## Pozvánka k TEXu

Co je TEX? Je to systém k přípravě dokumentů, který pochází ze 70. let, ale ještě dnes lepší dokumenty s ničím jiným nepřipravíte (několik programů jej dohání, často tím, že používají algoritmy vyvinuté autory TEXu). Liší se zásadně od programů typu Word, OpenOffice a podobných v tom, že neformátuje text pomocí zázračných tlačítek, ale příkazů ve zdrojovém souboru (podobná filosofie jako HTML), díky čemuž je pak mimo jiné snadné se v tom zpětně povrtat. V čem je jeho výhoda? Kromě zatím nepřekonané kvality výstupu spočívá jeho hlavní síla ve schopnosti vysázet takřka bezbolestně matematiku. Nabízí také mocné nástroje k formátování, které jsou (na základní úrovni) velice snadno naučitelné, takže je pro mnoho lidí univerzálním psacím strojem. Sám jsem od poloviny 90. let prakticky nenapsal žádný text jinak než v TEXu (s výjimkou webových stránek, vlastně i ty se dají dělat přes TEX, ale není to to pravé). Mimochodem, všimli jste si, že když chcete do Wiki zadat nějakou matematiku, tak se to dělá přes TEX?

V tomto textíku TEX představíme a zkusíme ukázat, proč by se vám mohl líbit a co je potřeba k jeho provozování. Jak je zvykem, začneme příkladem.

Představte si, že potřebuji vytisknout jméno, abych jej vystřihl a nalepil na zvonek. Otevřu si svůj oblíbený textový editor (což je náhodou DOSový QEdit, ale lze si vybrat zcela svobodně, třeba Notepad, emacs, pico, joe, SimpleText, ... ) a naťukám tyto řádky:

```
\bf Petr Habala
\bye
```

Pak to uložím například jako `vizitka.tex`. Mohl bych v nouzi použít i příšerku typu WordPerfect, Word, OpenOffice atd., pak si ale musím dát pozor, abych to uložil jako čistý textový (ASCII) soubor, nechceme tam žádné formátovací vymyšlenosti.

Teď je třeba udělat dvě věci. Nejprve zavoláme TEX, aby nám z toho udělal výstup. Často se to dělá voláním z terminálového řádku, například já bych teď odeslal `tex vizitka`, ale jsou i jiné implementace TEXu, některé z nich třeba klikací.

TEX náš vstup přežvýká a vidí následující. Je tam text „Petr Habala“ a dva příkazy (ty začínají zpětným lomítkem). První z nich zapíná tučné písmo neboli *boldface*. Druhý TEXu říká, že už je konec vstupu. Takže vytvoří stránku, na které je tučně vytištěno to jméno. Stránku zapíše do souboru, ale protože neví, co s tím pak budeme chtít dělat, tak nepoužije nějaký konkrétní formát, ale svůj vlastní zvaný `dvi` jako DeVice Independent čili nezávislý na zařízení. Je pak na nás, co s tím uděláme dále.

Třetí krok zpracování je tedy vybrat vhodný driver, který ze souboru `vizitka.dvi` udělá něco rozumného. Možná by nebylo špatné se na něj podívat. Naťukám tedy `xdvi vizitka` či `windvi vizitka` (podle toho, u kterého počítače zrovna sedím) a odešlu, otevře se okénko, ve kterém vidím výslednou stránku. Jsem spokojen, takže kliknu na obrázek tiskárny a je to hotovo. Se souborem `vizitka.dvi` bych ovšem mohl dělat i jiné věci. Mohl bych vytvořit postscript voláním `dvips -o vizitka.ps vizitka`, pak z něj udělat PDF pomocí `ps2pdf vizitka`, ale protože toto se dělá často, mám v TEXovém balíčku přímo příkaz `pdftex`, který pošlu na zdrojový soubor namísto příkazu `tex` (tedy odešlu příkaz `pdftex vizitka`) a on rovnou vyplivne `vizitka.pdf`.

