

# Manuál k $\text{\LaTeX}$

Petr Olšák

Autor  $\mathrm{T\!E\!X}$ u je profesor Donald Knuth.

$\mathrm{T\!E\!X}$  je ochranná známka American Mathematical Society.

Ostatní v manuálu použité názvy programových produktů, firem apod. mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

Tento text je volně šířen společně s balíkem  $\mathcal{C\!S\!T\!E\!X}$ . Jedná se o referenční manuál k tomuto balíku. Text je k dispozici ve formátech `tex` ( $\mathcal{C\!S\!plain}$ ), PostScript a PDF. Výchozí adresa textu je `ftp://math.feld.cvut.cz/pub/cstex/doc`.

Text můžete tisknout a jinak používat pro soukromé účely. V nezměněném stavu v elektronické podobě jej můžete také distribuovat bez dalšího omezení. Není dovoleno tento text zveřejňovat v pozměněném stavu a publikovat jej v papírové podobě bez předchozí dohody s autorem.

This document is free distributed as a part of the  $\mathcal{C\!S\!T\!E\!X}$  package. It is a reference manual of the  $\mathcal{C\!S\!T\!E\!X}$ . It is available in formats `tex` ( $\mathcal{C\!S\!plain}$ ), PostScript and PDF. The base URL of it is `ftp://math.feld.cvut.cz/pub/cstex/doc`.

You can print this document or use it in another way only as an individual end-user. If you don't do any change in this document then you can distribute it in electronical form without any limitation. It is not permitted to distribute this document in changed versions or to publish it in hardcopy form without previous agreement with the author.

This original is written in Czech language. The translation into other languages are welcome.

Verze textu 15. 5. 2003

Verze textu 30. 11. 2012

© RNDr. Petr Olšák, 2002, 2012

# Obsah

1. Úvod . . . . .	4
2. $\mathcal{C}\mathcal{S}$ fonty . . . . .	5
2.1 Odlišnosti od CM fontů . . . . .	8
2.2 Struktura METAFONTových zdrojových souborů $\mathcal{C}\mathcal{S}$ fontů . . . . .	8
2.3 Alternativní hyphenchar . . . . .	9
2.4 Anabáze UNIXových hard-linků . . . . .	9
2.5 Virtuální fonty přidané do balíčku $\mathcal{C}\mathcal{S}$ fontů . . . . .	10
2.6 Historie a budoucnost $\mathcal{C}\mathcal{S}$ fontů . . . . .	11
3. <code>cspsfonts.tar.gz</code> – 35 základních PostScriptových fontů . . . . .	13
3.1 Historie a budoucnost balíčku <code>cspsfonts.tar.gz</code> . . . . .	16
4. Formát $\mathcal{C}\mathcal{S}$ plain . . . . .	17
4.1 Překódování v input procesoru $\mathrm{T}\mathrm{E}\mathrm{X}$ u . . . . .	17
4.2 Inicializace formátu . . . . .	19
4.3 Výchozí nastavení v $\mathcal{C}\mathcal{S}$ plainu . . . . .	21
4.4 Makro <code>opmac.tex</code> . . . . .	22
4.5 Použití PostScriptových fontů v $\mathcal{C}\mathcal{S}$ plainu . . . . .	23
4.6 UTF-8 kódovaný $\mathcal{C}\mathcal{S}$ plain . . . . .	26
4.7 Použití fontů kódovaných podle Corku (T1 kódování) nebo v Unicode . . . . .	28
4.8 Vzory dělení slov v různých jazycích a kódováních . . . . .	30
4.9 Nové řídicí sekvence $\mathcal{C}\mathcal{S}$ plainu oproti Knuthovu plainu . . . . .	32
4.10 $\mathcal{C}\mathcal{S}$ plain a $\mathcal{A}\mathcal{M}\mathcal{S}\mathrm{T}\mathrm{E}\mathrm{X}$ . . . . .	36
4.11 $\mathrm{pdf}\mathrm{T}\mathrm{E}\mathrm{X}$ + $\mathcal{C}\mathcal{S}$ plain = $\mathrm{pdf}\mathcal{C}\mathcal{S}$ plain . . . . .	36
4.12 Historie a budoucnost $\mathcal{C}\mathcal{S}$ plainu . . . . .	36
5. Formát $\mathcal{C}\mathcal{S}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ . . . . .	38
5.1 Různé $\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ y . . . . .	38
5.2 Záhlaví $\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ ového dokumentu . . . . .	41
5.3 Vlastnosti $\mathcal{C}\mathcal{S}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ u . . . . .	42
5.4 Vlastnosti stylových souborů <code>czech.sty</code> a <code>slovak.sty</code> . . . . .	43
5.5 PostScriptové fonty v $\mathcal{C}\mathcal{S}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ u . . . . .	44
5.6 Použití fontů kódovaných podle Corku (T1 kódování) v $\mathcal{C}\mathcal{S}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ u . . . . .	45
5.7 $\mathrm{pdf}\mathrm{T}\mathrm{E}\mathrm{X}$ + $\mathcal{C}\mathcal{S}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ = $\mathrm{pdf}\mathcal{C}\mathcal{S}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ . . . . .	45
5.8 Historie a budoucnost $\mathcal{C}\mathcal{S}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ u . . . . .	45
6. Reference . . . . .	47

# 1. Úvod

Po skoro deseti letech se k tomuto textu vracím, abych jej aktualizoval. Veškeré změny jsou vyznačeny touto barvou. Černý text je naopak původní z roku 2003. Nabízím čtenáři dva bonbónky v jednom: podrobnou technickou dokumentaci i pohled do historie.

V tomto manuálu se pokusím vyčerpávajícím způsobem popsat vlastnosti  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$ u. Z tohoto důvodu to asi nebude jednoduché čtení pro  $\mathrm{T}\mathrm{E}\mathrm{X}$ ové začátečníky. Těm doporučuji nejprve přečíst [7].

$\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$  je sada  $\mathrm{T}\mathrm{E}\mathrm{X}$ ových maker, fontů, vzorů dělení slov a doplňujícího software pro podporu české a slovenské sazby v  $\mathrm{T}\mathrm{E}\mathrm{X}$ u. Je vytvořen tak, aby mohl být použit na libovolné  $\mathrm{T}\mathrm{E}\mathrm{X}$ ové distribuci na libovolném operačním systému.

Dříve byla slovem  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$  označována také kompletní  $\mathrm{emT}\mathrm{E}\mathrm{X}$ ová distribuce doplněná o zmíněná makra, fonty a vzory dělení. To bylo v roce 1993, kdy sdružení  $\mathcal{CS}\mathrm{TUG}$  rozesílalo pod tímto názvem  $\mathrm{emT}\mathrm{E}\mathrm{X}$ ovou distribuci na disketách svým členům. V té době většina členů používala DOS, takže  $\mathrm{emT}\mathrm{E}\mathrm{X}$  pro DOS byl pro ně vyhovující. V dnešní době uživatelé pracují s nejrůznějšími operačními systémy a s různými distribucemi  $\mathrm{T}\mathrm{E}\mathrm{X}$ u pro tyto systémy. Z toho důvodu se v tomto manuálu přidržíme jen užšího významu slova  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$ , definovaného v předchozím odstavci.

$\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$  sestává ze ~~tří~~ **dvou** základních pilířů:  $\mathcal{CS}$ fonty,  $\mathcal{CS}$ plain a  ~~$\mathcal{CS}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$~~ .  $\mathcal{CS}$ fonty jsou konzervativním rozšířením Knuthových Computer Modern fontů,  $\mathcal{CS}$ plain je konzervativním rozšířením Knuthova formátu plain a konečně  $\mathcal{CS}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$  je **byl** jistou modifikací běžně používaného formátu  $\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ . Tyto ~~tři~~ pilíře jsou podrobně dokumentovány v následujících kapitolách. K těmto pilířům  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$ u je ještě připojena podpora použití základních 35 PostScriptových fontů v češtině a slovenštině prostřednictvím virtuálních fontů.

Základní balíčky  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$ u přístupné na Internetu mají následující názvy:

<code>csfonts.tar.gz</code>	... $\mathcal{CS}$ fonty -- Metafontové soubory a metriky.
<code>csplain.tar.gz</code>	... Makra pro formát $\mathcal{CS}$ plain a vzory dělení slov.
<code>cslatex.tar.gz</code>	... Makra pro formát $\mathcal{CS}\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ . -- <b>zastaralé</b>
<code>cspsfonts.tar.gz</code>	... Virtuální fonty základních 35 PostScriptových fontů v~kódování podle $\mathcal{CS}$ fontů.

Pokud někdo řekne, že má na své distribuci  $\mathrm{T}\mathrm{E}\mathrm{X}$ u instalován  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$ , pak to znamená, že má určitě instalovány tyto čtyři balíky. Kromě toho k  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$ u řadíme i následující doplňující software, který už nemusejí mít všichni uživatelé  $\mathcal{CS}\mathrm{T}\mathrm{E}\mathrm{X}$ u instalován:

csfonts-t1.tar.gz .. Varianta CSfontů ve formátu PostScript Type1.  
 enctex.tar.gz ... Makra a návod na modifikaci tex.ch souboru na  
~~zahrnutí podpory přímého přístupu k vektorům~~  
~~xord/xchr prostřednictvím primitivů.~~  
 pro vícebytové překódování v input procesoru.  
 cstrip.tar.gz ... Test funkcionality formátu CSplain.  
 csindex-19980713.tar.gz ... Zdrojové soubory v jazyce C programu  
 csindex, který je modifikací programu  
 makeindex se zahrnutou podporou českého a  
 slovenského řazení.  
 vlna.tar.gz ... Zdrojové soubory v jazyce C a v jazyce cweb  
 programu vlna na automatické doplňování vlnek  
 za neslabičné předložky.

Na Internetu naleznete všechny uvedené balíky na

<ftp://math.feld.cvut.cz/pub/cstex/base>

Základní balíčky a csfonts-t1.tar.gz obsahují adresářovou strukturu podle TDS (T<sub>E</sub>X directory standard), takže pokud vaše distribuce T<sub>E</sub>Xu používá tento standard, pak rozbalení balíčků nad adresářem texmf povede automaticky k zařazení všech souborů na správná místa v texmf stromu.

Starší verze uvedených balíčků najdete na

<ftp://math.feld.cvut.cz/pub/cstex/base/old>

Dokumentaci k C<sub>S</sub>T<sub>E</sub>Xu (včetně tohoto manuálu) najdete na

<ftp://math.feld.cvut.cz/pub/cstex/doc>

Adresář <ftp://math.feld.cvut.cz/pub/cstex/old/> obsahuje kromě zmíněných věcí kompletní distribuci emT<sub>E</sub>Xu s podporou C<sub>S</sub>T<sub>E</sub>Xu (v podadresáři emtex), minimalizovanou distribuci emT<sub>E</sub>Xu+C<sub>S</sub>T<sub>E</sub>Xu pro pouhé tři instalační disky (v podadresáři minitex), návody a software na instalaci web2c T<sub>E</sub>Xu (v podadresáři web2c) a RPM balíky teT<sub>E</sub>Xu podporující C<sub>S</sub>T<sub>E</sub>X pro různé distribuce Linuxu (v podadresáři tetex-rpm). Tyto podadresáře vznikaly postupně v poslední dekádě minulého tisíciletí podle potřeby a zájmu o uvedené distribuce T<sub>E</sub>Xu. Podle toho, jak rostla a klesala popularita jednotlivých distribucí, jsou tyto podadresáře aktualizovány (nebo spíše neaktualizovány). Na druhé straně mohou vzniknout další adresáře s balíčky dalších distribucí T<sub>E</sub>Xu obsahujících C<sub>S</sub>T<sub>E</sub>X – stačí, když je někdo udělá a pošle o tom informaci na mou elektronickou adresu. Mohu třeba založit podadresář miktex-zip analogicky k adresáři tetex-rpm, pokud se to někomu bude hodit a pokud ty archivy samozřejmě udělá.

## 2. C<sub>S</sub>fonty

C<sub>S</sub>fonty jsou konzervativním rozšířením CM fontů Donalda Knutha. Tím je míněno, že každý C<sub>S</sub>font má svůj protějšek v nějakém CM fontu, přičemž se naprosto shoduje v kódování, tvarech a šířkách znaků na prvních 128 pozicích (kódy 0 až 127),

tj. na všech pozicích, které jsou tímto CM fontem použity. ČSfont tedy pouze doplňuje další znaky do pozic s kódem větším než 127. Tam jsou umístěny znaky české a slovenské abecedy podle kódování ISO 8859-2. Některé pozice stále zůstávají neobsazeny. Níže uvádím tabulku ČSfontu. Druhý znak na pozicích 0B až 0F se vyskytuje namísto prvního znaku ve fontech bez ligatur typu fi (*csr5*, *cstt\** a *csslitt\**), druhý znak na pozicích 20, 3C, 3E, 5C, 5F a 7B až 7D najdeme ve strojopisných fontech (*cstt\** a *cslstt\**) a konečně libra místo dolaru na pozici 24 se vyskytuje v kurzívách (*csti\**). Tyto nejednoznačnosti v kódování mají samozřejmě i původní CM fonty.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x	Γ	Δ	Θ	Λ	Ξ	Π	Σ	Υ	Φ	Ψ	Ω	ff↑	fi↓	fl'	ffi <sub>j</sub>	ffl <sub>z</sub>
1x	ı	j	`	´	˘	˙	–	°	‚	ß	æ	œ	ø	Æ	Œ	Ø
2x	¬	!	”	#	\$ £	%	&	’	( )	*	+	,	-	.	/	
3x	0	1	2	3	4	5	6	7	8	9	:	;	ı<	=	ı>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	“\	]	^	’_
6x	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	–{	—	”}	~	¨
8x														‰		
9x									À				-	„	«	»
Ax						Ě				Š		Ť			Ž	
Bx						Ĭ			à	š		ť			ž	
Cx	Ř	Á			Ä	Í			Č	É			Ě	Í		Ď
Dx			Ň	Ó	Ô		Ö		Ř	Ů	Ú		Ů	Ý		
Ex	í	á			ä	í			č	é			ě	í		ď
Fx			ň	ó	ô		ö		ř	ů	ú		ü	ý	„	“

Následuje seznam všech CM fontů. Pokud není vpravo uveden alternativní název ČSfontu, jedná se o matematický font, který nemá v ČSfontech alternativu. Také ji nepotřebuje.

CM font	ČSfont
cmr17, 12, 10, 9, 8, 7, 6, 5	csr17, 12, 10, 9, 8, 7, 6, 5
cmbx12, 10, 9, 8, 7, 6, 5	csbx12, 10, 9, 8, 7, 6, 5
cmsl12, 10, 9, 8	cssl12, 10, 9, 8
cmtt12, 10, 9, 8	cstt12, 10, 9, 8
cmslitt10, cmvtt10	csslitt10, csvtt10
cmss17, 12, 10, 9, 8	csss17, 12, 10, 9, 8
cmssi17, 12, 10, 9, 8	csssi17, 12, 10, 9, 8
cmssdc10, cmssbx10	csssdc10, csssbx10
cmssqi8, cmssq8	csssqi8, csssq8

cmdunh10, cmbxsl10, cmb10 cmff10, cmfib10	csdunh10, csbxsl10, csb10 csff10, csfib10
cmti12, 10, 9, 8, 7 cmbxti10, cmitt10 cmu10, cmfi10	csti12, 10, 9, 8, 7 csbxti10, csitt10 csu10, csfi10
cmcsc10, cmtcsc10	cscsc10, cstcsc10
cminch10	csinch10
cmmi12, 10, 9, 8, 7, 6, 5 cmmib10	
cmtex10, 9, 8	
cmsy10, 9, 8, 7, 6, 5 cmbsy10	
cmex10	

Mezi soubory metrik  $\mathcal{C}\mathcal{S}$ fontů navíc najdeme metriky vytvořené Sauterovou extrapolací, které nemají přímou obdobu mezi CM fonty:

$\mathcal{C}\mathcal{S}$ font
csb17, 12, 9, 8, 7, 6, 5 csbxsl12, 5, 6, 7, 8, 9 csbxti17, 12 cscsc17, 12 csdunh17, 12, 5, 6, 7, 8, 9 csfib12, 10, 9 csitt12, 17, 8, 9 cssl17, 5, 6, 7 cssltt12, 8, 9 csssbx12, 17, 9 cstcs12, 17 csti17 csu12, 17, 7, 8, 9 csvtt12, 8, 9

METAFONTová makra pro Sauterovu extrapolaci jsou součástí balíčku s  $\mathcal{C}\mathcal{S}$ fonty `csfonts.tar.gz`. Princip této extrapolace je například popsán v [5]. Fonty vytvořené Sauterovou extrapolací nedoporučujeme používat, protože nemají svou implementaci pomocí PostScriptového fontu. Do výstupního PDF se při použití takového fontu zavede bitmapový výstup z METAFONTu, což může vést ke komplikacím.

## 2.1 Odlišnosti od CM fontů

Nelze tvrdit, že text používající jen znaky z pozic 0–127 bude 100% shodně zpracován při použití CM fontů i  $\mathcal{C}$ sfontů. Odlišnosti existují, ale jsou tak nepatrné, že je velmi malá pravděpodobnost, že by při běžném užívání byla pozorovatelná rozdílnost. Nicméně přesto zde všechny odlišnosti uvádím včetně komentářů. Uvedené hodnoty jsou příkladem při srovnání fontu `csr10` s `cmr10`.

### 1. Kerningové páry

Mezi tečkami (..) je v `csr10` implicitní kern, aby bylo možno sázet elipsu. Kern 0,011111pt. V `cmr10` není.

Dvojice `ka` – `csr10`: –0,0027777pt, `cmr10`: –0,0055555pt.

Dvojice `P.` a `P`, – `csr10`: –0,0027777pt, `cmr10` není.

Dvojice `F.`, `F`,, `V.`, `V`,, `W.` a `W`, – `csr10`: –0,0055555pt, `cmr10` není.

Dvojice `Av` a `Aw` – `csr10`: –0,011111pt, `cmr10` není.

### 2. Ligatury

Dvojice `<<` vede v `csr10` na francouzské uvozovky, kód 158, v `cmr10` není.

Dvojice `>>` vede v `csr10` na francouzské uvozovky, kód 159, v `cmr10` není.

### 3. Výšky znaků

Formát `tfm` je omezen na maximálně 16 různých výšek znaků v jednom fontu. V `cmr10` je obsazeno všech 16 různých výšek. Přitom v `csr10` přicházejí další výšky znaků dané akcentovanými znaky. Proto METAFONT provedl v `csr10` jistá zao-krouhlení, která způsobí odlišnost výšek od výšek v `cmr10` maximálně o 0,007779pt. Jedná se o tyto znaky:

- $\Gamma$  až  $\Omega$ ,  $\mathcal{A}$ ,  $\mathcal{E}$  a všechny kapitálky: v `csr10` jsou menší o 0,00773pt.
- Nadřítka (kód 22), nadpuntík (kód 95) a přehláska (kód 127): v `csr10` větší o 0,007779pt.
- Písmena `i`, `j` jsou v `csr10` větší o 0,007779pt.
- Znak plus (+) je v `csr10` menší o 0,007778pt.

Rozdílnost výšek není kritická, protože při sazbě se většinou berou v úvahu jen šířky znaků. Pouze výjimečně promluví do sazby i výška (většinou když objekt v řádku je větší než `\baselineskip`).

