

Zpravodaj Československého sdružení uživatelů TeXu

Petr Olšák

Proč nerad používám LaTeX

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 7 (1997), No. 1-2, 89–99

Persistent URL: <http://dml.cz/dmlcz/149784>

Terms of use:

© Československé sdružení uživatelů TeXu, 1997

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>

V tomto článku bych chtěl uvést několik věcí, které se mi na koncepci L^AT_EXu nelíbí, a vysvětlit, proč jsem zůstal u plainu. Nechci tímto článkem vyvolávat nepokoje v řadách ČSTUGu¹. Vážím si samozřejmě lidí, kteří perfektně rozumějí L^AT_EXu, a tento článek nechce nijak znevažovat jejich práci. Nevylučuji ovšem, že článek vyvolá nějakou diskusi.

L^AT_EX je nadstavba postavená nad formátovacím programem T_EX. Podle mého názoru hlavní cíle L^AT_EXu jsou následující:

- Odstínit poměrně složitou problematiku T_EXu od koncového uživatele.
- Vytvořit vlastní jazyk vstupních textů.
- Umožnit formátovat jednoduché dokumenty podle předzpracovaných stylů.
- Umožnit snadnou výměnu dokumentů a jejich nové přeformátování.

Těchto cílů bylo dosaženo jen částečně. Pokusím se dokázat, že úplné dosažení stanovených cílů není možné z principiálních důvodů a že brzdou v této věci je, překvapivě, samotný T_EX.

Nedostatečnost jazyka T_EXu

T_EX nebyl stvořen se záměrem, aby se nad ním dělaly univerzální nadstavby na úrovni jeho jazyka maker. Rozhodně nelze tyto nadstavby prezentovat veřejnosti jako univerzální systémy. Na to není programovací jazyk T_EXu stavěný. Proto nechápu, proč se o to autoři různých vylepšení L^AT_EXu neustále snaží.

Můžete argumentovat, že vždy musí existovat hranice, pod níž nemusím zkoumat, jak systém funguje, mám-li si zachovat duševní zdraví. Například při jakékoli činnosti s počítačem potřebujeme určitou míru abstrakce. Programujeme-li třeba databázovou aplikaci, zajímají nás elementární nástroje vývojového prostředí a nezabýváme se tím, jak putují elektrony po polovodičích centrálního procesoru.

Pro tento způsob odstínění problému byla na každém stupni abstrakce vytvořena v počítačových vědách dobrá koncepce, která odstínění umožní. Musíme si ale přiznat, že T_EX takovou dobrou koncepci nenabízí. Nelze pomoci

¹Článek je publikován i přesto, že je dle recenzentů cíleně zkreslující, a tedy pro valnou část čtenářů i zavádějící. Též jazykové formulace a hrubé chyby nesplňují běžná kritéria pro publikování, nicméně redakce vyhověla autorovi a do textu nedělala již korekce. Pozn. red.

jeho makrojazyka vytvořit novou úroveň práce, při které není potřeba vracet se k vestavěným vlastnostem $\text{T}_{\text{E}}\text{X}$ u. Z toho pramení většina problémů $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u (jak uvidíme jednotlivě dále). Základní chybou $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u tedy je, že si stanovil cíle, pro které nebyly vytvořeny podmínky v $\text{T}_{\text{E}}\text{X}$ u samotném.

Uvedeme příklad. V běžných programovacích jazycích může programátor elementárními prostředky ošetřit všechny neočekávané situace programu. Všichni víme, že dobrý program by měl mít ošetřeny všechny alternativy vstupu, a považujeme za ostudu, když program havaruje se systémovou chybou. Makrojazyk $\text{T}_{\text{E}}\text{X}$ u ale nemá potřebné nástroje na ošetření všech závad na úrovni „ $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ error“ (srovnej: chyba ošetřená programem). Chyba často vyjde najevo až po činnosti vnitřních algoritmů $\text{T}_{\text{E}}\text{X}$ u a projeví se jako „ $\text{T}_{\text{E}}\text{X}$ error“ (srovnej: systémová chyba programu).

Pokud by se autor makra snažil vše ošetřit na úrovni svého makra, byl by vzniklý kód z 90 % o tom, jak ošetřit uživatelské výstřelky, a ve zbylých 10 % o vlastní sazbě. To je myslím zbytečná investice jednak lidského potenciálu (mám na mysli lidi, kteří taková makra dělají, a jiné, kteří jejich výtvořky musejí číst) a jednak strojového potenciálu (mám na mysli strojový čas a paměť počítačů).