Máme tedy tři základní etapy: Editace–TEXování–Vizualizace, které se v typickém případě (u složitějších dokumentů) musí několikrát zopakovat. Nejprve opakujeme cyklus E–T, protože jsme při rychlém psaní nadělali překlepy v příkazech, vynechali závorky a podobně a TEX nám při pokusu o TEXování nadává. Když je spokojen, opakujeme delší cyklus E–T–V, kdy pro změnu vrtáme do vzhledu dokumentu a opravujeme překlepy v textu, dokud nejsme spokojeni.

Vypadá to komplikovaně, ale většinou je to výrazně rychlejší, než dělat totéž ve Wordu a podobných obrech. Pokud například potřebuji napsat párřádkový dopis (třeba žádost na studijní), tak jej mám vytištěný dřív, než by Word vůbec naběhl.

Teď máte představu, jak TEX funguje. V následujících částech se nejprve podíváme trochu hlouběji na TEX a pak se pobavíme o nutném programovém vybavení.

**Hrajeme si s TEXem**

Jednou ze silných stránek TEXu je snadnost a přehlednost formátování. Možná jste si třeba všimli, že tu je na začátku odstavců malinké odsazení. Normálně je větší, jeho velikost je uložena v registru. Já si ovšem do tohoto registru mohu sám dát velikost, jakou chci. Globálně se to tradičně dělá na začátku souboru, ve zdrojáku tohoto textu je hned někde nahoře řádek

`\parindent=3 mm` (paragraph indent, odsazení odstavce)

Podobně je možno nastavit všechno, třeba šířku a výšku textu ve stránce, rozestupy mezi řádky, záhlaví, je toho spousta. Mnoho z věcí se dá měnit i v průběhu textu, třeba měnit *fonty*, což jsem zařídil tak, že jsem slovo fonty napsal takto: `{\it fonty}`. Ty závorky jsou tam proto, aby se omezila platnost kurzívy, pomocí nich definujeme skupiny. TEX se skupinami dobře pracuje, s radostí je vnořuje a možnost nastavovat věci lokálně je často velice příjemná.

Zkusme se teď podívat na matiku. I člověk TEXu neznalý by asi uměl odhadnout význam tohoto zdrojového textu: `Koukejte, $\int\sin(x+13)x^2dx$.` Ty dolary TEXu říkají, že nebude text, ale matematika. Teď to tedy naťukám: Koukejte, $\int \sin(x+13)x^2dx$. Teď zkusím naťukat tu matematickou část znovu, ale použiji dvoje dolary, čímž TEXu naznačím, že by měl dotyčný vzorec zvýraznit vysázením na samostatný řádek. Čili naťukám `Koukejte, $$\int\sin(x+13)x^2dx.$$` a dostanu Koukejte

$$\int \sin(x+13)x^2dx.$$

Všimněte si, že integrál je větší. U některých značek náročných na výšku si TEX volí velikost podle toho, jestli matiku umístím do řádku, nebo ji vystavím (tomu on říká, že je displayed, pak volí tzv. „d-size“, zatímco matika v textu je „t-size“). Podobně má úspornější verze pro řádkovou matiku ještě třeba sumační znaménko a zlomek. TEX je samozřejmě možné poprosit o jinou velikost, když napíšu `Koukejte, $\displaystyle\int\sin(x+13)x^2dx$.` tak dostanu Koukejte, $\displaystyle\int \sin(x+13)x^2dx$. Je to v řádku, ale on to vysázel, jako by to bylo displayed.

Tím se dostáváme k dalším ohromné věci u TEXu, a to jsou makra. Pokud něco používám často, mohu si pro to vytvořit zkratku. Pokud bych psal často integrály a chtěl je i v textu velké, dal bych někam na začátek souboru řádek

`\def\mujint{\displaystyle\int}`

a pak bych mohl psát `$\myint\sin(x+13)x^2dx$.` Když tohle TEX přečte, tak si najde význam toho `\mujint` a makro nahradí příslušným řetězcem z definice. Poznámka pro myslitele: Pak je displayed vše, co následuje (je to přepínač). Kdybych chtěl jen velký integrál, definice by byla `\def\mujint{{\displaystyle\int}}`.