## 2.2 Struktura METAFONTových zdrojových souborů $\mathcal{C}$ sfontů

METAFONT čte nejprve soubor, který má stejné jméno, jako je jméno generovaného fontu, a má příponu `.mf`. Například `csr10.mf`. Takovému souboru říkáme hlavní soubor generování fontu. V  $\mathcal{C}$ sfontech je takových hlavních souborů celkem 121. Všechny mají stejný dvouřádkový obsah:

```
input cscope
use_driver;
```

V souboru `cscope.mf` se načtou jména a kódy akcentovaných znaků (jsou uspořádány podle ISO 8859-2) a definuje se `use_driver` v závislosti na jménu hlavního



souboru tak, že se načte odpovídající hlavní soubor CM fontů (tj. odpojí se předpona `cs` a připojí předpona `cm`). V případě fontu `csr10` se tedy načte v tuto chvíli soubor `cmr10.mf`. Navíc je v souboru `cscode.mf` předefinováno `generate` (v CM fontech má význam `input`) tak, že poslední řádek `cmr10.mf` s textem

```
generate roman
```

neprovede `input roman.mf`, ale provede `input kmroman.mf`. Další průběh výpočtu je tedy zase v režii  $\mathcal{C}$ fontů. Kromě souboru `kmroman.mf` jako alternativy k souboru `roman.mf` z CM fontů najdeme analogické alternativy s názvy `kmtexit.mf`, `kmcsc.mf`, `kmtexset.mf` a `kmtitle.mf`.

Z uvedených souborů se postupně načítají soubory generující tvary jednotlivých znaků. Jedná o všechny odpovídající soubory z CM fontů a navíc soubory uvedené v následující tabulce.

- `csaccent.mf` definuje makra pro akcenty,
- `csacutl.mf` generuje á, é, í, ó, ř, ú, ý,
- `csacutu.mf` generuje Á, É, Í, Ó, Ř, Ú, Ý,
- `cshachel.mf` generuje č, ě, ň, ř, š, ž,
- `cshacheu.mf` generuje Č, Ď, Ě, Ň, Ř, Š, Ť, Ž,
- `csotherl.mf` generuje ô, û, à, í, ò, ð, ĩ, ä, ö, ü,
- `csotheru.mf` generuje Ô, Û, Â, Ì, Ä, Ö, Ü,
- `csadded.mf` generuje „, “, », «, ‰, ,
- `cshyph.mf` generuje alternativní hyphenchar.

V souboru `kmroman.mf` a jemu podobných souborech jsou ještě zapsány kernové páry a údaje pro tabulku ligatur. V tomto souboru činnost METAFONTu končí příkazem `bye`.

## 2.3 Alternativní hyphenchar

Na pozici 156 je v  $\mathcal{C}$ fontech spojovník s úplně stejnou kresbou a metrikou, jako na pozici 45. Nastavíme-li `\hyphenchar\beznyfont=156`, budeme mít zaručeno, že ve slovech „je-li“ nebude  $\text{\TeX}$  dělit slovo. Bez tohoto nastavení by  $\text{\TeX}$  rozdělil „je-/li“, což není v souladu s požadavky na českou sazbu. Mnoho dalších způsobů řešení tohoto problému najdeme v [6].

Další aplikací znaku 156 je modifikace jeho metriky tak, aby kresba přesahovala přes šířku znaku výrazně doprava. Pak nastavením `\hyphenchar\beznyfont=156` dosáhneme tzv. visící interpunkce, viz například [4] nebo [6]. V poslední době se visící interpunkce stává módní záležitostí, protože to umí i nejnovější verze programu Adobe InDesign. Metriku  $\mathcal{C}$ fontu můžete pro vlastní potřeby modifikovat například programy `tftopl` a `pltotf`. Takto modifikovanou metriku ovšem nesmíte distribuovat pod stejným názvem, jaký má původní metrika  $\mathcal{C}$ fontu.

## 2.4 Anabáze UNIXových hard-linků

Jak jsme uvedli v předchozí sekci, hlavní METAFONTové soubory  $\mathcal{C}$ fontů mají jednu zvláštnost: ačkoli zde existuje 121 různě nazvaných souborů, všechny obsahují stejný dvouřádkový text.

Rozhodl jsem se těch 121 stejných různě nazvaných souborů implementovat pro úsporu inodů v UNIXu jako hard linky. Takto jsem fonty zabalil do balíčku `cs-fonts.tar.gz` a dal k dispozici internetové veřejnosti.

Pokud je systém, na který se balíček `csfonts.tar.gz` instaluje, rovněž UNIXového typu, pak program `tar` vytvoří v cílovém adresáři 121 hard linků a je vše v pořádku. Systém ušetřil 120 inodů, které nemusel alokovat.

Jestliže ale cílový systém nepracuje s hard linky, mohou nastat potíže. Osobně se ale domnívám, že pokud tento systém provozuje nějakou implementaci programu `tar`, měla by se tato implementace s problémem hard linků vyrovnat například tak, že místo hard linků vytvoří při extrahování archivu stejné soubory.

Proto považuji stížnosti uživatelů MS Windows na podivnost archivu `cs-fonts.tar.gz` za neopodstatněné. Tito uživatelé si stěžují, že se jim rozbalí z těch 121 souborů jen jeden, přičemž používají jakýsi `Wintar`. Odpovídám: nechť si tito uživatelé stěžují u svého dodavatele implementace programu `tar`. Odmítám kvůli Windowsovým uživatelům balit  $\mathcal{C}\mathcal{S}$ fonty jinak a opustit tak možnost šetření inodů na UNIXových systémech. Navíc se mi doneslo, že Windowsovi uživatelé mají možnost použít jiné implementace programu `tar`, které popsanou chybu neobsahují, a skutečně na Windowsovém filesystému založí 121 různých souborů se stejným obsahem.

## 2.5 Virtuální fonty přidané do balíčku $\mathcal{C}\mathcal{S}$ fontů

V balíčku `csfonts.tar.gz` jsou přítomny dvě skupiny virtuálních fontů. Jedny mapují  $\mathcal{C}\mathcal{S}$ fonty na CM fonty a druhé mapují CM fonty na  $\mathcal{C}\mathcal{S}$ fonty. V tomto odstavci vysvětlím důvody existence obou skupin a způsob jejich použití.

Až donedávna bylo potřeba při prohlížení `dvi` souborů programem `xdvi` (a jemu podobnými programy) počkat, až se pomocí `METAFONTu` a programu `gftopk` vygenerují potřebné bitmapy fontů, a teprve pak jsme je viděli v prohlížeči. Teprve od roku 2002 má program `xdvi` schopnost přímo zobrazovat `Type1` verze fontů, ale popisované virtuální fonty vznikly v době, kdy tomu tak nebylo a kdy  $\mathcal{C}\mathcal{S}$ fonty ještě neměly svou `Type1` variantu. Pokud jsme tehdy nemuseli čekat na generování nových fontů `METAFONTem`, pak jen proto, že už byly fonty vygenerovány dříve a zůstaly uloženy na disku v podobě `pk` souborů.

Za této situace bylo výhodné udržovat na disku co nejméně `pk` souborů. Protože uživatel  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u obvykle používá  $\mathcal{C}\mathcal{S}$ fonty, dá se předpokládat, že jejich `pk` soubory má na disku v hojném množství. Když pak jednou za čas potřebuje takový uživatel prohlédnout `dvi` soubor odkazující na CM fonty, pak je přeci zbytečné pro ně generovat nové bitmapy, když  $\mathcal{C}\mathcal{S}$ fonty jsou nadmnožinou CM fontů a bitmapy  $\mathcal{C}\mathcal{S}$ fontů pravděpodobně na disku už vygenerované jsou. Stačí pro zobrazení znaků CM fontů čerpat z bitmap  $\mathcal{C}\mathcal{S}$ fontů.

Právě k tomu účelu slouží první skupina virtuálních fontů označovaná jako `cm2cs`. Po rozbalení `csfonts.tar.gz` je tato skupina virtuálních fontů uložena do adresáře `fonts/vf/public` a je tedy aktivní. Každý `dvi`-ovladač, který potřebuje vykreslit znak z CM fontu, pak použije tyto virtuální fonty, které jej nasměrují na  $\mathcal{C}\mathcal{S}$ fonty. Takže `dvi`-ovladač bude tyto znaky nakonec čerpat z  $\mathcal{C}\mathcal{S}$ fontů. K žádnému zkreslení

informace přitom nedochází, protože všechny znaky CM fontů jsou v odpovídajících  $\mathcal{C}\mathcal{S}$ fontech na stejných pozicích a vypadají úplně stejně.

Pokud  $\mathcal{C}\mathcal{S}$ fonty vůbec nepoužíváte (nebo jen občas), bude pro vás asi výhodné virtuální fonty `cm2cs` z adresáře `fonts/vf/public` odstranit. Stejně tak se nehodí mít tyto virtuální fonty aktivní v mezinárodních  $\mathrm{T}\mathrm{E}\mathrm{X}$ ových distribucích, kde většina uživatelů  $\mathcal{C}\mathcal{S}$ fonty nikdy nepoužije. Tito uživatelé by asi byli velmi překvapeni, že při prohlížení `dvi` souboru se standardními CM fonty jim distribuce generuje na disk bitmapy jakýchsi  $\mathcal{C}\mathcal{S}$ fontů.

Druhá skupina virtuálních fontů s označením `cs2cm` není po rozbalení balíčku s  $\mathcal{C}\mathcal{S}$ fonty aktivní, protože není v adresáři `fonts/vf`, ale v adresáři `fonts/vf-cnv`, který `dvi`-ovladače s obvyklou konfigurací neprocházejí.

Tuto skupinu virtuálních fontů použijeme v případě, že jsme vytvořili v  $\mathcal{C}\mathcal{S}\mathrm{T}\mathrm{E}\mathrm{X}$ u dokument odkazující na  $\mathcal{C}\mathcal{S}$ fonty, a nyní jej chceme ve formátovaném tvaru (jako soubor `dvi`) poslat například do zahraničí. Přitom předpokládáme, že příjemce nebude mít ve své  $\mathrm{T}\mathrm{E}\mathrm{X}$ ové distribuci  $\mathcal{C}\mathcal{S}$ fonty.

Virtuální fonty `cs2cm` mapují všechny znaky  $\mathcal{C}\mathcal{S}$ fontů na odpovídající znaky z CM fontů. Pokud znak v CM fontech neexistuje, virtuální font jej nahradí kompozitem (například písmeno+háček). Pokud tedy aplikujeme například program `dvicopy` s těmito virtuálními fonty, dostáváme nový `dvi` soubor, který už odkazuje jen na CM fonty, a akcentovaná písmena jsou v tomto `dvi` souboru nahrazena kompozity. Takový `dvi` soubor můžeme poslat do zahraničí a máme jistotu, že bude zpracovatelný na libovolné  $\mathrm{T}\mathrm{E}\mathrm{X}$ ové distribuci, protože každá distribuce obsahuje CM fonty. Kvalita sazby se ovšem zhorší, protože akcenty (coby samostatné znaky) jsou v CM fontech takové poněkud nedomrlé, zatímco v  $\mathcal{C}\mathcal{S}$ fontech jsou kresleny společně s písmenem s velkou péčí. Informace v dokumentu ovšem není uvedenou modifikací `dvi` souboru vůbec změněna. Výjimkou je případ výskytu jednoho z těchto čtyř znaků: `promile`, `ogonek` (ocásek pod polské `a`), levá a pravá francouzská uvozovka. Tyto znaky nelze nijak pomocí CM fontů nahradit. Pokud dokument některý z těchto znaků obsahuje, pak program `dvicopy` ohlásí „---missing character packet“ a ponechá místo těchto znaků prázdná místa.

Ve `web2c` distribuci  $\mathrm{T}\mathrm{E}\mathrm{X}$ u v `UNIX`u můžete pro uvedenou konverzi `dvi` souboru z  $\mathcal{C}\mathcal{S}$ fontů na CM fonty použít příkaz:

```
VFFONTS=\$TEXMF/fonts/vf-cnv// dvicopy vstup.dvi vystup.dvi
```

Poznamenejme, že v žádném případě nesmí `dvi` ovladač najít současně virtuální fonty `cm2cs` i `cs2cm`. V takovém případě dojde k havárii, neboť odkazy ve virtuálních fontech se dostanou do nekonečného cyklu.

## 2.6 Historie a budoucnost $\mathcal{C}\mathcal{S}$ fontů

Tvary akcentů  $\mathcal{C}\mathcal{S}$ fontů byly vytvořeny a implementovány v jazyce `METAFONT`u Petrem Novákem ve spolupráci s českými typografy někdy na začátku 90. let. Autor přenechal  $\mathcal{C}\mathcal{S}$ fonty  $\mathcal{C}\mathcal{S}\mathrm{TUG}$ u, který s nimi může libovolně nakládat.

METAFONTový kód byl pak v letech 1992–1993 dále upraven Karlem Horákem. Karel se inspiroval z METAFONTových zdrojů pro fonty vytvořené v Polsku. Zpracoval tam možnost nastavení kódování fontu a vytvořil makra umožňující mít všechny hlavní mf soubory se stejným dvouřádkovým obsahem.

Na schůzce tvůrců  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u na FEL v roce 1993 bylo rozhodnuto, že  $\mathcal{C}\mathcal{S}$ fonty budou mít kódování podle ISO 8859-2. Později, při implementaci  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u do UNIXových distribucí nepodporujících změny xord/xchr vektorů, se ukázalo, že to bylo velmi prozíravé rozhodnutí.

V roce 1993 jsem převzal údržbu  $\mathcal{C}\mathcal{S}$ fontů do svých rukou. Udělal jsem jen velmi drobné změny. Poslední 28. 9. 1996:

Písmeno Č a další akcentované kapitálky měly před tímto datem větší výšku než kresba o 1,2pt. Opravil jsem. Také jsem tehdy odstranil nevhodné záporné kerny: Tě, Tř, Tö, Tü, Tä, Tà (analogicky pro Ě, Y, Ý). Vě, Vř, Vö, Vü (analogicky pro F, W) a redukoval jsem přílišné záporné kerny: Té, Tó, Tů, Tí, Tá, Tú (analogicky pro Ě, Y, Ý).

Pak jsem vývoj  $\mathcal{C}\mathcal{S}$ fontů zmrazil podobným způsobem, jako Knuth přestal měnit CM fonty. Prioritním požadavkem je, aby dokument opírající se o  $\mathcal{C}\mathcal{S}$ fonty byl od roku 1996 formátován naprosto stejně dnes i kdykoli v budoucnu. Aby byl tento požadavek splněn, není tedy možné zasáhnout do rozměrů znaků, kernů a ligaturních tabulek.

V roce 1996 jsem do  $\mathcal{C}\mathcal{S}$ fontů přidal virtuální fonty podporující náhradu Computer Modern fonty a naopak.

V roce 1998 se podařilo dohodnout s autorem  $\text{te}\mathcal{T}\mathcal{E}\mathcal{X}$ u Thomassem Esserem, aby zařadil do své distribuce  $\mathcal{C}\mathcal{S}$ fonty a celý  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ . Od této chvíle jsou distribuce odvozené z  $\text{te}\mathcal{T}\mathcal{E}\mathcal{X}$ u implicitně vybaveny  $\mathcal{C}\mathcal{S}$ fonty.

V roce 1998 jsem také pro potřeby výstupu do formátu PDF vytvořil variantu  $\mathcal{C}\mathcal{S}$ fontů, tentokrát ve formátu PostScript Type1. Vyšel jsem z BaKoMa Type1 implementace CM fontů a vytvořil jsem si program `t1accent`, který k písmenkům přidával akcenty podle vzorových PostScriptových tahů generovaných z původních  $\mathcal{C}\mathcal{S}$ fontů METAPOSTem. Na mnoha místech jsem byl nucen přistoupit k mikrotypografickým kompromisům – v drobnostech se kresby některých znaků  $\mathcal{C}\mathcal{S}$ fontů z Type1 liší od svých originálních METAFONTových protějšků. Proto jsem distribuci Type1  $\mathcal{C}\mathcal{S}$ fontů označil jako „alpha“ a do komentáře jsem dal důrazné varování, že tyto fonty je možné používat na vlastní riziko. Z toho důvodu jsem také ponechal implicitní konfiguraci programu `dvips` tak, aby program používal léty osvědčený výstup z METAFONTu, tedy bitmapy formátu `pk`. Type1  $\mathcal{C}\mathcal{S}$ fonty byly původně konfigurovány jen pro `pdf\mathcal{T}\mathcal{E}\mathcal{X}`.

Rozhodnutí ponechat `dvips` pracovat implicitně s bitmapami naráželo na problémy. Neustále dokola se uživatelé ptali, jak je možné, že výstup z `pdf\mathcal{T}\mathcal{E}\mathcal{X}`u je dobrý, ale při cestě `dvips` – `pstopdf` dostávají roztřesená písmenka. Protože jsem byl uondán velmi častým odpovídáním na tuto otázku, ani jsem se nakonec nezlobil, když v roce 2001 autor  $\text{te}\mathcal{T}\mathcal{E}\mathcal{X}$ u rozhodl, že bude  $\mathcal{C}\mathcal{S}$ fonty pro `dvips` implicitně konfigurovat ve verzi Type1. Asi ty mikrotypografické kompromisy ani tak moc nevadí, zatímco roztřesená písmenka v PDF způsobovala oheň na střeše.

V současné době existují volně dostupné nástroje, jako například `texttrace` opírající se o `autotrace`. Tyto nástroje umožní převést METAFONTový font do Type1

„obtahováním bitmap“ skoro automaticky. Vyzkoušel jsem to na  $\mathcal{C}$ sfontech a s výsledkem jsem nebyl vůbec spokojen: výsledné `pfb` soubory byly asi pětikrát větší než ty moje „ručně“ vyrobené. Proto jsem zatím alpha verzi Type1 formátu  $\mathcal{C}$ sfontů z roku 1998 neopustil.

Do budoucna bych velmi rád do  $\mathcal{C}$ sfontů přidal znak euro a paragraf. Taková změna by byla zpětně kompatibilní, takže bych se jí nebránil. METAFONTové zdroje pro paragraf ověřené na všech  $\mathcal{C}$ sfontech už několik let mám, ale nezveřejnil jsem je. METAFONTové zdroje znaku euro by se snadno daly převzít z jiného METAFONTového fontu. Největší potíž je ovšem v tom, že s uvedením nové verze  $\mathcal{C}$ sfontů dnes nestačí zveřejnit jen METAFONTové zdroje a metriky, ale je třeba mít okamžitě s tím konzistentní Type1 varianty fontů. Do manuální práce na nové verzi Type1 varianty  $\mathcal{C}$ sfontů se mi ale moc nechce. Je to nevďěčná a rozsáhlá práce: `pfb` souborů je v balíčku 57 a každý je třeba disassemblovat, v editoru přidat nové znaky a znovu převést na `pfb`. Přitom s automatickými nástroji, jak jsem uvedl před chvílí, nejsem spokojen.

### 3. `cspsfonts.tar.gz` – 35 základních PostScriptových fontů

Každý PostScriptový RIP musí být vybaven aspoň 35 základními fonty ze sady „base 35“. Sada mimo jiné obsahuje sedm rodin textových fontů, každá rodina obsahuje čtyři fonty (základní, kurzíva, tučný a tučná kurzíva, **Helvetica navíc čtyři zúžené varianty**). Strojopis se obvykle kombinuje z rodiny Courier. Pro tyto fonty je vytvořena podpora ve formě metrik v kódování  $\mathcal{C}$ sfontů a virtuálních fontů, které tyto metriky mapují na skutečné fonty, kódované samozřejmě úplně jinak. Tato podpora je v balíčku `cspsfonts.tar.gz`, který je povinnou součástí  $\mathcal{C}$ TeXu.

Kódování těchto metrik je inspirováno kódováním  $\mathcal{C}$ sfontů, ale chybí matematické znaky ze začátku tabulky a znaky, které v běžných PostScriptových fontech nenajdeme (např. pozice 20: škrťátko polského l). Na druhé straně jsou přidány znaky s kódy 80–86 (hexadecimálně) a některé další (např. kompletní polské Ł a ł). Viz následující tabulku fontu `ptmr8z`, implementující písmo Times-Roman pro  $\mathcal{C}$ TeX.

