OPmac – rozšiřující makra plainTEXu

Petr Olšák, 2012 – 2019

http://petr.olsak.net/opmac.html

Obsah

Úvod 1
1 Výběr fontové rodiny 2
2 Velikosti fontů a řádkování 3
3 Členění dokumentu 4
4 Další číslované objekty a odkazy na ně 4
5 Odrážky 6
6 Tvorba automaticky generovaného obsahu 7
7 Sestavení rejstříku 8
8 Barvy, vodoznaky 9
9 Klikací odkazy 10
10 Verbatim texty 11
11 Jednoduché tabulky 12
12 Vkládání obrázků 14
13 PDF transformace 15
14 Poznámky pod čarou a na okraji 15
15 Bibliografické údaje 16
16 Matematická sazba 19
17 Okraje 20
18 Poslední strana 21
19 Další jazyk 21
20 Předdefinované styly dokumentů 22
21 Shrnutí 23

Úvod

OPmac je balík jednoduchých doplňujících maker k plainTEXu umožňující uživatelům základní LATEXovou funkcionalitu: změny velikosti písma, automatickou tvorbu obsahu a rejstříku, práci s `bib` databázemi, referencemi, možnost proložení referencí hyperlinkovými odkazy atd.

Hlavní zásady balíku OPmac jsou:

- V jednoduchosti je síla.
- Makra nejsou univerzální, ale jsou čitelná a srozumitelná.
- Uživatel si makra může snadno předefinovat k obrazu svému.

Každé makro je napsáno s cílem co největší srozumitelnosti pro lidi, kteří to budou chtít číst a měnit. Troufám si říci, že balík nabízí čtenáři inspiraci, jak se programují TEXová makra. Z kódu maker je cítit jistá elegance. Technická část dokumentace k OPmac by tedy mohla sloužit jako učebnice programování TEXových maker. To je zásadní rozdíl od koncepce LATEXu. Když se člověk podívá do LATEXového souboru `latex.ltx`, vystřeví se na něj množství zavináčů a makra, ze kterých je často cítit topornost a mnohdy nepochopení vnitřní koncepce TEXu. Skoro nikdo se v tom nevyzná. Soubor `latex.ltx` obsahuje 8000 řádků a schopnosti LATEXu jsou navíc

ukryty v desítkách různých dalších makro souborech, zatímco v OPmac vidíte vše pohromadě a názorně. Navíc v některých věcech OPmac výrazně překračuje možnosti LaTeXu: generování rejstříků bez externího programu, přímé čtení `*.bib` souborů bez bibTeXu, listingy externích souborů.

Balík OPmac nabízí podobně jako LaTeX autorům textů *rozhraní*, tj. smluvenou sadu značek na vymezení struktury dokumentu. Je jiná, než v LaTeXu, možná nabídne napsat zdrojový text článku poněkud přehledněji a oku více lahodícím způsobem. Balík OPmac ovšem neřeší typografický vzhled dokumentu. Bez doplňujících maker vyleze jednoduchý střízlivý dokument. Předpokládá se, že autor dodatečných plainTeXových maker ušije vzhled dokumentu na míru konkrétnímu požadavku.

Příklad začátku dokumentu:

```
\input opmac          % zavedení makra OPmac
\chyph                % zapnutí češtiny
\fontfam [LM fonts] % použití Latin Modern fontů
\typosize[12/14]    % nastavení základní velikosti sazby
```

Makro OPmac spolupracuje s běžnými plainTeXovými formáty: s Knuthovým klasickým plainTeXem nebo s $\mathcal{CS}$plainem nebo s plainTeXem doplněným makry z `etex.src` (ten je v běžných TeXových distribucích základním formátem pdfTeXu, XeTeXu a LuaTeXu).

Uživatelům OPmac nabízím konzultace po emailu a uvítám hlášení o chybách. Výsledky konzultací typicky zveřejňuji na http://petr.olsak.net/opmac-tricks.html, kde najdete desítky nejrůznějších řešení dílčích problémů.

1 Výběr fontové rodiny

OPmac implicitně rodinu fontů nenastavuje, tj. fonty jsou připraveny stejné jako v plainTeXu (CM fonty) nebo v $\mathcal{CS}$plainu (CS fonty). Je ale možné použít tzv. „fontové soubory“ pro zavedení jiné fontové rodiny, tj. typicky čtyř základních variant `\rm`, `\bf`, `\it` a `\bi`. Tyto soubory interně používají primitivní příkaz `\font` pro zavedení jednotlivých fontů.

Nemusíte si pamatovat název souboru, který je potřeba pro zavedení fontové rodiny použít. Stačí napsat `\fontfam[`⟨*NázevRodiny*⟩`]` a požadovaný soubor se načte. V argumentu ⟨*NázevRodiny*⟩ nezáleží na mezerách a velkých písmenech, takže `\fontfam[Times Roman]` je totéž jako `\fontfam[TimesRoman]` i jako `\fontfam[timesroman]`. Také je připraveno několik běžných aliasů, takže třeba `\fontfam[times]` rovněž zavede rodinu Times Roman.

Použijete-li `\fontfam[?]`, zobrazí se na terminálu a v log souboru přehled všech fontových rodin dostupných pomocí fontových souborů. Část seznamu vypadá takto:

```
[LM Fonts]  {\caps \sans \ttset ...} {\rm \bf \it \bi } +AMS (8z 8t U)
[TG Heros]  {\caps \cond } {\rm \bf \it \bi } +TX (8z 8t)
```

Nejprve je uveden ⟨*NázevRodiny*⟩, pak následuje seznam modifikátorů základních selektorů následovaný základními selektory pro výběr variant v dané rodině, pak za znakem plus je uvedena implicitní matematická sada fontů užitá k vybrané rodině a konečně v kulaté závorce je seznam kódování, které rodina fontů podporuje. Více informací o makru `\fontfam` lze najít v souboru `fontfam.tex`.

Fonty přepínáme pomocí základních selektorů (`\rm`, `\bf`, `\it`, `\bi`). Bezprostředně před základním selektorem mohou předcházet *modifikátory* které dále pozměňují variantu písma. Například `\caps\it` zapne malé kapitálky v kurzívě nebo `\cond\caps\bf` zapne kondenzovanou modifikaci s malými kapitálkami tučné varianty. Modifikátory následované příkazem `\fam` (např. `\caps\fam`) modifikují všechny později použité základní selektory (uvnitř stejné skupiny). Modifikátory následované sekvencí `\one` (např. `\caps\one`) modifikují jen aktuálně užívanou variantu písma. Modifikátory pracují typicky nezávisle na sobě a každá rodina fontů může disponovat jinou sadou modifikátorů. Více se o nich píše v souboru `cs-heros.tex` nebo v článku `kpfonts-plain.pdf`.

Použijete-li `\fontfam[Catalog]`, vytiskne se katalog dostupných fontových rodin.

2 Velikosti fontů a řádkování

Všechna makra popsaná v této i předchozí sekci nastavují změny ve fontech a dalších parametrech jen lokálně, takže jsou-li ve skupině, za ní se nastavení vrací k původním hodnotám.

Makro `\typosize[`⟨*velikost fontu*⟩`/`⟨*řádkování*⟩`]` nastaví velikost textových i matematických fontů a řádkování. Je-li některý z parametrů prázdný, makro nastaví jen údaje plynoucí z neprázdného parametru. Parametry neobsahují jednotku, jednotka pt se doplní v makru. Příklady

```
\typosize[10/12]    % to je implicitní nastavení
\typosize[11/12.5]  % font velikosti 11pt, řádkování 12.5pt
\typosize[8/]       % font velikosti 8pt, řádkování nezměněno.
```

Začátek dokumentu tedy může vypadat takto:

```
\input opmac \typosize[11.5/13]  % sazba v písmu 11.5pt s řádkováním 13pt
```

Makro `\typoscale[`⟨*faktor-font*⟩`/`⟨*faktor-řádkování*⟩`]` zvětší nebo zmenší velikost textových i matematických fontů resp. řádkování ⟨*faktor*⟩krát aktuální velikost fontů resp. řádkování. Faktor je celé číslo, přitom 1000 znamená faktor jedna ku jedné (jako za slovem `scaled` v příkazu `\font`). Je-li parametr prázdný, je to stejné, jako by byl roven 1000.

```
\typoscale[800/800]    % fonty i řádkování se zmenší na 80 %
\typoscale[\magstep2/] % \magstep2 je 1440, tj. fonty se zvětší 1,44krát
```

Někdy je žádoucí (např. při přechodu na poznámky pod čarou) zmenšit font vzhledem ke stále stejné velikosti písma. Stačí psát `\typobase\typoscale[`⟨*font*⟩`/`⟨*řádkování*⟩`]`. Pak se provede zvětšení/zmenšení vzhledem k *základnímu písmu*, což je písmo nastavené po prvním použití `\typosize` nebo `\typoscale`.

Pokud zavedete font příkazem `\font\prepinac=`⟨*metrika*⟩ ⟨*nic-nebo-at-nebo-scaled*⟩, pak `\prepinac` přepíná do pevně stanoveného fontu, který není ovlivněn makry na nastavování velikosti. Ale co není, může být. Stačí font registrovat pomocí `\regfont\prepinac` a nyní i `\prepinac` přepíná do fontu podle velikosti nastavené pomocí `\typosize` nebo `\typoscale`. Příklad:

```
\font\zapfchan=pzcmi8z  \regfont\zapfchan
\typosize[20/] Taky \zapfchan přepne do Zapf-Chancery ve velikosti 20pt.
```

OPmac registruje pět fontových přepínačů ukrytých v makrech `\rm`, `\it`, `\bf`, `\bi`, `\tt`. Takže tato makra implicitně nastavují font do stanovené velikosti.

Na místo `\typosize` a `\typoscale` je možno použít makra na změnu velikosti jen aktuálního fontu `\thefontsize[`⟨*velikost-fontu*⟩`]` a `\thefontscale[`⟨*faktor*⟩`]`. Tato makra nemění matematické fonty ani řádkování.

Všechna zde uvedená makra na změnu velikosti fontů jsou vybavena inteligencí: hledají metriku, která má svou designovanou velikost nejblíže požadované velikosti. Takže při požadavku na velikost 13pt se použije metrika `csr12 at13pt`, zatímco při velikosti 7.5pt se použije metrika `csr8 at7.5pt`. Data pro tuto inteligenci jsou přečtena ze souboru `ams-math.tex`, kde je najdete u příkazů `\regtfm`.

Poslední poznámka k fontům se týká makra `\em`, které přepíná variantu písma specifickým způsobem. Je to kontextové makro, které pracuje v závislosti na aktuálně zvoleném fontu. Implicitně přepíná na `\it`. Pokud ale je aktuálním fontem `\it`, přepne na `\rm`. Je-li aktuálním fontem `\bf` přepne na `\bi` a obráceně. Makro navíc správně doplní italické korekce ke slovu před jeho použitím a za jeho použitím. Takže se o italické korekce není nutno starat. Příklad:

```
To je {\em zdůrazněný} text.     % jako: To je {\it zdůrazněný\/} text.
\it To je {\em zdůrazněný} text. % jako: To je\/ {\rm zdůrazněný} text.
\bf To je {\em zdůrazněný} text. % jako: To je {\bi zdůrazněný\/} text.
\bi To je {\em zdůrazněný} text. % jako: To je\/ {\bf zdůrazněný} text.
```

3 Členění dokumentu

Dokument se může skládat z kapitol, kapitola ze sekcí a sekce z podsekcí. Titul dokumentu vyznačte pomocí `\tit` ⟨*titul*⟩⟨*konec-řádku*⟩, kapitolu zahajte `\chap` ⟨*titul*⟩⟨*konec-řádku*⟩ a podobně novou sekci zahajte `\sec` ⟨*titul*⟩⟨*konec-řádku*⟩ a podsekci `\secc` ⟨*titul*⟩⟨*konec-řádku*⟩. Takže třeba:

```
\chap Brouci

\sec  Chrousti

\secc O nesmrtelnosti chroustů

Bla bla bla bla ...
Bla bla bla a ještě bla.
```

Kapitoly se automaticky číslují jedním číslem, sekce dvěma čísly (číslo kapitoly.sekce) a podsekce třemi čísly. Pokud dokument neobsahuje kapitoly, číslo kapitoly chybí, tj. sekce má jedno číslo a podsekce dvě.