Plná síla maker se ovšem ukáže, když začneme používat makra s argumenty. Například názvy kapitolek jsou tu dělány pořád stejně, takto: `\title{Hrajeme si s \TeX em}` a co s tím dělat se TEX dozví na začátku souboru, kde je definice makra

`\def\title#1{\bigskip\centerline{\bf #1}\medskip}`

Ten kriminálek je značka pro proměnnou. Když TEX vidí `\title{Hrajeme si s \TeX em}`, tak pozná, že má použít řetězec z definice a na místo značky `#1` má dát to, co jsem dal já, interpretuje to tedy jako

`\bigskip\centerline{\bf Hrajeme si s \TeX em}\medskip`

Překlad: Velká svislá mezera, pak je text „Hrajeme si s TEXem“ tučně a vycentrovaný a pak je střední svislá mezera.

Tohle je naprosto úžasná věc. Všechny formátovací věci si lze takto předpřipravit na začátku souboru a pak už se jen na ně stručnými příkazy odvolávat. Další výhodou je, že když se pak rozhodnete, že názvy mají vypadat jinak, tak stačí změnit jednu definici na začátku souboru a je to všude správně.

Jedna z nejpopulárnějších definic souvisí s matematikou. Standardní příkaz na zlomek $\frac{1}{2}$ je `{1\over2}`. Mnoha uživatelům to nepřijde dobře čitelné, snad většina TEXařů rychle přejde k definici

`\def\frac#1#2{{#1\over#2}}` (ta má dva argumenty)

Teď tu půlku napíšeme jako `\frac{1}{2}`, u složitějších výrazů je to výrazně lepší. Zkusme si jeden takový, pro zvýraznění si jej dáme jako displayed.

```
$$
13^{\frac{1}{x+1}}+\frac{1}{x+1}+\sum_{i=1}^{\infty}a_i
+x^2\cdot\frac{\cos(\pi x)+2e^x\frac{1}{x^2}}{\ln(x^2+1)+\frac{1}{x+1}}
$$
```

udělá

$$13^{\frac{1}{x+1}}+\frac{1}{x+1}+\sum_{i=1}^{\infty}a_i+x^2\cdot\frac{\cos(\pi x)+2e^x\frac{1}{x^2}}{\ln(x^2+1)+\frac{1}{x+1}}$$

Poznáte to ve zdrojovém textu? Mimochodem, nemuseli jsme psát `x^{2}` či `a_{i}`, protože TEX bere jako argument skupinu, pokud není tak příkaz, a pokud není ani ten tak první znak. To je velice pohodlné, půlku napíšeme jako `\frac12`, ale třináctinu už musíme psát jako `\frac1{13}`, protože `\frac113` dělá $\frac{1}{1}3$. Zkuste si vůbec rozmyslet, že všechny závorky definující jednotlivé skupiny jsou logicky správné a některé lze vynechat, jmenovitě ty kolem {1} a {`\infty`}.

Všimněte si také, jak pěkně TEX pracuje s velikostmi. Můžete porovnat velikost zlomku $\frac{1}{x+1}$ jak vypadá zde (textová velikost), jak vypadá v d-size a jak se velikost změní, když je ten zlomek součástí exponentu či jiného zlomku. Podobně se mění velikost „$x^2$“. Ve zdrojovém textu jsme to přitom nemuseli řešit, zadali jsme výraz logicky (významově) a TEX si všechno rozmyslel sám. Matiku prostě umí.

S makry se dají dělat úžasná kouzla. Připomeňme si makro na psaní nadpisu. Dá se upravit tak, aby se tam automaticky dávalo číslo kapitoly, které se s každým použitím o jedno zvětší. Podobně lze napsat makro pro napsání **Věta 13.**, u kterého se číslo automaticky zvyšuje, uživatel jen volá `\Veta` (nebo jak si to nazval). To je ohromně užitečné, nejen proto, že se o číslování nemusíme starat, ale pokud se autor časem rozhodne, že by určité větě slušelo jiné místo, tak ji přesune a čísla se automaticky zase udělají správně.

