooks) do TEXovského mechanizmu zalamovania riadkov na preformátovávanie odstavcom „za behu“. Týmto spôsobom môže vykresliť výstup editovaného TEXovského kódu interaktívne na obrazovku a poskytuje možnosť editovania priamo v okne náhľadu.

Takéto množstvo týchto mierne odlišných systémov môžete vidieť porovnaním riešení založených na *emacs*e (od zvýrazňovania syntaxe až po okamžité náhľady): všetky môžu byť aktivované naraz bez toho, aby si do svojich úloh vzájomne zasahovali.

Rozličné prístupy poskytujú rozličné možnosti líšiace sa rýchlosťou odozvy, oblasťou obrazovky, v ktorej pracujú (zdrojové alebo zvláštne okno), stupňom vernosti zobrazovaného voči výstupu a vyváženosťou medzi vizuálnou pomocou a vizuálnym rozptylovaním pozornosti.

`preview-latex`: `support/preview-latex`
`texmacs`: `systems/unix/TeXmacs`

U. Možná zde *není* odpověď

303. Co dělat, když najdete chybu

Nejprve bychom si měli být jisti, že jsme *opravdu* našli chybu. Dvakrát se přesvědčme v knize o TEXu, LATEXu (či o tom formátu, který používáme), zda tam není napsáno něco o chování, které pozorujeme. Srovnejme to, co vidíme, s odpověďmi, jež jsou uvedeny výše, zeptejme se všech lidí, o nichž víme, že mají zkušenosti s TEXem. Důvodů k takové opatrnosti je mnoho.

Nalezne-li někdo chybu v samotném TEXu, potom se zřejmě jedná o zcela výjimečnou osobu. Don Knuth si je tak jistý kvalitou svého programu, že nabízí nálezci chyby peněžitou odměnu; šeky však vyplňuje velice zřídka. Nalezne-li se opravdová chyba v samotném TEXu (METAFONTu, v CM fontech, v TEXbooku), není dobré psát ihned Knuthovi, jelikož on se na chyby dívá tak jednou, dvakrát do roka, a i v těchto chvílích se zabývá chybami, které mu doporučí tým jeho spolupracovníků. Nejprve je dobré kontaktovat Barbaru Beeton z AMS (`bnb@math.ams.org`), anebo skupinu uživatelů TEXu (TUG).

Jestliže je nalezena chyba v LATEXu 2_ε měla by být postoupena týmu LATEX3. K sepsání zprávy o chybě slouží soubor `latexbug.tex`, který je součástí distribuce LATEX 2_ε. Tento soubor necháme zpracovat LATEXem. Během zpracování jsme instruováni o tom, co je třeba dělat.

Prosíme, šetřte časem této skupiny, která se snaží pracovat pro dobro celé LATEXovské pospolitosti. Veškerý čas, který tito lidé stráví analýzou domnělých chyb, pak chybí při psaní a ladění nových maker.

Nalezne-li se chyba v LATEX2.09, nebo v nějakém jiném neudržovaném softwaru, nelze toho udělat mnoho. Pomoc (podporu tohoto softwaru) lze nalézt v newsgroup jako například `comp.text.tex` nebo v diskusních listech `texhax@tex.ac.uk`, ale prezentací domnělých chyb na těchto fórech se vystavujete možnému posměchu! Jinak je potřeba si najít ochotného konsultanta[8] v otázkách spojených s TEXem (viz `http://www.tug.org/consultants.html`).

V. Lokalizace pro češtinu a slovenštinu, CSTUG

304. Co je to CSTUG

CSTUG je občanské sdružení uživatelů TEXu. Sdružuje individuální a kolektivní členy a jeho oficiální název je „Československé sdružení uživatelů TEXu". Členství je dobrovolné. Stanovy sdružení jsou dostupné na `ftp://ftp.cstug.cz/pub/tex/local/cstug/info/`

305. Jaká je adresa a kontakt na sdružení CSTUG

Adresa sdružení (sídlo) je CSTUG, c/o FEL ČVUT, Technická 2, 166 27 Praha 6. Sem můžete posílat korespondenci týkající se CSTUGu. Kontaktní email je `cstug@cstug.cz` nebo `secretary@cstug.cz`.