Snaha $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u o odstínění složitosti $\text{T}_{\text{E}}\text{X}$ u

Jedním z cílů $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u bylo umožnit nepoučenému uživateli používat $\text{T}_{\text{E}}\text{X}$, aniž by musel pronikat do složitostí tohoto programu. Nakonec ale sami $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ oví odborníci přiznávají, že bez $\text{T}_{\text{E}}\text{X}$ booku to prostě nejde. Takže snaha o odstínění problematiky $\text{T}_{\text{E}}\text{X}$ u je vlastně zbytečná.

Představte si, že si nějaký uživatel přečte $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ovou příručku a nabude dojmu, že mu bude stačit rozumět problematice sazby na úrovni této příručky. Pak se jednou překlepne třeba při sestavování tabulky a na terminálu na něj $\text{T}_{\text{E}}\text{X}$ křičí: **Extra alignment tab has been changed to \cr**. Uživatel začne znovu listovat ve své příručce a zjistí, že tam o žádném `\cr` není jediná zmínka. Má pak tři možnosti: (1) Zmáčkne Enter a podobně se zachová i u dalších chyb. Pomyslí si, že ten $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ je něco tajemného a mystického. (2) Propadne zoufalství a jde od toho. Dojde k závěru, že je lepší zůstat u Wordu. Vždyť stačí vzít tabulku v Excelu a jednoduše ji přemístit do Wordu a jaképak smolení se s nějakým podezřelým `\cr`. (3) Pořídí si $\text{T}_{\text{E}}\text{X}$ book a po intenzivním studiu nakonec řekne: „aha“. V tuto chvíli ale už nepotřebuje, aby mu $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ zakrýval složitost $\text{T}_{\text{E}}\text{X}$ u.

Nebudu podrobně popisovat zoufalý výraz v očích každého začínajícího uživatele $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u, když po použití `\vspace` tak, že nad značkou ani pod ní není prázdný řádek, dostane nečekaný výsledek. Ve své příručce nemá žádnou zmínku o tom, co to je `\vadjust`, takže mu připadá chování `\vspace` jako tajuplné. Podobně zoufale se uživatelé dívají na obrazovku svého počítače, pokud jim uplavou plovoucí objekty (tabulky a obrázky) úplně jinam, než předpokládali.

Přitom oni chtěli jen použít značku `\caption` a to je přinutilo definovat objekt v prostředí, v němž může uplavat na jinou stránku.

Utajení skutečností

Utajení některých důležitých vlastností $\text{T}_{\text{E}}\text{X}$ u v $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ových příručkách v největší míře souvisí s předchozím bodem o snaze odstínit složitost. Ovšem některé chybějící informace jsou těžko odůvodnitelné. Za všechny uvedu jen dva příklady.

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ový uživatel se nikde nedočte o makru `\choose`, které je elementární zkratkou použití primitivu `\atopwithdelims` a vytváří binomické koeficienty. Můžeme sice v $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u napsat

```
\left(\begin{array}{c} n \\ k \end{array}\right)
```

ale lidé aspoň s nepatrnou mírou vkusu tuší, že to je podivná snůška $\text{T}_{\text{E}}\text{X}$ ových primitivů (`\left`, `\right`) kombinovaná s $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ovým monstrem `array`.

Druhým příkladem je chybějící zmínka o primitivu `\eqno` a `\leqno` pro značkování rovnic. Skutečně, v celém $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ importovi to nenajdete a v jiných příručkách taky ne. Autoři těchto příruček zřejmě žili v domněnání, že jejich čtenáři si za všech okolností vystačí s automatickým značkováním rovnic, které jim poskytuje $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. Ovšem chyba lávky. Zúčastnil jsem se jednoho $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ového kurzu, kde řešili problém, jak dostat na okraj rovnice konkrétní značku, nepřidělenou žádným automatem. Například značku (P). Řešení tohoto úkolu bylo impozantní. Nejprve se vysázela rovnice, pak se pomocí záporného `\vspace` vycouvalo zpět na úroveň rovnice, dále se použilo `\begin{flushright}`, napsala se značka a konečně se to uzavřelo pomocí `\end{flushright}`. To asi nepotřebuje dalšího komentáře.