Fonty rodiny Courier mají kódování inspirováno  $\mathcal{C}$ fontem `cstt10`, takže se na pozicích 3C, 3E, 5C, 5F a 7B až 7D znaky poněkud liší. Toto kódování je označeno 8u na rozdíl od kódování ostatních PostScriptových fontů z balíčku `cspsfonts.tar.gz`, které je nazváno 8z. Tuto nejednotnost kódování známe už u  $\mathcal{C}$ sfontů a prapůvodně existuje u Computer Modern fontů. Znak vlevo na zmíněných pozicích odpovídá kódování 8z a znak vpravo kódování 8u. Na ostatních pozicích je kódování 8z a 8u shodné.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x													fi	fl		
1x	ı		`	´	˘	˙	-	°	,	ß	æ	œ	ø	Æ	Œ	Ø
2x		!	”	#	\$	%	&	’	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	ı<	=	ı>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	“\	]	^	’_
6x	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	- {	—	~ }	~	..
8x	...	†	‡	•	£	¶	€		™	©	®			‰	<	>
9x									À		,	‘	-	˘	«	»
Ax		À		Ł	Ꞥ	Ł		§		Š		Ť			Ž	
Bx		à		ł		ŀ			à	š		ť			ž	
Cx	Ŕ	Á	Â		Ä	Í		Ç	Č	É		Ë	Ě	Í	Î	Ď
Dx			Ň	Ó	Ô		Ö		Ř	Ů	Ú		Ü	Ý		
Ex	í	á	â		ä	í		ç	č	é		ë	ě	í	î	ď
Fx			ň	ó	ô		ö		ř	ů	ú		ü	ý	„	“

**Upozornění:** Ve výše uvedené tabulce jsou prázdná místa, na kterých mohou být umístěny nedokumentované znaky. Když si vytisknete například skutečnou tabulku fontu `ptmr8z`, zjistíte, že tomu tak skutečně je. Pokud chcete mít dokument zpracovatelný i v budoucích verzích  $\text{\LaTeX}$ , nemůžete se o tyto nedokumentované znaky opírat. Tyto znaky byly obsazeny v rámci soukromé iniciativy Zdeňka Wagnera, který potřeboval použít font pro více jazyků. Bohužel umístil některé znaky na pozice, které nejsou pro budoucí rozšíření fontů vhodné. Každý si může do volných pozic doplnit co potřebuje, ovšem s tím rizikem, že jeho dokumenty využívající tyto pozice nemusejí být kompatibilní s  $\text{\LaTeX}$ .

Sada metrik balíčku `cspsfonts.tar.gz` zahrnuje následující fonty:

Metrika	odkazuje na	Odpovídající PostScriptový font
pagk8z	<code>rpagk</code>	AvantGarde-Book
pagko8z	<code>rpagko</code>	AvantGarde-BookOblique
pagd8z	<code>rpagd</code>	AvantGarde-Demi
pagdo8z	<code>rpagdo</code>	AvantGarde-DemiOblique
pagkc8z	* <code>rpagk</code>	AvantGarde-Book Caps & Small Caps
pagdc8z	* <code>rpagd</code>	AvantGarde-Demi Caps & Small Caps
pbkl8z	<code>rpbkl</code>	Bookman-Light
pbkli8z	<code>rpbkli</code>	Bookman-LightItalic
pbkd8z	<code>rpbkd</code>	Bookman-Demi
pbkdi8z	<code>rpbkdi</code>	Bookman-DemiItalic
pbklc8z	* <code>rpbkl</code>	Bookman-Light Caps & Small Caps
pbkdc8z	* <code>rpbkd</code>	Bookman-Demi Caps & Small Caps

pcrr8u	rperr	Courier
pcrro8u	rperro	Courier-Oblique
pcrb8u	rperb	Courier-Bold
pcrbo8u	rperbo	Courier-BoldOblique
pcrrc8u	* rperr	Courier Caps & Small Caps
pcrbc8u	* rperb	Courier-Bold Caps & Small Caps
phvr8z	rphvr	Helvetica
phvro8z	rphvro	Helvetica-Oblique
phvb8z	rphvb	Helvetica-Bold
phvbo8z	rphvbo	Helvetica-BoldOblique
phvrc8z	* rphvr	Helvetica Caps & Small Caps
phvbc8z	* rphvb	Helvetica-Bold Caps & Small Caps
phvrn8z	* rphvrrn	Helvetica-Narrow
phvron8z	* rphvron	Helvetica-Narrow-Oblique
phvbn8z	* rphvbrn	Helvetica-Narrow-Bold
phvbon8z	* rphvbon	Helvetica-Narrow-BoldOblique
phvrnc8z	* rphvrrn	Helvetica-Narrow Caps & Small Caps
phvbnc8z	* rphvbrn	Helvetica-Narrow-Bold Narrow C & Small C
pncr8z	rpncr	NewCenturySchlbk-Roman
pncri8z	rpncri	NewCenturySchlbk-Italic
pncb8z	rpncb	NewCenturySchlbk-Bold
pncbi8z	rpncbi	NewCenturySchlbk-BoldItalic
pncrc8z	* rpncr	NewCenturySchlbk-Roman Caps & Small Caps
pncbc8z	* rpncb	NewCenturySchlbk-Bold Caps & Small Caps
pplr8z	rpplr	Palatino-Roman
pplri8z	rpplri	Palatino-Italic
pplb8z	rpplb	Palatino-Bold
pplbi8z	rpplbi	Palatino-BoldItalic
pplrc8z	* rpplr	Palatino-Roman Caps & Small Caps
pplbc8z	* rpplb	Palatino-Bold Caps & Small Caps
ptmr8z	rptmr	Times-Roman
ptmri8z	rptmri	Times-Italic
ptmb8z	rptmb	Times-Bold
ptmbi8z	rptmbi	Times-BoldItalic
ptmrc8z	* rptmr	Times-Roman Caps & Small Caps
ptmbc8z	* rptmb	Times-Bold Caps & Small Caps
pzcmi8z	rpzcmi	ZapfChancery-MediumItalic

V tabulce jsou hvězdičkou označeny ty metriky, které neodkazují přímo na speciální PostScriptový font, ale jsou implementovány pomocí virtuálního fontu (Caps & Small Caps). ~~Nebo se jedná o Narrow varianty fontu Helvetica, které jsou konfigurovány v psfonts.map pomocí PostScriptové transformace.~~

České a slovenské znaky jsou byly ve fontech 8z a 8u implementovány jako kompozity (tj. virtuální font je poskládal ze samostatného akcentu a základního písmene). Tento přístup přinášel problémy: ve výsledném PDF nebylo možno vyhledávat český text a nebylo možno přenést text clipboardem do jiné aplikace.



Proto jsem v roce 2012 zveřejnil novou verzi fontů, které odkazují přímo na PostScriptový font, který překóduvávají pomocí /Encoding vektoru. Bývalý argument, že nemáme jistotu, zda ve fontu naše akcentované znaky jsou, už dávno neplatí. Použité segmenty jsou přítomny v každém PostScriptovém fontu. Proto virtuální fonty po sestavení kompozitů odkazují na metriky fontů (s písmenem `r` na začátku), které jsou již kódovány podle StandardEncoding. Fonty v PostScriptovém RIPu tedy nepotřebujeme mít počestšeny ani poslovenšteny a také nepotřebujeme měnit Encoding vektor těchto fontů na úrovni PostScriptu.

Program `dvips` je obvykle konfigurován tak, že do výstupního PostScriptového kódu nezavádí žádný z fontů „base 35“ a ponechá tam jenom odkaz. Starost o font přenechá PostScriptovému RIPu. PostScriptové fonty z „base35“ tedy principiálně není nutné pro provoz v `TeXu` instalovat na disk a nejsou tedy ani obsaženy v balíčku `cspsfonts.tar.gz`. Programu `dvips` je doporučeno nabídnout mapovací soubor `cs-a35-nodownload.map`.

Program `pdftex` je naopak konfigurován tak, že do výstupního PDF vkládá všechny použité fonty. Pokud je použit Adobe font z „base35“, je za něj připravena a zavedena náhrada z dílny URW, která je přítomna v běžných `TeXových` distribucích. Programu `pdftex` je tedy potřeba nabídnout jiný mapovací soubor `cs-a35-urwdownload.map`.

Pokud si chcete prohlížet ~~jste si prohlíželi~~ `dvi` soubor odkazující na fonty z „base 35“ například starší verzi programu `xdvi`, pak tento program potřebuje ~~oval~~ (alespoň na přechodnou dobu) bitové mapy těchto fontů. Pokud bitové mapy v instalaci ještě ~~neexistují~~ ~~nebyly~~, program zavolá ~~al~~ skript `mktexpk`, který pro vytvoření rastru volá ~~al~~ dále Ghostscript. To je implementace PostScriptového RIPu, která musí obsahovat fonty z „base 35“.\* Odtud se fonty konvertují do bitových map `pk` a odtud je nakonec použije program `xdvi`.

Poznamenejme ještě, že novější verze programu `xdvi` obsahují tzv. modul `t1lib`, díky němuž `xdvi` umí přímo číst PostScriptové fonty a vykreslovat je na obrazovku. Není tedy potřeba vůbec vyrábět `pk` bitmapy fontů, ani na přechodnou dobu. Náhrázky fontů od firmy URW z „base 35“ jsou proto v novějších distribucích `TeXu` přímo instalovány a nespolehá se v tomto případě na Ghostscript. Nemusíte pak ani spoléhat na přítomnost fontů v PostScriptovém RIPu v tiskárně. Novější verze `web2c TeXu` nabízejí pro takový případ možnost použít u programu `dvips` přepínač `-Pdownload35`.

### 3.1 Historie a budoucnost balíčku `cspsfonts.tar.gz`

Balíček `cspsfonts.tar.gz` má méně bohatou historii než `CSfonty`. Tento balíček začal vznikat v září roku 1994. Tehdy jsem zjistil, že popisy kompozitů v `AFM` metrice pomocí řádků `CC` jsou správně převáděny programem `afm2tfm` na odpovídající kompozity ve vytvářeném virtuálním fontu. Problém byl jen v tom, že originální `AFM` metriky od Adobe neobsahovaly popisy všech kompozitů potřebných pro český

---

\* Z licenčních důvodů Ghostscript neobsahuje originální fonty od Adobe, ale jen jejich hodně dobré náhrázky od firmy URW. Rozdíl většinou pouhým okem nerozeznáte.



a slovenský jazyk. Z toho důvodu jsem si vytvořil program `a2ac` [10], který na základě přehledné tabulky kompozity do AFM metrik doplnil a současně doplnil kerningové informace pro nově vytvářené znaky. Za použití tohoto programu pak vznikla sada metrik a virtuálních fontů s písmenem `c` na začátku (např. `cptmr`).  
**České a slovenské znaky měly plovoucí akcenty.**

V roce 1996 pak uveřejnil pan Wagner nové metriky generované stejným způsobem, ovšem opravil několik estetických nedostatků a navíc metriky nazval podle doporučení Karla Berryho (`8z` a `8t` na konci). Od té doby jsou v balíčku `cspfonts.tar.gz` obsaženy metriky pana Wagnera.

~~Konečně~~ **V** roce 1999 jsem musel po konzultaci s Karlem Berryem metriky pro rodinu Courier přejmenovat z původního `*8t` na nynější `*8u`, protože názvy s `8t` na konci nám kolidovaly s názvy stejných fontů v kódování podle Corku. To je zatím poslední změna v tomto balíčku.

Chystám **al jsem** doplnění balíčku `cspfonts.tar.gz` o nové metriky kódované jako `8z`, které mapují další běžně používané PostScriptové fonty. Jako první na řadě se nabízí přidání metrik `8z` pro volně šířenou rodinu fontů Charter písmolijný BitStream.

V roce 2012 jsem přidal do merik fontů z tohoto balíčku znak Euro a další znaky (viz vybarvené znaky v tabulce). Bohužel po letech už nebyly k dispozici zdroje, ze kterých metriky generoval kdysi pan Wagner. Zejména nebyl k dohledání správný soubor `x12.enc`. V metrikách tedy bylo plno nedokumentovatelných slotů, o kterých jsem těžko mohl soudit, co byly zač. Takové sloty považuji za chybu. Proto jsem ustoupil od zpětné kompatibility a přegeneroval všechny metriky kompletně znovu. Je možné že sazba těmito fonty někdy výjde mírně jinak než kdysi, ale aspoň máme metriky, které přesně korespondují souboru `x12.enc`. Protože se mi nepodařilo dohledat AFM metriky fontů přímo od adobe takové, které obsahují znak `rcaron`, fonty jsem vygeneroval z metrik URW. Generující skript je obsahem balíčku.

Z balíčku `cspfonts.tar.gz` jsem přemístil soubory `ctimes.tex` atd. do balíčku `csplain.tar.gz`, kam jsem přidal další podobné soubory.

## 4. Formát `Csplain`

Formát `Csplain` má svou stránku na <http://petr.olsak.net/csplain.html>. Je tam poněkud stručněji řečeno víceméně totéž, jako zde.

Popíšu jenom **zejména** odlišnosti formátu `Csplain` (povel `csplain`) od Knuthova formátu `plain` (povel `tex`), popsaného v `TEXbooku` [4].

### 4.1 Překódování v input procesoru `TEXu`

Vstupní text pro formát `Csplain` se předpokládá 8bitový. Kódování češtiny nebo slovenštiny vstupního textu je takové, jaké běžně používáte v systému, kde je váš `TEX` instalován. O tomto kódování budeme nadále mluvit jako o *vstupním kódování*.

Ze vstupního kódování je třeba text překódovat do *vnitřního kódování*, se kterým interně pracuje `TEX`. Toto vnitřní kódování je v `Csplainu` implicitně nastaveno

na ISO-8859-2 nezávisle na operačním systému. V tomto kódování musí být připraveny fonty použité v dokumentu.  $\mathcal{C}$ sfonty a fonty zaváděné pomocí `\input ctimes`, `\input cbookman` atd. tuto vlastnost mají.

Překódování textu ~~do vnitřního kódování můžete udělat ručně před spuštěním  $\text{\TeX}$ u (to moc nedoporučuji)~~ nebo je tato konverze implementována ~~o~~ do input procesoru samotného  $\text{\TeX}$ u. Jak je to uděláno závisí na použité distribuci  $\text{\TeX}$ u. Nejběžnější metody nastavení input procesoru budou zmíněny níže.

Problém vztahu vstupního a vnitřního kódování ilustruji na příkladě. Nechť je ve vstupním souboru napsáno `\char174=Ž`. Po zpracování  $\mathcal{C}$ splainem musím na výstupu dostat:  $\hat{Z}=\hat{Z}$ . Tímto příkladem jsem naznačil dvě věci. Za prvé: některá makra mohou být závislá na zvoleném vnitřním kódování  $\text{\TeX}$ u (zde makro `\char174`, které má vytisknout písmeno  $\hat{Z}$ ). Pokud použiju v systému povel `csplain` bez doplňujících maker, pak předpokládám, že kód 174 znamená písmeno  $\hat{Z}$ .

Za druhé: pokud vpravo od rovnítko vidím v editoru, kterým zpracovávám vstupní dokument, písmeno  $\hat{Z}$ , pak se tento znak musí dostat do vnitřních částí  $\text{\TeX}$ u pod kódem 174, ačkoli v tom editoru je třeba kódován úplně jinak. Konverze by měla být implementována (závisle na použitém vstupním kódování) v input procesoru  $\text{\TeX}$ u. Důležité je, že to ve svém editoru *vidím jako  $\hat{Z}$*  a očekávám na základě této vizuální informace jednotné chování  $\mathcal{C}$ splainu na všech systémech, kde existují editory, kterými se mohu do zdrojového textu podívat a vidět  $\hat{Z}$  (ačkoli různé operační systémy mohou toto  $\hat{Z}$  kódovat různě).

Pokud se zdrojový soubor  $\mathcal{C}$ splainu přenáší mezi systémy s různým kódováním, pak předpokládáme, že příjemce se na soubor podívá ve svém editoru a když uvidí rozsypaný čaj, nejprve si obdrženy soubor opraví.

Jak bylo řečeno, konverze mezi vstupním kódováním a vnitřním kódováním ~~probíhá na úrovni input procesoru  $\text{\TeX}$ u. To Konverzi do jednotného vnitřního kódování~~ je možné realizovat v různých distribucích  $\text{\TeX}$ u různě. Pročtete si dokumentaci k použité distribuci. V  $\text{em}\text{\TeX}$ u se pro tyto účely používá tzv. TCP tabulka zaváděná při inicializaci formátu. Ve velmi starých web2c distribucích  $\text{\TeX}$ u (léta 1992–1997) můžete najít tzv. Škarvadovu záplatu, která nastavovala kódování podle proměnné prostředí systému. Později se pro překódování používal `enc\text{\TeX}` (rok 1997), který nastavoval `xord/xchr` vektor input procesoru  $\text{\TeX}$ u pomocí přidáných primitivů (viz [9] a [8]). ~~Dnes~~ **Ještě později** (po roce 1998) se v distribucích web2c, `te\text{\TeX}`, `\text{\TeX}live` a odvozených používají tzv. TCX tabulky, které se vyvolají z příkazového řádku pomocí přepínače `-translate-file` nebo `-default-translate-file`. ~~Dnes~~ **doporučuji se k `enc\text{\TeX}`u vrátit, protože jeho současná verze nabízí (na rozdíl od TCX tabulek) novou možnost: konverzi vícebytového kódování na vnitřní jednobytové, takže zvládne vstup kódovaný v UTF-8. Toto kódování je od roku 2012 doporučeno jako implicitní vstupní kódování  $\mathcal{C}$ splainu.**

Se správným nastavením překódovací tabulky úzce souvisí schopnost  $\text{\TeX}$ u zapisovat přímo znaky použité abecedy ~~8bitový text~~ do logů a do `\write` souborů ~~zpětně ve vstupním kódování. (tam musí být text zpětně konvertován do vstupního kódování).~~  $\text{\TeX}$  implicitně pro výstup do těchto souborů používá pro znaky s kódem větším než 127 hexadecimální přepis uvozený dvěma znaky  $\wedge$ . To je pro  $\mathcal{C}$ splain nepřipustné, a proto je nutné v  $\text{\TeX}$ u rozšířit tzv. tabulku znaků povolených pro tisk

do logů a pracovních souborů. Ta je ve web2c distribuci rozšířena automaticky zavedením TCX tabulky: znaky, které jsou v této tabulce zmíněny, se stávají povolenými pro tisk. To vysvětluje nutnost používání tabulky `il2-cs.tcx` v UNIXových distribucích, přestože tabulka pouze deklaruje překódování „jedna ku jedné“. **Detekuje-li  $\mathcal{C}$ Splain při generování formátu aktivovaný  $\text{enc}\text{T}_{\text{E}}\text{X}$ , nastaví se tisknutelnost znaků s kódem větším než 127 automaticky.**

Věnujte prosím při implementaci  $\mathcal{C}$ Splainu do jiných  $\text{T}_{\text{E}}\text{X}$ ových distribucí zvýšenou pozornost právě problémům překódování v input procesoru a problému množiny povolených znaků pro tisk do logů a pracovních souborů. Pokud se chcete přesvědčit o správnosti implementace  $\mathcal{C}$ Splainu, vyzkoušejte test `cstrip` (viz [11]).

## 4.2 Inicializace formátu

Pokud se vám podařilo  $\mathcal{C}$ Splainem zpracovat tento dokument, pak máte formát  $\mathcal{C}$ Splain již inicializovaný a můžete směle tuto sekci přeskočit. Pokud používáte nejnovější verzi  $\text{T}_{\text{E}}\text{X}$ live, pak stačí napsat na příkazový řádek `csplain` dokument a **použije se předgenerovaný formát, nebo se pokud tak činíte poprvé, formát se automaticky vytvoří.** Také můžete inicializovat formát  $\mathcal{C}$ Splain pomocí nástroje `tex-config` například tak, že odstraníte komentářové znaky u slova `csplain` v souboru `texmf/web2c/fmtutil.cnf` a spustíte `texconfig init`.