Implicitní vzhled nadpisů kapitol, sekcí a podsekcí je definován v makrech `\printchap`, `\printsec` a `\printsecc`. Můžete se na obsah těchto maker podívat do technické dokumentace nebo do `opmac.tex`. Můžete se těmi makry inspirovat a třeba je předefinovat podle vlastního typografického návrhu.

První odstavec za titulem kapitoly, sekce a podsekce není odsazen. Pokud jej chcete mít odsazen jako ostatní odstavce, napište `\let\firstnoindent=\relax`.

Jestliže je název kapitoly, sekce nebo podsekce příliš dlouhý, rozlomí se do řádků. V takovém případě je někdy lepší rozdělit název do řádků manuálně. K tomu slouží makro `\nl`, které odřádkuje v místě použití (newline). Toto makro se navíc v obsahu chová jako mezera.

Kapitola, sekce, podsekce se nečísluje, předchází-li `\nonum`. Kapitola, sekce, podsekce se neobjeví v obsahu, předcházlí-li `\notoc`.

4 Další číslované objekty a odkazy na ně

Kromě kapitol, sekcí a podsekcí se automaticky číslují ještě rovnice a popisky pod obrázky a pod tabulkami.

Pokud je na konci display módu uvedeno `\eqmark`, tato rovnice bude číslovaná. Formát číslování je implicitně jediné číslo uzavřené v kulaté závorce resetované při každém zahájení nové sekce. Příklad: `$$ a^2 + b^2 = c^2 \eqmark $$` vytiskne

$$a^2 + b^2 = c^2 \eqmark$$

Je-li potřeba očíslovat jedním číslem více rovnic sestavených pomocí `\eqalignno`, pak použijte `\eqmark` v posledním sloupci, například takto:

```
$$
  \eqalignno{a^2+b^2 &= c^2 \cr
                   c &= \sqrt{a^2+b^2} & \eqmark \cr}
$$
```

Ukázka dává tento výsledek:

$$a^2 + b^2 = c^2$$
$$c = \sqrt{a^2 + b^2} \tag{2}$$

Dalšími číslovanými objekty jsou popisky. Popisek pod obrázky je potřeba uvést slovem `\caption/f` a popisek pod nebo nad tabulkami slovem `\caption/t`. Pak následuje text popisku ukončený prázdným řádkem. Příklad:

```
\hfil\table{rl}{věk   & hodnota \crl\noalign{\smallskip}
                0--1  & neměřitelná \cr
                1--6  & projevující se \cr
                6--12 & výrazná \cr
               12--20 & extrémní \cr
               20--60 & mírnější \cr
               60--$\infty$ & umírněná} % vytvoření tabulky
\par\nobreak\medskip
\caption/t Závislost závislosti na počítačích na věku.
```

Tato ukázka vytvoří:

věk	hodnota
0–1	neměřitelná
1–6	projevující se
6–12	výrazná
12–20	extrémní
20–60	mírnější
60–∞	umírněná

Tabulka 4.1 Závislost závislosti na počítačích na věku.

Vidíme, že makro `\caption/t` doplnilo slovo „Tabulka“ následované číslem. Toto číslo přebírá číslo sekce a za tečku doplňuje ještě číslo tabulky. Podobně se chová `\caption/f`, jen místo slova „Tabulka“ se v textu zjeví slovo „Obrázek“. Obrázky a tabulky jsou číslovány nezávisle. Popisek je centrován. Je-li popisek delší na více řádcích, je centrován poslední řádek.

Způsob číslování lze změnit jinou definicí makra `\thednum` (pro rovnici), `\thetnum` (pro tabulky) a `\thefnum` (pro obrázky). Makro OPmac je definuje implicitně takto:

```
\def\thednum{(\the\dnum)}
\def\thetnum{\thesecnum.\the\tnum}
\def\thefnum{\thesecnum.\the\fnum}
```

Makro OPmac vloží slovo „Tabulka“ v závislosti na nastaveném jazyce příkazem `\chyph`, `\shyph`, `\ehyph`. Při `\shyph` dostaneme „Tabuľka“ a při `\ehyph` máme „Table“. Podobně se chovají slova „Obrázek/Obrázok/Figure“ a „Kapitola/Kapitola/Chapter“. Jiná automaticky generovaná slova OPmac nepoužívá.

Předefinovat tato slova lze pomocí `\sdef`, jak ukazuje následující příklad, který zamění celá slova za zkratky.

```
\sdef{mt:t:cs}{Tab.}  \sdef{mt:t:sk}{Tab.}  \sdef{mt:t:en}{Tab.}
\sdef{mt:f:cs}{Obr.}  \sdef{mt:f:sk}{Obr.}  \sdef{mt:f:en}{Fig.}
```

LaTeXoví uživatelé jsou zvyklí, že jim tabulky a obrázky plavou v dokumentu, přičemž inklinují k horní části stránky. To se při použití OPmac implicitně neděje, ale je možno plavání zařídit pomocí plainTeXového makra `\topinsert` resp. `\midinsert`. Například:

```
\topinsert
   \hfil\table{rl}{...} % vytvoření tabulky
   \medskip
   \caption/t Závislost závislosti na počítačích na věku.
\endinsert
```

Na automaticky číslované objekty je nutno se občas v textu odkazovat. Protože dopředu nevíme, pod jakým číslem se rovnice, sekce, tabulka atd. vytiskne, je potřeba použít interní lejblíky k označení odkazovaných objektů. K tomu slouží makro `\label[`⟨*lejblík*⟩`]`, které musí předcházet makru, jež generuje číslo. Není nutné, aby `\label` předcházel těsně danému makru. Tedy například:

```
\label[chroust] \sec O nesmrtelnosti chroustů

\label[zavislaci]
\hfil\table{rl}{...} % vytvoření tabulky
\caption/t Závislost závislosti na počítačích na věku.

\label[pythagoras]
$$ a^2 + b^2 = c^2 \eqmark $$

Nyní můžeme hovořit o~sekci~\ref[chroust] na straně~\pgref[chroust] nebo
taky o~rovnici~\ref[pythagoras] na straně~\pgref[pythagoras]. Dále bude
potřeba upozornit na tabulku~\ref[zavislaci] na straně~\pgref[zavislaci],
která shrnuje jistý druh závislosti.
```

Text z ukázky vytvoří zhruba toto: „Nyní můžeme hovořit o sekci 2.1 na straně 13 nebo taky o rovnici (1) na straně 15. Dále bude potřeba upozornit na tabulku 5.3.1 na straně 42, která shrnuje jistý druh závislosti.“

Jestliže se v textu vyskytují dopředné reference (tj. odkazujeme na objekt, který ještě není vytištěn) nebo text odkazuje na stránky (`\pgref`), je nutné TEXovat dokument aspoň dvakrát.

Pomocí `\label[`⟨*lejblík*⟩`]\wlabel{`⟨*text*⟩`}` se dá vytvořit kdekoli obecný cíl ⟨*text*⟩, na který je možné odkazovat makry `\ref[`⟨*lejblík*⟩`]` nebo `\pgref[`⟨*lejblík*⟩`]`.

5 Odrážky

Jednotlivé myšlenky je občas potřeba vypíchnout odrážkami. Prostředí s odrážkami se vymezuje sekvencemi `\begitems` a `\enditems`. Uvnitř tohoto prostředí je hvězdička aktivním znakem, který zahajuje odrážky. Prostředí s odrážkami je možné vnořit do sebe. Pomocí `\style` ⟨*znak*⟩ hned za slovem `\begitems` je možné vymezit některé z předdefinovaných vzhledů odrážek:

```
\style o % malý puntík
\style O % velký puntík $\bullet$ (implicitní volba)
\style - % spojovník
\style n % odrážky číslované 1., 2., 3., ...
\style N % odrážky číslované 1), 2), 3), ...
\style i % odrážky číslované (i), (ii), (iii), (iv), ...
\style I % odrážky číslované I, II, III, IV, ...
\style a % odrážky s písmeny a), b), c), ...
\style A % odrážky s písmeny A), B), C), ...
\style x % malý čtvereček
\style X % velký čtvereček
```

Příklad:

```
\begitems \style n
* Tady je první myšlenka.
* A tady druhá, která je rozdělena na
  \begitems \style a
  * podmyšlenku
  * a hned následuje další podmyšlenka,
  * poslední podmyšlenka.
  \enditems
* Tady je třetí myšlenka.
\enditems
```

vytvoří následující výstup:

1. Tady je první myšlenka.
2. A tady druhá, která je rozdělena na

   a) podmyšlenku
   b) a hned následuje další podmyšlenka,
   c) poslední podmyšlenka.

3. Tady je třetí myšlenka.

Chcete-li uvnitř prostředí s odrážkami vytisknout hvězdičku, pište `\char‘\*`.

Pomocí `\sdef{item:`⟨*písmeno*⟩`}{`⟨*text*⟩`}` si můžete dodefinovat vzhled odrážek podle svých představ. Implicitní odrážku můžete předefinovat pomocí `\def\normalitem{`⟨*text*⟩`}`.

Jednotlivá prostředí s odrážkami se odsazují podle velikosti registru `\iindent`, který je nastaven na hodnotu `\parindent` v době čtení souboru `opmac.tex`. Pokud později změníte `\parindent`, doporučuji na stejnou hodnotu nastavit `\iindent`. Vertikální mezera nad a pod prostředím s odrážkami je řízena makrem `\iiskip`.

6 Tvorba automaticky generovaného obsahu

Makro `\maketoc` vytiskne v místě svého použití obsah dokumentu bez nadpisu, jen jednotlivé řádky obsahu. Odsazení jednotlivých řádků je nastaveno na násobky registru `\iindent`. Často je potřeba dokument TEXovat vícekrát, než se obsah zjeví a než se čísla stran srovnají správně, protože po prvním zjevení obsahu se mohou stránky posunout jinam.

Titulek k obsahu by neměl být číslovaný a neměl by se zjevit v obsahu, takže jej zapíšeme třeba pomocí

```
\nonum\notoc\sec Obsah
```

Titulky kapitol, sekcí a podsekcí zapisuje OPmac pro účely sestavení obsahu do externího souboru `*.ref`. Může se stát, že uživatel v těchto textech použije nějaké komplikované makro, které se pak v souboru „rozsype“ do takového stavu, že nejde vzápětí přečíst. V takovém případě je potřeba makro zabezpečit proti expanzi při zápisu do souboru pomocí deklarace `\addprotect\makro`. Takto deklarované makro je pak zabezpečené proti expanzi do `*.ref` souboru. Například OPmac deklaruje:

```
\addprotect~ \addprotect\TeX \addprotect\thefontsize \addprotect\em
```

a mnoho dalších. Není možné ale předvídat všechno, co může uživatel nacpat do titulku sekce nebo kapitoly. Pokud se tedy „rozsype“ REF soubor, je potřeba si tímto způsobem zabezpečit používané makro.

Poznámka: Při přechodu na novější verzi OPmac se může stát, že REF soubor vygenerovaný starou verzí způsobí chyby při dalším zpracování. Pak je potřebné nejprve REF soubor smazat.

7 Sestavení rejstříku

Makro pro zanášení slov do rejstříku je navrženo s ohledem na optimalizaci počtu úhozů na klávesnici. Autor už napsal své dílo, má daný termín odevzdání a nyní ho čeká úmorná práce vyhledávání slov v textu, která by měla přijít do rejstříku, a jejich vyznačování. Je třeba mu tuto práci co nejvíce usnadnit.

