Zpravodaj Československého sdružení uživatelů TeXu

Petr Tesařík

S češtinou a slovenštinou do Babylónu

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 17 (2007), No. 1, 2–11

Persistent URL: http://dml.cz/dmlcz/150020

Terms of use:

© Československé sdružení uživatelů TeXu, 2007

Institute of Mathematics of the Czech Academy of Sciences provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use*.

This document has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library* http://dml.cz

projekt podpořily (Dante, NTG, TUG, TUG India). Podobně výbor rozhodl v budoucnu podpořit projekt mplib (Metapost Library), který by měl vyústit do tvorby programu METAPOST nové generace.

Poslední informace se týká konference EurobachoTEX, na jejíž organizaci se CSTUG podílí. Vzhledem ke skutečnosti, že zájem o autobus, který by „posbíral“ účastníky z České republiky, Slovenska, Maďarska, Rakouska a Německa, byl mizivý, muselo se od jeho organizování upustit. Z členů CSTUGu se aktivně zúčastní pánové K. Horák, P. Sojka a K. Píška. Výbor proto rozhodl, že jim CSTUG částečně přispěje na pokrytí nákladů na účast na konferenci.

Na závěr bych vám všem chtěl popřát v letošním roce, kdy nás čeká mimo jiné volba nového výboru CSTUGu, hodně zdraví a spoustu osobních i TEXovských úspěchů.

Jaromír Kuben
předseda CSTUGu

S češtinou a slovenštinou do Babylónu

Petr Tesařík

Tento článek je jedním z výstupů grantu CSTUGu na vytvoření české a slovenské podpory pro balík Babel, jehož jsem byl řešitelem. Možná se někdo podiví, jak to myslím – vždyť podpora češtiny i slovenštiny je v Babelu již řadu let! A má pravdu: Bylo by vlastně přesnější mluvit o přesunu CSLATEXu pod křídla Babelu.

Něco je špatně

S trochou nadsázky se dá říci, že na počátku veškerého zmatku stál LATEXový amerikanocentrismus či spíše snaha si jej nevšímat. Přestože totiž LATEX nebyl původně určen pro sazbu jiných než anglických textů, objevily se v průběhu let rozličné snahy, jak jej přizpůsobit pro sazbu dokumentů v jiných jazycích, případně i ve více jazycích současně. Pro češtinu a slovenštinu upravili LATEX Jiří Zlatuška a Zdeněk Wagner; výsledný produkt dostal jméno CSLATEX a o něco později jej dále vylepšil Jaroslav Šnajdr. Paralelně se již od počátku devadesátých let vyvíjel balík Babel, který se snažil sjednotit či alespoň zastřešit lokalizaci LATEXu pro nejrůznější jazyky.

Soudě podle názvů některých maker se domnívám, že CSLATEX se snažil být s Babelem spíše kompatibilní. Původní záměr byl možná dokonce takový, aby

styly `czech.sty` a `slovak.sty` byly do Babelu začleněny, jako se to již předtím stalo například se styly německým a francouzským. Nicméně vývoj se ubíral jinou cestou, a tak dnes existují dva soubory `czech.sty` a dva soubory `slovak.sty`, které jsou navzájem nekompatibilní.

Největší závadou je však samotná existence dvou stejně pojmenovaných souborů. Jedním z nejčastějších problémů je tak správné nastavení prohledávaných cest pro babelizovaný LaTeX a pro CSLaTeX. Komu nepřipadá absurdní, že pro správné fungování stylu `czech.sty` je občas nutné nejdříve smazat soubor `czech.sty`, nejspíš výsledky mé práce neocení. Ti ostatní si mohou přečíst, jak nový styl funguje a proč právě tak.

Ach, ty uvozovky...

...aneb proč nelze vytvořit dokonalé makro `\uv`. Jak asi všichni víte, existují v CSLaTeXu dvě verze tohoto makra, jejichž rozdíl bývá vysvětlován následovně:

- „staré“, které umožňuje umístit do uvozovek `\verb`, ale nezachovává implicitní kerny a
- „nové“, které sice neumožňuje použití `\verb`, zato zachovává implicitní kerny na obou stranách.