Následující text popisuje možnost vytvoření formátu „ručně“.  $\mathcal{C}$ Splain v  $\text{T}_{\text{E}}\text{X}$ live se vstupním kódováním UTF-8 vygenerujete za použití  $\text{enc}\text{T}_{\text{E}}\text{X}$ u (přepínač `-enc`) z příkazové řádky takto:

```
pdfetex -ini -enc "\let\enc=u \input csplain.ini"
```

Takto vygenerovaný formát implicitně vystupuje do DVI. Dále `pdf $\mathcal{C}$ Splain` vygenerujete z příkazové řádky:

```
pdfetex -jobname=pdfcsplain -ini -enc "\let\enc=u \input csplain.ini"
```

Pozorování: těsně před `\dump` je ve výpisu a v log souboru napsáno toto hlášení: `jobname=pdfcsplain, PDF output initialised`. Takže nyní je implicitně nastaven výstup do PDF.

Soubory `csplain.fmt` a `pdfcsplain.fmt` je třeba umístit někam, kde je  $\text{T}_{\text{E}}\text{X}$ ová distribuce najde. Místo použití výše uvedených příkazů se můžete pokusit editovat konfigurační soubor v  $\text{T}_{\text{E}}\text{X}$ ové distribuci a spustit generování formátu prostředky, které k tomu účelu nabízí distribuce.

Spuštění  $\mathcal{C}$ Splainu (resp. `pdf $\mathcal{C}$ Splainu`) provedete jedním z příkazů

```
pdfetex -fmt=csplain.fmt dokument
csplain dokument
pdfetex -fmt=pdfcsplain.fmt dokument
pdfcsplain dokument
```

Druhý a čtvrtý řádek obsahuje typické zkratky implementované různým způsobem v závislosti na operačním systému a  $\text{T}_{\text{E}}\text{X}$ ové distribuci.

Další možností je vygenerovat `csplain` s jednobytovým vstupním kódováním pomocí `tcx` tabulek. To se dělá přepínačem `-translate-file`, za kterým následuje

název použité `tcx` tabulky. Zbýlý text v této sekci většinou dokresluje situaci před deseti a více lety.

Příkazem `initex` označuji v tomto textu spuštění `TEX`u v inicializačním módu, kdy `TEX` dokáže načítat tabulky vzorů dělení pro různé jazyky a ukládat nabyté vědomosti příkazem `\dump` do binárních souborů, tzv. formátů. Dříve na to existoval zvláštní program `iniTEX`, dnes se často používá nějaký přepínač: `tex -ini`, `tex /i` atd.

Formát `Čsplain` (soubor `csplain.fmt`) vygenerujete jednoduše:

```
initex csplain.ini
```

Používat jej pak můžete pomocí příkazu

```
tex &csplain dokument
```

V UNIXu musíte použít `tex \&csplain dokument`, aby se znak `&` neinterpretoval na úrovni shellu. Nezapomeňte ale nastavit správnou překódovací `tcx` tabulku způsobem obvyklým v použité `TEX`ové distribuci. Níže uvádím několik příkladů.

- **te<sub>T</sub>E<sub>X</sub>, T<sub>E</sub>XLive, vstupní kódování ISO-8859-2:**

inicializace: ~~tex -ini csplain.ini~~

obsah skriptu ~~csplain:~~

```
tex -fmt=csplain -default-translate-file=il2-cs $@
```

inicializace: `tex -ini -translate-file=il2-cs csplain.ini`

instalace příkazu `csplain`: `ln -s tex csplain`

spuštění `Čsplainu`: `csplain dokument`

- **te<sub>T</sub>E<sub>X</sub>, T<sub>E</sub>XLive, vstupní kódování podle Kamenických (hypotetický příklad):**

inicializace: `tex -ini csplain.ini`

obsah skriptu `csplain`:

```
tex -fmt=csplain -default-translate-file=kam-cs $@
```

spuštění `Čsplainu`: `csplain dokument`

- **web2<sub>T</sub>E<sub>X</sub> s enc<sub>T</sub>E<sub>X</sub>em, vstupní kódování ISO-8859-2:**

inicializace: `tex -ini csplain.ini`

instalace příkazu `csplain`: `ln -s tex csplain`

spuštění `Čsplainu`: `csplain dokument`

- **web2<sub>T</sub>E<sub>X</sub> s enc<sub>T</sub>E<sub>X</sub>em, vstupní kódování podle Kamenických:**

inicializace: `tex -ini -enc '\let\enc=k \input csplain.ini'`

instalace příkazu `csplain`: `ln -s tex csplain`

spuštění `Čsplainu`: `csplain dokument`

- **em<sub>T</sub>E<sub>X</sub>, vstupní kódování podle Kamenických:**

inicializace: `htex386 /i /8 /mt26000 /Ckamenic.tcp csplain.ini`

obsah dávky `csplain.bat`:

```
htex386 /mt26000 &csplain %1 %2 %3 %4 %5 %6 %7
```

Zastavím se na chvíli u `emTEX`u. Parametr `/mt` zde zvětšuje prostor pro ukládání vzorů dělení slov tak, aby se do formátu vešlo celkem pět vzorů dělení: anglické vzory

dělení a dále české i slovenské vzory dělení obě v kódování ISO-8859-2 a Cork\*. Paměť pro vzory dělení v `TeXu` se skládá ze dvou částí. První je nastavována interní `TeXovou` proměnnou `trie.size` (ta koresponduje s přepínačem `/mt`) a druhá proměnnou `trie.op.size` (viz [6], str. 301). Bohužel, hodnota `trie.op.size` není v `emTeXu` měnitelná a je ve verzích `tex.exe` a `tex386.exe` nastavena na malou hodnotu: pět vzorů dělení slov se tam nevejde. Proto jsem v příkladu použil `htex386.exe`, ve kterém je `trie.op.size` nastavena na dostatečně velkou hodnotu. Pokud chcete v `emTeXu` použít `tex.exe` nebo `tex386.exe`, pak musíte generovat formát `CSplain` jen se třemi vzory dělení – vyloučit alternativní kódování Cork. Toho lze dosáhnout jednak tím, že použijete verzi `CSplainu` starší než `<Feb. 2000>`, nebo tak, že potlačíte opakované volání vzorů dělení pro Cork pomocí `\let\Cork=\relax` takto:

```
tex /i/8/mt17000/Ckamenic.tcp \let\Cork=\relax \input csplain.ini
```

Poznamenejme, že potlačit načítání vzorů dělení v alternativním kódování (Cork) lze pomocí `\let\Cork=\relax` ve všech distribucích `TeXu`. Nejedná se tedy jen o `emTeXovou` záležitost. Nicméně v ostatních mě dostupných distribucích nejsou problémy s výchozí hodnotou paměti `TeXu` pro vzory dělení a je možno načíst bez nutnosti cokoli nastavovat všech pět vzorů dělení.

### 4.3 Výchozí nastavení v `CSplainu`

Aby byla zachována co největší kompatibilita s mezinárodním formátem `plain` a aby se necítili Češi nebo Slováci vzájemně utlačováni, jsou při startu `CSplainu` inicializovány americké vzory dělení slov a zapnuto větší mezerování za tečkami (tzv. `\nonfrenchspacing`).

Pro přepnutí na české vzory dělení slov použijte povel `\chyph` a do slovenštiny přepnete pomocí `\shyph`. Oba povelů navíc zapnou stejnoměrné mezerování i za tečkou (tzv. `\frenchspacing`). Takové mezerování je v české a slovenské sazbě obvyklejší. Zpět na americké vzory dělení a větší mezerování za tečkou přejdete pomocí povelu `\ehyph`.

Příklad: Chceme-li v `CSplainu` zpracovat český dokument, stačí na začátku dokumentu uvést `\chyph`:

```
\chyph
Tady už bude vše fungovat česky.
\bye
```

Přepínače `\chyph`, resp. `\shyph`, resp. `\ehyph` je možné v dokumentu používat kdekoli (například i uvnitř odstavce) na oddělení částí dokumentu, které podléhají českým resp. slovenským resp. anglickým pravidlům dělení slov.

Příkazy pro akcenty `\v`, `\^`, `\'`, `\'`, `\v`, `\"` mají implicitně původní význam z `plainu`, tj. pracují s primitivem `\accent`, který sestavuje znak z akcentu a základu. To nemusí být žádoucí, protože pak nefunguje dělení slov. Příkazem `\csaccents` se uvedené příkazy předefinují tak, že expandují přímo na akcentované

---

\* Možnost zapnout kódování Cork jako alternativní vnitřní kódování `CSplainu` je implementována od verze `<Feb 2000>`. Podrobněji se o tom zmiňuji v sekci 4.5.

znaky jako celek. Začne fungovat dělení slov v případech, kdy jsou slova napsána `takov\’ym podivn\’ym zp\r usobem`. Nově je po `\csaccents` definován příkaz `\r` pro akcent kroužku. Konečně se správně vysázejí i sekvence `\v d` nebo `\’i`, které původně činily potíže.

**Upozornění:** nedoporučuji v dokumentech pro  $\text{\LaTeX}$  používat styl `czech.sty` resp. `slovak.sty`. Tím se totiž dokument zcela zbytečně stává závislým na externím balíku `maker`, který není příliš stabilní. Pokud k tomu nemáte vážné důvody, styl *nepoužívejte*. Styl je vyvíjen spíše pro uživatele  $\text{\LaTeX}$ u, zatímco uživatel  $\text{\LaTeX}$ u si vystačí s povely `\chyph` a `\shyph`. V budoucnu přestěhuji tyto styly z archivního souboru `csplain.tar.gz` (kam nepatří a jsou tam jen z historických důvodů) do archivu `cslatex.tar.gz`.

Rozměr tiskového zrcadla (výška a šířka textu) je implicitně nastaven tak, aby všechny čtyři okraje měly velikost 1 palec na papíru formátu A4. To je významný rozdíl oproti nastavení ve formátu `plain`, kde se sice také předpokládají okraje 1 in, ale na papíru US-letter. Níže uvádím tabulku nastavení rozměrů zrcadla v  $\text{\LaTeX}$ u a pro srovnání tytéž parametry v originálním `plainu`:

$\text{\LaTeX}$	<code>plain</code>
<code>\hsize = 159.2mm</code>	<code>\hsize = 6.5in (165.1mm)</code>
<code>\vsize = 239.2mm</code>	<code>\vsize = 8.9in (226.06mm)</code>
<code>\hoffset = 0mm</code>	<code>\hoffset = 0in</code>
<code>\voffset = 0mm</code>	<code>\voffset = 0in</code>

Kromě znaků anglické abecedy jsou v  $\text{\LaTeX}$ u za písmena považovány i všechny akcentované znaky české a slovenské abecedy, takže i tyto znaky mají nastaven `\catcode=11` a dále mají nastaveny odpovídající `\lccode` a `\uccode`.

Výchozí fonty zavedené do formátu  $\text{\LaTeX}$ u jsou  $\text{\LaTeX}$ fonty. Pro matematiku jsou zavedeny původní Computer Modern fonty. Na rozdíl od `plainu` formát  $\text{\LaTeX}$ u ne-zavádí do paměti při inicializaci desítky dalších fontů označených jako `\preloaded`, protože další fonty se dají zavést povelem `\font` až v době potřeby. Na starodávných strojích bylo možná užitečné zavést fonty „do rezervy“ už při inicializaci formátu, aby pak příkaz `\font` použitý v dokumentu moc nezdržoval. Při dnešních rychlostech počítačů je toto opatření zcela zbytečné, a proto v  $\text{\LaTeX}$ u nepoužité.

Matematická sazba funguje v  $\text{\LaTeX}$ u zcela stejně, jako v originálním `plainu`. Je to díky tomu, že implicitně zavedené  $\text{\LaTeX}$ fonty jsou konzervativním rozšířením Computer Modern fontů. Při zavádění jiných fontů do dokumentu je potřeba počítat při matematické sazbě s některými obtížemi (viz následující dvě sekce).

## 4.4 Makro `opmac.tex`

Makro `opmac.tex` je od konce roku 2012 součástí balíčku  $\text{\LaTeX}$ u. Napíšete-li do záhlaví dokumentu

```
\chyph \input opmac
```

pak máte k dispozici další makra, která řeší následující věci: automatickou tvorbu obsahu, číslování, kapitoly, sekce, křížové reference, verbatim prostředí, vkládání



obrázků, bibtexové odkazy. Tuto sadu svých maker dávám k dispozici s tím, že jsem si vědom, že makra neřeší všechno a ve všech souvislostech, nicméně kladu důraz na jednoduchost a srozumitelnost maker. Předpokládám, že si je uživatelé předefinují k obrazu svému, nicméně usnadním začínajícímu uživateli první kroky směrem k  $\mathcal{C}\mathcal{S}$ plainu.

WWW stránka nabízející OPmac je <http://petr.olsak.net/opmac.html>. Uživatelká dokumentace k makrům je v souboru `opmac-u.pdf` a technická dokumentace v souboru `opmac-d.pdf`.

## 4.5 Použití PostScriptových fontů v $\mathcal{C}\mathcal{S}$ plainu

Součástí  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u v balíčku `cspsfonts.tar.gz` jsou kromě metrik také soubory, které předefinují textové fonty z výchozích  $\mathcal{C}\mathcal{S}$ fontů na fonty některé rodiny z „base 35“. Jak jsme již uvedli v předchozí kapitole, fonty z balíčku `cspsfonts.tar.gz` jsou kódovány v ISO-8859-2, tedy v souladu s vnitřním kódováním použitým v  $\mathcal{C}\mathcal{S}$ plainu. Níže je tabulka souborů, které zavádějí PostScriptové fonty do  $\mathcal{T}\mathcal{E}\mathcal{X}$ u primitivem `\font`:

soubor	Rodina fontů
-----	-----
<code>cavantga.tex</code>	Avantgarde Book
<code>cbookman.tex</code>	Bookman
<code>chelvet.tex</code>	Helvetica
<code>cncent.tex</code>	New Century
<code>cpalatin.tex</code>	Palatino
<code>ctimes.tex</code>	Times Roman

Chcete-li například přepnout do písma Bookman, stačí napsat na začátek dokumentu:

```
\input cbookman
```

Pokud sami pracujete s primitivem `\font` například pro zavedení větších velikostí fontů u nadpisů, doporučuji použít následující konstrukci:

```
\font\titulfont=\fontname\tenbf\space scaled \magstep2
```

Tato konstrukce není závislá na konkrétním fontu, takže když později změníte před takovou konstrukcí `\input bookman` například na `\input cpalatin`, změní se automaticky i font pro nadpisy.

Uvedené soubory `cavantga.tex`, ..., `ctimes.tex` zavádějí čtyři běžné varianty textových fontů `\tenrm`, `\tenit`, `\tenbf`, `\tenbi` a k nim pátou variantu `\tentt` implicitně ve velikosti 10pt. Uživatel místo přímého použití uvedených přepínačů používá obvykle makra `\rm`, `\it`, `\bf`, `\bi`, `\tt`.

Je možné tyto jednou zavedené fonty kdykoli později změnit do jiné velikosti. Například po `\def\sizespec{at12pt}\resizeall` bude všech pět variant fontů připraveno ve velikosti 12pt. Individuálně lze změnit velikost každému fontu zvlášť příkazem `\resizefont` jehož parametr je fontový přepínač (nikoli makro). Přepínač změní velikost svého fontu podle makra `\sizespec`. Pozor: makra `\resizefont`

a `\resizeall` sice změní velikosti fontů odpovídajícím přepínačům, ale nenastaví aktuální sazbu novým fontem, takže bez následného použití přepínače nevidíte výsledek. Je tedy nutno například psát: `\def\sizespec{at12pt}\resizeall\tenrm`.

Změna velikosti fontů má lokální platnost, takže třeba makro pro sazbu titulu může vypadat takto:

```
\def\titul#1{\centerline{\def\sizespec{at20pt}\resizefont\tenbf\tenbf#1}}
```

Makro `\resizeall` lze rozšiřovat o další fontové přepínače mimo uvedených pět. O toto rozšíření se postará makro `\regfont <přepínač>`. Po použití tohoto makra je `<přepínač>` tzv. „registrovaný“, takže bude měnit velikost svého fontu po spuštění makra `\resizeall`. Fontové soubory `cavantga.tex`, ..., `ctimes.tex` nastavují obvykle jen základních pět variant a žádné nové přepínače neregistrují. Zavádí-li ale fontový soubor fonty i pro další varianty, měl by použít `\regfont`.

Fontové soubory `cavantga.tex`, ..., `ctimes.tex` automaticky načtou další soubor maker `tx-math.tex`. Tato makra zavedou do matematické sazby TXfonty. Ty jsou vizuálně kompatibilní s TimesRoman a mnoha dalšími fonty.

Po zavedení makra `tx-math.tex` jsou k dispozici stovky příkazů pro jednotlivé znaky matematické sazby (nadmnožina znaků z AMST<sub>E</sub>Xu) a dále jsou (kromě přepínače `\cal` z plainT<sub>E</sub>Xu) k dispozici následující přepínače matematických abeced:

```
\frac    % matematická fraktura
\script  % skript mnohem zakroucenější než \cal
\bbchar  % písmena kreslená dvěma tahy
\bf      % bold sans-serif písmo
\bi      % bold slanted sans-serif písmo, někdo tím značí vektory
```

Kompletní soubor matematických fontů je makrem `tx-math.tex` zaveden ve velikosti 10/7/5pt (základní/indexová/indexindexová velikost). Ovšem tuto implicitní volbu je možné kdykoli v dokumentu změnit pomocí (například):

```
\setmathsizes[12/8.4/6]\normalmath
```

Příkaz `\normalmath` pozmění všechny matematické fonty do potřebných nových velikostí deklarovaných pomocí `\setmathsizes`. Příkaz `\boldmath` dělá totéž, ale připraví místo „normálních“ fontů jejich tučné varianty. Tento příkaz se hodí použít do maker s nadpisy. Změny fontů mají lokální platnost.

Kromě `tx-math.tex` je k dispozici analogické makro `ams-math.tex`, které ponechává v matematické sazbě  $\mathcal$ fonty a přidává k nim kompletní sadu  $\mathcal{AMS}$  fontů. Toto makro nabízí pro práci s fonty (volba matematické abecedy, změna velikostí a duktu) stejné příkazy jako makro `tx-math.tex`.

Obě makra navíc nabízejí dva přepínače, které mění chování proměnných, číslíc a fontů pro sazbu `sin`, `cos`, `lim`, atd.:

```
\itvariables % proměnné a \rm z aktuálního textového fontu
\mitvariables % proměnné ze speciálního fontu, \rm taky.
```

Volba `\itvariables` je implicitní v `tx-math.tex`, protože se předpokládá, že „kolem“ je textový font se kterým by měly být proměnné v matematice v souladu.



Naopak v `ams-math.tex` je implicitní `\mitvariables`, aby se využilo mírně modifikované kurzívy vyvinuté speciálně pro matematickou sazbu v Computer Modern. „Kolem“ se totiž předpokládá běžný text v  $\mathcal{S}$ fontech nebo ve vizuálně kompatibilních fontech.

Kromě souborů `cavantga.tex`, ..., `ctimes.tex` jsou v balíčku  $\mathcal{S}$ plainu připraveny další obdobné, jejichž názvy většinou začínají na `cs-`. Například `cs-antt.tex` zavádí fonty Antykwy Toruńské. Přehled o všech takových souborech by se měl objevit na terminálu po `\input cs-all`. Toto makro nevykoná nic, jen vypíše seznam dostupných fontových souborů. Iniciativě k přidání dalších fontových souborů se mezi nekladou.