Nerozlišování mezi $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ em a $\text{T}_{\text{E}}\text{X}$ em

Skoro v žádné $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ové příručce se nedozvíme přesnou souvislost mezi $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ em a $\text{T}_{\text{E}}\text{X}$ em. Občas v některé najdeme mlhavé přirovnání rozebírající vztah závodního vozu s pohodlným rodinným korábem, ale vyložení technických souvislostí považují asi autoři příruček za složité.

Z uvedených důvodů se v $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ových příručkách nerozlišuje mezi vlastnostmi $\text{T}_{\text{E}}\text{X}$ u a $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u. Pro jednoduchost se všude mluví o $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u. Například $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ formátuje odstavce, $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ rozdělí text na více stránek. Víme, že to není pravda. Když pak někteří $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ oví uživatelé přijdou na to, že kromě $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u existuje ještě nějaký podivný $\text{T}_{\text{E}}\text{X}$, jsou obvykle zmateni. Své zmatení dávají často na jevo novotvarem $(\text{L}^{\text{A}})\text{T}_{\text{E}}\text{X}$.

Vlastní jazyk vstupních textů

L^AT_EX se pokusil definovat vlastní jazyk pro značkování vstupních textů. Snaha byla, aby značky podléhaly pokud možno jednotným principům a byly pro uživatele pochopitelné. Jazyk má také pomoci uživateli vyhnout se různým syntaktickým úskalím jazyka T_EXu.

Srovnáme způsoby zápisu, které zhruba(!) znamenají totéž:

T _E X	L ^A T _E X
<code>\hspace=30pc</code>	<code>\setlength{\textwidth}{30pc}</code>
<code>\vskip1cm</code>	<code>\vspace{1cm}</code>
<code>{a\over b}</code>	<code>\frac{a}{b}</code>
<code>\hbox{abc}</code>	<code>\mbox{abc}</code>
<code>\vbox{...}</code>	<code>\begin{minipage} ... \end{minipage}</code> nebo <code>\parbox{...}</code>

Bohužel (nebo bohudík?), v L^AT_EXu je možno psát i některé přímé T_EXové konstrukce. To je další nedůslednost v odstínění problematiky T_EXu před L^AT_EXovým uživatelem. Jakmile to L^AT_EXový uživatel zjistí, začne většinou míchat oba dva způsoby zápisu: T_EXový i L^AT_EXový. To dokazuje, že snaha o vytvoření lepšího, přehlednějšího, a přitom stejně mocného jazyka, se zcela nepovedla.

T_EX nám například hlásí `Overfull \hbox` a L^AT_EX jako makro nemá prostředky toto hlášení změnit na `Overfull \mbox`. Záznam o sazbě je tedy L^AT_EXovému uživateli předkládán pomocí T_EXového a nikoli L^AT_EXového slovníku. Považuji to za matení nebohých uživatelů.

Strukturované značkování

Pod tímto pojmem rozumím značky, které člení dokument do struktury (například kapitola, citát, poznámka) a nezabývají se typografií dokumentu (zda použít `\medskip` nebo `\bigskip` a jaký zavést do sazby font).

L^AT_EXoví zastánci zdůrazňují, že koncepce vstupního jazyka L^AT_EXu vede uživatele ke strukturovanému značkování. To není pravda.

Uživatele, který teprve začíná pracovat s T_EXem nebo s jakýmkoli jiným sázečím systémem, velmi těžko přesvědčíme o výhodách strukturovaného značkování. Chce to dlouhodobou praxi, než člověk na výhody tohoto způsobu značkování přijde. Většina lidí se rozhodne pořídit první dokument proto, aby jednotlivé jeho části vypadaly přesně tak a tak. Nasadí si tak na oči klapky, které se těžko sundávají.

Mé zkušenosti s dokumenty různých autorů ukazují, že v používání strukturovaného značkování není rozdíl mezi uživatelem plainu a L^AT_EXu. Spíš bych řekl,

že dlouhodobá praxe vede ke strukturovanému značkování bez závislosti na tom, jakými prostředky je text pořízen.

Domnívám se také, že člověk snáze dospěje ke koncepci strukturovaného značkování, pokud má vzhled dokumentu absolutně pod kontrolou svých vlastních maker. Používat značky vymezující strukturu dokumentu a nevědět, co se *přesně* stane v době, kdy mě bude zajímat vzhled dokumentu, může být pro uživatele, který je zodpovědný i za typografii dokumentu, frustrující².