Dokonce lze naprogramovat, aby šly k větám dát názvy, třeba `\Veta\label{Rolle}`, a když se na ni chce autor někde jinde odkázat, tak napíše `viz věta \ref{Rolle}` a automaticky se tam objeví správné číslo věty. Podobně lze pojmenovat automaticky číslované kapitoly, takže autor většího textu už čísla vůbec nemusí řešit, pracuje čistě na úrovni obsahu, třeba `viz kapitola \ref{derivace}`. Přeházení kapitol pak čísla nerozhodí, vše včetně odkazů se automaticky přečísluje správně. Lze tedy zcela osvobodit formu od obsahu, což se někdy silně hodí.

Samozřejmě se nedá čekat, že by si uživatel všechno programoval sám. Užitečnější věci již někdo dávno udělal a dal k dispozici v podobě souborů s makry, které si každý může přihrát do svého souboru. Například American Mathematical Society připravila balík maker `amstex.tex`, který rozšiřuje možnosti matematiky (různé rovnání rovnic a podobně, má také v sobě `\frac`), já už ze zvyku každý TEXový soubor začínám řádkem `\input amstex`. Jsou balíčky umožňující automatické číslování a odkazování popisované výše, balíčky jazykové, balíčky pro zahrnutí obrázků (nejlépe se vkládá postscript) i balíčky na automatické dělání indexu či seznamu literatury s odkazováním. Novějším vynálezem jsou balíčky, pomocí kterých lze dosáhnout toho, že TEXový dokument po převodu do pdf obsahuje klikací odkazy.

Typický zdrojový soubor tedy začíná nejprve případným načtením souborů s makry, pak bývá zvykem nastavit obecný formát pro celý dokument a poté následuje houf definic maker. Nakonec je samotný text. Jak už vyplývá z řečeného, konkrétní způsob, jakým je ten text do zdrojového souboru napsán, nehraje ve výsledku roli, formátování je určeno příkazy. Je tu jedna podstatná výjimka, prázdný řádek označuje nový odstavec.

Od maker se dostáváme k rozhodnutí, kterému dřív či později čelí každý TEXař. V zásadě se uživatelé TEXu dělí do dvou skupin.

Jedna skupina jsou lidé, kteří vycházejí ze samotného TEXu a případně jej modifikují balíčky maker. To je i můj případ, obvykle používám `amstex` kvůli psaní matiky, na kreslení obrázků

načítám balíček `tikz`, komplikovanější obrázky (grafy) dělám jinde a načítám je jako postscripty pomocí balíčku `epsf`. Výhodou tohoto přístupu je flexibilita, mám k dispozici plnou sílu TEXu a mohu si vybírat z balíčků v zásadě dle libosti (pokud tedy mezi nimi nedojde ke konfliktu, definicemi je dokonce možné přepisovat původní makra TEXu). Nevýhodou může být to, že o některé věci se člověk musí trochu víc starat. TEXu jako takovému se říká `plain` (ve smyslu bez ozdob).

Druhá skupina jsou LATEXáři. LATEX vznikl tak, že jeden člověk TEX totálně předefinoval z hlediska formátování. Základ je stejný, ale všechny příkazy týkající se vzhledu jsou změněny (většinou delší). Například šířku stránky v TEXu nastavíme pomocí `\hsize=180mm` (horizontal size), v LATEXu byste měli použít `\setlength{textwidth}{180mm}`. Proto také musí LATEXáři občas používat speciální verze balíčků maker, udělat makra fungující pod plainem i LaTEXem není triviální.