Po zavedení rodiny PostScriptových fontů je potřeba dát pozor na matematickou sazbu. Jedná se o veškerou sazbu realizovanou ve zdrojovém textu mezi dolary. Ta i po změně textových fontů pracuje s původními  $\mathcal{S}$ fonty a Computer Modern fonty. Jedině tak je dosaženo, že realizace všech matematických symbolů z plainu zůstává zachována. Napíšete-li po zavedení nového textového fontu v textu například číslovku 2 bez použití dolarů, dostanete dvojku z nového fontu. Použijete-li ale 2, na výstupu bude dvojka z  $\mathcal{S}$ fontů. Toto míchání fontů není estetické. Chcete-li se mu aspoň částečně vyhnout, použijte po zavedení rodiny fontů povel `\setssimplemath`, například:

```
\input cbookman \setssimplemath
```

Příkaz zavede i pro fonty, které  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  používá při sazbě mezi dolary ( $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ová rodina 0 a 1), odkazy na nově použité PostScriptové fonty. Tento příkaz je ale potřeba použít s velkou opatrností, protože bude fungovat zřejmě jen jednoduchá matematika. Například řecké symboly, které použité rodiny fontů neobsahují, nenávratně ztratíte. Navíc matematické symboly a natahovací závorky ( $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ová rodina 2 a 3) zůstávají i nadále v Computer Modern, takže se míchání typů fontů zcela nevyhnete.

Pokud chcete používat složitější matematickou sazbu kombinovanou s PostScriptovými fonty, pak máte dvě možnosti:

1. Nastavit `\mathcode` všech matematických symbolů tak, aby byly použity například znaky z PostScriptového fontu Symbol. Matematické znaky z tohoto fontu jsou navrženy jako doplněk k fontu Times Roman, ale hodí se i k jiným PostScriptovým fontům. Bohužel, natahovací závorky a velké operátory (jedna ze specialit  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ u) v tomto fontu nejsou, takže tyto objekty je nutné ponechat v Computer Modern. Abyste nemuseli `\mathcode` pracně nastavovat, doporučuji použít makro OFS [12].

2. Zakoupit nějaký komerční matematický font včetně natahovacích závorek. Vhodný je například MathTimes firmy Y&Y. Pro šťastné majitele této sady fontů je v  $\mathcal{S}$ plainu distribuován soubor `cmt.tex`, takže pokud na začátku dokumentu uvedete

```
\input ctimes  
\input cmt
```

pak budete mít i matematickou sazbu v „Times-Roman stylu“ se vším všudy včetně řeckých symbolů a natahovacích závorek. Jen některé drobnosti (například

skákavé číslice v plainu dosažené pomocí ) musíte udělat jinak. Přečtěte si manuál k zakoupené sadě fontů.

Matematické fonty v „Times-Roman stylu“ se dají kombinovat i s mnoha textovými PostScriptovými fonty dynamické antikvy s dostačujícím estetickým výsledkem. Podstatně horší je kombinovat dynamickou antikvu textového PostScriptového fontu se statickou antikvou Computer Modern. To je důvod, proč se nákup sady fontů MathTime vyplatí.

Pokud chcete používat rozsáhlé množství fontů sofistikovaným způsobem v matematické i textové sazbě, můžete použít balíček OFS [12].

## 4.6 UTF-8 kódovaný $\mathcal{C}$ splain

Od roku 2012 je doporučeno implicitně generovat  $\mathcal{C}$ splain se vstupním kódováním UTF-8 pomocí encTeXu. Každý znak mimo ASCII je v UTF-8 kódován dvěma nebo více byty. Tato sekce pojednává o vlastnostech takto generovaného formátu. Poznáte jej podle toho, že do logu a na terminál napíše:

```
The utf8->iso8859-2 re-encoding of Czech+Slovak alphabet
activated by encTeX
```

Vygenerovaný formát zaručí správné zpracování ASCII znaků. Dále je při výchozím nastavení zaručeno správné zpracování jen znaků české a slovenské abecedy, tedy znaků:

Á á Ä ä Č č Ď ě É é Ě ě Í í Ĺ ĺ Ľ ľ Ň ň Ó ó Ŏ ŏ Ô ô Ř ř Š š Ť ť Ú ú Ů ů Ű ű Ý ý Ž ž

Tyto znaky jsou z UTF-8 kódu mapovány na interní jeden byte podle kódování  $\mathcal{C}$ sfontů. Dále jsou implicitně mapovány UTF-8 kódy všech znaků, které mají svou řídicí sekvenci definovanou v plainu nebo v  $\mathcal{C}$ splainu. Jsou to tyto znaky:

```
plain: \ss ß \l ł \L Ł \ae æ \oe œ \AE Æ \OE Œ
       \o ø \O Ø \i í \j j \aa å \AA Å
       \S § \P ¶ \copyright © \dots ... \dag † \ddag ‡
csplain: \clqq „ \crqq “ \flqq « \frqq » \promile ‰
```

Pokud je na vstupu jakýkoli jiný UTF-8 kód, implicitně je nemá  $\mathcal{C}$ splain mapován, takže se vypíše na terminál a do logu následující varování:

```
WARNING: unknown UTF-8 code: ‘X = ^^xx^^xx’ (line: ??)
```

a do tiskového výstupu se vloží na dané místo černý čtvereček. Uživatel si musí takový znak dodefinovat. Například po hlášení:

```
WARNING: unknown UTF-8 code: ‘Ñ = ^^c3^^91’ (line: 42)
```

si uživatel do záhlaví dokumentu například doplní definici:

```
\mubyte\Ntilde ^^c3^^91\endmubyte % UTF-8 kód mapován na \Ntilde
\def\Ntilde{{\~N}} % sekvence \Ntilde definována
```

Po této úpravě se při zpracování dokumentu už varování neobjeví a vstupní kód se zpracuje jako řídicí sekvence `\Ntilde`, která expanduje na `\~N`, takže v tiskovém výstupu se objeví Ñ.

Pokud se řídicí sekvence mapovaná pomocí `\mubyte` objeví v argumentu `\write` souboru, nebude expandovat, ale promění se při zápisu do souboru zpětně do odpovídajícího UTF-8 kódu.

Během `\write` a při zápisu do logu se také zpět na UTF-8 kódy proměňují interní byte, které byly mapovány. Implicitně tedy jsou to znaky české a slovenské abecedy vyjmenované výše. Ostatní nemapované interní byte s kódem větším jak 127 se přepisují podle dvouzobákové konvence např. takto: `^^ad`. Jestliže do deklarace dokumentu napíšete `\xprncodes=1`, budou se ostatní nemapované byty vypisovat přímo.

Příkazy `\mubytein`, `\mubyte` a `\endmubyte` jsou součástí `encTeXu` a jsou popsány v jeho dokumentaci. Stručně řečeno `\mubytein=0` vypíná přechodně použití kódovací tabulky a `\mubyte token string\endmubyte` zařadí do kódovací tabulky další údaj: `string` bude na vstupu převeden na interní `token` v `TeXu` a při činnosti `\write` bude `token` zpětně převeden na `string`.

Při použití jiných fontů pomocí `\input ctimes` (atd., viz sekci 4.5) je zavolán soubor `chars-8z.tex`, který mapuje další UTF-8 kódy na řídicí sekvence, jež jsou pomocí `\chardef` propojeny se znakem fontu. Jsou to tyto znaky:

```
\euro €      \trademark ™    \registered ®    \ellipsis ...
\textbullet • \sterling £    \currency ¤
```

Soubor `chars-8z.tex` navíc předefinovává makra `plainu` `\P`, `\S`, `\dag`, `\ddag`, `\copyright`, `\lslash`, `\Lslash` pomocí `\chardef`, aby tyto řídicí sekvence vedly přímo na znak ve fontu.

V balíčku `enc tex.tar.gz` (a podruhé i v `csplain.tar.gz`) jsou připraveny soubory, které mapují UTF-8 kódy celých bloků UNICODE tabulky na řídicí sekvence a definují jejich výchozí chování. Můžete použít:

```
\input utf8lat1 % Latin-1 Supplement U+0080--U+00FF
\input utf8lata % Latin Extended-A U+0100--U+017F
```

a možná v budoucnu i další. Po zavedení těchto souborů se nově deklarované UTF-8 kódy mapují na řídicí sekvenci, např. `ě` se mapuje na `\edieresis` a tato sekvence expanduje na příslušné sestavení akcentu, v tomto příkladě na `"e`. Na interní byte zůstávají mapovány jen znaky české a slovenské abecedy. Pokud ale *před zavedením těchto souborů* použijete `\input t1code` (přechod na interní kódování T1, tj. podle Corku), pak se na interní byte mapují také všechny znaky definované v T1 kódování. Pro nově mapované řídicí sekvence jsou použity definice ze souborů `utf8lat*.tex` jen tehdy, pokud tyto řídicí sekvence nejsou definovány dříve.

Pokud předložíte `CSplainu` (generovaném pro UTF-8 kodování vstupu) dokument kódovaný jinak, než v UTF-8, skoro jistě se dočkáte chybového hlášení:

```
! UTF-8 INPUT IS CORRUPTED ! May be you are using another input encoding.
```

Je ovšem možné snadno přejít do módu, při němž UTF-8 kódovaný `ℒplain` pracuje stejně jako `ℒplain` s kódováním ISO-8859-2. Stačí na začátek dokumentu napsat:

```
\mubytein=0 \mubyteout=0 \mbytelog=0 \xprncodes=1
```

Tuto práci vykoná také soubor `utf8off.tex`, tj. stačí napsat `\input utf8off`. Tento soubor navíc definuje makro `\clearmubyte`, které vymaže data deklarovaná pomocí `\mubyte... \endmubyte`.

Je dokonce možné mít na vstupu při sazbě jediného dokumentu různé soubory různě kódované. Není nutné při přechodu z jednoho kódování přepínat na druhé. Stačí na začátek dokumentu napsat:

```
\input mixcodes
```

a od této chvíle je možné na vstupu střídat texty kódované dle UTF-8, ISO-8859-2 nebo CP1250 dle libosti. Všechny výstupy pomocí `\write` jsou kódovány podle UTF-8 a jsou tedy opět připraveny k načtení.

V předchozím odstavci není zcela přesná formulace „střídat kódování dle libosti“. Platí to jen pro český text. Slovákům nefunguje »Ľ« v CP1250. Pokud jim to vadí, napíšu `\shyph` před voláním `\input mixcodes`. Pak ale zase nefunguje »ž« v ISO-8859-2. Tyto dva znaky jsou v uvedených dvou kódováních poněkud v konfliktu. UTF-8 vstup ale funguje bez problémů.

## 4.7 Použití fontů kódovaných podle Corku (T1 kódování) nebo v Unicode

Od verze `ℒplainu` <Feb. 2000> načítá tento formát při inicializaci tabulky vzorů dělení nejen v ISO-8859-2, ale též v kódování podle Corku (tzv. T1 kódování). Tyto tabulky „čekají“ na případ, kdy uživatel bude chtít použít ve svém dokumentu fonty, jejichž metriky jsou kódované v T1. Může se totiž stát, že uživatel nebude chtít připravovat metriky a virtuální fonty pro nově zakoupené PostScriptové fonty ručně (pomocí programů `a2ac` a `afm2tfm` nebo pomocí `fontinst`), ale bude chtít využít již hotové metriky nabízené v `TeX`ových archivech. Tyto hotové metriky jsou ale většinou v T1 kódování.

Od verze `ℒplainu` <Nov. 2012> načítá tento formát při své inicializaci tabulky vzorů dělení také v UNICODE, pokud detekuje, že je použit `TeX`ový program, který interně dovede pracovat v UNICODE. Takovým `TeX`ovým programem může být `LuaTeX`, `XeTeX` nebo `Omega`.

Upozorňuji na určitá omezení použití `ℒplainu` s T1 kódovanými fonty:

- Všechny fonty v celém dokumentu musejí být kódovány jednotně v ISO-8859-2 nebo v T1 nebo v UNICODE. V dokumentu nelze fonty oboutěchto kódování jednoduše míchat.
- `ℒplain` *implicitně* nastavuje vnitřní kódování do ISO-8859-2. Změníte-li tuto vlastnost, pak použitý povel v systému se už nesmí nazývat `csplain`.
- Změnit vnitřní kódování `TeXu` příkazem `%&` na prvním řádku dokumentu je možné ve web2c distribucích `TeXu` a odvozených. V jiných instalacích nemusí být tato vlastnost možná. Proto je třeba počítat s tím, že dokument opírající se o tento příkaz nemusí být zcela přenositelný na jiné implementace `TeXu`.

Přepnutí na vnitřní kódování podle Corku uděláte příkazem `\input t1code`. Přepnutí na kódování UNICODE provedete pomocí `\input ucode`. Toto by měl být první příkaz v dokumentu, pak teprve následuje `\chyp` nebo `\input opmac` nebo načítání fontových souborů.

Kromě toho, Nemáte-li UTF-kódovaný `csplain`, se musíte se postarat o to, aby se vstupní kódování správně konvertovalo na vnitřní kódování  $\TeX$ u. Například ve web2c distribuci použijete volbu `-translate-file`. Příklad použití T1 kódovaného fontu:

```
%&csplain -translate-file=il2-t1
\input t1code    %% Některé definice závislé na kódování
\chyp            %% Příkaz nyní zapne tabulku 15 místo tabulky 5
\font\font=ptmr8t %% Zavedení T1 kódovaného fontu Times-Roman
\font\font
Tady je český text zpracovaný uvnitř  $\TeX$ u v kódování T1
včetně použití správné tabulky pro dělení slov.
\end
```

Takto připravený soubor  $\TeX$ ujete na web2c instalaci nikoli повеlem `csplain`, ale raději повеlem `tex`, tedy:

`tex dokument`

Program `tex` z web2c implementace se podívá do prvního řádku dokumentu a pokud tam najde dvojici `%&`, zavede formát a TCX tabulku (která pozměňuje xord/xchr vektor) podle toho, co je za touto dvojicí napsáno. Pokud byste použili skript `csplain` obsahující přepínač `-translate-file=il2-cs`, nebude to fungovat, protože přepínač na příkazovém řádku má vyšší prioritu než v dokumentu. V novějších implemetacích příkazu `csplain` je použit skript, který obsahuje přepínač `-default-translate-file=il2-cs`. Ten má prioritu nižší, takže pak lze při zpracování dokumentu s konstrukcí `%&` použít i příkaz `csplain`.

Do distribuce  $\mathcal{S}$ plainu jsou zařazeny soubory `dcfonts.tex`, `ecfonts.tex` a pro inspiraci též `ttimes.tex`. Pokud na začátku dokumentu například napíšete:

```
%&csplain -translate-file=il2-t1
\input ecfonts
\chyp
```

převádíte celou sazbu do EC fontů kódovaných podle Corku. V zaváděném souboru `ecfonts.tex` je proveden `\input t1code`, takže se o to přímo v dokumentu už není potřeba starat.

Při zavádění PostScriptových fontů (například použitím ~~`ttimes.tex`~~) nastávají stejné problémy s matematickou sazbou, jako při použití PostScriptových fontů kódovaných v ISO-8859-2. Tyto problémy jsou popsány v předchozí secei.

Pokud dáte `\input t1code` před `\input ctimes`, (nebo `\input cavantga`, atd.), budou požadované fonty zavedeny v T1 kódování. Takže k sazbě Bookmanem v T1 kódování stačí v UTF-8 kódovaném  $\mathcal{S}$ plainu napsat:

```
\input t1code \input cbookan
```

Bohužel T1 kódované fonty jsou generovány programem fontinst, který má ve svých konfiguračních souborech chybu pro písmena *ť* a *ď*. Tato písmena jediná nejsou v těchto fontech realizovaná jako přímé znaky a jsou poskládána z komponent naprosto odstrašujícím způsobem. Kdo opraví tuto chybu ve fontinstu, nechtě o tom, prosím, reportuje na příslušných místech, aby mohli správci distribucí přegenerovat 8t metriky.

Pokud dáte v 16bitovém T<sub>E</sub>Xu (tj. XeT<sub>E</sub>X, LuaT<sub>E</sub>X nebo v něčem podobném) `\input ucode` před `\chyph` a `\input lmfnts`, bude čeština uvnitř T<sub>E</sub>Xu zpracována v Unicode a tištěna Latin Modern Unicode fonty. Jiné soubory pro čtení fontů zatím pro 16bitový T<sub>E</sub>X upraveny nejsou.

## 4.8 Vzory dělení slov v různých jazycích a kódováních

Implicitně načítá C<sub>S</sub>plain při generování formátu následující vzory dělení:

- `\USenglish=0` ... (implicitní vzor z plainT<sub>E</sub>Xu) v ASCII kódování
- `\czILtwo=5` ... čeština v ISO-8859-2
- `\skILtwo=6` ... slovenština v ISO-8859-2
- `\czCork=15` ... čeština v T1 kódování
- `\skCork=16` ... slovenština v T1 kódování
- `\czUnicode=115` ... čeština v Unicode (jen, pokud je detekován XeT<sub>E</sub>X, LuaT<sub>E</sub>X, nebo jiný 16bitový T<sub>E</sub>X)
- `\skUnicode=116` ... slovenština v Unicode (jen, pokud je detekován XeT<sub>E</sub>X, LuaT<sub>E</sub>X, nebo jiný 16bitový T<sub>E</sub>X)

Jednotlivé vzory dělení se v dokumentu zapínají pomocí `\uslang`, `\czlang` a `\sklang`. Příkaz `\uslang` zapne také `\nonfrenchspacing` a ostatní zapínají `\frenchspacing`. Jsou zachovány staré názvy těchto přepínačů: `\ehyph=\uslang`, `\chyph=\czlang` a `\shyph=\sklang`. Přejít k novým názvům s aspoň dvěma písmenky pro jazyk je důsledkem nové možnosti v C<sub>S</sub>plainu použít 54 různých jazyků.

V roce 2012 byl zcela přepracován soubor `hyphen.lan`, který je použit C<sub>S</sub>plainem pro čtení vzorů dělení slov. Na řádcích 48 až 160 tohoto souboru jsou za dvojtečkami skryty jazyky a kódování zhruba ve tvaru:

```
\let\czCork=y      % Czech
\let\skCork=y      % Slovak
:\let\deCork=y     % German
:\let\frCork=y     % French
:\let\plCork=y     % Polish
:\let\cyCork=y     % Welsh
:\let\daCork=y     % Danish
...
:\let\saUnicode=y  % Sanskrit
:\let\ruUnicode=y  % Russian
:\let\ukUnicode=y  % Ukrainian
:\let\hyUnicode=y  % Armenian
:\let\asUnicode=y  % Assamese
```



Stačí si vybrat jazyky, odstranit příslušnou dvojtečku a znovu vygenerovat formát `ČSplain`. Nebo můžete přidat vzor bez zásahu do souboru `hyphen.lan` vhodné zvolenou zprávou na příkazovém řádku při generování formátu. Například:

```
pdftex -ini -enc "\let\plCork=y \let\enc=u \input csplain.ini"
nebo
pdftex -ini -enc "\let\allpatterns=y \let\enc=u \input csplain.ini"
```

Jakmile je načten vzor dělení pro nový jazyk, je automaticky připraven k tomu odpovídající přepínač, tedy například `\delang` po načtení `\deCork`. Seznam všech načtených vzorů dělení je v makru `\pattlist`.

Na začátku zpracování dokumentu jsou inicializovány vzory dělení `\..ILtwo`. Vzory dělení typu `\..Cork` budou fungovat až po `\input t1code` a vzory dělení `\..Unicode` budou fungovat až po `\input ucode`. Pozor: `ČSplain` neřeší správné nastavení `\lccode` a `\uccode` v 16bitovém `TeXu` v rozsahu větším, než jsou znaky z české a slovenské abecedy. Takže při použití exotických jazyků si uživatel musí nastavit k tomu správná `\lccode`.