Idea strukturovaného značkování se nejlépe projeví při dělbě práce: písárka a sazeč. Pokud sazeč chce postavit sazbu dokumentu na plainu, dohodne se s písárkou na některých značkách definujících strukturu dokumentu. Jestliže chce písárka průběžně vidět, jak dokument vypadá, dají se některé značky narychlo nadefinovat (například `\let\značka=\relax`). Při vlastním zpracování dokumentu pak sazeč definuje jednotlivé značky podle požadavků na typografii dokumentu. Vzhled knihy mu vzniká rovnou pod rukama a před očima.

Pokud sazeč při návrhu značek vychází z plainu, má písárka většinou podstatně méně práce. Sazeč totiž může s výhodou využít mocný nástroj v podobě separátorů parametrů v makrech `TeXu`, který není v `LATeXu` pro potřeby uživatele využit. Ve smluvených značkách je pak daleko nižší výskyt kučeravých závorek a dalších drobletek, které nejsou zcela potřeba. Například pro název kapitoly se ti dva mohou dohodnout na:

```
\kap Název kapitoly  
⟨prázdný řádek⟩
```

Je to jednak přehlednější a jednak to vyžaduje od písárky méně námahy, než kdyby postupovala podle `LATeXu`.

Ideální dělba práce

V předchozí úvaze jsme naznačili ideální dělbu práce. Autor nebo písárka pořídí text se značkami podle pokynů sazeče. Sazeč, který se naučí `TeX`, pak sám doplní makra podle pokynů typografa. Domnívám se, že to byl původní záměr autora programu `TeX`.

`LATeX` se snaží uvedenou dělbu práce obejít. Autor může nabýt dojmu, že mu stačí napsat text podle jednoduché příručky k `LATeXu` a sazečem se stává samotný `LATeX`. Protože schází lidský faktor mezi autorem a konečným vzhledem dokumentu, trpí tím často autor i dokument.

Z jedné strany dává `LATeX` autorovi najevo, že typografii zvládne automat sám a dobře (což vůbec není pravda), a z druhé strany nakonec nutí autora získat znalosti, které by při lidské dělbě práce nebyly pro něj potřebné. Pokud má autor

²Všeobecně rozšířený názor je právě opačný. Autor by se měl při psaní plně abstrahovat od vizuální podoby a koncentrovat se na logické značkování. Pozn. red.

realizovat přesně požadavky typografa, musí proniknout nejen do problematiky $\text{\TeX}$ u, ale též do problematiky složitých maker $\text{\LaTeX}$ u, ve kterém původně svůj dokument napsal. Tím se z autora stává jakýsi polyhistor. Autor na počátku pořizování svého dokumentu určitě netušil, že na něj budou nakonec kladeny tak vysoké požadavky.

Předzpracované styly

Mezi zastánci $\text{\LaTeX}$ u se může rozšířit klamná vize, že všechny požadavky na sazbu, které kdy budou potřebovat, zvládnou výběrem vhodného stylu. Vůbec tedy není potřeba rozumět $\text{\TeX}$ u, stačí zjistit, jaký styl použít, napsat jeho název do hlavičky a máme vystaráno.

Chtěl bych zdůraznit, že základní styly v $\text{\LaTeX}$ u jsou po typografické stránce poměrně dost nevhodné. Především jde o volbu velikostí nadpisů a jejich umístění v sazbě. Rovněž existence pružných vertikálních výplňků je ve většině případů naprosto nepřipustná. Zbavit se jich lze změnou poměrně malého množství parametrů, ale ve svém důsledku to znamená přepsat si aspoň `\output` rutinu podle vlastní koncepce. Jestliže to ale uděláme a provedeme to dobře, jsme na nejlepší cestě vytvořit si vlastní makro, které nazveme například `M\jTeX` nebo jinak. $\text{\LaTeX}$ jako takový už nepotřebujeme.

Svémi základními styly dělá navíc $\text{\LaTeX}$ $\text{\TeX}$ u negativní reklamu. Dovedu si představit, že kupříkladu nějaký nakladatel projeví nezávazný zájem o $\text{\TeX}$ a udělá povrchní průzkum věci. Přitom jako první narazí na $\text{\LaTeX}$ a prefabrikované dokumenty jednoho typu. Může se stát, že průzkum na této úrovni skončí a nakladatel o $\text{\TeX}$ přestane mít další zájem. Je pravda, že může ztratit zájem i o plain, dívá-li se na problém takto: plain bez dodatečných maker neumí vytvořit obsah ani rejstřík, zatímco Ventura to udělá kliknutím myši.