Pro zanesení slova do rejstříku slouží makro `\ii`. Je to zkratka za „insert to index“. Jeho parametr je ⟨*slovo*⟩ bez mezery ukončené mezerou (obecnější tvar parametru uvedeme později). Toto slovo se přepíše do rejstříku, ve kterém jsou všechna takto deklarovaná slova seřazena podle abecedy a jsou k nim připojena čísla stránek, na kterých bylo použito odpovídající makro `\ii` ⟨*slovo*⟩. Příklad:

```
Tady mluvím o jistém
\ii kroutilík
kroutilíku, který provokoval moji zvědavost.
```

Makro `\ii` viditelně neudělá v sazbě nic. Přilepí se na následující slovo (v našem příkladě slovo „kroutilíku“) jako skrytá značka. Číslo strany, kde se ta značka objeví, bude v rejstříku vedle slova „kroutilík“.

Je-li `\ii` zapsáno ve vertikálním módu, zahájí se v daném místě odstavec, aby se mohla neviditelná značka z `\ii` nalepit na následující slovo. Pokud si to z nějakých důvodů nepřejete, použijte interní variantu makra `\iindex{`⟨*slovo*⟩`}`, která nezahajuje odstavec.

Pokud se v rejstříku má objevit stejné slovo jako v textu, není nutno je psát dvakrát. Stačí použít makro `\iid` (zkratka za `\ii` double):

```
Hlavní zásady jsou \iid nestrannost , \iid pravdomluvnost a \iid odvaha .
```

To povede ke stejnému výsledku jako

```
Hlavní zásady jsou \ii nestrannost nestrannost,
\ii pravdomluvnost pravdomluvnost a \ii odvaha odvaha.
```

Povšimněte si, že čárky a tečky jsou odstrčeny od dublovaného slova, protože mezera je ukončovací znak parametru `\iid`. Do textu se mezera vrátí právě tehdy, když nenásleduje tečka nebo čárka. V našem příkladě před spojkou „a“ mezera ve výsledku je, ale před tečkou nebo čárkou mezera není.

Vlastnosti makra `\iid` jsou tímto popsány zcela. Vraťme se k makru `\ii`, které poskytuje další možnosti.

Parametr `\ii` je vždy ukončen mezerou. Může obsahovat čárky (bez mezer), které naznačují, že se do rejstříku dává více slov:

```
{\bf Definice.}
\ii lineární~prostor,vektorový~prostor
{\em Lineárním prostorem} (nebo též vektorovým prostorem) rozumíme ...
```

Dostaneme totéž jako při `\ii lineární~prostor \ii vektorový~prostor` a tato ukázka demonstruje ještě jednu věc: je-li potřeba do parametru `\ii` dostat mezeru, pište vlnku nebo napište heslo uzavřené ve svorkách.

Pokud se v rejstříku objeví hesla skládající se z více slov, obvykle chceme, aby u hesla, které opakuje první slovo, se toto slovo v rejstříku nevypisovalo opakovaně, ale aby bylo nahrazeno pomlčkou. Například:

lineární podprostor 12, 16, 18, 29
— prostor 12, 16–32, 51
— závislost 18–20, 34

Při takovém požadavku pište místo vlnky mezi slovy lomítko. Příklad:

```
\ii lineární/prostor,vektorový/prostor
```

Někdy je vhodné kromě hesla lineární/prostor zařadit i heslo prostor/lineární. Aby se to nemuselo psát dvakrát, je k dispozici zkratka `@` napsaná za čárku na konci parametru:

```
\ii lineární/prostor,vektorový/prostor,@
% je totéž jako \ii lineární/prostor,vektorový/prostor
%               \ii prostor/lineární,prostor/vektorový
```

Počet lomítek v hesle pro rejstřík není omezen. Můžete tedy vytvořit víceúrovňový rejstřík. Nicméně je třeba vědět, že zkratka `@` nevytváří všechny permutace, ale jen přesune první údaj před lomítkem za všechny ostatní. Takže `\ii a/b/c,@` je totéž jako `\ii a/b/c \ii b/c/a`.

Samotný rejstřík vznikne v místě příkazu `\makeindex`. Rejstřík obsahuje data z předchozího zpracování dokumentu TEXem, takže je potřeba TEXovat aspoň dvakrát. Makro `\makeindex` abecedně seřadí data v rejstříku podle českých a slovenských pravidel řazení a upraví odkazy na stránky (aby se stránky neopakovaly a inklinovaly k zápisu ve tvaru 26–28). Makro `\makeindex` se nestará o prostředí, do kterého sazbu vyvrhne, ani o nadpis. To musíme udělat sami. OPmac nabízí pro sazbu do více sloupců makra `\begmulti` ⟨*počet-sloupců*⟩ `...` `\endmulti`. Příklad:

```
\sec Rejstřík
\begmulti 3 \makeindex \endmulti
```

Do rejstříku musejí být zařazena jen „čistá“ slova, která neobsahují makra expandující na primitivní příkazy TEXu. Pokud chcete vytisknout v rejstříku něco komplikovanějšího, můžete sestavit slovník výjimek pomocí maker `\iis` ⟨*heslo*⟩⟨*mezera*⟩`{`⟨*tisk*⟩`}` (název makra můžeme číst jako `\ii` speciální). Funkci si vysvětlíme na příkladu:

```
\iis chikvadrat {$\chi$-kvadrát}
\iis relax {{\tt \char‘\\relax}}
\iis Goedelova/věta/o~neúplnosti {G\"odelova/věta/o~neúplnosti}
\iis věta/o~neúplnosti/Goedelova {věta/o~neúplnosti/G\"odelova}
```

Lze pak psát `\ii relax`, `\ii chikvadrat` nebo `\ii Goedelova/věta/o~neúplnosti,@`. OPmac abecedně řadí podle těchto hesel, ale když dojde na potřebu heslo vytisknout do rejstříku, vytiskne místo těchto hesel materiál, který je uveden na pravé straně slovníku. Příklad ukazuje, že tím lze řešit nejen tisk hesel, která je potřeba ošetřit speciálními makry (v příkladu slovo relax), ale také výjimky abecedního řazení. Slovník výjimek je možný zapsat kamkoli před `\makeindex`, typicky se píše na začátek dokumentu.

Výjimku z řazení dvojhlásky ch (například ve slově mochnátý, tj. mnohonohý) je možné zařídit pomocí tečky, která má stejně jako ostatní interpunční znaky, nulovou řadicí platnost (OPmac hesla řadí, jakoby tam interpunkce nebyla). Takže třeba takto:

```
... \ii moc.hnátý ...
\iis moc.hnátý {mochnátý}
```

Je-li při zpracování `\makeindex` zapnutý anglický jazyk (implicitní nastavení nebo po přepínači `\ehyph`), pak se ch neinterpretuje jako dvojhláska. Ostatní pravidla řazení zůstávají nezměněna.

Pro různé speciální znaky můžete využít znak `@`, který se řadí před celou abecedou. Speciální znak pak nahradíte až ve slovníku výjimek. Takže třeba `\ii Ernst~@~Young` pro řazení a `\iis Ernst~@~Young {Ernst \& Young}` pro tisk.

8 Barvy, vodoznaky

Makra uvedená v této sekci nastavují barvy jen při přímém vytváření do PDF. Takže při výstupu do DVI tato makra neudělají nic. Barvu textu můžete nastavit pomocí přepínačů `\Blue`, `\Red`, `\Brown`, `\Green`, `\Yellow`, `\Cyan`, `\Magenta`, `\White`, `\Grey`, `\LightGrey` a `\Black`.

Implicitně tyto přepínače pracují globálně nezávisle na TEXové skupině. Barvu jinou než černou je pak potřeba ukončit explicitním přepínačem `\Black`. Toto chování je možné změnit

uvedením příznaku `\localcolor`. Tento příznak je možné nastavit globálně (například na začátku dokumentu) nebo lokálně uvnitř skupiny. Při globálním nastavení se sazba vrací k původní barvě za všemi TEXovými skupinami a při lokálním nastavení se k původní barvě vrací sazba za skupinou začínající příznakem `\localcolor` (a za všemi vnořenými skupinami). Příklad:

```
Černý {\localcolor \Blue modrý {\Green zelený \Red červený} modrý} černý.
```

Další příklad vytvoří podbarvený text:

```
\def\podbarvi#1#2#3{\setbox0=\hbox{#3}\leavevmode
   {\localcolor\rlap{#1\strut\vrule width\wd0}#2\box0}}
   \podbarvi\Yellow\Brown{Tady je hnědý text na žlutém pozadí.}
```

V původní verzi OPmac bylo možné rozlišovat barvy tenkých linek a písma. Tuto vlastnost jsem opustil od verze Dec. 2014. Zdůvodnění najdete v technické dokumentaci.

Kromě uvedených barevných přepínačů si můžete „namíchat“ v režimu CMYK i barvy vlastní. Stačí se inspirovat, jak jsou uvedené přepínače definovány:

```
\def\Red{\setcmykcolor{0 1 1 0}}
\def\Brown{\setcmykcolor{0 0.67 0.67 0.5}} ...
```

Aktuální barvu ve formě čtyř čísel CMYK je možné přečíst z makra `\currentcolor` například pomocí `\let\savedcolor=\currentcolor` a později je možné se k této barvě vrátit pomocí `\setcmykcolor\savedcolor`.

Vodoznakem je míněn šedý text opakující se na každé stránce, který je vytištěn pod obvyklým textem. Například OPmac nabízí makro `\draft`, které způsobí, že každá stránka obsahuje šikmo napsaný veliký šedý nápis DRAFT. Můžete se inspirovat v technické dokumentaci, jak je to uděláno.

9 Klikací odkazy

Pokud napíšete na začátek dokumentu `\hyperlinks{`⟨*color-in*⟩`}{`⟨*color-out*⟩`}`, pak se v dokumentu při výstupu do PDF stanou klikacími:

- čísla generovaná pomocí `\ref` a `\pgref`,
- čísla kapitol, sekcí, podsekcí a stránek v obsahu,
- čísla nebo značky generované pomocí `\cite` (odkazy na literaturu),
- texty tištěné pomocí makra `\url` nebo `\ulink`.

Poslední z uvedených odkazů je externí a bude mít barvu ⟨*color-out*⟩, zatímco ostatní čísla jsou interními odkazy a budou mít barvu ⟨*color-in*⟩. Příklad:

```
\hyperlinks{\Blue}{\Green}   % vnitřní odkazy modré, URL zelené
```

Je možné zobrazit rámečky ohraničující aktivní plochu pro klikání. Tyto rámečky jsou viditelné jen v PDF prohlížeči, při tisku na tiskárně se nezobrazují. Stačí těmto rámečkům „namíchat“ barvu (tentokrát RGB) a definovat některé ze sekvencí `\pgborder`, `\tocborder`, `\citeborder`, `\refborder` a `\urlborder`. První část jména kontrolní sekvence určuje, jakých odkazů se to týká. Příklad:

```
\def\tocborder{1 0 0}  % odkazy v obsahu vlevo budou mít červený rámeček
\def\pgborder{0 1 0}   % odkazy na stránky budou mít zelený rámeček
\def\citeborder{0 0 1} % odkazy na publikace budou mít modrý rámeček
```

Implicitně tato makra nejsou definována, což znamená, že se rámečky netvoří.

Manuálně je možno vytvořit cíl odkazu makrem `\dest[`⟨*typ*⟩`:`⟨*lejblík*⟩`]` a klikací text makrem `\link[`⟨*typ*⟩`:`⟨*lejblík*⟩`]{`⟨*color*⟩`}{`⟨*text*⟩`}`. Parametr ⟨*typ*⟩ je typ odkazu (toc, pg, cite, ref nebo další).