Skutečnost je ještě o něco barvitější, neboť v nějaké verzi CSLaTeXu byla definice `\crqq` změněna následovně:

```
\DeclareTextCommand{\crqq}{IL2}%
  {{\edef\@SF{\spacefactor\the\spacefactor}\char255 \@SF\relax}}%
```

Nevím, zda je v české či slovenské typografii zvykem, aby se velikost mezery za uvozovkami řídila posledním znakem před uvozovkami, nicméně uvedený kód má za následek, že se mezi tento poslední znak a uvozovky dostane povel hlavního procesoru, takže TeX pak tuto dvojici nehledá v tabulce kerningových párů. Případný implicitní kern u pravé uvozovky se tedy neuplatní (schválně si zkuste jak se Vám s CS-fonty vysází kód `\uv{teď}`). Ironií osudu je, že by bylo celkem snadno možné vytvořit takové makro `\uv`, které by zachovávalo kerny pouze u levé uvozovky a zároveň umožňovalo použít `\verb`...

Rozumně složité makro, které by umožňovalo použít `\verb` a zároveň zachovávalo implicitní kerny na *obou* stranách, vytvořit nelze. Zkusím to jednou pro vždy dokázat. Pro začátek si připomeňme, jak TeX zachází s kerningovými páry (TBN, str. 103):

> Každá dvojice znaků ze společného fontu může mít v tabulce kerningových párů fontu nějakou hodnotu, která se mezi znaky vloží v podobě implicitního kernu automaticky. [...] Pokud se mezi vložením dvojice znaků do horizontálního seznamu vykoná jakýkoli jiný povel hlavního procesoru, implicitní kern se nevloží a ligatura se nevytvoří.

Očividně tedy nelze otevřít skupinu a uvozovky vysázet pomocí `\aftergroup` (trik použitý ve „staré“ definici makra `\uv`), protože mezi vysázením posledního znaku a zpracováním povelů z `\aftergroup` musí hlavní procesor vykonat ukončení skupiny.

Protože `\verb` nemůže být použito v parametru makra (metoda zvolená v „nové“ definici), zbývají jenom následující možnosti:

1. předefinovat kategorii znaku `}`, nebo
2. zpracovat celý obsah uvozovek pomocí `\let`, případně `\futurelet`.

V prvním případě narazíme na to, že je třeba aktivovat také znak `{`. Pokud je totiž uvnitř uvozovek vložena skupina, musí odpovídající znak `}` tuto skupinu ukončit, ovšem bez uvozovek. Problém nastává s tím, že znak `{` se používá také jako levé ohraničení parametru makra. Pokud je tento znak aktivní, použije se jako parametr pouze tento znak. Jakmile je parametr jednou přiřazen, není možné ho změnit, takže tímto směrem se dále nedostaneme.

V druhém případě narazíme na to, že zadání není jednoznačné – není totiž jasné, který znak uvozovky ukončuje. Uvažujme např.:

```
\uv{\LaTeX{} \verb/tu zpracuje {nespárovanou závorku/}
```

Text je vždy ukončen pravou složenou závorkou, ale nemusí to být ani závorka první, ani závorka párová. Šlo by samozřejmě definovat speciální pravidlo pro makro `\verb`, nicméně problém by se nám opakoval v bledě modrém, pokud uživatel např. definuje:

```
\def\mujverb{\verb}
```

Mimoto by se případná implementace nedokázala vyrovnat s libovolným makrem, které mění kategorii některého znaku. Dokonale by to bylo možné vyřešit jedině tak, že makro `\uv` by provádělo plnou expanzi a postupně provádělo povely hlavního procesoru jeden po druhém (např. zmíněné `\catcode`), přičemž slova by se musela zpracovávat jako jeden celek (kvůli implicitním kernům). Takové makro lze ovšem sotva označit jako rozumně složité.