`ČSplain` definuje pro každý jazyk s načtenými vzory dělení makro `\lan:<číslo>` jako `<zkratku jazyka>`, například `\lan:5` stejně jako `\lan:15` expandují na `cz`. Programátor maker toho může využít. Programátor maker dále může využít toho, že makra `\czlang`, `\delang` atd. volají makro `\initlanguage{<zkratka jazyka>}`. Toto makro implicitně neudělá nic, ale programátor maker si je může předefinovat dle svého. Protože je `\initlanguage` zavolán těsně za `\language=<číslo>` v kontextu:

```
\..lang -> \language=<číslo>\relax
            \initlanguage{<značka>}\frenchspacing
            \lefthyphenmin=<lhm>\righthyphenmin=<rh>%
            \message{<text>}
```

je možné, aby programátor maker odebral další inteligenci maker `\..lang` a převzal je do vlastních rukou. Třeba definuje:

```
\def\initlanguage #1#2#3\message#4{#3\csname lg:#1\endcsname}
```

Toto řešení přebírá z makra `\..lang` jen nastavení registrů `\lefthyphenmin` a `\righthyphenmin`, ale ruší implicitní `\message` i nastavení `\frenchspacing`. Místo toho se předpokládá, že budou definována makra `\lg:<značka>`, ve kterých budou tyto věci (a možná mnoho dalších jako třeba nastavení implicitního fontu) řešeny pro každý jazyk individuálně.

Přepínání mezi kódováními funguje pomocí `\corklangs ... \czlang` (nyní je aktivní vzor dělení `\czCork`) a dále třeba `\iltwolangs ... \czlang` (nyní je aktivní vzor dělení `\czILtwo`). Makra `\iltwolangs`, `\corklangs`, a `\unicodelangs` jsou spuštěna při `\input il2code`, `\input t1code` a `\input ucode`, takže uživatel to nemusí řešit.

Je potřeba si uvědomit, že 8bitovým `TeXem` sice můžete vynutit načtení Unicodových vzorů dělení slov, ale bude vám to málo platné, tedy k ničemu. Dále je potřeba vědět, že v 16bitovém `TeXu` můžete načíst vzory dělení `\..ILtwo` a `\..Cork` korektně jen pro češtinu a slovenštinu, protože další vzory dělení jsou čteny pomocí

triku s aktivními znaky, který dekoduje 8bitový vstup. Nicméně typicky v 16bitovém T<sub>E</sub>Xu není inicializován 8bitový vstup, takže se to poláme.

## 4.9 Nové řídicí sekvence **Č**splainu oproti Knuthovu plainu

**\austrian**

~~Registr rezervovaný pro číslo tabulky vzorů dělení slov pro rakouskou němčinu (hodnota 2). Vzory dělení pro tento jazyk implicitně nejsou zavedeny.~~

**\bi**

Přejde na tučnou krurzívu. Příkaz doplňuje sadu příkazů **\rm**, **\it** a **\bf** definovanou v plainT<sub>E</sub>Xu.

**\cmaccents**

Nastaví řídicí sekvence **\^**, **\‘**, **\’**, **\v** a **\"** tak, aby fungovaly stejně, jako v originálním plainu (použití primitivu **\accent**). Sekvence **\r** je nedefinována. Toto je defaultní nastavení. Pokud slovo obsahuje akcentovaný znak zapsaný některou z těchto sekvencí, pak pro ně nefunguje automatické dělení.

**\cmaccentsmessage**

Způsobí výpis informace do logu a na terminál o použití příkazu **\cmaccents**. Pokud vás tato zpráva obtěžuje, nastavte **\let\cmaccentsmessage=\relax** před vyvoláním příkazu **\cmaccents**.

**\corklangs**

Inicializace vzorů dělení podle T1 kódování (Cork).

**\csaccents**

Nastaví řídicí sekvence **\^**, **\‘**, **\’**, **\v**, **\"** a **\r** tak, aby expandovaly na odpovídající 8bitové reprezentace znaků podle ISO-8859-2 (po **\input t1code** pak expandují na kódování podle Corku). Sekvence **\r** u expanduje na ů. Pokud slovo obsahuje akcentovaný znak zapsaný některou z těchto sekvencí, pak pro ně funguje i automatické dělení slov.

**\csaccentsmessage**

Způsobí výpis informace do logu a na terminál o použití příkazu **\csaccents**. Pokud vás tato zpráva obtěžuje, nastavte **\let\csaccentsmessage=\relax** před vyvoláním příkazu **\csaccents**.

**\clqq**

Vytiskne levé české uvozovky („).

**\crqq**

Vytiskne pravé české uvozovky (“).

**\chyph =\czlang**

Aktivuje tabulku dělení slov podle registru **\czech** a nastaví stejnoměrné mezerování za tečkami (tzv. **\frenchspacing**). Dále nastavuje **\lefthyphenmin=2** a **\righthyphenmin=3**. Druhý registr lze změnit na dvojku, vzory dělení jsou na to připraveny.

**\czech**

Registr rezervovaný pro číslo tabulky vzorů dělení slov pro češtinu (hodnota **\iltwoczech** nebo **\toneczech** podle toho, zda nebylo nebo bylo použito **\input t1code**).



`\ehyph = \uslang`

Aktivuje tabulku dělení slov podle registru `\USenglish` a nastaví větší mezerování za tečkami (tzv. `\nonfrenchspacing`). Toto nastavení je výchozí.

`\english`

~~Registr rezervovaný pro číslo tabulky vzorů dělení slov pro britskou angličtinu (hodnota 4). Vzory dělení pro tento jazyk implicitně nejsou zavedeny.~~

`\extrahyphenchar`

Alternativní znak spojovníku (-). Kód 156. Po `\input t1code` není toto makro definováno.

`\extrahyphens`

Nastaví rozdělovací znak pro běžné textové fonty na `\extrahyphenchar`. Defaultně je v plainu jako rozdělovací znak nastaven znak minus (ASCII 45), což může činit v českém textu komplikace.  $\TeX$  totiž v případě rozdělovacího znaku přímo ve vstupním textu může dělit tak, že znak neopakuje na dalším řádku. Po `\extrahyphens` bude mít dělicí znak kód 156, což je kód, který se na vstupu nevyskytuje. Máme tím zaručeno, že ve slovech „je-li“ nebude  $\TeX$  dělit vůbec. Jinou aplikací je použití speciální metriky nebo dokonce speciálního tvaru pro `\extrahyphenchar`.

`\flqq`

Levé francouzské uvozovky («), které se v češtině a němčině používají vpravo.

`\frqq`

Pravé francouzské uvozovky (»), které se v češtině a němčině používají vlevo, tedy »takto«.

`\french`

~~Registr rezervovaný pro číslo tabulky vzorů dělení slov pro francouzštinu (hodnota 3). Vzory dělení pro tento jazyk implicitně nejsou zavedeny.~~

`\german`

~~Registr rezervovaný pro číslo tabulky vzorů dělení slov pro němčinu (hodnota 1). Vzory dělení pro tento jazyk implicitně nejsou zavedeny.~~

`\iltwoczech = \czILtwo`

Konstanta 5 rezervovaná pro číslo tabulky vzorů dělení slov pro češtinu při kódování fontů podle ISO-8859-2.

`\iltwolangs`

Inicializace vzorů dělení podle ISO-8859-2 (implicitní).

`\iltwoslovak = \skILtwo`

Konstanta 6 rezervovaná pro číslo tabulky vzorů dělení slov pro slovenštinu při kódování fontů podle ISO-8859-2.

`\ogonek`

Vytvoří polský akcent pod písmenem: ą.

`\pattlist`

Seznam všech načtených vzorů dělení.

`\promile`

Vytiskne znak ‰.

`\regfont <přepínač>`

Provede registraci fontového přepínače. Tedy přidá do existujícího makra `\resizefontall` další dva tokeny: `\resizefont <přepínač>`.

## `\resizeall`

Provede `\resizefont` podle aktuálně nastaveného `\sizespec` na sekvence `\tenrm`, `\tenit`, `\tenbf`, `\tenbi` a `\tentt` a případně další registrované fontové přepínače. Příklad:

```
\def\sizespec{at12pt}\resizeall \tenrm
```

Nastavená `\it` velikost 12pt `\bf` funguje `\bi` všude.

## `\resizefont <přepínač>`

Argumentem musí být fontový `<přepínač>`, tj. sekvence dříve deklarovaná pomocí `\font`. Tento `<přepínač>` po činnosti příkazu `\resizefont` svůj význam: odkazuje na stejný font ale ve velikosti nastavené podle obsahu makra `\sizespec`. Naříklad: `\def\sizespec{at14pt}\resizefont\tenbf`. Od této chvíle je `\tenbf` tučný font (stejný, jako byl původní `\tenbf`) ve velikosti 14pt. Změna významu je lokální.

## `\resizefontskipat`

Pomocné makro pro `\resizefont`.

## `\shyph=\sklang`

Aktivuje tabulku dělení slov podle registru `\slovak` a nastaví stejnoměrné mezerování za tečkami (tzv. `\frenchspacing`).

## `\sizespec`

Makro spolupracující s příkazem `\resizefont`. Makro `\sizespec` je buď prázdné, nebo obsahuje „`at<dimen>`“, nebo „`scaled<number>`“. Nezapomeňte na klíčová slova `at` resp. `scaled`.

## `\slovak`

Registr rezervovaný pro číslo tabulky vzorů dělení slov pro slovenštinu (hodnota `\iltwslovak` nebo `\toneslovak` podle toho, zda nebylo nebo bylo použito `\input t1code`).

## `\tenbi`

Font-selector definovaný jako `\font\tenbi=csbxti10`. Je totiž užitečné mít pre-loadovanou fontovou čtveřici `\tenrm`, `\tenit`, `\tenbf` a `\tenbi`.

## `\toneczech=\czCork`

Konstanta 15 rezervovaná pro číslo tabulky vzorů dělení slov pro češtinu při kódování fontů podle Corku.

## `\toneslovak=\skCork`

Konstanta 16 rezervovaná pro číslo tabulky vzorů dělení slov pro slovenštinu při kódování fontů podle Corku.

## `\unicodelangs`

Inicializace vzorů dělení v Unicode.

## `\USenglish`

Registr rezervovaný pro číslo tabulky vzorů dělení slov pro americkou angličtinu (hodnota 0).

## `\uv`

Makro pro uvozovky. Používá se `\uv{takto}`. Makro je definováno tak, aby se uvnitř jeho argumentu dala použít verbatim konstrukce. To ale způsobí, že je potlačen kerning před pravou uvozovkou. Pokud nepoužíváte verbatim konstrukce v argumentu uvozovek, doporučuji makro předefinovat jednoduše na:

`\long\def\uv#1{\clqq#1\crqq}.`

**Poznámka:** `CSplain` nedefinuje makro pro ‚jednoduché‘ uvozovky, protože `CSfonty` neobsahují pro tyto uvozovky speciální znaky. Můžete ale místo nich použít čárku a apostrof třeba takto:

`\long\def\singleuv#1{,#1'}`

Při generování UTF-8 `CSplainu` se použije `encTeX`, který přidává primitivy `\mubyte`, `\endmubyte`, `\mubytein`, `\mubyteout`, `\mubytelog`, `\specialout`, `\noconvert`, `\xordcode`, `\xchrcode` a `\xprncode`. Během generování je načten soubor `utf8unkn.tex`, který mapuje neznámé kódy na `\warntwobytes` a `\warnthreebytes` a tyto sekvence definuje jako makra. Dále definuje makra `\badutfinput`, `\missingutfchar` a `\xprncodes`.

## Přidání vzorů dělení slov dalších jazyků

`CSplain` implicitně načítá jen anglické vzory dělení (US) a české a slovenské. Od verze <Feb. 2000> navíc načítá české a slovenské vzory dělení nejen v kódování ISO-8859-2, ale i v kódování podle Corku. Anglické vzory dělení jsou implicitně aktivní a přepínáme na ně pomocí `\ehyph`. České vzory dělení se zapínají pomocí `\chyph` a slovenské pomocí `\shyph`.

Pokud potřebujete přidat další jazyk, pak před generováním formátu `CSplain` editujte soubor `hyphen.lan`. Tam najdete příklady doplnění vzorů dělení němčiny nebo francouzštiny (stačí odkomentovat odpovídající řádky). Podle těchto příkladů můžete analogicky zařadit potřebný další jazyk. Nezapomeňte v souboru `hyphen.lan` také definovat přepínač na nové vzory dělení. Pro němčinu definujeme například přepínač `ghyph` takto:

```
\def\ghyph{\language=\german
\lccode'\='\'
\frenchspacing
\lefthyphenmin=2
\righthyphenmin=2}
```

Nakonec přegenerujte formát způsobem popsáním v sekci 4.2.

Vzory dělení západoevropských jazyků jsou většinou kódovány podle ISO-8859-1, přičemž kódování Cork je nadmnožinou tohoto kódování. Je tedy možné přepnout pomocí `\input t1code` na začátku dokumentu do Corku, použít fonty kódované podle Corku a pak přepínat mezi češtinou a třeba němčinou střídavě pomocí přepínačů `\chyph` a `\ghyph`.

Vzory dělení exotických jazyků jsou kódovány ve svém specifickém kódování, ke kterému musíme sehnat font stejně kódovaný. Po přepnutí do nového jazyka musíme přepnout i na tento font. Dále je nutno řešit otázku možné kolize vstupního kódování češtiny/slovenštiny se vstupním kódováním pro nový jazyk. Je proto bezpečnější si pro speciální znaky jazyka udělat makra, která expandují na znak podle kódování fontu, a psát vstupní text pomocí těchto maker.

## 4.10 $\mathcal{C}\mathcal{S}$ plain a $\mathcal{A}\mathcal{M}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$

$\mathcal{A}\mathcal{M}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$  můžete používat v plainu i  $\mathcal{C}\mathcal{S}$ plainu bez nutnosti generovat kvůli tomu formátový soubor. Máte-li dokument v  $\mathcal{A}\mathcal{M}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u, napište jednoduše na jeho začátek:

```
\input amstex
```

a dále můžete dokument zpracovat příkazem  $\mathcal{C}\mathcal{S}$ plain.

## 4.11 pdf $\mathcal{T}\mathcal{E}\mathcal{X}$ + $\mathcal{C}\mathcal{S}$ plain = pdf $\mathcal{C}\mathcal{S}$ plain

Pokud vygenerujete formát  $\mathcal{C}\mathcal{S}$ plain pdf $\mathcal{T}\mathcal{E}\mathcal{X}$ em, je potřeba jej nazvat jinak, aby byl odlišen od formátu  $\mathcal{C}\mathcal{S}$ plain pro originální  $\mathcal{T}\mathcal{E}\mathcal{X}$ . Proto je v  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u zvoleno jméno `pdfcsplain.fmt`. Dále je potřeba připravit příkaz `pdfcsplain`, který spustí pdf $\mathcal{T}\mathcal{E}\mathcal{X}$  s formátem `pdfcsplain.fmt` a zajistí případné překódování na úrovni input procesoru pdf $\mathcal{T}\mathcal{E}\mathcal{X}$ u stejně, jako to dělá příkaz `csplain`. Pokud uživatel nepoužil ve svých makrech konstrukce měnící sazbu při použití pdf $\mathcal{T}\mathcal{E}\mathcal{X}$ u, měl by dostat příkazem `pdfcsplain` naprosto stejný výstup jako při použití příkazu `csplain`. Pouze s tím rozdílem, že výstup nebude v `dvi` souboru, ale v `pdf`.

Ve web2c  $\mathcal{T}\mathcal{E}\mathcal{X}$ u a odvozených distribucích lze generovat formát příkazem

```
pdftex -ini -fmt pdfcsplain csplain.ini
```

a příkaz `pdfcsplain` v UNIXu implementovat jako skript s obsahem

```
pdftex -fmt pdfcsplain -default-translate-file=il2-cs $@
```

Toto je pouze příklad implementace. Na jiných operačních systémech a jiných  $\mathcal{T}\mathcal{E}\mathcal{X}$ ových distribucích se situace může trochu lišit.

Poznamenejme, že v  $\mathcal{T}\mathcal{E}\mathcal{X}$ u a odvozených distribucích většinou stačí napsat `pdfcsplain dokument`, a pokud jste `pdfcsplain` ještě nepoužili, vygeneruje se automaticky.

## 4.12 Historie a budoucnost $\mathcal{C}\mathcal{S}$ plainu

$\mathcal{C}\mathcal{S}$ plain vznikl v roce 1992 jako jednoduché a minimální rozšíření Knuthova plainu používající  $\mathcal{C}\mathcal{S}$ fonty a akceptující 8bitový vstup. Jeho vytvoření bylo motivováno zařazením do  $\mathcal{e}\mathcal{m}\mathcal{T}\mathcal{E}\mathcal{X}$ ové distribuce, která se připravovala k rozesílání členům  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ u.

$\mathcal{C}\mathcal{S}$ plain se opíral a stále opírá o starší makra `hyphen.lan` a `plaina4.tex`, která už měl Olin Ulrych vytvořena dříve.  $\mathcal{C}\mathcal{S}$ plain v době svého vzniku načítal české vzory dělení, které vytvořil Láďa Lhotka heuristicky bez použití slovníků a programu `patgen`. Já jsem pro  $\mathcal{C}\mathcal{S}$ plain vytvořil `csplain.ini` a makro `csfonts.tex`, umožňující při generování formátu číst přímo Knuthovo makro `plain.tex`, a přitom natáhnout místo  $\mathcal{C}\mathcal{M}$ fontů  $\mathcal{C}\mathcal{S}$ fonty.

V roce 1994 byly vzory dělení slov Ládi Lhotky vyměněny za nové české vzory dělení od Pavla Ševečka, který na to použil slovník a `patgen`. V rámci své firmy tyto vzory dělení prodává komerčním firmám pro potřeby DTP programů, jako

byly Ventura, PageMaker nebo Quark. Dnes jsou tyto vzory dělení také například ve Wordu. Aby Pavel Ševeček odlišil volně šířené vzory dělení pro  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$  od komerčně šířených, volně šířené vzory dělení mírně modifikoval. Tvrdí se, že běžný uživatel nepozná rozdíl v kvalitě vzorů dělení komerčních a volně šířených. Ševečkovy vzory dělení slov jsou výrazně kvalitnější než původní Lhotkovy, a proto jsme u těchto vzorů dělení zůstali.

Opuštěním Lhotkových vzorů dělení došlo k poslední změně v  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$ , která může způsobit zpětnou nekompatibilitu: tj. dokumenty vytvořené v  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  před rokem 1994 mohly dopadnout jinak než dnes, protože některá slova mohla být rozdělena jinak. Od této doby je  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  fixován a stabilní podobně, jako Knuthův plain.

Protože jsem autorem názvu  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$ , souborů `csplain.ini`, `csfonts.tex` a množství dokumentace k  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  a protože jej od jeho vzniku udržuji, rozhodl jsem se přísně dbát na zpětnou kompatibilitu. Změny do  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  dělám jen takové, které jsou opravdu nezbytné. To se stává jednou za několik let (viz historické poznámky v `csplain.ini`). Změny dělám tak, že pouze přidám další nejnnutnější makra, ale stávající makra a jejich význam nechávám nezměněna. Uživatelům  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  ručím, že jejich dokumenty napsané v  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  a opírající se o neměnné fonty (např.  $\mathcal{C}\mathcal{S}$ fonty nebo base 35 PostScriptové fonty, metriky z  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ ), budou i v budoucnu  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$ em formátovány naprosto stejně, jako dnes.

Abych mohl takovou záruku uživatelům poskytnout, není  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  zveřejněn pod GNU GPL, ale jedná se o licenci velmi podobnou Knuthově. Přesné znění licence je uvedeno na konci souboru `csplain.ini`. Zhruba řečeno, jedná se o „patent na název“.  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  můžete svobodně distribuovat, používat a měnit, ale pokud jej změníte, nesmíte jej dále distribuovat pod názvem  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$ . Změny v  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  může dělat jen tzv. „současný administrátor  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ “, což jsem zatím stále já. Pokud bych v budoucnu toto brímě někomu předal, pak určitě jen takovému člověku, který má na budoucnost  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  stejný názor jako já.