Dalším problémem $\text{\LaTeX}$ u je snášenlivost stylů různých výrobců mezi sebou. Toto není možné elementárními prostředky $\text{\TeX}$ u nikdy uspokojivě vyřešit. Vždy budou existovat detaily, ve kterých se mohou některé styly vzájemně pohádat. Přitom výsledný efekt a případné chybové hlášení je velmi často dosti daleko od podstaty věci a náprava se těžko hledá. Odstranění takových chyb pak vyžaduje nejen perfektní znalost $\text{\TeX}$ booku, ale též důkladnou znalost kódů jednotlivých stylů a kódu samotného $\text{\LaTeX}$ u. A především notnou dávku trpělivosti. Dospějete postupně k závěru, že na úrovni plainu byste se stejnými znalostmi a s podstatně menším úsilím vytvořili makro, které veškeré typografické požadavky řeší také. Důležitá je efektivita práce.

Vždy budou existovat požadavky, které nepůjdou vyřešit pouze zavedením dalšího stylu do $\text{\LaTeX}$ u. Pak se uživatel musí rozhodnout, zda modifikovat styl, nebo si problém vyřešit sám.

Složitost maker $\text{\LaTeX}$ u

Většina stylů musí být aspoň částečně schopna pracovat univerzálně v různých podmínkách. Mají proto ošetřeno plno věcí, které by autor makra „na míru“ nikdy nedělal. Kód takových stylů se pak stává zbytečně složitým a nepřehledným. Příkladem takového nepřehledného a zbytečně složitého kódu (protože chce být univerzální) je samotné jádro $\text{\LaTeX}$ u. Navíc, jak plyne z nedostatečnosti makrojazyka $\text{\TeX}$ u, je požadavek na univerzálnost $\text{\LaTeX}$ u principiálně těžko dosažitelný.

Ve většině případů složitost $\text{\LaTeX}$ u ve svém dokumentu uživatelé v plné šíři nevyužijí. Určitě plno maker $\text{\LaTeX}$ u zůstává při sazbě dokumentu v nečinnosti. Ta makra, která něco činí, by mohla stejnou práci vykonávat efektivněji a jednodušeji. To snese přirovnání o dělové kouli proti mouše. Přitom si ta moucha pohodlně uletí, protože než se tam ta dělová koule přivalí, to chvíli trvá.

Zkuste se namátkou zeptat nějakého $\text{\LaTeX}$ isty, kterými cestami se ubírá napsané „ř“ ve vstupním kódu dokumentu, než se dopravuje k výstupnímu písmenu „ř“ v $\text{\dv}$ i. Bazírujte na naprosto přesném popisu expanze všech maker, která jsou při této činnosti použita. Většina lidí vám na tuto otázku nebude schopna dát odpověď. Ona totiž ta odpověď vůbec není jednoduchá. Dal jsem si tu práci a jeden půlden jsem v kódu $\text{\LaTeX}$ u tuto cestu hledal. Když jsem zjistil, jakými zákoutími a skulinami tato cestička vede, zhrozil jsem se a začal jsem vážně pochybovat, jak to může vůbec stabilně fungovat. Věc je závislá na velkém množství dalších okolností, takže v žádné chvíli nemáme absolutní jistotu, zda „ř“ povede vždy na „ř“. Zákonitě to musí zlobit při použití primitivu `\uppercase`. Proto $\text{\LaTeX}$ tento primitiv dost nepřehledně předefinovává.

Když použiji věci, o kterých vím do posledního šroubu, jak fungují v $\text{\TeX}$ u samotném, nebudu mít nikdy jistotu, co to udělá v $\text{\TeX}$ u zavírovaném $\text{\LaTeX}$ em. Tato nejistota mě od $\text{\LaTeX}$ u poměrně značně odrazuje.

K veřejnému použití jsem například nabídl makro na čarové kódy EAN. Občas mi nějaký uživatel napíše, že to nefunguje, a přitom to použil v $\text{\LaTeX}$ u. Odpovídám stručně: „Použijte (cs)plain, za $\text{\LaTeX}$ já neručím. Pokud to náhodou funguje v $\text{\LaTeX}$ u, považujte to za zázrak³. Stejně si nebudete jistý, zda to za půl roku v tom $\text{\LaTeX}$ u ještě bude fungovat.“ Umíte si představit, jak mi povstaly vlasy hrůzou na hlavě, když mi jeden dobrák ze zahraničí oznámil, že z mého makra `ean.tex` vytvořil odrůdu `ean.cls`!