Nabízí se ještě jedno jednoduché řešení, a to „přemapování“ anglických uvozovek (`‘‘` a `’’`) na české uvozovky. Ani to není ideální. Pro angličtinu totiž probíhá nahrazení těchto dvojic znaků na úrovni fontu (ligatura), zatímco pro češtinu by bylo nutné aktivovat znaky `‘` a `’`. To s sebou v každém případě nese některé nežádoucí důsledky. Pro dvojice znaků je v zásadě možné postupovat dvěma způsoby:

1. definovat první znak jako makro bez parametrů a druhý znak načíst pomocí `\let`, resp. `\futurelet`, nebo
2. definovat první znak jako makro s jedním parametrem (touto cestou se vydala implementace `\declare@shorthand` v novějších verzích Babelu).

V prvním případě se pro načtení druhého znaku provádí příkaz hlavního procesoru, takže nebude fungovat např. následující kód:

`\catcode‘-\active`

Vzhledem k tomu, že právě takovýto kód zapisuje Babel do souboru `.aux` při aktivaci znaku pomocí `\initiate@active@char`, je tento postup nepřijatelný. Se znakem ’ je stejný problém, i když se neprojevuje tak často (uvozuje oktalová čísla v syntaktickém pravidle <*number*>)). Mimoto je zde opět problém s implicitními kerny…

V druhém případě se vše odehrává na úrovni expand procesoru. Tím se odstraní výše uvedené nedostatky, ale problém nastane s mezerami – expand procesor totiž všechny mezery mezi jménem makra a parametrem odstraní, takže například takovýto text: „Ty mrchy furt utíkaj’ do lesa“ se vysází jako: „Ty mrchy furt utíkaj’do lesa.“ Lze to samozřejmě obejít tak, že se mezi znak ’ a následnou mezeru vloží např. sekvence `{}`. Mimoto jsou odsuvníky na konci slov v češtině čím dál vzácnější, takže toto řešení by bylo v zásadě přijatelné. Nakonec jsem se však rozhodl ještě trochu jinak (viz dále).

CSLaTeX umožňuje předefinovat znaky uvozovek prvním způsobem, a to povelem `\csprimeson` (mimochodem, tento název je poněkud zavádějící, protože správná expanze znaků ’ na povely `\prime` v matematickém módu je z hlediska českých uvozovek okrajovou záležitostí). Nový `czech.sty` tento povel nabízí pouze v módu zpětné kompatibility s CSLaTeXem.

Po zvážení všech výše uvedených možností jsem dospěl k názoru, že nejlepší bude vyřešit celou záležitost stejně jako v němčině: pro zápis levé uvozovky používat sekvenci `"‘` a pro zápis pravé uvozovky `"’`. Jednak je znak " aktivní i v jiných jazycích (takže se zbytečně neaktivují další znaky), jednak za situace, kdy je beztak potřeba vytvořit novou konvenci, jsem dal přednost řešení, které se používá i jinde. Neaktivní znak " je k dispozici v makru `\dq`.

Pro úplnost bych měl dodat, že existuje ještě jedno řešení, které funguje perfektně: upravit tabulku ligatur, aby posloupnost ‘‘ vedla na znak „ a posloupnost ’’ na znak “. Samozřejmě, že takovým fontem pro změnu není snadné vysázet správně anglický text, nehledě na to, že porušuje kompatibilitu s fonty Computer Modern a neumím si představit, že by třeba tyto ligatury Petr Olšák pozměnil v oficiální verzi CS-fontů.

Spojovníky, rozdělovníky a pomlčky

Dalším velkým tématem je zpracování znaku -. CSLaTeX definuje dvě makra – `\splithyphens` a `\standardhyphens`. Při `\splithyphens` je znak - aktivní a snaží se správně rozdělovat slova, která obsahují spojovník. Navíc se snaží neoddělovat příklonku „-li“. Při vší úctě k tvůrcům tohoto makra se sluší poznamenat, že jeho implementace není moc sofistikovaná, protože oddělení příklonky „-li“ předchází tak, že nedovolí zlom v místě spojovníku, pokud za ním následuje písmeno „l“, tj. nezlomí se ani slova jako „česko-latinský“. To je však detail.