$\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  i nadále považuji za minimální rozšíření Knuthova plainu a nikdy do něj nepřidám žádné složitější makro vylepšující uživatelský komfort (jako například `eplain`). Zastávám názor, že uživatel plainu a  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  chce mít všechna makra pod svou vlastní kontrolou a raději si je udělá sám, než aby spoléhal na hotová, ale méně stabilní, řešení. Uživatel plainu/ $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  si vytváří vlastní stále znovu používaná makra, která mu musí fungovat ve všech i budoucích verzích  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$ . Proto se snažím  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  pokud možno neměnit.

Lákavá je například změna definice uvozovek v  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  jednoduše na

```
\long\def\uv#1{\clqq#1\crqq}
```

aby fungoval automatický kerning s oběma stranami uvozovek. Tuto změnu ale nikdy v  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  neudělám, protože není zpětně kompatibilní se stávajícím řešením. Raději budu psát do omrzení do dokumentace, že taková jednoduchá definice je asi lepší, než ta z  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$ , a že si ji každý může zařadit do svých maker.

Změnu v  $\mathcal{C}\mathcal{S}\mathcal{P}\mathcal{L}\mathcal{A}\mathcal{I}\mathcal{N}$  z <Feb. 2000> považuji za asi největší, kterou jsem byl ochoten udělat. K zařazení alternativního kódování Cork (které samozřejmě není a nikdy

nebude v  $\mathcal{C}\mathcal{S}\text{plain}$  implicitní), mě motivovala skutečnost, že se kolem mě pohybovalo mnoho uživatelů  $\text{plain}$ , kteří rádi používají fonty v tomto alternativním kódování. Svým krokem jsem jim umožnil používat  $\mathcal{C}\mathcal{S}\text{plain}$ , takže si nemusejí vytvářet své vlastní formáty.

## 5. Formát $\mathcal{C}\mathcal{S}\text{LaTeX}$

Významní uživatelé  $\text{LaTeX}$ u sice v elektronické konferenci v roce 2012 přiznali, že už  $\mathcal{C}\mathcal{S}\text{LaTeX}$  dávno nepoužívají, nicméně se stále přimlouvají, aby v  $\mathcal{C}\mathcal{S}\text{TeX}$ u zůstal. Nebude ale dále rozvíjen a všem uživatelům je kladeno silné doporučení, aby přešli na „normální“  $\text{LaTeX}$ . Navrhoval jsem  $\mathcal{C}\mathcal{S}\text{LaTeX}$  zrušit úplně (aby to lidi nepletlo), ale nebyl jsem vyslyšen.

Od roku 2012 napíše  $\mathcal{C}\mathcal{S}\text{LaTeX}$  při každém spuštění na terminál a do logu toto:

```
**** WARNING **** CSLaTeX is obsolete, don't use it please.  
**** You can use 'normal' LaTeX+Babel or XeLaTeX+polyglossia.
```

Pokud použijete  $\text{LaTeX}$ , pak záhlaví dokumentu může vypadat třeba takto:

```
\documentclass[a4paper,12pt,twoside]{article}  
\usepackage[utf8]{inputenc}  
\usepackage[T1]{fontenc}  
\usepackage[czech]{babel}
```

Při použití  $\text{XeLaTeX}$ u s balíčkem `polyglossia` může záhlaví dokumentu být například takové:

```
\documentclass[a4paper,12pt,twoside]{article}  
\usepackage{polyglossia}  
\setdefaultlanguage{czech}  
\usepackage{fontspec}  
\usepackage{xunicode}
```

Následující text až do konce kapitoly bych mohl celý škrtnout, ale nechám jej beze změny. Čtenář to možná ocení jako materiál ke studiu historie  $\mathcal{C}\mathcal{S}\text{TeX}$ u.

### 5.1 Různé $\text{LaTeX}$ y

Do roku 1992 byl  $\text{LaTeX}$  udržován Leslie Lamportem. Naposledy měl jeho  $\text{LaTeX}$  verzi 2.09. Typické pro tento „starý“  $\text{LaTeX}$  bylo použití příkazu `\documentstyle` namísto `\documentclass` v záhlaví dokumentu. Tato větev  $\text{LaTeX}$ u není dále udržována a podporována. Dnes převzali iniciativu nad  $\text{LaTeX}$ em Němci Frank Mittelbach a Rainer Schöpf a nazvali jej  $\text{LaTeX} 2_{\epsilon}$ . Každý půlrok vytvářejí novou verzi (většinou obsahující jen opravy předchozí verze) a názvem dávají najevo, že se jedná o předchůdce  $\text{LaTeX}$ u 3, na němž už deset let pracují. Dokument napsaný pro  $\text{LaTeX} 2_{\epsilon}$  poznáme vesměs podle toho, že začíná příkazem `\documentclass` místo `\documentstyle`.  $\text{LaTeX} 2_{\epsilon}$  ovšem také akceptuje příkaz `\documentstyle`, přechází přitom do pokusu o emulaci starého  $\text{LaTeX}$ u 2.09, a navíc uživatele upozorní na to, že použil zastaralé záhlaví dokumentu.

V  $\text{\LaTeX}$ <sub>2 $\epsilon$</sub>  je zabudovaná „časovaná bomba“ s rozbuškou na rok a půl. Ta se projeví při generování formátu, nikoli při běžném provozu. Pokud generujete formát ze zdrojů  $\text{\LaTeX}$ <sub>2 $\epsilon$</sub> , které jsou starší než rok a půl, generování končí chybou a na terminálu se objeví varování, že máte instalován příliš starý  $\text{\LaTeX}$  a že je vhodné si obstarat nový. Pokud na toto hlášení odpovíte klávesou Enter, formát se přesto vygeneruje.

V tomto manuálu se budeme dále zabývat jen dnes asi nejpoužívanějším  $\text{\LaTeX}$ <sub>2 $\epsilon$</sub> . Pojmeme  $\text{\LaTeX}$  tedy budeme rozumět  $\text{\LaTeX}$ <sub>2 $\epsilon$</sub> . I za těchto okolností budeme nuceni rozlišovat mezi třemi „druhy“  $\text{\LaTeX}$ ů:

- vanilla  $\text{\LaTeX}$
- babelizovaný  $\text{\LaTeX}$
- $\text{\LaTeX}$

Vanilla  $\text{\LaTeX}$  (nedotčený  $\text{\LaTeX}$ ) vzniká vygenerováním formátu  $\text{\LaTeX}$  pouze za přítomnosti souborů z adresáře `latex/base`, tj. při generování nejsou čteny žádné soubory ovlivňující generování formátu. V distribuci tedy zcela chybí tyto soubory: `fonttext.cfg`, `fontmath.cfg`, `preload.cfg` a `hyphen.cfg`. Tento formát obsahuje pouze anglické dělení slov. Pro český nebo slovenský jazyk není tedy příliš použitelný. Není často používán.

Babelizovaný  $\text{\LaTeX}$  vzniká generováním formátu  $\text{\LaTeX}$  za přítomnosti souborů `fonttext.cfg`, `fontmath.cfg` a `preload.cfg` z balíčku `generic` a `hyphen.cfg` z balíčku `Babel`. Tento formát obsahuje vzory dělení těch jazyků, které jsou v době generování formátu uvedeny v konfiguračním souboru `Babelu` s názvem `language.dat`. Je běžně používán.

Formát  $\text{\LaTeX}$  vzniká generováním formátu  $\text{\LaTeX}$  za přítomnosti souborů `fonttext.cfg` a `hyphen.cfg` z balíčku  $\text{\LaTeX}$ . Ostatní soubory ovlivňující generování formátu mohou chybět nebo být z balíčku `generic`. Obvykle je formát  $\text{\LaTeX}$ u nazván `cslatex.fmt` a nikoli `latex.fmt`, aby bylo možno jej odlišit od běžně používaného babelizovaného  $\text{\LaTeX}$ u. Z toho důvodu je v balíčku  $\text{\LaTeX}$ u k dispozici soubor `cslatex.ini`, který provede `\input latex.ltx`.  $\text{\LaTeX}$  obsahuje vzory dělení anglického, českého a slovenského jazyka. Navíc příkaz `cslatex` musí zajistit konverzi ze vstupního kódování (podle použitého systému) na vnitřní kódování ISO 8859-2 na úrovni vstupního preprocesoru  $\text{\TeX}$ u. Tento princip je tedy shodný s  $\text{\Csplain}$ em.

Je patrné, že absolutní jistotu o tom, jaký  $\text{\LaTeX}$  vlastně používáme, získáme jen pečlivým pročtením logu po vygenerování formátu. Koncept, kdy vlastnosti formátu jsou závislé na obsahu nějakých dodatečných souborů v distribuci v době generování formátů, se mi nelíbí, ale nelze proti tomu nic dělat. Takto to navrhli tvůrci  $\text{\LaTeX}$ u a je to tedy jejich věc. Důsledek tohoto rozhodnutí je, že pokud řekneme: „používám  $\text{\LaTeX}$ “, nikdy není zcela přesně řečeno, co se tím vlastně myslí. Tím se tento koncept diametrálně liší od Knuthova `plainu` nebo též  $\text{\Csplain}$ u.

Co se stane, pokud jsou například v  $\text{\TeX}$ ové distribuci přítomny hned dva soubory `hyphen.cfg` – jeden z balíčku `Babel` a druhý z balíčku  $\text{\LaTeX}$ ? Není-li taková situace v distribuci řešena speciální konfigurací, není definováno, který z těchto souborů se vlastně načte, a tudíž není dopředu známo, jaký vygenerujeme  $\text{\LaTeX}$ .



V novějších web2c distribucích T<sub>E</sub>Xu lze konfigurovat způsob prohledávání texmf stromu podle názvu použitého programu nebo podle názvu použitého (resp. generovaného) formátu. Nejčastěji jsou tyto rozdílnosti konfigurovány v souboru `web2c/texmf.cnf`. Tam je (mimo jiné) řečeno:

```
% cstex, from Petr Olsak
TEXINPUTS.cslatex= .;$TEXMF/tex/{cslatex,csplain,latex,generic,}//
TEXINPUTS.csplain= .;$TEXMF/tex/{csplain,plain,generic,}//
TEXINPUTS.pdfcslatex= \
    .;$TEXMF/{pdftex,tex}/{cslatex,csplain,latex,generic,}//
TEXINPUTS.pdfcsplain= \
    .;$TEXMF/{pdftex,cstex,tex}/{csplain,plain,generic,}//
```

což znamená, že pokud použijeme nebo generujeme formát C<sub>S</sub>L<sub>A</sub>T<sub>E</sub>X, pak se vstupní soubory přednostně čtou z adresáře `tex/cslatex` a `tex/csplain` a teprve potom v ostatních adresářích. Takže `hyphen.cfg` a `fonttext.cfg` si iniT<sub>E</sub>X vezme přednostně z adresáře `tex/cslatex`. Při provozu C<sub>S</sub>L<sub>A</sub>T<sub>E</sub>Xu pak bude stylový soubor `czech.sty` nebo `slovak.sty` při použití formátu C<sub>S</sub>L<sub>A</sub>T<sub>E</sub>X přečten z adresáře `tex/csplain` a nikoli z adresáře `tex/generic/babel`, kde se nalézá soubor stejného jména, ovšem pro balík Babel.

Aby se hledací algoritmy přizpůsobily názvu generovaného formátu, je nutné tento formát při generování C<sub>S</sub>L<sub>A</sub>T<sub>E</sub>Xu explicitně specifikovat na příkazový řádek. Nestáčí tedy napsat:

```
$ tex -ini cslatex.ini
```

ale musíme psát

```
$ tex -ini -fmt cslatex cslatex.ini
```

Při provozu formátu pak stačí psát

```
$ tex -fmt cslatex dokument
```

Ve web2c distribuci jsou k babelizovanému L<sub>A</sub>T<sub>E</sub>Xu v konfiguračním souboru `web2c/texmf.cnf` uvedeny následující řádky:

```
TEXINPUTS.latex = .;$TEXMF/tex/{latex,generic,}//
TEXINPUTS.pdflatex = .;$TEXMF/{pdftex,tex}/{latex,generic,}//
TEXINPUTS = .;$TEXMF/tex/{generic,}//
```

což znamená, že formát přednostně hledá vstupní soubory v adresáři `tex/generic` a tam je podadresář `babel`. Dokonce je babel upřednostněn i v případě, kdy nevedeme na příkazové řádce žádný název formátu.

S uvedenou konfigurací je možné bezproblémové soužití babelizovaného L<sub>A</sub>T<sub>E</sub>Xu s C<sub>S</sub>L<sub>A</sub>T<sub>E</sub>Xem v jediné distribuci. C<sub>S</sub>L<sub>A</sub>T<sub>E</sub>X pak obvykle voláme skriptem `cslatex`, který provede `tex -fmt cslatex`, zatímco babelizovaný L<sub>A</sub>T<sub>E</sub>X voláme skriptem `latex`, což spustí `tex -fmt latex`.

Pokud nelze ve vaší T<sub>E</sub>Xové distribuci konfigurovat prohledávací algoritmy podle názvu formátu, pak je možné soužití babelizovaného L<sub>A</sub>T<sub>E</sub>Xu s C<sub>S</sub>L<sub>A</sub>T<sub>E</sub>Xem za předpokladu, že budete dodržovat následující zásady:



- Babelizovaný L<sup>A</sup>T<sub>E</sub>X vygenerujte v době, kdy není v distribuci přítomen C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>X. Tak máte jistotu, že iniT<sub>E</sub>X nenajde soubor `hyphen.cfg` z C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>Xu, ale z balíku Babel.
- C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>X pak vygenerujte nad adresářem `cslatex`. Aktuální adresář má totiž ve všech distribucích T<sub>E</sub>Xu přednost před ostatními adresáři v `texmf` stromu. Proto si iniT<sub>E</sub>X přečte soubory `fonttext.cfg` a `hyphen.cfg` z balíčku C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>X.
- Po instalaci C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>Xu do `texmf` stromu je nutno ještě vymazat soubory

```
tex/generic/babel/czech.sty
tex/generic/babel/slovak.sty
```

protože soubory s těmito názvy jsou v `texmf` stromu dvojmo. Přitom v balíčku Babel je nikdy nevyužijete, pokud budete psát korektní záhlaví dokumentu pro babelizovaný L<sup>A</sup>T<sub>E</sub>X (o tom pojednám podrobněji v následující sekci). Na druhé straně soubory `czech.sty` a `slovak.sty` z C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>Xu budete potřebovat velmi často, takže tyto soubory nemažte.

## 5.2 Záhlaví L<sup>A</sup>T<sub>E</sub>Xového dokumentu

Pokud používáte C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>X, pak záhlaví dokumentu může mít tvar

```
\documentclass{article}
\usepackage{czech} % nebo \usepackage{slovak}
... další příkazy záhlaví dokumentu
\begin{document}
...
\end{document}
```

Jestliže zapomenete na `\usepackage{czech}` nebo `\usepackage{slovak}`, pak v dokumentu nebudou fungovat akcentovaná písmena, nebude zapnuto české ani slovenské dělení slov a automaticky generované názvy (kapitoly, sekce, obrázky, ...) nebudou přeloženy do národního jazyka. V tomto případě tedy C<sup>S</sup>L<sup>A</sup>T<sub>E</sub>X bude pracovat stejně jako vanilla L<sup>A</sup>T<sub>E</sub>X.

Pokud naopak používáte babelizovaný L<sup>A</sup>T<sub>E</sub>X, pak záhlaví dokumentu může mít tvar

```
\documentclass{article}
\usepackage[czech]{babel} % nebo \usepackage[slovak]{babel}
\usepackage[T1]{fontenc}
\usepackage[kódování-vstupu]{inputenc}
... další příkazy záhlaví dokumentu
\begin{document}
...
\end{document}
```

Všimněte si jiného způsobu volání stylového souboru `czech` nebo `slovak`. V tomto případě se slovo `czech` či `slovak` zapíše pouze jako volba balíčku `babel`.

Prakticky je to realizováno načtením souboru `czech.lfd` nebo `slovak.lfd`. Pro fungování Babelu tedy není vůbec nutná existence souboru `czech.sty` nebo `slovak.sty`. Tyto soubory v balíčku Babel sice existují, ale jejich funkce spočívá pouze v tom, že „vynadají“ uživateli za nesprávné použití příkazu `\usepackage` a dále načítají odpovídající `lfd` soubor.

Volba `[czech]` nebo `[slovak]` v balíčku Babel ještě nemusí zajistit přepnutí na odpovídající dělení slov. K tomu je navíc potřeba, aby byl babelizovaný  $\text{\LaTeX}$  s těmito vzory dělení už vygenerován.

Na rozdíl od  $\text{\LaTeX}$ u není v babelizovaném  $\text{\LaTeX}$ u řešena synchronizace vstupního kódování dokumentu s vnitřním kódováním  $\text{\LaTeX}$ u.  $\text{\LaTeX}$  implicitně pracuje s vnitřním kódováním podle CM fontů (tzv. `OT1`), což pro většinu jazyků není užitečné. Je tedy třeba při použití babelizovaného  $\text{\LaTeX}$ u přepnout do vnitřního kódování podle Corku (tzv. `T1`), protože v tomto kódování jsou načteny vzory dělení. Použijeme tedy balíček `fontenc` s volbou `[T1]`. Pak se automaticky použijí též fonty v tomto kódování. Babelizovaný  $\text{\LaTeX}$  nepodporuje (na rozdíl od  $\text{\LaTeX}$ u) žádné jiné vnitřní kódování vhodné pro češtinu nebo slovenštinu.

Dále je potřeba v babelizovaném  $\text{\LaTeX}$ u na vnitřní kódování `[T1]` navázat vstupní kódování dokumentu. Pokud například máte dokument napsán v kódování ISO 8859-2, pak je potřeba použít balíček `inputenc` s volbou `[latin2]`. Pokud máte dokument v kódování MS Windows CP 1250, napište volbu `[cp1250]`.

### 5.3 Vlastnosti $\text{\LaTeX}$ u

Uvedeme podrobně rozdíly mezi vanilla  $\text{\LaTeX}$ em a  $\text{\LaTeX}$ em.

Oba formáty deklarují vnitřní kódování CM fontů jako implicitní (tzv. `OT1`) a oba načítají ještě deklaraci kódování podle Corku (tzv. `T1`).  $\text{\LaTeX}$  navíc načítá deklaraci vnitřního kódování podle  $\text{\LaTeX}$ fontů (tzv. `IL2`).

$\text{\LaTeX}$  má ve svém souboru `hyphen.cfg` u příkazu `\DeclareLanguage` pro jazyky `czech` a `slovak` možnost použít kromě volby `IL2` ještě volbu `T1`. V novějších distribucích  $\text{\LaTeX}$ u jsou již obě volby implicitně napsány. V takovém případě se při generování formátu načtou vzory dělení češtiny a slovenštiny nejen v kódování `IL2`, ale také v kódování `T1`.  $\text{\LaTeX}$  dále definuje mechanismy přepínání mezi těmito vzory dělení poté, co uživatel přepne vnitřní kódování z `T1` na `IL2` nebo naopak standardními  $\text{\LaTeX}$ ovými prostředky. Kvůli tomu je v  $\text{\LaTeX}$ u předefinováno interní makro  $\text{\LaTeX}$ u `\DeclareFontEncoding`, protože vanilla  $\text{\LaTeX}$  nepředpokládá, že je po přepnutí kódování nutné přepnout i vzory dělení slov.

V  $\text{\LaTeX}$ u je navíc definováno makro `\splithyphens` a `\standardhyphens`. Po použití makra `\splithyphens` se nastaví znak `-` jako aktivní a funguje podobně jako `\discretionary{-}{-}{-}`. Znamená to, že slova jako „je-li“ se rozdělí správně česky: „je-/-li“. Spojovník se opakuje na následujícím řádku. Makro navíc složitě ošetřuje výskyt `--` a `---`, který tiskne „normálně“ jako za sebou následující znaky s kódem 45, které se promění v ligaturu pomlčky nebo dlouhé pomlčky. Makro `\standardhyphens` dává vše do původního stavu, tj. po jeho použití znak `-` není aktivní. (Uživatelé  $\text{\LaTeX}$ u mohou pro tyto potřeby použít makro podle [6] na straně 217.)