306. Jaké mám výhody z členství v CSTUGu

Každý individuální člen se svým členstvím hlásí k poslání sdružení, které je specifikováno ve stanovách sdružení. Má právo dostávat občasník zvaný Zpravodaj, který vychází zpravidla alespoň čtyřikrát do roka.

Členové mají slevy na publikace vydávané nebo dotované sdružením, slevy na akcích a školeních pořádaných sdružením.

307. Jak se stanu členem CSTUGu

Přihlásit se můžete pomocí formuláře na `http://www.cstug.cz/clenstvi/reg.html`. Vyplňte, prosím, zejména všechny údaje, které jsou ve formuláři označeny jako „povinné". Jejich vyplnění je nezbytné k tomu, abychom dokázali identifikovat Vaše platby a abychom znali adresu, na niž máme zasílat Zpravodaj a jiné materiály, na které máte jako člen nárok.

Přihlásit se můžete také písemně e-mailem nebo pozemní poštou na adrese sdružení. Je třeba sdělit jméno a příjmení, rodné číslo (pro identifikaci plateb), adresu pro korespondenci, telefon, adresu a telefon do zaměstnání a email.

Formulář přihlášky k vytisknutí na papír je také v `ftp://ftp.cstug.cz/pub/local/cstug/info/`.

Informaci o členských příspěvcích a způsobu jejich placení lze nalézt na `http://www.cstug.cz/clenstvi/index.html#prispevky`

308. Jak si objednám CD-ROM TEXLive či 4AllTEX

Závazné objednávky posílejte na adresu sdružení nebo `orders@cstug.cz`.

309. Jak si objednám publikace vydávané sdružením nebo další TEXovou literaturu

Objednávejte na adrese knihkupectví Mareček, Botanická 68a, 602 00 Brno, `bookorders@cstug.cz`. Pokud uplatňujete členskou slevu, přiložte kopii dokladu o zaplacení členského příspěvku CSTUGu na kalendářní rok. Je zde možno objednávat i zahraniční TEXovou literaturu a stará čísla Zpravodaje, pokud nejsou rozebraná.

[8] TUG udržuje registr konsultantů a UKTUG vytváří další.

310. Co to je Zpravodaj a jak ho získám

Zpravodaj je tištěný občasník sdružení. Zpravidla vycházejí 4 čísla ročně. Zpravodaj je rozesílán řádným členům sdružení, kolektivní členové dostávají po pěti výtiscích od každého čísla.

311. Jak mám postupovat, když chci napsat článek do Zpravodaje?

Korespondence týkající se Zpravodaje se posílá na adresu `zpravodaj@cstug.cz`. Příspěvky ukládejte do adresáře `ftp://ftp.icpf.cas.cz/wagner/incoming/`. Uvedený adresář je uvolněn pro zápis. Pokud nemáte přístup na Internet, můžete zaslat příspěvek na disketě na adresu Zdeněk Wagner, Vinohradská 114, 130 00 Praha 3. Disketa musí být naformátována pro DOS. Nezapomeňte přiložit všechny soubory, které dokument načítá.

Pro sazbu Zpravodaje byl vytvořen a je udržován a zdokumentován styl `csbul.sty` z `http://hroch486.icpf.cas.cz/bulletin/csbuldoc.zip`. Pro usnadnění finální sazby editoři uvítají, když jej autoři budou pro psaní svých příspěvků používat. Kopie je zrcadlena v adresáři `ftp://ftp.cstug.cz/pub/tex/local/cstug/zpravodaj/`

312. Kde je nejbližší zrcadlo archívu CTAN

Pro uživatele v České republice je nejbližší kopie CTANu v Brně na adrese `ftp://ftp.cstug.cz/pub/tex/CTAN/`. Na tomto serveru je také lokální archív CSTUGu — `ftp://ftp.cstug.cz/pub/tex/local/cstug/`.

Uživatelé ze západního Slovenska mají nejblíže na bratislavský server SAV `http://www.savba.sk/cstug/`, kam je zrcadlen brněnský archív.

Nejbližší zrcadlo CTANu na východě Slovenské republiky je na universitě v Košicích na `ftp://oko.fei.tuke.sk/pub/cstex/`.

Pro střední Slovensko je nejbližší zrcadlo CTANu v Banské Bystrici na `ftp://sunsite.uakom.sk/pub/mirrors/CTAN/`.