Při návrhu nového řešení jsem vycházel z toho, že jediným důvodem, proč by se příklonka „-li“ neměla oddělovat, je příliš malá délka zbývajícího úseku slova na druhém řádku. Obecně jde tedy o vztah mezi délkou slova za spojovníkem a nastavením registru `\righthyphenmin`. Aktivní znak `-` se proto nyní chová tak, že načítá do pomocného makra znaky, dokud jich není alespoň `\righthyphenmin`, nebo dokud nenarazí na konec slova. To znamená, že i slova jako „je-li“ mohou být rozdělena, pokud nastavíte `\righthyphenmin` na hodnotu 2 (či méně). Naopak, i u delších slov může být dělení potlačeno, pokud hodnotu `\righthyphenmin` zvětšíte.

Implementace této funkce není zcela dokonalá, ale měla by postačovat pro většinu situací. Provádí se postupně úplná expanze kódu, přičemž se uvažují znaky, primitiv `\char` a primitivy definované pomocí `\chardef`. Algoritmus se také vyrovná s otevřením a uzavřením skupiny, ale ostatní primitivy považuje za konec slova, takže nezvládne například následující konstrukci:

```
česko-sl\relax ovenský
```

Takto by to samozřejmě nikdo nenapsal, ale umím si představit, že při použití maker mohou podobné situace nastat.

Nové řešení pro Babel také nemění nic na tom, že aktivní znak `-` bude působit problémy jako součást čísel v horizontálním módu a nadále se místo něj musí v těchto případech používat zástupné makro `\minus`.

Na rozdíl od CSLaTeXu je znak `-` aktivní vždy. Bez dalšího nastavení však expanduje na neaktivní znak `-`, takže jeho funkce je naprosto shodná s neaktivní definicí. Pokud chcete aktivovat automatické dělení slov se spojovníkem, musíte buď použít makro `\splithyphens` (stejně jako v CSLaTeXu), nebo uvést jazykový atribut „split“:

```
\languageattribute{czech}{split}
\languageattribute{slovak}{split}
```

Rozdíl je v tom, že pokud je ve stejném dokumentu použita čeština i slovenština, makro `\splithyphens` ovlivňuje oba jazyky, zatímco `\languageattribute` pouze ten z jazyků, pro který je definován. Kombinace obou způsobů v jednom dokumentu může mít nečekané následky.

Na tomto místě je asi vhodné vysvětlit, jak obě makra z CSLaTeXu vlastně fungují. Makro `\splithyphens` je možné použít kdekoli v textu. Pokud je aktuálním jazykem čeština nebo slovenština, znak se rovnou také aktivuje, jinak si CSLaTeX pouze zapamatuje, že aktivace se má provést při nejbližším přepnutí do českého či slovenského jazyka. Pokud je poté jazyk přepnut zpět např. na angličtinu, znak `-` zůstává v CSLaTeXu aktivní. Makro `\standardhyphens` vždy nastavuje kategorii znaku `-` na 12 (ostatní znaky) a vypíná aktivaci znaku `-` při přepnutí na češtinu či slovenštinu.

Nová implementace se v tomto ohledu chová trochu konzistentněji. Makra `\splithyphens` a `\standardhyphens` lze také použít kdekoli a v češtině a slo-

venštině také rovnou mění chování znaku -. V ostatních jazycích si zapamatuje, jaký mód se má použít při dalším přepnutí na češtinu nebo slovenštinu. V každém případě však deaktivuje rozšířenou funkci znaku - pro jiné jazyky. Za zmínku ještě stojí, že nastavení příznaku se neprovádí globálně (stejně jako v $\mathcal{CS}$LaTeXu), takže nastavení `\splithyphens` například uvnitř prostředí `otherlanguage` platí pouze do uzavření této skupiny.