Příkaz `cslatex` musí také zajistit konverzi z kódování češtiny/slovenštiny obvyklé v použitém systému do vnitřního kódování ISO 8859-2 alias IL2. Ve web2c distribuci  $\TeX$ u je proto ve skriptu `cslatex` resp. dávce `cslatex.bat` použit přepínač `-default-translate-file`. Pro systémy MS Windows je vedle tohoto přepínače použita hodnota `cp1250cs`, zatímco pro UNIXové systémy se používá hodnota `il2-cs`, což nastavuje konverzi „jedna ku jedné“. V jiných distribucích se musí implementovat konverze na úrovni vstupního procesoru  $\TeX$ u způsobem závislým na použité distribuci. Například v  $\text{em}\TeX$ u je možné použít TCP tabulky.

## 5.4 Vlastnosti stylových souborů `czech.sty` a `slovak.sty`

V této sekci budu z důvodu stručnosti mluvit o češtině a souboru `czech.sty`, ale to samé samozřejmě platí i pro slovenštinu a stylový soubor `slovak.sty`.

Načtení souboru `czech.sty` z  $\text{CS}\text{L}\text{A}\text{T}\text{E}\text{X}$ u pomocí `\usepackage{czech}` způsobí následující změny:

- Nastaví se vnitřní kódování  $\text{L}\text{A}\text{T}\text{E}\text{X}$ u na IL2, takže se implicitně použijí  $\mathcal{C}$ fonty.
- Inicializuje se české dělení slov v kódování IL2.
- Mezerování se nastaví na `\frenchspacing`, tj. rovnoměrné mezerování mezi slovy i za tečkami.
- Nastaví se české názvy pro jména „Kapitola“, „Obsah“ atd. Viz následující tabulka.
- Makro `\today` expanduje na český datum.
- Definují se makra `\clqq` a `\crqq` pro dvojité české uvozovky, a to i jejich nouzová varianta pro případ, kdy není použito kódování IL2. Definují se též makra `\clq` a `\crq` pro jednoduché české uvozovky.
- Definuje se makro `\uv` jako `\def\uv#1{\clqq#1\crqq}`.
- Definují se přepínače `\csprimeson` a `\csprimesoff` (popis viz níže).

Tabulka automaticky generovaných slov  $\text{L}\text{A}\text{T}\text{E}\text{X}$ u, která jsou změněna po načtení `czech.sty` na české názvy:

Původní název	<code>czech.sty</code>	<code>slovak.sty</code>
Preface	Předmluva	Predhovor
References	Reference	Literatúra
Abstract	Abstrakt	Abstrakt
Bibliography	Literatura	Literatúra
Chapter	Kapitola	Kapitola
Appendix	Příloha	Dodatok
Contents	Obsah	Obsah
List of Figures	Seznam obrázků	Zoznam obrázkov
List of Tables	Seznam tabulek	Zoznam tabuliek
Index	Rejstřík	Register
Figure	Obrázek	Obr.
Table	Tabulka	Tabuľka
Part	Část	Časť

encl	Příloha	Príloha
cc	Na vědomí	cc.
To	Komu	Pre
Page	Strana	Str.
see	viz	viď
see also	viz také	viď tiež

Kromě toho styl `czech.sty` akceptuje následující volby:

**T1**

místo implicitního vnitřního kódování  $\text{\LaTeX}$  IL2 se použije kódování T1.

**IL2**

nemusíte psát, je to implicitní volba.

**OT1**

použije se vnitřní kódování podle CM fontů. Pak samozřejmě nefunguje dělení slov češtiny a slovenštiny.

**split**

zapne `\splithyphens`.

**nocaptions**

výstup makra `\today` a automaticky generovaná slova zůstanou v angličtině.

**olduv**

použije se stará definice makra `\uv`, která umožňuje použít v argumentu verbatim konstrukce.

Například `\usepackage[split,olduv]{czech}` zapne navíc zdvojování spojovníku při rozdělení a definuje makro `\uv` tak, že jsou možné verbatim konstrukce uvnitř argumentu.

Nyní vysvětlím vlastnosti přepínačů `\csprimeson` a `\csprimesoff`. Po použití příkazu `\csprimeson` jsou přiděleny znakům „“ a „’“ aktivní kategorie, aby `‘‘takto po anglicku’’` zapsané uvozovky se převedly na „takové“ uvozovky. Rovněž `‘jednoduché’` anglické uvozovky jsou pak vytištěny „jednoduše“. Makro `\csprimesoff` vrací vše do původního stavu, kdy jsou uvedené znaky neaktivní. Po načtení `czech.sty` je implicitně nastaveno `\csprimesoff`.

Uvedený popis chování stylového souboru `czech.sty` se týká jen případu, kdy je tento stylový soubor načten z  $\text{\LaTeX}$ . Pokud je načten z babelizovaného  $\text{\LaTeX}$ , pak se provede `\input czech.lfd`, takže veškeré definice „češtiny“ jsou v režii balíku Babel. Tam například vůbec není definováno makro `\uv`. Pokud je `czech.sty` načten z  $\text{\LaTeX}$ , pak se provede `\chyph`, definuje se `\csprimeson`, `\csprimesof` a makro `\today` expanduje na datum po česku. Je-li načten `czech.sty` z originálního plainu, pak se provede totéž jen s tím rozdílem, že místo `\chyph` se objeví na terminálu varování o nemožnosti přepnout na české vzory dělení.

## 5.5 PostScriptové fonty v $\text{\LaTeX}$

Při práci s PostScriptovými fonty v  $\text{\LaTeX}$  stačí použít standardní nástroje NFSS. Například pro zapnutí do fontů rodiny Times Roman stačí napsat do záhlaví dokumentu `\usepackage{times}`. Protože  $\text{\LaTeX}$  při `\usepackage{czech}`

implicitně pracuje s vnitřním kódováním podle ISO 8859-2, použijí se v tomto případě virtuální fonty z balíčku `cspsfonts.tar.gz`. Z toho důvodu jsou v balíčku `CSLATEX` přítomny potřebné `fd` soubory. Následující tabulka ukazuje parametry příkazu `\usepackage` pro rodiny fontů ze standardní skupiny 35 PostScriptových fontů.

Rodina fontů	parametr
-----	-----
Avantgarde Book	<code>avant</code>
Bookman	<code>bookman</code>
Helvetica	<code>helvet</code>
New Century	<code>newcent</code>
Palatino	<code>palatino</code>
Times Roman	<code>times</code>

## 5.6 Použití fontů kódovaných podle Corku (T1 kódování) v `CSLATEX`

Máte-li vzory dělení v `CSLATEX` načteny i pro kódování T1 (viz příkaz `\DeclareLanguage` v souboru `hyphen.cfg`), pak můžete místo implicitního vnitřního kódování IL2 použít kódování T1. V takovém případě se automaticky použijí místo `CSfontů` fonty kódované podle Corku. Vnitřní kódování T1 inicializujete zápisem:

```
\usepackage[T1]{czech} % nebo \usepackage[T1]{slovak}
```

Protože ale input procesor `TEX` při použití příkazu `cslatex` konvertuje vstupní kódování dokumentu na ISO 8859-2, je v tomto případě nutné navázat na to další konverzí pomocí balíčku `inputenc` takto:

```
\usepackage[latin2]{inputenc}
```

Zde je potřeba vždy psát `[latin2]`, ať je vstupní kódování dokumentu jakékoli, protože vstupní procesor `TEX` nám toto kódování převedl na ISO 8859-2.

Po zapnutí vnitřního kódování na T1 lze použít PostScriptové fonty stejným způsobem jako předtím (například `\usepackage{times}`). Nyní se ale použijí metriky dodávané v mezinárodních distribucích `TEX` a kódované podle Corku.

## 5.7 `pdfTEX` + `CSLATEX` = `pdfCSLATEX`

V případě spojení `CSLATEX` s `pdfTEX` platí vše naprosto stejně, jako bylo řečeno v sekci „`pdfTEX` + `CSplain` = `pdfCSplain`“. Nahraďte v této sekci slovo `CSplain` slovem `CSLATEX` a slovo `pdfCSplain` slovem `pdfCSLATEX` a přečtěte si tuto sekci ještě jednou.

## 5.8 Historie a budoucnost `CSLATEX`

`CSLATEX` vytvořil zhruba v roce 1992 Jiří Zlatuška. Od něj pochází myšlenka načtení vzorů dělení stejného jazyka v různých kódováních a předefinování vnitřního `LATEX`ového makra `\DeclareFontEncoding`. Na své implementaci `TEX`u tehdy

provozoval mimo jiné fonty kódované v KOI-8, takže přepínání vnitřního kódování  $\text{\LaTeX}$ u si vlastně udělal pro svoje potřeby. Jiří Zlatuška je také autorem maker  $\text{\splithyphens}$  a  $\text{\standardhyphens}$ . Veškerá makra napsal dobře dokumentovaná pro použití v systému docstrip. Svou práci zveřejnil pod licencí GPL mimo jiné podle jeho slov proto, že pokud to bude někoho zajímat, tak to může dále udržovat a zvelebovat podle svých vlastních představ. On sám se kvůli své zaneprázněnosti v jiné oblasti tímto problémem později zřejmě nezabýval.

Po schůzce tvůrců  $\text{\CSTeX}$ u v roce 1993 převzal starost o  $\text{\CSL\TeX}$  podle dohody Zdeněk Wagner, který vytvořil definice kódování IL2. Vytvořil také pro  $\text{\CSL\TeX}$  definiční soubory  $\text{fd}$  jednak pro  $\text{\CS}$ fonty a jednak pro PostScriptové fonty z balíčku  $\text{cspfonts.tar.gz}$ . Od něj také pochází implementace  $\text{\CSL\TeX}$ u pro  $\text{\LaTeX}$  2.09. V  $\text{emTeX}$ ové distribuci  $\text{\CSTeX}$ u je stále tato implementace obsažena (pod označením  $\text{latex209}$ ).

Soubor  $\text{czech.sty}$  má asi podstatně delší historii než  $\text{\CSL\TeX}$ . Pochází z dílny Olina Ulricha, který se zřejmě inspiroval podobným stylovým souborem pro německý jazyk. Olin rovněž vytvořil makra  $\text{\csprimeson}$  a  $\text{\csprimesoff}$ . Zdeněk Wagner pak převzal Olinův stylový soubor a upravil jej pro provoz v  $\text{\CSL\TeX}$ u. Slovenskou část včetně vzorů dělení slov vytvořila Janka Chlebíková. Soubor  $\text{slovak.sty}$  je přesnou kopií souboru  $\text{czech.sty}$  s výjimkou slovensky specifických částí.

V duchu licence GPL převzal zhruba v roce 1997 údržbu  $\text{\CSL\TeX}$ u Jaroslav Šnajdr. Udělal několik úprav stylových souborů  $\text{czech.sty}$  a  $\text{slovak.sty}$  včetně přechodu na novou definici uvozovek, uvnitř jejichž argumentu nefungují verbatim konstrukce. Tím kuriózně způsobil, že  $\text{\CSL\TeX}$ em od této doby nejde bez chyb formátovat český překlad úvodu do  $\text{\LaTeX}$ u, který je pod názvem balíčku  $\text{csuvodlat.tar.gz}$  součástí dokumentace  $\text{\CSTeX}$ u. Je to názorná ukázka toho, co může způsobit změna kódu, která nerespektuje zpětnou kompatibilitu. Pan Šnajdr rovněž napsal  $\text{html}$  dokumentaci k  $\text{\CSL\TeX}$ u, která popisuje instalaci  $\text{\CSL\TeX}$ u ze zdrojových souborů použitím docstripu. Použijete-li ale balíček  $\text{cslatex.tar.gz}$ , pak nemusíte docstrip aplikovat, protože vedle zdrojových souborů jsou tam už přítomny i všechny soubory, které vznikají po aplikaci docstripu.

Já osobně jsem o  $\text{\LaTeX}$  a tím pádem  $\text{\CSL\TeX}$  jevil od začátku malý zájem, protože celý projekt je závislý na  $\text{\LaTeX}$ u samotném. Nemám tedy jistotu, jaké změny v něm současný  $\text{\LaTeX}$ -team udělá a jaké budou existovat do budoucna potíže se zpětnou kompatibilitou. To je zásadní odlišnost od Knuthova  $\text{plain}$ u a  $\text{\TeX}$ u samotného. Raději jsem se tedy do  $\text{\LaTeX}$ ových věcí nemíchal.

V roce 1999 jsem nicméně přidal pár řádek maker do souboru  $\text{czhyphen.tex}$  tak, aby byl použitelný v babelizovaném  $\text{\LaTeX}$ u. Do té doby totiž tato větev  $\text{\LaTeX}$ u používala Lhotkovy vzory dělení, zatímco v  $\text{\CSL\TeX}$ u jsme už dávno měli daleko kvalitnější Ševečkovy vzory dělení. Tyto novější vzory dělení jsou totiž napsány za použití  $\text{\TeX}$ ových sekvencí, což je sice nezávislé na kódování češtiny, ale balíček Babel to implicitně nedokáže strávit a očekává vzory dělení v kódování T1. Upravený soubor jsem nazval Babelovsky:  $\text{czhyph.tex}$ , zatímco v  $\text{\CSTeX}$ u zůstává původní soubor  $\text{czhyphen.tex}$ . Sjednocení názvů těchto souborů by stejně nevyřešilo nejednotnost vývoje  $\text{\CSL\TeX}$ u a babelizovaného  $\text{\LaTeX}$ u.

Na výborové schůzi v roce 1999 jsem dostal za úkol prověřit možnost spojení babelizovaného  $\text{\LaTeX}$  s  $\text{\CSLaTeX}$ em. Neustálé dotazy začínajících uživatelů, kteří si pletou tyto dva  $\text{\LaTeX}$ y, nás utvrzují v tom, že by se pro sloučení mělo něco udělat.

Analyzoval jsem proto makra Babelu a udělal návrh na možné zapracování funkcionality  $\text{\CSLaTeX}$ u do Babelu. Domnívám se, že  $\text{\CSLaTeX}$  klidně může přestat existovat, ale babelizovaný  $\text{\LaTeX}$  musí bezpodmínečně převzít všechny vlastnosti  $\text{\CSLaTeX}$ u tak, aby dokumenty dříve zpracovávané  $\text{\CSLaTeX}$ em byly naprosto stejně a bez jediné úpravy zpracované novým babelizovaným  $\text{\LaTeX}$ em. Kvůli tomuto požadavku musí babelizovaný  $\text{\LaTeX}$  umět načítat vzory dělení stejného jazyka ve více kódováních, jako to nyní dělá  $\text{\CSLaTeX}$ . Dospěl jsem k závěru, že čistým řešením tohoto problému je jediné zásah do jádra  $\text{\LaTeX}$ u samotného, aby dokázal při změně kódování fontů přepnout automaticky i vzory dělení. Zlatuška kvůli tomu předefinoval makro jádra  $\text{\LaTeX}$ u `\DeclareFontEncoding`. Tato záplata, či jinak řečeno odmítnutí původního kódu tohoto makra, je na úrovni Babelu podle mého názoru velmi nečisté řešení. Skutečnost, že přepínání vzorů dělení při přepnutí kódování fontů  $\text{\LaTeX}$ ové jádro neřeší, považuji totiž za chybu  $\text{\LaTeX}$ u. V roce 1999 jsem tedy požádal  $\text{\LaTeX}$ -team, aby zapracoval změnu v duchu Zlatuškovu návrhu do  $\text{\LaTeX}$ ového jádra. Můj návrh nebyl  $\text{\LaTeX}$ -teamem akceptován. Za těchto okolností nejsem schopen zapracovat funkcionalitu  $\text{\CSLaTeX}$ u do Babelu, protože to prostě nejde. Uživatelé  $\text{\LaTeX}$ u se budou muset nadále potýkat s tím, že jejich oblíbený formát trpí určitou schizofrenií.

Společně se sloučením  $\text{\CSLaTeX}$ u s Babelem jsem připravoval zásadní revizi stylů `czech.sty` a `slovak.sty` – v podstatě jsem měl v úmyslu jejich totální přepsání. Tyto stylové soubory obsahují množství reliktních z dob minulých, plno zcela nepoužívaných větví ve složitém větvení pomocí `\if` a stávají se totálně nepřehlednými. Protože ale ke sloučení  $\text{\CSLaTeX}$ u s Babelem nakonec nedošlo, upustil jsem zatím od plánu pracovat na těch stylových souborech. Není ale vyloučeno, že k tomu dojde v budoucnosti. V takovém případě počítám s tím, že makra `\splithyphens` a `\standardhyphens` přesunu z formátu do stylového souboru, kam přirozeně patří. Dokumenty, které tato makra používají, a přitom nemají v záhlaví `\usepackage{czech}` ani `\usepackage{slovak}`, pak nebudou fungovat. Předpokládám, že takových dokumentů není mnoho, protože  $\text{\CSLaTeX}$  a stylové soubory jsou většinou používány současně.

Protože pan Šnajdr se přestal  $\text{\CSLaTeX}$ em zabývat, byl jsem nucen v roce 2002 zanést do stylových souborů jednu opravu podle požadavku pana Kubena. Neznamená to ale, že bych se ujal iniciativy nad  $\text{\CSLaTeX}$ em. Jak jsem už vysvětlil, jsem ochoten převzít iniciativu jen tehdy, když bude  $\text{\LaTeX}$ ové jádro umět přepínat mezi různě kódovanými vzory dělení stejného jazyka. Přitom členové  $\text{\LaTeX}$ -teamu jsou toho názoru, že to možná bude zapracováno až do  $\text{\LaTeX}$ u 3.

## 6. Reference

- [1] WWW stránka  $\text{\CSTeX}$ u <http://petr.olsak.net/cstex/>.
- [2] WWW stránka  $\text{\CSplain}$ u <http://petr.olsak.net/csplain.html>.



- [3] WWW stránka makra OPmac <http://petr.olsak.net/opmac.html>.
- [4] Donald Knuth. *The T<sub>E</sub>Xbook*. Addison Wesley Publishing Company. Eleventh printing, revised, May 1991, ISBN 0-201-13447-0
- [5] Petr Olšák. *Typografický systém T<sub>E</sub>X*. Konvoj, Brno 2000, ISBN 80-85615-91-6
- [6] Petr Olšák. *T<sub>E</sub>Xbook naruby*. Konvoj, Brno 2001, ISBN 80-7302-007-6.
- [7] Petr Olšák. *První setkání s T<sub>E</sub>Xem*. Volně šířený dokument ve formátech `tex`, `ps`, `pdf` na [ftp://math.feld.cvut.cz/pub/cstex/doc/prvni.\\*](ftp://math.feld.cvut.cz/pub/cstex/doc/prvni.*), 22 stran.
- [8] Petr Olšák. *Putování písmene ř z klávesy na papír*. Zpravodaj C<sub>S</sub>TUGu, 3/1997, strany 109–118.
- [9] Petr Olšák. *Rozšíření T<sub>E</sub>Xu encT<sub>E</sub>X*. Rozšíření ve formě změnového souboru `tex.web` je volně šířeno na <ftp://math.feld.cvut.cz/pub/olsak/enc tex/>.
- [10] Petr Olšák. *Program a2ac*. Program včetně zdrojových kódů v jazyce C je volně šířen na <ftp://math.feld.cvut.cz/pub/olsak/a2ac/>.
- [11] Petr Olšák. *Test cstrip*. Test je nepovinnou součástí C<sub>S</sub>T<sub>E</sub>Xu. Je volně šířen na <ftp://math.feld.cvut.cz/pub/cstex/base/cstrip.tar.gz>.
- [12] Petr Olšák. *Makro OFS*. T<sub>E</sub>Xové makro OFS pro práci s rozsáhlými kolekcemi fontů je volně šířeno na <ftp://math.feld.cvut.cz/pub/olsak/ofs/>.