Výhodou LATEXu je, že má jakoby některé balíčky zahrnuty v sobě, autoři knih oceňují zejména automatické číslování a odkazování či správu literatury. Od uživatele tyto vymoženosti nevyžadují žádnou práci navíc, jsou již integrovány do systému, proto je LATEX silně populární zejména mezi lidmi, kteří se tomu nechtějí příliš mnoho věnovat. Pár základních nastavení se tam také dělá jednodušeji.

Nevýhoda je, že pokud se vám některá z nastavení autora LATEXu nelíbí, tak může být obtížné až nemožné to změnit, zatímco v TEXu to často jde jedním dvěma příkazy. V zásadě je filosofie LATEXu podobná Windows, běžnému uživateli nabízí pohodlí, ale pokud se vám nelíbí, co vám připravili, tak máte smůlu. Díky tomu je LATEX vysoce oblíben nakladateli, autoři jsou jím svázáni a chtě nechtě pak jejich odborné články i knihy musejí vypadat tak, jak chce nakladatel. I já jsem svou první větší knihu musel kvůli nakladateli převádět z plainu do LATEXu.

V případě menších dokumentů pak má také nevýhodu ve zbytečné komplikovanosti. Vzpomínáte si na ten dvouřádkový příklad, se kterým jsme tu začínali? Pokud byste to chtěli udělat v LATEXu, museli byste napsat

```
\documentclass{article}
\begin{document}
\textbf{Petr Habala}
\end{document}
```

Rozhodnutí mezi plainem a LATEXem se tedy odvíjí od toho, jak rozsáhlé texty hodláte tvořit (u malých se LATEX nevyplatí, u velkých pomůže s odkazy), jak velkou kontrolu nad podobou textu chcete a jak moc se tomu chcete věnovat.

## Programové vybavení

Základem je samozřejmě TEX. Autor jej již od začátku koncipoval jako volně přístupný, takže si dnes může každý stáhnout nějaký distribuční balík zdarma. Takový balík obsahuje samotný program `tex`, fonty, rozličné drivery pro zpracování dvi souborů, speciální úpravy pro národní abecedy (lokalizace) a také velké množství rozšiřujících balíčků maker.

Pokud si vyberete nějaký populární balíček (třeba TeXLive či MikTeX), tak se dá čekat, že se zdárně nainstaluje a pak budete moci z příkazové řádky dělat vše, co potřebujete (já pod Windows pracuji s TEXem v DOSím okénku, pod Linuxem v terminálu).

Kolik vám to zabere? Typická distribuce je jedno DVD (TeXLive se dá stáhnout jako `iso` a má asi Giga), ale obsahuje úplně všechno, tři čtvrtiny nikdy nepoužijete a nemusíte si je ani instalovat. V polovině devadesátých let se TEX vlezl na 11 disket ($11 \times 1.44 = 15.8$ mega), ale protože jsem měl malý HDD, vykostil jsem to a žil jsem s TEXem o velikosti 5MB. Většinu toho, co je na mých stránkách, jsem schopen s touto miniverzí udělat (kromě převodu na pdf a obrázky, to tenkrát ještě nebylo). Nároky tedy nejsou vysoké, pokud si necháte nainstalovat základní verzi o velikosti pár set MB, nejspíše vám ještě nadlouho vystačí.

Existují implementace TEXu, které mají na ono jádro TEXu popsané výše ještě naroubován speciální klikací GUI zjednodušující práci. V typickém případě je to editor, který je již prolinkován se vším, takže něco napíšete, kliknete na tlačíto TEX, výsledek se zobrazí, a pokud si na ten obrázek někde kliknete, tak vás to v editoru automaticky přenese na odpovídající místo ve zdrojovém

souboru, ať to můžete hned upravit. Sám to nepoužívám, ale je to věc osobní preference. Tyto implementace již zdarma nebývají.

V Linuxu bývá TEX často již standardní součástí základní distribuce systému.

I ne-GUI verze většinou nabízejí jistý komfort, například prográmek na zviditelňování dokumentu (klikací `windvi` u Windows či `xdvi` pod Linuxem) bývá nastaven tak, že když na něj kliknete, tak automaticky natáhne nejnovější verzi `dvi` souboru. Díky tomu jej stačí při ladění souboru zavolat jednou a nechat viset, po úpravě zdroje a TEXování se v zobrazovači objeví již opravený dokument.