Kódování fontu

Pro začátek si zkuste odpovědět na následující otázku: „Jakým způsobem lze v LaTeXu s NFSS přepínat mezi CM-fonty, EC-fonty a $\mathcal{CS}$-fonty?“ Pokud jste to zatím nevěděli, možná vás trochu překvapí následující kód:

```
\fontencoding{OT1}\selectfont
Použitý font: \fontname\font % cmr10

\fontencoding{T1}\selectfont
Použitý font: \fontname\font % ecrm1000

\fontencoding{IL2}\selectfont
Použitý font: \fontname\font % csr10
```

Problém je v tom, že jádro LaTeX 2_ε počítá pouze s kódováním T1, ale základními fonty pro $\mathcal{CS}$LaTeX jsou $\mathcal{CS}$-fonty. Bylo by sice možné vytvořit variantu $\mathcal{CS}$-fontů s kódováním T1, ale moc bychom si tím nepomohli, protože by bylo nutné změnit také soubor `t1cmr.fd` (standardní součást LaTeX 2_ε) nebo mít dvě verze tohoto souboru, což by zase přinášelo obdobné problémy jako dvě verze `czech.sty` a `slovak.sty`.

Ponechme tedy $\mathcal{CS}$-fonty v kódování IL2 a pro kódování T1 používejme EC-fonty (či dokonce LM-fonty). K tomu je nutné zajistit následující:

1. donutit Babel, aby při inicializaci formátu zavedl vzory dělení pro češtinu a slovenštinu ve dvou kódováních,
2. při přepnutí kódování nastavit správné vzory dělení a upravit `\uccode`, `\lccode` a `\sfcode` a
3. při přepnutí na český, resp. slovenský jazyk nastavit vzory dělení podle právě zvoleného kódování.

Pokud jde o první bod, v Babelu již určitá podpora jiných kódování existovala. Kódování lze zapsat do `language.dat` jako nepovinný parametr oddělený dvojtečkou od jména jazyka. Takto lze v souboru `language.dat` uvést stejný jazyk několikrát pro různá kódování. Aby to fungovalo, je ještě potřeba pro jedno z těchto kódování vytvořit alias bez dvojtečky, tj. výsledek vypadá nějak takto:

```
czech:IL2       czhyph.tex
```

```
czech:T1        czhyph.tex
=czech
slovak:IL2      skhyph.tex
slovak:T1       skhyph.tex
=slovak
```

K tomu je ovšem nutné odstranit ze souborů `czhyph.tex` a `skhyph.tex` explicitní nastavení kódování T1. Kdyby nám nevadilo, že LaTeX bude při startu vypisovat dvakrát „Loading CZ hyphenation patterns...“ a dvakrát „Loading SK hyphenation patterns...“, mohli bychom místo toho použít přímo soubory `czhyphen.tex` a `skhyphen.tex`.

Bohužel to ještě není úplně všechno, protože Babel podporuje pouze kódování fontu, která jsou již zavedena, tj. nenačítá v případě potřeby nové definice ze souboru. Potřebná úprava Babelu není nijak složitá a Johannesi Braamsovi jsem ji poslal již v dubnu, takže doufám, že se brzy objeví i v oficiální verzi. Na okraj poznamenám, že definice kódování vyžadují některá interní LaTeXová makra, pro která Babel nedefinuje žádnou náhradu pro případ, že je použit s formátem plain (např. `\ProvidesFile`). Z toho mimo jiné plyne, že (alespoň prozatím) nebude možné takto Babelem nahradit csplain.

Pokud jde o druhý bod, kódování IL2 je definováno tak, že při přepnutí na IL2 se potřebné kódy změní. Problém však nastává při přepnutí z IL2 do jiného kódování. LaTeX 2_ε v tomto směru moc možností nenabízí, a tak jsem zvolil variantu předefinovat makro `\@@enc@update` tak, aby se v případě, že aktuální kódování je IL2, nejdříve obnovily hodnoty `\uccode`, `\lccode` a `\sfcode` podle T1. Poté se provede změna kódování v LaTeXu pomocí původní definice makra `\@@enc@update` a nakonec se přepnou vzory dělení, pokud jsou pro daný jazyk a kódování zavedeny. Potom už zbývá jen jedna drobná komplikace, a to, že předefinování uc/lc/sf kódů v souboru `il2enc.def` je prováděno nesprávně, ale to lze napravit snadno.