³Makro EAN je beze změny používáno v redakci. $\text{\LaTeX}$ 2_ε přináší jedinou obtíž, která již byla ve Zpravodaji zmíněna. Ve snaze o zavedení českých pravidel pro dělení slov obsahujících spojovník je znak „minus“ aktivní, pokud se při načítání `czech.sty` použije parametr „split“. Proto se `ean.tex` musí načítat v preambuli a před vlastním voláním `\ean` je nutno nastavit `\catcode'\-=12`.

Většina stylů psaných v plain $\text{\TeX}$ u je v $\text{\LaTeX}$ u přímo použitelná. Není to žádný zázrak, spíše ukázka možností modularity výstavby dokumentu. Pozn. red.

Přenositelnost dokumentů

Je potřeba odlišit dva typy přenosu dokumentů. V první řadě se budeme ptát po přenositelnosti textu dokumentu včetně struktury, ale bez přesných specifikací formátu sazby. Příjemce pak dokument formátuje podle svých požadavků. Výsledek může vypadat docela jinak, než jak dokument viděl na obrazovce autor. Takový způsob přenosu se používá například při sestavování sborníků. V druhém typu přenosu budeme chtít přenést zcela věrně nejen text dokumentu, ale i formát sazby. Požadavkem je, aby příjemce viděl na obrazovce totéž co autor.

L^AT_EX je poměrně dobře koncipován pro první typ přenosu. Svým jazykem totiž deklaruje jednotný způsob značkování dokumentů, který by se mohl s trochou nadsázky nazvat mezinárodním standardem. Praxe ale ukazuje, že nic není tak růžové, jak se na první pohled zdá. Už jste někdy vytvářeli sborník příspěvků rozličných autorů? Uvedu příklad ze své zkušenosti: 50 % příspěvků přišlo ve Wordu či něčem podobném. Po troše konverzní práce je využitelný většinou jen holý text. 40 % příspěvků přišlo v různých verzích L^AT_EXu. Posledních 10 % přišlo v plainu nebo jako holý ASCII text. Nakonec se ukázalo, že s těmito příspěvky bylo nejméně práce.

L^AT_EX jako standardní jazyk pro značkování struktury dokumentu nám příliš nepomohl. Příspěvky v L^AT_EXu totiž byly „každý pes jiná ves“. To je dáno jednak různými verzemi současného L^AT_EXu a jednak neschopností většiny přispěvatelů oddělit strukturu dokumentu od sazby. V jejich příspěvcích se to hemžilo značkami jako `\vspace{5mm}`, `\clearpage`, `\`, `{\Large Nadpis}` atd. Aby to k něčemu bylo, museli by všichni přispěvatelé přistoupit na společný způsob značkování. Nestačí říci, že je to L^AT_EX. Je potřeba vyjmenovat všechny povolené značky vymezující strukturu a pro jistotu uvést též všechny obraty, které jsou pro konečnou typografii sborníku nevhodné. Ukazuje se, že těch pár obrátů, které zůstávají povoleny, lze zpracovat jednoduchým makrem. Ačkoli tyto obraty vypadají L^AT_EXovsky, není nakonec nutno použít L^AT_EX.

Pokud jde o přenositelnost dokumentů včetně formátu, je na tom L^AT_EX dost špatně. Především různé verze L^AT_EXu mohou vést k různým výsledkům. Rovněž koncepce stylů v tuto chvíli naráží na problémy. Většina uživatelů L^AT_EXu totiž implicitně předpokládá, že příjemce jeho dokumentu má povinnost mít instalovány všechny stylové soubory, které jsou v jeho dokumentu použity. Je pravda, že L^AT_EX má nástroje k vložení všech pomocných zdrojů do dokumentu. To je vhodné pro výměnu dokumentů mezi uživateli, ale ti to většinou nepoužívají. Méně znalí uživatelé se navíc chovají k záhlaví svého dokumentu jako k nějakému tabu, které někde obkreslili a někdo poučenější jim do toho dopsal volání nějakých nových stylů. Výsledkem často je, že zaslaný dokument používá v záhlaví dvacet různých stylů, přičemž se nakonec ukáže, že jsou potřeba jen tři. Při obdržení takového dokumentu je proto nejprve

nutné minimalizovat počet použitých stylů, dále zvážit, zda kvůli jednomu efektu v dokumentu má cenu toulat se po Internetu a shánět styly, které ještě v naší instalaci $\text{\LaTeX}$ u nemáme. Mnohdy je jednodušší chyby způsobené nezavedením stylu prostě přeskocit, vždyť ono to nějak dopadne. Pak přijde velké překvapení: ono to dopadlo většinou uspokojivě. Proč tam ten styl vlastně byl?