Makro `\url` vytiskne odkaz do internetu. Příklad: `\url{http://petr.olsak.net}` vytvoří http://petr.olsak.net. Text je psán strojopisem a může se lámat do řádků za lomítky. Je-li nastaveno `\hyperlinks`, stává se tento text aktivním vnějším odkazem. Vyskytují-li se v argumentu `\url` znaky `%`, `\`, `#`, `$`, `{` a `}`, je třeba použít `\%`, `\\`, `\#`, `\$`, `\{` a `\}`. Ostatní speciální znaky `~`, `_`, `^`, `&` lze napsat do parametru `\url` přímo. Dále je možno do parametru `\url` napsat `\|` k označení místa, kde je dovoleno zlomit řádek. Libovolný text odkazovaný na webovou stránku lze vložit pomocí `\ulink[`⟨*URL*⟩`]{`⟨*text*⟩`}`. Například odkaz na stránky OPmac jsou vytvořeny pomocí `\ulink[http://petr.olsak.net/opmac.html]{stránky OPmac}`.

Obsah dokumentu se dá přesunout do levé záložky PDF prohlížeče tak, že klikáním na něj se přechází v dokumentu na požadované místo. Ve specifikaci PDF se tomu říká „outlines“. Makro, které uvedenou věc zařídí, se jmenuje `\outlines{`⟨*úroveň*⟩`}`. Záložky budou implicitně rozevřeny do ⟨*úrovně*⟩ včetně. Takže třeba při ⟨*úroveň*⟩=0 jsou vidět jen úrovně kapitol. Bohužel písmo v záložkách typicky nezvládá správně česká a slovenská písmena. Proto OPmac konvertuje texty do záložek tak, že tam jsou bez hacku a carek. Chcete-li vypnout tuto konverzi, napište `\def\toasciidata{}`. Chcete-li akcenty zachovat, použijte doplňkové makro `pdfuni.tex`.

Samotný řádek do záložek vložíte makrem `\insertoutline{`⟨*text*⟩`}`. Text v tomto případě nepodléhá konverzi. V sazbě se neobjeví nic, jen se stane cílem, když uživatel na záložku s ⟨*textem*⟩ klikne. Obsah se do záložek vloží celý během činnosti makra `\outlines`, takže další řádky vložené pomocí `\insertoutline` tomuto obsahu předcházejí nebo následují podle toho, zda předcházejí nebo následují místu, kde je použito `\outlines`.

10 Verbatim texty

Vytisknout část textu verbatim „tak jak je“ bez interpretace speciálních znaků lze v prostředí vymezeném makry `\begtt` a `\endtt`. Příklad:

```
\begtt
Tady je vše
   napsáno bez   interpretace speciálních znaků, jakými
 jsou mezera, %, $, \, ~, ^, _, {, }, #, &.
\endtt
```

Ve výstupu se objeví:

```
Tady je vše
   napsáno bez   interpretace speciálních znaků, jakými
 jsou mezera, %, $, \, ~, ^, _, {, }, #, &.
```

Není-li za `\endtt` prázdný řádek, nemá následující odstavec výchozí odsazení.

Je-li před zahájením `\begtt` nastaven registr `\ttline` na nezápornou hodnotu, bude makro číslovat řádky. První řádek má číslo `\ttline+1` a po práci makra se registr `\ttline` posune na číslo posledního vytištěného řádku. Takže v dalším prostředí `\begtt` … `\endtt` číslování pokračuje tam, kde přestalo. Implicitně je `\ttline=-1`, takže číslování neprobíhá.

Levé odsazení každého řádku v `\begtt...\endtt` je nastaveno na `\ttindent`. Tento registr má výchozí hodnotu rovnu `\parindent` (v době čtení souboru `opmac.tex`). Vertikální mezera nad a pod verbatim výpisem je vložena makrem `\ttskip`.

Makro `\begtt` zahájí skupinu a v ní nastaví všem speciálním znakům plainTEXu kategorii 12. Pak spustí makro `\tthook`, které je implicitně prázdné. V něm je možno nastavit další kategorie znaků podle potřeby. Definici aktivních znaků je potřeba udělat pomocí `\adef`⟨*znak*⟩`{`⟨*text*⟩`}`. Normální `\def` nefunguje, důvod je vysvětlen v TBN, str. 26. Příklad:

```
\def\tthook{\adef!{?}}
\begtt
Nyní se každý vykřičník promění v otazník. Že nevěříte? Vyzkoušejte!
\endtt
```

Definovaný `\tthook` funguje ve všech verbatim výpisech, dokud jej nepředefinujete jinak. Tipy:

```
\def\tthook{\typosize[9/11]}    % jiná velikost verbatim výpisů
\def\tthook{\ttline=0}          % všechny výpisy číslovány od jedničky
\def\tthook{\adef{ }{\char`\ }} % místo mezer budou vaničky
```

Verbatim lze tisknout i v řádku uvnitř odstavce. Pomocí `\activettchar`⟨*znak*⟩ si uživatel zvolí znak, který bude aktivní a bude zahajovat i končit verbatim výpisy uvnitř odstavce. Verbatim výpis se v odstavci nikdy nerozlomí (je v boxu). Autor makra OPmac obvykle nastavuje `\activettchar"`, takže pak může psát třeba toto:

```
Je-li před zahájením "\begtt" nastaven registr "\ttline" na nezápornou...
```

Znak nastavený pomocí `\activettchar` má lokální platnost a ruší se také pozdějším nastavením `\activettchar` na jinou hodnotu. Při zahájení každého řádkového verbatim výpisu se spustí makro `\intthook`, které je implicitně prázdné. **Upozornění:** deklaraci `\activettchar`⟨*znak*⟩ proveďte až po přečtení všech makrosouborů. Důvod: `\activettchar` nastavuje ⟨*znak*⟩ jako aktivní, což může při čtení souborů maker vadit. Není taky rozumné aktivovat znak pro derivaci nebo jiný v textu používaný znak.

Verbatim výpisy je možné tisknout z externího souboru. Například

```
\verbinput (12-42) program.c
```

vytiskne ve stejné úpravě, jako při použití `\begtt`, ... `\endtt`, řádky 12 až 42 ze souboru `program.c`. Parametry v kulaté závorce mohou vypadat také takto:

```
\verbinput (-60) program.c   % výpis od začátku souboru do řádku 60
\verbinput (61-) program.c   % výpis od řádku 61 do konce souboru
\verbinput (-) program.c     % výpis celého souboru
\verbinput (70+10) program.c % výpis od řádku 70, tiskne 10 řádků
```

V dalších ukázkách OPmac čte od řádku, který následuje za naposledy přečteným řádkem souboru z předchozího volání `\verbinput`. Je-li soubor čten poprvé, začíná číst prvním řádkem. Tento prvně čtený řádek je označen v komentářích jako `n`.

```
\verbinput (+10) program.c  % výpis deseti řádků od řádku n
\verbinput (+) program.c    % výpis od řádku n do konce souboru
\vebrinput (-5+7) program.c % vynechá 5 řádků, od n+5 tiskne dalších 7
\verbinput (-3+) program.c  % vynechá 3 řádky, tiskne do konce souboru
```

Narazí-li čtení na konec souboru dřív, než je vytištěno vše, co si žádá uživatel, přepis souboru je ukončen a žádná chyba se nezjeví.

Výpisy provedené makrem `\verbinput` jsou ovlivněny registrem `\ttindent` a makrem `\tthook` stejně, jako prostředí `\begtt`...`\endtt`. Při `\ttline<-1` se netisknou čísla řádků. Je-li `\ttline=-1`, čísluje se podle řádků souboru. Je-li `\ttline` nezáporné, čísluje se od `\ttline+1`.

11 Jednoduché tabulky

LaTeXoví uživatelé jsou zvyklí při vymezení pravidel zarovnávání v tabulce používat deklarace typu `{cclr}`. Každé písmeno vymezí jeden sloupec v tabulce, přitom písmeno `c` znamená centrovaný sloupec, `l` je sloupec zarovnaný doleva a `r` sloupec zarovnaný doprava. Podobnou možnost deklarace jednoduchých tabulek nabízí OPmac v makru `\table{`⟨*deklarace*⟩`}{`⟨*data*⟩`}`. Příklad:

```
\table{||lc|r||}{\crl
   měsíc     & zboží     & cena\hfil  \crli \tskip.2em
   leden     & noťas     & 14 kKč     \cr
   únor      & skejt     &  2 kKč     \cr
   červenec  & jachtička & 3,4 MKč    \crl}
```

Uvedený příklad povede na následující výsledek:

měsíc	zboží	cena
leden	noťas	14 kKč
únor	skejt	2 kKč
červenec	jachtička	3,4 MKč

Ve skutečnosti výsledek nebude uprostřed řádku, ale tam, kam `\table` napíšete.

Kromě písmen `c`, `l`, `r` se v ⟨*deklaraci*⟩ mohou objevit znaky „svislítko“, které vymezují svislou čáru mezi sloupci. Dále kromě deklarátorů jednotlivých sloupců `c`, `l`, `r` je v OPmac od verze Jun 2017 připraven deklarátor `p{`⟨*rozměr*⟩`}`, který vymezí sloupec tabulky s pevnou šířkou sloupce ⟨*rozměr*⟩. Delší text se formátuje jako odstavec dané šířky bez odstavcové zarážky, kratší jednořádkový text je vlevo. Například `p{42mm}` vytvoří sloupec pro text široký 42 mm. Při velmi úzkých sloupcích nastávají obvykle potíže s formátováním odstavce do bloku. V takovém případě můžete hned za ⟨*rozměr*⟩ přidat třeba `\raggedright` a odstavec bude formátován s nezarovnaným pravým okrajem, například `p{42mm\raggedright}`.

Pokud v ⟨*deklaraci*⟩ použijete `[`⟨*text*⟩`]`, je tento text aplikován v každém řádku tabulky v místě odpovídajícím umístění v ⟨*deklaraci*⟩. Například `r[\kern10pt]l` vloží mezi sloupce `r` a `l` dodatečnou desetibodovou mezeru.

V ⟨*deklaraci*⟩ je možné místo opakování stejného deklarátoru použít číslo následované deklarátorem, tedy třeba `4c` je totéž jako `cccc`. Opakovat se mohou celé úseky ⟨*deklarace*⟩, pokud za číslem pokračuje úsek ve svorkách. Například `c 3{|c}` je totéž jako `c|c|c|c`. Mezery v ⟨*deklaraci*⟩ se ignorují a je možné je použít pro zvýšení přehlednosti.

V datové části tabulky musí být přesně tolik sloupců, kolik jich bylo deklarováno. Jsou odděleny znakem `&` nebo symbolem pro konec řádku `\cr`. Z toho vyplývá, že na každém řádku musí být v datové části o jeden znak `&` méně, než je počet sloupců. Nedodržíte-li toto pravidlo, TEX se pomstí chybovou hláškou

```
! Extra alignment tab has been changed to \cr
```

nebo vytvoří nedomrlou tabulku.

Místo symbolu pro konec řádku `\cr` je možno použít `\crl` (přidá jednoduchou vodorovnou čáru) nebo `\crll` (přidá dvojitou čáru), `\crli` (přidá čáru přerušenou svislými dvojitými linkami, tj. **i**nterrupted) a `\crlli` (přidá dvojitou čáru přerušenou svislými dvojitými linkami). Konečně lze použít `\crlp{`⟨*seznam*⟩`}`, což vloží přerušovanou čáru jako `\crli`, ale jen ve sloupcích, jejichž pořadová čísla oddělená čárkami jsou vyjmenována v ⟨*seznamu*⟩. Takže třeba užití `\crlp{1,2}` v ukázce tabulky výše (místo `\crli`) vytvoří čáru jen pod slovy „měsíc“ a „zboží“. V ⟨*seznamu*⟩ lze používat zkratky typu ⟨*od*⟩`-`⟨*do*⟩, například `1-3,5` je totéž jako `1,2,3,5`.