Pokud jde o třetí bod, bylo nutné pozměnit Babel, aby při přepínání vzorů dělení bral ohled i na aktuální kódování. Tato změna se (doufám) také objeví v nejbližší verzi Babelu.

Na závěr ještě dodám, že stejně jako v CSLaTeXu se CS-fonty použijí i v matematickém módu, pokud jsou ovšem zvoleny jako primární fonty dokumentu. Jestli dojde k předefinování matematických fontů, závisí na tom, zda je na začátku dokumentu vybrané kódování IL2, tj. pokud je celý dokument sázen např. s EC-fonty, CS-fonty se do matematického režimu necpou. Při použití pdflatexu to má samozřejmě také vliv na množství fontů přibalených do výsledného PDF.

Co zbývá z CSLaTeXu?

Předcházející text možná vyzněl tak, jako by se teď o vše, co souvisí s českou sazbou, staral jazykový soubor v Babelu a CSLaTeX už nebyl vůbec potřeba. To není úplně přesné. CSLaTeX totiž mimo jiné obsahuje také soubor definice kódování IL2 pro LaTeX (`il2enc.def`) a soubory pro mapování CS-fontů (`il2cm*.fd`). Tyto soubory jsou nezbytně nutné, pokud chcete sázet své dokumenty CS-fonty (to asi chcete), a musí být někde dále spravovány, protože kódování IL2 nebylo do LaTeXu nikdy oficiálně přijato a pokud dokážu posoudit, ani nikdy nebude.

Dalšími soubory, které je třeba spravovat, jsou `czhyph.tex` a `skhyph.tex`, které vytvořil Petr Olšák, aby se i v Babelu mohly používat Ševečkovy vzory dělení (do té doby se tam používaly starší vzory od p. Lhotky). Tyto soubory kupodivu nejsou ani součástí Babelu, ani součástí CSLaTeXu. V distribuci TeXlive se například instalují ze souboru `hyphen-czechslovak.zip`, který (dá-li se věřit hlavičkám) sestavil sám Sebastian Rahtz.

Kompatibilita

Při veškerých úpravách hrála zpětná kompatibilita s CSLaTeXem velkou roli. O něco menší roli pak hrála kompatibilita se stávající podporou češtiny a slovenštiny v Babelu. Konkrétně to znamená, že pokud je definiční soubor jazyka zaveden pomocí `\usepackage{czech}` nebo `\usepackage{slovak}`, použije se mód zpětné kompatibility s CSLaTeXem, který je téměř stoprocentně kompatibilní s CSLaTeXem. Jsou mi známy tyto odlišnosti:

- Aktivní znak `-` vždy respektuje nastavení `\righthyphenmin` a rozděluje i slova, v nichž po spojovníku následuje znak `1`.
- V módu `\splithyphens` nezůstává znak `-` aktivní v jiných jazycích než češtině nebo slovenštině.
- Aktuální datum vkládané pomocí `\today` v CSLaTeXu umožňuje pro češtinu řádkový zlom mezi číslem dne a jménem měsíce, zatímco v nové verzi je zde zlom zakázán. Pro úplnost dodávám, že pro slovenštinu byl na stejném místě zlom řádku zakázán odjakživa.
- Veškerá funkčnost je aktivní až po zavedení `czech.sty`, resp. `slovak.sty`. CSLaTeX používá upravený `hyphen.cfg` a `fonttext.cfg`, takže některé věci jsou zavedené již ve formátu, např. CS-fonty, přepínání vzorů dělení podle zvoleného kódování nebo aktivní znak `-`.
- Nový český a slovenský styl nebude fungovat s LaTeXem 2.09. Tím mám na mysli skutečný LaTeX2.09, nikoli kompatibilní mód LaTeX 2_ε, tj. pokud váš soubor začíná např. nějak takto:

  ```
  \documentstyle[czech]{article}
  ```

je stále možné jej zpracovat s LaTeX 2$_\varepsilon$ a novou verzí `czech.sty`. Případné zájemce bych však chtěl upozornit, že tuto variantu jsem testoval jen velmi okrajově.
- Makra `\crq` a `\crqq` ve vertikálním módu nezpůsobují chybu, ale přejdou do horizontálního módu.