Na druhé straně uživatelé plainu dobře vědí, jaká makra jsou v zasílaném dokumentu potřeba. Vycházejí z toho, že příjemce bude mít standardní a v jednotlivých verzích skoro neměnný plain nebo csplain. Všechny definice, které rozšiřují uvedené formáty, pak přibalí naprosto vědomě do svého dokumentu. Dokumenty pak jsou bezproblémově přenositelné. Schopnost přesně rozlišovat mezi jádrem formátu a nadstavbou je především vlastností uživatelů plainu, protože jsou sami nuceni tomu pořádněji rozumět.

Uvedu jeden kuriózní příklad. Návrh stanov $\text{\LaTeX}$ u byl napsán v plainu a bylo jej tedy možné vystavit na síti tak jak je. Bez obav, že bude mít příjemce při zpracování problémy. Na druhé straně volební kandidátka byla napsána v $\text{\LaTeX}$ u a na síti jsme ji v podobě zdrojového textu neviděli. Víte proč? Autor tohoto dokumentu použil pro čtverečky na zakřížkování zvolených jmen speciální font. To ovšem v $\text{\LaTeX}$ u znamená, že je potřeba mít správnou verzi souboru `fd` pro tento font a plno dalších záležitostí. Jinak to nebude fungovat. Autor kandidátky mě proto explicitně požádal, abych z důvodů komplikací s přenositelností dokument ve zdrojové podobě nevystavoval. Já sám jsem půl hodiny pracoval na tom, než jsem byl schopen kandidátku bezproblémově vytisknout. Z Internetu jsem musel přitáhnout použité styly, `fd` toho fontu a font samotný. Kdybych tušil, že se jedná o přihlouplé čtverečky, měl bych pro ně makro postavené na primitivech `\hbox`, `\vbox`, `\vrule` a `\hrule` uděláno během pár vteřin. A dokument by byl přenositelný.

Nestálost verzí $\text{\LaTeX}$ u

To je kapitola sama pro sebe. Divím se, že pohyb verzí a zmatek, který kolem toho vznikl, neodradil uživatele od používání $\text{\LaTeX}$ u. Všichni uživatelé $\text{\LaTeX}$ u 2 ϵ jsou vlastně pokusnými králíky a jsou zúčastněni na velkém projektu, který směřuje snad někdy k $\text{\LaTeX}$ u 3.

Jako poměrně velký podraz vůči uživatelům považuji moment, kdy se pojmem $\text{\LaTeX}$ začalo nazývat něco, co se od původního a hojně používaného makra tohoto jména ($\text{\LaTeX}$ 2.09) poměrně zásadně liší. Jaký terminologický chaos vznikl po tomto dravém nástupu $\text{\LaTeX}$ u 2 ϵ , to jistě každý čtenář dobře ví. Když nyní řeknu slovo $\text{\LaTeX}$, nikdo neví, co tím mám na mysli.

Každý půlrok přichází na svět nová verze $\text{\LaTeX}$ u 2 ϵ , která je s předchozí slučitelná jen na uživatelské úrovni. Pokud ovšem uživatel přesáhne svými znalostmi běžné příručky k $\text{\LaTeX}$ u a tyto znalosti použije a podepře je vlastnostmi

nedokumentovaných funkcí konkrétního jádra $\text{\LaTeXu}$, má za půl roku smůlu.

Mnozí z nás byli svědky neslučitelnosti různých verzí počestění $\text{\LaTeXu}$ s různými verzemi jádra $\text{\LaTeXu}$ 2 ϵ . Také víme, že tyto problémy vycházely najevo na naprosto neočekávaných místech (například použití makra `\ldots`). To samo o sobě dokumentuje obtížnou kontrolu nad složitými makry⁴ $\text{\LaTeXu}$ a nesnadnou implementaci národní nadstavby k $\text{\LaTeXu}$.

Půlroční změny verzí $\text{\LaTeXu}$ musí udržovat uživatele i administrátory systémů v neustálé pozornosti. To je úmorné a vyčerpávající. Přiznávám se, že já jako administrátor systému pro naši katedru tuto rychlost pohybu nestačím sledovat a docela mě unavuje.