Těsně za `\cr`, `\crl` atd. může následovat `\tskip`⟨*dimen*⟩, což vytvoří vertikální mezeru velikosti ⟨*dimen*⟩, přitom se nepřeruší svislé čáry v tabulce.

Za povšimnutí stojí, že v ukázce u slova „cena“ je připojeno `\hfil`, což vloží pružnou mezeru vpravo od položky. Protože sloupec `r` obsahuje implicitní stejnou pružnou mezeru vlevo, je slovo „cena“ centrováno, zatímco ostatní údaje ve sloupci jsou zarovnány napravo.

Makro `\table` pracuje s předdefinovanými hodnotami, které můžete změnit, pokud chcete dosáhnout jiný vzhled tabulky:

```
\def\tabiteml{\enspace} % co vkládá vlevo každé datové položky
\def\tabitemr{\enspace} % co vkládá vpravo každé datové položky
\def\tabstrut{\strut}   % podpěra vymezující výšku řádků
\def\vvkern{1pt}   % velikost mezery mezi dvojitou svislou linkou
\def\hhkern{1pt}   % velikost mezery mezi dvojitou vodorovnou linkou
```

Vyzkoušejte si tabulku po `\def\tabiteml{}\def\tabitemr{}`. Sloupce budete mít na sebe nalepeny bez mezer. Příklad definice `\tabstrut`:

```
\def\tabstrut{\vrule height11pt depth3pt width0pt}
```

Tento příklad vymezuje v tabulce vzdálenost mezi účařím 14pt, z toho 11pt je rezervováno pro přetahy nad účařím a 3pt pro přetahy pod účařím. Vyskytne-li se větší písmeno, zvětší to v daném místě řádkování.

OPmac definuje `\strut` závislý na zvoleném řádkování (při použití příkazu `\typosize`) zhruba takto:

```
\def\strut{\vrule height.709⟨baselineskip⟩ depth.291⟨baselineskip⟩ width0pt}
```

Tip: vyzkoušejte si `\def\tabiteml{$\enspace} \def\tabitemr{\enspace$}`. Ty dolary způsobí, že každá datová položka bude zpracována v matematickém módu. Makro `\table` se nyní podobá LaTeXovému prostředí `array`.

Chcete-li přesáhnout jedním údajem více sloupců tabulky, můžete použít plainTeXové makro `\multispan{`⟨*číslo*⟩`}` nebo makro z OPmac `\mspan`⟨*číslo*⟩`[`⟨*deklarace*⟩`]{`⟨*text*⟩`}`, které přesáhne ⟨*číslo*⟩ sloupců a ⟨*text*⟩ v tomto prostoru formátuje podle ⟨*deklarace*⟩, která obsahuje právě jedno písmeno `c`, `l` nebo `r` (označující způsob formátování) a dále může obsahovat jeden nebo více znaků `|` (označující svislé čáry). Máte-li v tabulce svislé čáry a chcete, aby svislé čáry z `\mspan` na ně navazovaly, pak použijte v `\mspan` čáry před údajem `c`, `l` nebo `r` jedině v případě, že `\mspan` zasahuje do prvního sloupce. V ostatních případech používejte čáry výhradně na konci ⟨*deklarace*⟩, protože každý sloupec (s výjimkou prvního) přidává případné svislé čáry jen na svůj konec.

Makro `\frame{`⟨*text*⟩`}` vytvoří rámeček kolem ⟨*textu*⟩ s vnitřními okraji o velikostech `\vvkern` a `\hhkern`. Například `\frame{ahoj}` vytvoří ahoj. Povšimněte si, že účaří rámovaného textu zůstalo nezměněno. Pokud chcete mít tabulku s dvojitými čarami, je výhodné ji vytvořit po stranách a nahoře a dole s jednoduchými čarami a celou ji zabalit do `\frame`:

```
\frame{\table{|c||l||r||c|}{\crl
  \mspan4[|c|]{\bf Nadpis} \crl
  \noalign{\kern\hhkern}\crli
  první & druhý & třetí  & čtvrtý \crlli
  sedmý & osmý  & devátý & desátý \crli}}
```

Nadpis			
první	druhý	třetí	čtvrtý
sedmý	osmý	devátý	desátý

Kromě předdefinovaných znaků `c,l,r,p` lze v ⟨*deklaraci*⟩ `\table` použít libovolný další symbol pro deklaraci sloupce, stačí připravit `\def\tabdeclare`⟨*symbol*⟩`{`⟨*vlevo*⟩`##`⟨*vpravo*⟩`}`. Toto je podrobněji popsáno v technické dokumentaci.

Tloušťka všech čar je v TeXu implicitně 0,4pt. OPmac umožňuje tuto implicitní tloušťku nastavit jinak pomocí `\rulewidth=`⟨*šířka*⟩, například `\rulewidth=1.5pt`.

Další příklad použití makra `\table` najdete v sekci 4. Pokud potřebujete vytvořit komplikovanější tabulky, můžete se inspirovat sekcí OPmac triků věnující se tabulkám. Můžete si také definovat případně další vlastní tabulkové deklarátory nebo použít přímo primitivní `\halign`, což vyžaduje zřejmě prostudovat TBN, kapitolu čtvrtou.

12 Vkládání obrázků

Makro `\inspic` ⟨*jméno*⟩`.`⟨*přípona*⟩⟨*mezera*⟩ vloží obrázek. Je-li předem nastaven registr `\picw`, má obrázek požadovanou šířku. Implicitní hodnota registru je 0 pt, což znamená, že bude obrázek vložen ve své přirozené velikosti. Analogicky lze nastavit výšku obrázku registrem `\picheight`. Přípony souboru s obrázkem mohou být `png`, `jpg`, `jbig2`, `pdf`.

Obrázek je vyhledán v adresáři `\picdir`. Toto makro je implicitně prázdné, tj. obrázek je vyhledán v aktuálním adresáři.

O umístění obrázku v sazbě se musíte postarat vlastními prostředky. Například:

```
\picw=.5\hsize \centerline{\inspic hodiny.jpg }\nobreak\medskip
\caption/f Hodiny na brněnském náměstí Svobody.
```

Makro není vhodné použít při opakovaném použití stejného obrázku v dokumentu (opakující se grafika na každé straně nebo obrázek jako odrážka ve výčtu položek). V takovém

případě je vhodnější natáhnout obrázek do PDF dokumentu jen jednou pdfTEXovým příkazem `\pdfximage` a dále opakovat jeho zobrazení na různých místech dokumentu pomocí `\pdfrefximage`. Dokumentace k pdfTEXu řekne víc.

Makro `\inspic` pracuje jen při výstupu do PDF. Pokud máte nastaven výstup do DVI, můžete použít makro `epsf.tex`. Vzhledem k omezeným možnostem (obrázek jen ve formátu EPS) není tento způsob práce s obrázky v makru OPmac podporován.

Chcete-li „programovat“ obrázky přímo ve zdrojovém textu TEXu, lze použít plainTEXové rozhraní vynikajícího makra Ti*k*Z, se kterým je OPmac kompatibilní.

13 PDF transformace

Veškerá sazba v pdfTEXu může podléhat lineární transformaci, která je daná transformační maticí `\pdfsetmatrix{`⟨*a*⟩ ⟨*b*⟩ ⟨*c*⟩ ⟨*d*⟩`}`. Tato matice se v lineární algebře zapisuje do dvou řádků:

$$\begin{pmatrix} a & c \\ b & d \end{pmatrix}, \quad \text{např. zvětšení: } \begin{pmatrix} s_1 & 0 \\ 0 & s_2 \end{pmatrix}, \quad \text{nebo rotace: } \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}.$$

PdfTEXové primitivy `\pdfsetmatrix`, `\pdfsave` a `\pdfrestore` bohužel nejsou v dokumentaci pdfTEXu uvedeny, tak je musím dokumentovat aspoň zde. Příkaz `\pdfsave` uloží stávající transformační matici a aktuální bod sazby. V době konání příkazu `\pdfrestore` se matice vrátí do původní podoby a aktuální bod sazby v té době musí být na stejném místě, jako byl v době `\pdfsave`, jinak se nám sazba rozjede a pdfTEX nadává. Toho se dá docílit např. pomocí `\pdfsave...\rlap{`⟨*text*⟩`}\pdfrestore`. Transformační matice se nastavují pomocí `\pdfsetmatrix`. Opakované použití `\pdfsetmatrix` způsobí pronásobení transformační matice novou maticí, takže to funguje jako skládání zobrazení. OPmac nabízí dvě užitečná makra `\pdfscale{`⟨*vodorovně*⟩`}{`⟨*svisle*⟩`}` a `\pdfrotate{`⟨*úhel*⟩`}`. Parametr ⟨*úhel*⟩ je interpretován ve stupních. Tato makra provedou odpovídající `\pdfsetmatrix`.

Aplikujeme-li více matic za sebou, je potřeba vědět, že výchozí text prochází transformací jednotlivých matic „odzadu dopředu“, takže například:

```
První: \pdfsave \pdfrotate{30}\pdfscale{-2}{2}\rlap{text1}\pdfrestore
      % text1 je zvětšen dvakrát a překlopen podél svislé osy,
      % dále je otočen o 30 stupňů doleva a konečně je vytištěn.
druhý: \pdfsave \pdfscale{-2}{2}\pdfrotate{30}\rlap{text2}\pdfrestore
      % text2 je otočen o 30 stupňů doleva, dále zvětšen a překlopen
      % podél svislé osy, nakonec vytištěn.
třetí: \pdfsave \pdfrotate{-15.3}\pdfsetmatrix{2 0 1.5 2}\rlap{text3}%
       \pdfrestore % nejprve zkosení, pak otočení o 15.3 stupňů doprava
```

Ukázka dává následující výsledek. První, druhý, třetí: 

14 Poznámky pod čarou a na okraji

Poznámku pod čarou vytvoříte pomocí `\fnote{`⟨*text*⟩`}`. V místě tohoto zápisu v textu se objeví automaticky generovaná značka a pod čarou dole na stránce je tato značka zopakována a vedle ní je ⟨*text*⟩.

Značka je implicitně definovaná jako číslo v exponentu. Číslování poznámek je na každé stránce započato jedničkou[1]. Čísla jsou vygenerována správně až po opakovaném TEXování. Při prvním zpracování jsou místo čísel otazníky.

[1] Toto chování se dá změnit vložením příkazu `\runningfnotes` na začátek dokumentu. V takovém případě se poznámky číslují od jedné v celém dokumentu. Další možnosti číslování jsou uvedeny v technické dokumentaci.

Implicitní značkování je možné změnit předefinováním makra `\thefnote`. Například po

```
\def\thefnote{\ifcase\locfnum\or
   *\or**\or***\or$^{\dag}$\or$^{\ddag}$\or$^{\dag\dag}$\fi)}
```

bude první poznámka mít hvězdičku se závorkou, druhá dvě hvězdičky se závorkou atd. Uvedená definice předpokládá, že na jedné stránce nebudete mít více než 6 poznámek.

Makro `\fnote` je možné zapsat jen v běžném textu odstavce, nikoli v boxu (například v tabulce). Chcete-li odkazovat třeba z tabulky, je nutné v tabulce vytvořit jen značky a mimo tabulku (ovšem tak, aby to neuteklo na jinou stránku) zapíšete texty poznámek. K vytvoření značky použijte `\fnotemark`⟨*číslo*⟩ a text (bez značky) vytvoří `\fnotetext{`⟨*text*⟩`}`. Příklad:

```
{\typoscale[/1200]\table{||lc|r||}{\crl
     měsíc     & zboží                & cena\hfil \crli \tskip.2em
     leden     & noťas\fnotemark1     & 14 kKč    \cr
     únor      & skejt\fnotemark2     &  2 kKč    \cr
     červenec  & jachtička\fnotemark3 & 3,4 MKč   \crl}}
\par\nobreak \fnotetext{notebook}\fnotetext{skateboard}\fnotetext{jachta}
```

Čísla za slovy `\fnotemark` je třeba psát od jedné v každé tabulce/boxu. Nemusejí souviset se skutečným číslem poznámky. Například, je-li na stejné stránce nad tabulkou z ukázky normální `\fnote`, bude mít vytištěno číslo 1, odkazy v tabulce budou mít čísla 2, 3, 4 a případná další poznámka pod tabulkou na stejné stránce obdrží číslo 5.