Zavedení stylu povelem `\usepackage{czech}` nevypisuje standardní varování Babelu. Je to proto, že v různých (i tištěných) dokumentech se doporučuje zareagovat na toto varování smazáním příslušného souboru `czech.sty`, což by teď problémy spíše přidělalo.

Kvůli kompatibilitě s Babelem jsou zachovány dva povely: `\q` a `\w`. První se používá pro vytvoření kličky (např. u „ď“), druhý pro vytvoření kroužku (např. nad „ů“). V současné době jsou to pouze aliasy pro standardní makra `\v` a `\r`, která jsou patřičně nadefinována pro kódování OT1.

Slovenská podpora v Babelu umožňovala také sázet slovenské znaky pomocí digrafů, např. `"a`↦ä, `^o`↦ô, apod. Tato možnost je zachována v režimu Babelu, v kompatibilním režimu tyto digrafy nemají žádný speciální význam (tj. sází se jako "a, ^o, apod.).

Posledním detailem je verze těchto souborů. Poslední babelovský `czech.sty` měl verzi 1.3k, `slovak.sty` verzi 1.3a. Verze souborů z CSLaTeXu měly číslo 2.4. Abych naznačil, že nová verze nahrazuje oba tyto soubory, očísloval jsem ji jako 3.0. Aktuální verze (s jazykovými atributy a opravami několika chyb) má pak číslo 3.1.

Přehled

Následující tabulka přehledně shrnuje funkce nových českých a slovenských stylů. Vlevo je kód pro kompatibilní mód (CSLaTeX), vpravo pro přirozený mód (Babel).

• inicializace s CS-fonty:

```
\usepackage{czech}
```

```
\usepackage[czech]{babel}
\usepackage[IL2]{fontenc}
```

• inicializace s EC fonty (výhledově LM fonty):

```
\usepackage[T1]{czech}
```

```
\usepackage[czech]{babel}
\usepackage[T1]{fontenc}
```

• uvozovky bez `\verb`:

```
\uv{text v uvozovkách}
```

```
\uv{text v uvozovkách}
```

• uvozovky kolem `\verb`:

```
\clqq \verb|verbatim text|\crqq
```

```
"‘\verb|verbatim text|"’
```

• přemapování anglických uvozovek na české:

```
\csprimeson
```

```
(není implementováno)
```

• francouzské uvozovky:

```
\flqq text\frqq
```

```
"<text">
```

• použití staré definice makra \uv:

```
\usepackage[olduv]{czech}
```

```
(není implementováno)
```

• aktivní znak -:

```
\usepackage[split]{czech}
```

```
\usepackage[czech]{babel}
\languageattribute{czech}{split}
```

• (dočasná) aktivace znaku - kdekoli v textu:

```
\splithyphens
```

```
\splithyphens
```

• (dočasná) deaktivace znaku - kdekoli v textu:

```
\standardhyphens
```

```
\standardhyphens
```

• správně rozdělený spojovník bez aktivního znaku -:

```
(není implementováno)
```

```
česko"=slovenský
```

• záporné číslo v horizontálním módu při \splithyphens:

```
\kern\minus1pt
```

```
\kern\minus1pt
```

• české dělení bez přeložených textových řetězců:

```
\usepackage[nocaptions]{czech}
\begin{document}
český text, anglické datum: \today
\end{document}
```

```
\usepackage[czech,english]{babel}
\begin{document}
\begin{otherlanguage*}{czech}
český text, anglické datum: \today
\end{otherlanguage*}
\end{document}
```

• slovenské znaky pomocí digrafů:

```
(není implementováno)
```

```
m"ak^cene zo znakov ASCII
```

• slovenské digrafy s čárkou:

```
(není implementováno)
```

```
\usepackage
  [activeacute,slovak]{babel}
\begin{document}
teraz aj d'l^zne zo znakov ASCII
```