Jednorázová investice do vzdělání

Naučit se rozumět vestavěným algoritmům $\text{\TeXu}$ a správnému použití primitivů nebo maker plainu je činnost, kterou stačí udělat jednou pro vždy. Naopak, v případě stylů $\text{\LaTeXu}$ člověk musí být neustále ve střehu a musí se postupně stát pochodujícím slovníkem na styly, aby dokázal ke konkrétnímu účelu vybrat a správně použít ten správný styl té jediné správné verze. Přiznávám, že to přesahuje kapacitu mého mozkového potenciálu, a proto zůstávám při zemi: pouze se znalostmi algoritmů $\text{\TeXu}$, jeho primitivů a maker plainu. Tyto věci nejsou v pohybu, takže se nemusím učit každý rok něco nového.

Závěr

$\text{\TeX}$ je koncipován jako otevřený systém. Kdo má chuť a sílu si osvojit jeho detaily, má v ruce mocný nástroj na vytváření nadstaveb podle svých představ. Není podle mého názoru férové vnucovat uživatelům „jedinou správnou nadstavbu“ a v příručkách tvrdit, že při jejím použití nebudou muset vědět, co to je $\text{\TeX}$. $\text{\LaTeX}$ se tím snaží zastřešit otevřený základ univerzální nadstavbou, čímž neguje výhody základu. Protože je toto zastřešení navíc děravé z důvodů nedostatečnosti samotného základu, utrpí tím nakonec oba. Jednak základ a jednak nadstavba. A to je škoda.

Domnívám se, že kdyby se více úsilí věnovalo výkladům a vysvětlování principů samotného $\text{\TeXu}$, bylo by to jen ku prospěchu věci. Udržovat neustále

⁴Tato makra nejsou složitá. Problémy plynou z toho, že $\zeta\text{\TeX}$ se podporou ζ -fontů odchýlil od mezinárodního standardu. $\text{\LaTeX}$ 2.09 byl psán dosti nepřehledně a nestrukturovaně, makra jsou často implementována na nečekaných místech. $\text{\LaTeX}$ 2 ϵ tyto nečtenosti postupně odstraňuje a opravy, které postupně zavádí $\text{\LaTeX}$ 3 team, se musí promítnout i do souborů podporujících národní podporu. Jinak dochází k nečekanému předefinování „českého“ makra na „mezinárodní standard“, což způsobí nějaký typ katastrofy. Pozn. red.

přehled o vlastnostech kódu L^AT_EXu a všech jeho současných stylů a verzí je podle mého názoru daleko náročnější, než se jednou pro vždy seznámit se samotným T_EXem.

Příprava a tisk jednoduché brožury s případným využitím stylu *fancyheadings*

JOSEF BARÁK

Úvod

Mnoho dokumentů v T_EXu je navrženo ve formátu A4, který je však pro běžné čtení poměrně nepohodlný. Častá velikost učebnic a brožur je proto přibližně A5. Tímto formátem docílíme nejen větší přehlednosti, ale i úspory papíru.

Vzhledem k tomu, že jsem nedávno připravoval brožuru s fyzikální tematikou a musel přitom vyřešit některé problémy, chtěl bych se podělit o své zkušenosti s uživateli T_EXu, kteří dosud takovouto práci nikdy nedělali. Protože používám L^AT_EX 2_ε, zaměřím se ve svém popisu právě na něj.

Při vytváření brožury musíme vyřešit dva hlavní problémy: vložit vhodné příkazy do preambule dokumentu (tj. před `\begin{document}`) a rovněž nakonfigurovat prohlížeč tak, abychom si mohli výsledky své práce prohlédnout (a samozřejmě rovněž vytisknout) v reálné podobě. Moje požadavky na vytvářenou brožuru byly následující:

- tisk čtyř stran A5 na list papíru A4 (oboustranně)
- pořadí stran tak, aby se z potisknutých listů papíru dala přímo složit brožura
- na každé straně musí být podtržené záhlaví, ve kterém bude vždy číslo strany a dále střídavě názvy kapitol a podkapitol
- bude počítáno s korekcí tloušťky papíru (jinak by po složení a ořezu brožury byl levý okraj textu na každé straně opticky posunutý)

Příkazy v preambuli dokumentu

Přestože pro vytváření běžné brožury vystačíme s prostředky standardní instalace G_T_EXu, je dle mých zkušeností lépe využít styl *fancyheadings*¹ od Pieta van

¹Například <ftp.cstug.cz/pub/tex/CTAN/macros/latex209/contrib/fancyheadings>