Poznámku na okraji stránky vytvoříte pomocí řídicí sekvence `\mnote{`⟨*text*⟩`}`. Poznámka je vlevo (na pravou zarážku) na sudé stránce a je vpravo (na levou zarážku) na liché stránce. Tuto vlastnost mají poznámky až po opakovaném TEXování. Při prvním TEXování jsou všechny poznámky vpravo. Chcete-li mít poznámky i při opakovaném TEXování jen vpravo nebo jen vlevo, pište do úvodu dokumentu `\fixmnotes\right` nebo `\fixmnotes\left`.

Řídicí sekvenci `\mnote{`⟨*text*⟩`}` můžete napsat do odstavce nebo před odstavec. S odstavcem samotným to nic neudělá. Řádek odstavce, kde je `\mnote` vložena jako neviditelná značka, je na stejné úrovni, jako první řádek textu poznámky.

Text poznámky je od sazby odsazen o `\mnoteindent` a maximální šířka poznámky je `\mnotesize`. Text poznámky se rozlomí do více řádků, aby nepřesáhl `\mnotesize`.

Není ošetřen případ, kdy je `\mnote` víceřádková a je umístěna na úroveň například posledního řádku strany. Pak text poznámky přečuhuje poněkud dolů ze strany. Nebo se poznámky mohou překrývat. Je tedy nutné `\mnote` použít jen na velmi krátké poznámky a případně si tento jev pohlídat a ošetřit při definitivní sazbě manuálně. Pro manuální ošetření vertikální polohy poznámek slouží `\mnoteskip`, což je registr, který udává, o kolik se má následující poznámka (a jen ta) posunout nahoru. Při záporné hodnotě se posune dolů. Například `\mnoteskip=2\baselineskip \mnote{`⟨*text*⟩`}` posune poznámku o dva řádky výše.

15 Bibliografické údaje

Odkazy. Pomocí `\cite[`⟨*lejblík*⟩`]` nebo `\cite[`⟨*lejblík1*⟩`,`⟨*lejblík2*⟩`,`⟨*lejblík3*⟩`]` atd. vytvoříme v textu odkazy na položky v seznamu literatury. V seznamu literatury je třeba uvést záznamy, které mají odkazované lejblíky. Tyto záznamy dostanou v seznamu automaticky vygenerovaná čísla a sekvence `\cite` se pak promění na číselné odkazy, například [27] nebo [18, 24, 42] atd. Souvislé řady čísel [1, 2, 3, 5, 6] se promění v intervaly [1–3, 5–6] jen tehdy, když je v úvodní deklaraci dokumentu napsáno `\shortcitations`. A seřadí se podle velikosti při `\sortcitations`.

Při `\nonumcitations` se odkazy nepřevádějí na čísla. K tomu je potřeba použít navazující BibTEXový styl (např. `alpha`, `apalike`) nebo rozšířenou formu příkazu `\bib`, viz níže. Odkazy vypadají při stylu `alpha` takto [Nov08] a při stylu `apalike` takto [Novák, 2008].

Příkaz `\rcite[`⟨*lejblíky*⟩`]` funguje jako `\cite[`⟨*lejblíky*⟩`]`, ale kolem odkazů nejsou přidány závorky. Možnost využití: `[\rcite[novak08],~s.~213]` vytvoří například odkaz [17, s. 213]. Závorky kolem musíte napsat sami.

Příkaz `\ecite[`⟨*lejblík*⟩`]{`⟨*text*⟩`}` vytiskne pouhý ⟨*text*⟩, který se chová jako odkaz na literaturu. Příklad použití: `z~výsledů Nováka [\ecite[novak08]{2008},~s.~213] plyne...` z výsledků Nováka [2008, s. 213] plyne… Přitom `novak08` je registrován do seznamu citovaných položek a třeba při `\hyperlinks` bude číslo 2008 prolinkováno s odpovídající položkou v seznamu literatury.

Příklady redefinice `\cite` pro alternativní formátování odkazů:

```
\def\cite[#1]{(\rcite[#1])}     % \cite[lejblík] vytvoří (27)
\def\cite[#1]{$^{\rcite[#1]}$}  % \cite[lejblík] vytvoří^{27}
```

Seznam literatury je možné vložit do dokumentu čtyřmi různými způsoby:

- Manuálně: pomocí jednotlivých položek `\bib[`⟨*lejblík*⟩`]` přímo v dokumentu.
- S využitím BibTEXu makrem `\usebibtex{`⟨*bib-báze*⟩`}{`⟨*bst-styl*⟩`}`.
- Využitím jednou vygenerované databáze makrem `\usebbl/`⟨*typ*⟩ ⟨*bbl-báze*⟩.
- Přímým čtením `.bib` databáze makry TEXu bez využití BibTEXu.

Jednotlivé způsoby jsou níže probrány podrobněji.

Manuálně vložený seznam literatury v dokumentu vypadá například takto:

```
\bib [tbn] Petr Olšák. {\it\TeX{}book naruby.} 468~s. Brno: Konvoj, 2001.
\bib [tst] Petr Olšák. {\it Typografický systém \TeX.}
                       300~s. Brno: Konvoj, 2000.
```

Výše uvedená ukázka dá následující výstup:

[1] Petr Olšák. *TEXbook naruby.* 468 s. Brno: Konvoj, 2001.

[2] Petr Olšák. *Typografický systém TEX.* 300 s. Brno: Konvoj, 2000.

Je možné použít rozšířený způsob zápisu `\bib [`⟨*lejblík*⟩`] = {`⟨*značka*⟩`}` ⟨*text záznamu*⟩. Údaj ⟨*značka*⟩ se použije do odkazů při zapnutém `\nonumcitations`. Kolem rovnítka musejí být mezery. Například:

```
\bib [tbn] = {Olšák, 2001}
     OLŠÁK, P. {\it\TeX{}book naruby.} 468~s. Brno: Konvoj, 2001.
```

Využití BibTEXu. Předpokládá se, že uživatel disponuje souborem ⟨*bib-báze*⟩`.bib`, ve kterém jsou nashromážděny bibliografické údaje ve formátu, v jakém je čte program BibTEX. V TEXové distribuci jistě najdete nějaký `*.bib` soubor, tak se do něj podívejte. Lejblíkem je první údaj u každého bibliografického záznamu. Soubor ⟨*bib-báze*⟩`.bib` by měl obsahovat bibliografické údaje, které jsou nadmnožinou toho, co potřebuje uživatel vypsat ve svém dokumentu. Na místo, kde budete chtít vypsat seznam literatury, vložte následující pokyn:

`\usebibtex{`⟨*bib-báze*⟩`}{`⟨*bst-styl*⟩`}`

Parametr ⟨*bib-báze*⟩ je jméno souboru bez přípony `.bib`, ve kterém jsou připraveny bibliografické záznamy. Parametr ⟨*bst-styl*⟩ je jméno stylového souboru bez přípony `.bst`, který použije BibTEX pro konverzi ze zdroje ⟨*bib-báze*⟩`.bib` do výstupu ⟨*dokument*⟩`.bbl`. Tento výstup pak bude makrem `\usebibtex` přečten a vložen do dokumentu. Typicky používané ⟨*bst-styly*⟩ jsou `plain`, `alpha`, `apalike`, `ieeetr`, `unsrt`. Existují desítky, možná stovky dalších `.bst` stylů, viz internet.

Při prvním zpracování dokumentu TEXem makro `\usebibtex` připraví vstupní pokyny pro BibTEX do souboru ⟨*dokument*⟩`.aux` a zjistí, že soubor ⟨*dokument*⟩`.bbl` zatím neexistuje. To dá najevo na terminálu:

`WARNING: .bbl file doesn't exist. Use the ``bibtex` ⟨*dokument*⟩`'' command.`

Přejděte tedy na příkazový řádek a napište `bibtex` ⟨*dokument*⟩. Tím se spustí program BibTEX, který přečte ze souboru ⟨*dokument*⟩`.aux` vstupní pokyny (kterou otevřít `.bib` databázi, který `.bst` styl a jaké lejblíky jsou požadovány) a na základě toho vygeneruje soubor ⟨*dokument*⟩`.bbl`, který obsahuje výběr jen těch záznamů, které byly uživatelem citovány pomocí `\cite`. Soubor ⟨*dokument*⟩`.bbl` je navíc zkonvertovaný z `.bib` formátu do formátu čitelného TEXem. Tato konverze je řízena stylem `.bst`.

Když znovu TEXujete dokument, makro `\usebibtex` v tomto případě shledá, že soubor ⟨*dokument*⟩`.bbl` existuje, načte jej a vytvoří seznam literatury. Seznam obsahuje jen citované položky. Druhé spuštění TEXu obvykle nestačí, protože příkazy `\cite` jsou typicky dopřednými referencemi, takže zatím nemají ponětí o přiřazení čísel k ⟨*lejblíkům*⟩ v seznamu literatury. To se dozvědí až v místě použití `\usebibtex`, což je typicky na konci dokumentu. Takže teprve třetí TEXování dá vše do pořádku.

Seznam literatury obsahuje po použití BibTEXu jen citovaná dílka. Pokud chcete do seznamu zařadit další položky, které nejsou v textu explicitně odkazovány příkazem `\cite`, použijte `\nocite[`⟨*lejblík*⟩`]`. Toto makro dá BibTEXu pokyn, aby do seznamu zahrnul i položku s ⟨*lejblíkem*⟩, ale v místě použití tohoto makra se nevytiskne nic. Konečně pomocí `\nocite[*]` dáváme BibTEXu vzkaz, že chceme mít v seznamu literatury celou `.bib` databázi.

Zdroj bibliografických záznamů může být ve více `.bib` souborech. V takovém případě stačí jejich názvy oddělit čárkou: `\usebibtex{`⟨*bib-báze1*⟩`,`⟨*bib-báze2*⟩`}{`⟨*bst-styl*⟩`}`.

Někdy se stane, že autoři `.bib` databází nebo `.bst` stylů neopustili při tvorbě těchto souborů LATEXový způsob myšlení a občas jim uklouzne nějaká LATEXová konstrukce z prstů až do počítače. Odtud se dostane do čteného `.bbl` souboru a náš plainTEX si s tím nebude vědět rady. K tomu slouží seznam `\bibtexhook`, kde můžete uvést definice těchto LATEXových konstrukcí. Tyto definice budou mít lokální platnost jen při čtení `.bbl` souboru. Například

```
\def\bibtexhook{\def\emph##1{{\em##1}}\def\frac##1##2{{##1\over##2}}}
```

Využití jednou vygenerované databáze. Tvorba seznamů literatury BibTEXem má jistou nevýhodu. Pokud později do dokumentu vložíte další `\cite[`⟨*lejblík*⟩`]`, musíte veškerou anabázi s bibTEXem provést znovu. A protože v současné době probíhá inflace odborných publikací způsobená tím, že se podle kvanta publikací a citací daňový poplatník rozhodl odměňovat vědce, je každé zjednodušení práce s bibliografickými záznamy přínosné. Makro OPmac navrhuje řešení, při kterém stačí použít BibTEX pro mnoho nových článků jen jednou.

1. Vytvořte si zvláštní dokument ⟨*mojebáze*⟩`.tex`, do kterého napíšete:

   `\input opmac  \genbbl{`⟨*bib-báze*⟩`}{`⟨*bst-styl*⟩`}  \end`

2. Po TEXování dokumentu ⟨*mojebáze*⟩`.tex` spusťte `bibtex` ⟨*mojebáze*⟩. Tím se vytvoří soubor ⟨*mojebáze*⟩`.bbl`.
3. Zpracujte TEXem soubor ⟨*mojebáze*⟩`.tex` ještě jednou. Vytvoří se seznam veškeré literatury, který byl v souboru ⟨*bib-báze*⟩`.bib`, přitom každá položka je označena svým ⟨*lejblíkem*⟩. Vytiskněte si tento výstup a dejte si jej na nástěnku.
4. Uložte soubor ⟨*mojebáze*⟩`.bbl` někam, kde jej umí přečíst TEX bez ohledu na to, v kterém pracujete adresáři.
5. Přejděte k editaci svého dokumentu, pište `\cite` nebo `\nocite` podle potřeby a v místě seznamu literatury dejte sekvenci `\usebbl/`⟨*typ*⟩ ⟨*mojebáze*⟩ . Údaj ⟨*typ*⟩ má tyto možnosti:

   `\usebbl/a` ⟨*mojebáze*⟩ `% vypsat kompletně celou` ⟨*mojebaze*⟩ `(a=all),`
   `\usebbl/b` ⟨*mojebáze*⟩ `% jen \(no)cite údaje řadit dle` ⟨*mojebaze*⟩ `(b=base),`
   `\usebbl/c` ⟨*mojebáze*⟩ `% jen \(no)cite řadit podle pořadí citace (c=cite).`

Kroky 2 až 4 budete muset opakovat pouze tehdy, když budete chtít přidat do ⟨*mojebáze*⟩`.bbl` další údaj, tj. po upgradu souboru ⟨*bib-báze*⟩`.bib`. Prudí-li různí odběratelé vaší vědecké činnosti požadavky na různé ⟨*bst-styly*⟩, stačí si vygenerovat podle různých stylů různé soubory typu `mybbl-plain.bbl`, `mybbl-ieeetr.bbl`.

Přímé čtení `.bib` databáze je možné pomocí makra `\usebib`. Je-li toto makro použito poprvé, natáhne se dodatečný modul `opmac-bib.tex`, který navíc potřebuje externí balíček na čtení `.bib` souborů `librarian.tex` od Paula Isamberta. Užití je podobné jako při `\usebbl`:

```
\usebib/c (⟨style⟩) ⟨bib-báze⟩ % řadit podle pořadí citace (c=cite),
\usebib/s (⟨style⟩) ⟨bib-báze⟩ % řadit podle klíče ve stylu (s=style).
```

Zde ⟨*bib-báze*⟩ je jeden nebo více `.bib` souborů oddělených čárkou bez mezery a bez přípony. Prametr ⟨*style*⟩ udává část jména souboru `opmac-bib-`⟨*style*⟩`.tex`, ve kterém je specifikace formátování položek. Součástí balíčku je styl `simple`, tedy soubor `opmac-bib-simple.tex`, který si může uživatel zkopírovat na jiný název a modifikovat dle svých představ. Více informací je na konci souboru `opmac-bib.tex`.

Formátování seznamu literatury je řízeno makrem `\printbib`, které je vloženo na začátek každé položky v seznamu. Implicitně makro tiskne čísla položek do hranatých závorek a při použití `\nonumcitations` předsadí první řádek položky a nepřidává nic. Makro může využít `\the\bibnum` pro tisk čísla nebo `\the\bibmark` pro tisk značky (při `\nonumcitations`). Příklady:

```
% Číslování položek bez hranatých závorek:
\def\printbib{\hangindent=\parindent \indent \llap{\the\bibnum. }}

% Tisk zkratek při použití bibTeXového stylu alpha a \nonumcitations:
\def\printbib{\hangindent=\parindent \noindent [\the\bibmark]\quad}
```

Další příklady (třeba jak TEX změří šířku největšího čísla a podle toho vypočítá odsazení celého seznamu) jsou uvedeny na http://petr.olsak.net/opmac-tricks.html.

16 Matematická sazba

Následující text popisuje vlastnosti souboru maker `ams-math.tex` (resp. `tx-math.tex` při použití některých PostScriptových fontů). Toto makro je makrem OPmac načítáno, takže uvedené vlastnosti jsou k dispozici též uživatelům OPmac.

V matematické sazbě (mezi dolary) nefungují přepínače textových fontů deklarovaných příkazem `\font`. Místo toho se tam přepíná mezi tzv. *matematickými abecedami*. Jakmile začnete psát písmena mezi dolary bez použití přepínače, je použita implicitní matematická abeceda `\mit`. Celkově jsou k dispozici následující abecedy:

`\mit`	`% matematická kurzíva`	$abc{-}xyz, ABC{-}XYZ$
`\it`	`% textová kurzíva`	$abc{-}xyz, ABC{-}XYZ$
`\rm`	`% textová antikva`	abc−xyz, ABC−XYZ
`\cal`	`% jednoduché cal. znaky`	$\mathcal{ABC}{-}\mathcal{XYZ}$
`\script`	`% kroucenější cal. znaky`	$\mathscr{ABC}{-}\mathscr{XYZ}$
`\frak`	`% fraktura`	$\mathfrak{abc}{-}\mathfrak{xyz}, \mathfrak{ABC}{-}\mathfrak{XYZ}$
`\bbchar`	`% zdvojené tahy`	$\mathbb{ABC}{-}\mathbb{XYZ}$
`\bf`	`% sans serif bold`	$\mathbf{abc}{-}\mathbf{xyz}, \mathbf{ABC}{-}\mathbf{XYZ}$
`\bi`	`% sans serif bold slanted`	$\boldsymbol{abc}{-}\boldsymbol{xyz}, \boldsymbol{ABC}{-}\boldsymbol{XYZ}$

Dále jsou v matematické sazbě k dispozici stovky symbolů dostupných pomocí \⟨*něco*⟩, například `\alpha` α, `\geq` $\geq$, `\sum` $\sum$, `\sphericalangle` $\sphericalangle$, `\bumpeq`, $\bumpeq$. Seznam všech těchto symbolů najdete v dokumentaci k AMSTEXu, která se typicky jmenuje `amsguide.ps`.

Matematická sazba funguje v nastavené velikosti písma podle `\typosize` nebo `\typoscale`. V nadpisech se doporučuje použít boldifikovanou sadu všech fontů, která se zapíná pomocí makra `\boldmath`. Například:

```
\def\nadpis#1\par{\centerline{\typosize[17/]\bf\boldmath #1}}
\nadpis Tady je nadpis včetně vzorce $\int_a^b f(x) {\rm d}x$
```

Makro `ams-math.tex` nastavuje jako implicitní abecedu matematickou kurzívu, která je mírně jinak kreslena než kurzíva textová. Naopak makro `tx-math.tex` nastavuje jako implicitní abecedu textovou kurzívu převzatou z „okolního“ textového fontu. Ta se ve vzorečcích spolu s textovým fontem bude esteticky snášet daleko lépe. Chcete-li toto implicitní chování změnit, je možné použít následující globální přepínače:

```
\itvariables    % implicitní abeceda bude textová kurzíva,
\mitvariables   % implicitní abeceda bude matematická kurzíva.
```

Poznámka: Vlastnosti popsané v této kapitole (AMS symboly, `\bbchar`, `\script`, `\frak`, sans serifový font po přepínačích `\bf`, `\bi` v math módu) vyžadují zavést speciální fonty (AMS fonty, EC fonty). Chcete-li se vyhnout této závislosti na fontech, můžete využít OPmac trik 0111.

17 Okraje

PlainTEX nastavuje levý okraj 1 in a šířku sazby (`\hsize`) nastavuje tak, aby i pravý okraj při formátu papíru „letter“ byl 1 in. Také horní a dolní okraj (do kterého přesahuje záhlaví a čísla stran) jsou nastaveny na 1 in při formátu papíru „letter“ a tím je určena výška sazby (`\vsize`). $\mathcal{CS}$plain dělá totéž, tj. okraje jsou 1 in, ale formát papíru je A4.[1] OPmac umožňuje toto nastavení změnit příkazem:

`\margins/`⟨*pg*⟩ ⟨*formát*⟩ `(`⟨*levý*⟩`,`⟨*pravý*⟩`,`⟨*horní*⟩`,`⟨*dolní*⟩`)`⟨*jednotka*⟩
`například:`
`\margins/1 b5 (2,2,2,2)cm % nastaví všechny okraje na 2 cm pro papír b5.`
⟨*pg*⟩`... 1 = shodné okraje pro všechny stránky,`
⟨*pg*⟩`... 2 = okraje pro liché stránky, sudé mají prohozeny` ⟨*levý*⟩`/`⟨*pravý*⟩`,`
⟨*formát*⟩`... a3, a4, a5, b5, letter, a3l, a4l, a5l či uživatelem definovaný,`
⟨*levý*⟩`,`⟨*pravý*⟩`,`⟨*horní*⟩`,`⟨*dolní*⟩`... velikosti okrajů,`
⟨*jednotka*⟩`... mm, cm, in, pt, pc, bp, dd, cc.`

Každý z parametrů ⟨*levý*⟩, ⟨*pravý*⟩, ⟨*horní*⟩, ⟨*dolní*⟩ může být prázdný. Jsou-li prázdné oba ⟨*levý*⟩ i ⟨*pravý*⟩, je zachováno nastavení `\hsize` a levý i pravý okraj je stejný. Je-li jen jeden z parametrů ⟨*levý*⟩, ⟨*pravý*⟩ prázdný, zůstává zachováno `\hsize` a neurčený okraj se dopočítá. Jsou-li ⟨*levý*⟩ i ⟨*pravý*⟩ neprázdné, jsou oba okraje určeny a je podle nich upraveno `\hsize`. Analogické pravidlo platí pro ⟨*horní*⟩, ⟨*dolní*⟩ v souvislosti s výškou sazby `\vsize`. Například

```
\margins/2 a4 (,18,,)mm  % vnější okraj na dvojstraně 2*a4 je 18mm
                         % \hsize, \vsize beze změny.
```

Údaj ⟨*formát*⟩ může být též ve tvaru `(`⟨*šířka*⟩`,`⟨*výška*⟩`)`⟨*jednotka*⟩, kde ⟨*jednotka*⟩ je nepovinná a pokud chybí, použije se jednotka za údaji s okraji. Tedy třeba

```
\margins/1 (100,200) (7,7,7,7)mm
```

deklaruje papír o rozměru 100×200 mm a s okraji 7 mm po každé straně. Mezery před a za údajem ⟨*formát*⟩ nelze vynechat.

Uživatel může také před použitím `\margins` definovat vlastní ⟨*formát*⟩ papíru pomocí příkazu `\sdef{pgs:`⟨*formát*⟩`}{(`⟨*šířka*⟩`,`⟨*výška*⟩`)`⟨*jednotka*⟩`}`.

Celou sazbu na úkor okrajů je možno zvětšit/zmenšit příkazem `\magscale[`⟨*factor*⟩`]`. Například `\magscale[500]` zmenší sazbu na polovinu. Při této změně zůstává na místě „Knuthův bod“, tj. bod o souřadnicích 1 in, 1 in od horního a levého okraje. Sazba samotná je zalomena zcela stejně. Jednotky použité v dokumentu jsou od této chvíle relativní. Například po `\magscale[2000]` je použitá jednotka v dokumentu `1mm` ve skutečnosti `2mm`. Makro `\magscale` ponechává nezměněny jen rozměry stránek dané formátem stránek (a4, a3, atd.). Možnost použití makra: `\magscale[1414] \margins/1 a4 (,,,)mm` umístí sazbu, která je určena pro tisk na a5, doprostřed stránky a4 a odpovídajícím způsobem ji zvětší, aby se to korektorům lépe četlo.

[1] Přesněji: $\mathcal{CS}$plain nastavuje výšku sazby `\vsize=239.2mm`, což vede k dolnímu okraji o 7 mm většímu než 1 in.

18 Poslední strana

Číslo poslední strany dokumentu (to nemusí být počet stran) je uloženo při opakovaném zpracování TEXem v registru `\lastpage`. K tomu musí být otevřen soubor REF s daty pro křížové odkazy, rejstřík a obsah. Pokud pracujete s těmito daty, je soubor REF automaticky otevřen. Pokud ne, můžete si vynutit jeho otevření příkazem `\openref`. Není-li soubor REF otevřen, je hodnota registru `\lastpage` rovna nule. Číslování stránek ve tvaru ⟨*číslo*⟩/⟨*počet stran*⟩ zajistíte například takto:

```
\footline={\hss \rm \thefontsize[10]\the\pageno/\the\lastpage \hss}
```

19 Další jazyk

OPmac přímo podporuje češtinu, slovenštinu a angličtinu. Ostatní jazyky lze použít také, ale je potřeba vynaložit nepatrné úsilí. Především je potřeba použít TEXový stroj a formát takový, aby v něm byly podporovány vzory dělení příslušného jazyka. $\mathcal{CS}$plain pro pdfTEX podporuje v implicitním nastavení vzory dělení 16 jazyků (od března 2019) a $\mathcal{CS}$plain pro LuaTEX podporuje vzory dělení všech jazyků dostupných v TEXové distribuci (kolem 57 jazyků). Doporučuji tedy pro více jazyků použít LuaTEX.

Následuje ukázka pro španělštinu. Ukázku můžete zpracovat $\mathcal{CS}$plainem v LuaTEXu, tedy příkazem `luatex -fmt pdfcsplain dokument`.

```
\input opmac
\input lmfonts  % Unicode fonty

\sdef{mt:chap:es}{Capítulo}   % "Kapitola" v jazyku es
\sdef{mt:t:es}{Cuadro}        % "Tabulka" v jazyku es
\sdef{mt:f:es}{Figura}        % "Obrázek" v jazyku es

\eslang % Španělské vzory dělení slov + deklarovaná slova v jazyku "es".

\sec Mañana

Mañana.

\caption/f Test % vytvoří text "Figura 1.1 Test"

\bye
```

V ukázce je patrné, že nová tři slova ve španělštině je třeba přidat pod zkratku jazyka `es` (podle ISO 639-1). Tato slova začnou být používána automaticky po zapnutí vzorů dělení slov, tedy po použití přepínače `\eslang`.

Pokud používáte OPmac s formátem generovaným z `etex.src` (tj. příkaz `xetex dokument` nebo `luatex dokument`), pak nefunguje přepínač `\eslang` a místo něj zapnete vzory dělení pomocí `\uselanguage{espanol}`. Navíc v takovém případě musíte přiřadit dlouhému názvu jazyka jeho zkratku podle ISO 639-1 příkazem `\isolangset{espanol}{es}`.

Při abecedním řazení rejstříku pomocí `\makeindex` může OPmac ohlásit (není-li použit $\mathcal{CS}$plain) varování: “falling back to ASCII sorting”. Pokud chcete dodržet pravidla řazení daného jazyka, je proto nutné definovat makro `\sortingdata`⟨*iso-kód*⟩. Dále můžete přidat makro `\specsortingdata`⟨*iso-kód*⟩. Příklad:

```
\def\sortingdataes {aAäÄáÁ,bB,cCçÇ,^^P^^Q^^R,dD,...,zZ,.}
\def\specsortingdataes {ch:^^P Ch:^^Q CH:^^R}
```

Prvé makro obsahuje skupiny znaků oddělené od sebe čárkami a ukončené čárkou-tečkou. Znaky v každé skupině jsou nerozlišitelné při prvém průchodu (primární řazení). Jsou-li z tohoto

pohledu některé údaje shodné, je na tyto údaje použito sekundární řazení, ve kterém už jsou všechny znaky rozlišeny a seřazeny v pořadí, jako v makru. Poznámka: tři tečky v příkladu si musí španělsky mluvící uživatel doplnit za reálná data sám.

Řadicí algoritmus umožňuje skupiny znaků (spřežky Ch, Dz atd.) interpretovat jako jediný znak. K tomu slouží makro `\specsortingdata`⟨*iso-kód*⟩. V něm jsou mezerami od sebe odděleny údaje ve tvaru ⟨*skupina*⟩`:`⟨*token*⟩. OPmac provede výměnu ⟨*skupina*⟩ za ⟨*token*⟩ před zahájením řadicího algoritmu, takže ⟨*token*⟩ je možné vložit do `\sortingdata`⟨*iso-kód*⟩ jako jeden znak. Je nutné používat znaky, které nejsou součástí abecedy jazyka, například `^^A`, `^^B` atd. Přitom je potřeba se vyhnout znakům `^^I` a `^^M` , protože tyto mají speciální kategorii.

OPmac definuje interní makro `\sortingdata`, které je možné použít pro češtinu, slovenštinu a angličtinu. Makro `\sortingdata`⟨*iso-code*⟩ má přednost před interním makrem `\sortingdata`, je-li definováno a jazyk odpovídající ISO kódu má aktivovány vzory dělení slov.

Konečně můžete pozměnit seznam ignorovaných znaků (z hlediska řazení), který je uveden v makru `\setignoredchars` takto:

```
\def\setignoredchars{\setlccodes ,.;.?.!.:.'."|.(.).[.].<.>.=.+.{}{}}
```

Toto makro můžete předefinovat, ale ponechte v předu `\setlccodes`, vzadu `{}{}` a za každým znakem pište tečku, která je implicitním ignorovaným znakem.

20 Předdefinované styly dokumentů

Implicitní styl dokumentu při použití OPmac zahrnuje výchozí parametry a vzhled sazby jako v plainTEXu. Pochopitelně uživatel si styl dokumentu obvykle upraví pomocí svých maker a třeba s využitím maker OPmac.

Od března 2019 nabízí OPmac kromě toho zkratky s výchozím nastavením jiných stylů, což je možné použít, pokud chce uživatel vytvořit běžný typ dokumentu a nechce se zdržovat nastavováním parametrů a definicí vlastních maker. Jsou připraveny deklarační makra `\report` a `\letter`, která je možné použít v záhlaví dokumentu.

Deklarace `\report` je určena pro psaní zpráv. Písmo je nastaveno na 11 bodů, odsazení odstavců je 1,2 em. Makro `\tit` použije menší font než obvykle, protože se nepředpokládá, že bude použita úroveň „kapitola“. První stránka je nečíslovaná, ostatní jsou číslovány (od strany 2). Poznámky pod čarou jsou číslovány vzestupně v celém dokumentu. Při použití tohoto makra v záhlaví dokumentu je připraveno též makro `\author`⟨*text*⟩⟨*konec řádku*⟩, které umístí ⟨*text*⟩ v kurzívě doprostřed řádku. Při použití `\nl` je text rozdělen do více řádků, každý je centrován.

Deklarace `\letter` je určena pro psaní dopisů. Písmo je nastaveno na 11 bodů, odsazení není žádné a mezi odstavci je půlřádková mezera. Stránky nejsou číslovány. Chcete-li je číslovat, můžete použít například kód uvedený v sekci 18. Při použití tohoto makra je dále připraveno makro `\address` pro psaní víceřádkových adres a makro `\subject` které vypíše tučně slovo „Věc:“ nebo „Vec:“ nebo „Subject:“ v závislosti na použitém jazyce. Užití makra `\address` vypadá takto:

```
\address
  ⟨první řádek adresy⟩
  ⟨druhý řádek adresy⟩
  ⟨atd.⟩
  ⟨prázdný řádek⟩
```

tedy řádky se neukončují žádným speciálním znakem, ale konec řádku na vstupu je koncem řádku i v sazbě. Makro vytvoří `\vtop` se sazbou řádků tak, že jeho šířka odpovídá nejširšímu řádku. Chcete-li mít víceřádkovou adresu u pravého okraje stránky stačí tedy psát `\hfill\address...` A před adresu lze také do prvního řádku předsadit text jednoduše pomocí ⟨*text*⟩`\address...`

Analogický deklarátor \book není zahrnut, protože kniha vyžaduje individuální péči typografa a každá kniha může být jiná. Na takový projekt je účelné si připravit vlastní makra. Deklarátor \slides také není zahrnut, ale je možné použít OPmac triky 0017 a 0022.

21 Shrnutí

```
\tit  ⟨Název titulu (ukončený koncem řádku)⟩
\chap ⟨Název kapitoly (ukončený koncem řádku)⟩
\sec  ⟨Název sekce (ukončený koncem řádku)⟩
\secc ⟨Název podsekce (ukončený koncem řádku)⟩

\maketoc          % generování obsahu
\ii ⟨heslo⟩,⟨heslo⟩ % vložení hesel do rejstříku
\makeindex        % generování rejstříku

\label[⟨lejblík⟩] % deklarace před cílem odkazu
\ref[⟨lejblík⟩]   % odkaz na kapitolu, sekci nebo podsekci
\pgref[⟨lejblík⟩] % odkaz na stránku

\caption/t      % číslovaný popisek tabulky
\caption/f      % číslovaný popisek obrázku
\eqmark         % číslovaná rovnice

\begitems           % začátek výčtu položek
\enditems           % konec výčtu položek
\begtt              % začátek verbatim výpisu
\endtt              % konec verbatim výpisu
\activettchar ⟨znak⟩ % inicializace znaku pro in-text verbatim
\verbinput          % verbatim výpis ze souboru
\begmulti           % začátek vícesloupcové sazby
\endmulti           % konec vícesloupcové sazby

\cite[⟨lejblíky⟩]  % místo, odkud se odkazuje do seznamu literatury
\rcite[⟨lejblíky⟩] % jako \cite, ale bez závorek kolem
\sortcitations \shortcitations \nonumcitations % deklarace typu odkazů
\bib[⟨lejblík⟩]    % položka v seznamu literatury
\usebibtex{⟨bib-báze⟩}{⟨bst-styl⟩}  % použití bibTeXu pro seznam literatury
\genbbl{⟨bib-báze⟩}{⟨bst-styl⟩}     % předgenerování seznamu literatury
\usebbl/? ⟨bbl-báze⟩  % použití předgenerované databáze, ? je z {a,b,c}
\usebib/? (⟨style⟩) ⟨bib-báze⟩ % přímé čtení .bib bez BibTeXu, ? je z {c,s}

\fontfam[⟨název fontové rodiny⟩]                  % výběr fontové rodiny
\typosize[⟨font-velikost⟩/⟨řádkování⟩]            % nastavení velikosti sazby
\typoscale[⟨faktor-font⟩/⟨faktor-řádkování⟩]      % škálování velikosti sazby
\thefontsize[⟨velikost⟩] \thefontscale[⟨faktor⟩]  % velikost aktuálního fontu

\inspic ⟨soubor⟩.⟨přípona⟩ % vložení obrázku, přípony: jpg, png, pdf
\table{⟨pravidlo⟩}{⟨data⟩}  % makro pro tabulku
\frame{⟨text⟩}             % orámovaný text
\fnote{⟨text⟩}  % poznámka pod čarou (lokální číslování na každé stránce)
\mnote{⟨text⟩}  % poznámka na okraji (pravém nebo levém podle stránky)

\hyperlinks{⟨barva-in⟩}{⟨barva-out⟩}  % v PDF budou odkazy klikací
```

```
\outlines{⟨úroveň⟩}                    % PDF bude mít záložku s obsahem

\magscale[⟨faktor⟩]  % zvětšení/zmenšení sazby beze změny zlomu
\margins/⟨pg⟩ ⟨formát⟩ (⟨levý⟩,⟨pravý⟩,⟨horní⟩,⟨dolní⟩)⟨jednotka⟩  % okraje

\report \letter % inicializace předdefinovaného stylu dokumentu
```