Čtenáře teď možná napadá, jak je tomu s češtinou. V LATEXu je to relativně snadné, u plainu zase existuje mutace `csplain`, což je původní TEX, do kterého kolega Olšák přidal podporu češtiny (je součástí TeXLive, kdybych měl ve jméně akcenty, tak namísto `tex vizitka` volám `csplain vizitka`). Lidé motající se kolem počítačů jsou nicméně na komplikace s češtinou zvyklí, takže to není nic nového.

Druhou důležitou částí sestavy je editor. To je jednou z výhod TEXu, můžete si zvolit libovolný editor dle libosti (pokud tedy nepoužíváte nějaký super integrovaný komplet). Jsou jich desítky zdarma ke stažení, další za mírný poplatek, každý si určitě najde nějaký, který mu bude nejlépe vyhovovat.

Možná by vás pro inspiraci zajímalo, co se osvědčilo mi.

Malý editor se rychle spustí. Samozřejmostí jsou práce s blokem (kopírování, přesun), různé metody pohybu v dokumentu (posun o slovo, skok nahoru či dolů o odstavec), vyhledávání a nahrazování. Užitečná je možnost skočit na řádek podle čísla, protože TEX se na čísla řádků odkazuje při chybových hlášeních. Další výhodou je možnost otevírat více dokumentů najednou a přesouvat text mezi nimi. Pro delší psaní pomůže možnost nastavit barvy tak, aby byly šetrnější k očím, doporučuje se například žluté písmo na modrém pozadí. Některé editory umí barevně zvýrazňovat příkazy (syntax highlighting), mě to spíš rušilo, ale zejména programátoři jsou na to často zvyklí a některé editory jim to umí nabídnout i pro TEX.

Osobně nedám dopustit na možnost si editor překonfigurovat. Význam libovolných kláves a jejich kombinací si mohu dle libosti nadefinovat, takže jsem si nejčastěji používané funkce napojil na F-klávesy a teď jsou přístupné jedním úderem. Další příjemnou vychytávkou mého oblíbence je možnost si hodit kotvu. Určitě se vám stalo, že píšete, píšete, a pak si vzpomenete, že byste potřebovali něco opravit o několik obrazovek dříve. Já si v takové situaci hodím kotvu (navázal jsem si to na Ctrl-1), vrátím se v souboru, opravím si co potřebuji a pak mě klávesa Alt-1 vrátí tam, kde jsem si hodil kotvičku. Mohu si tak naházet kotvičky tři a už se mi to mnohokrát vyplatilo.

Protože často měním systémy, volil jsem editor DOSový, protože jej teď mohu používat v DOSím okně pod Windows i pod Linuxem, nosím jej s sebou na flashce.

Mimochodem, díky tomu, že stačí editovat v ASCII, nejste závislí jen na svém editoru, protože každý systém nějaký jednoduchý editor má. Ať si tedy sednete k jakémukoliv počítači, vždy si tam dokážete něco v TEXu udělat: Ve Windows je Notepad, pod Linuxem zase populární emacs či pico, opravdoví drsoni si to mohou zkusit rozdat s `vi`. V Applících býval SmartText, určitě něco je i teď. Není problém pracovat ani na dálku při terminálovém připojení (často tak opravuji soubory uložené ve škole přímo z domu).

Tolik rady středně pokročilého TEXaře. Pro první hlubší nahlédnutí si můžete na netu najít *Olšák: První setkání s TEXem na jeden večer.*

Pokud se chcete TEX naučit, můžete si vybrat z mnoha tlustých knih. Pro ty, kteří nechtějí jít LATEXovou cestou, je zajímavou alternativou překvapivě útlá brožurka s názvem Jemný úvod do TEXu, popřípadě její originál Gentle Introduction to TEX. Je k dispozici na netu a po jejím přečtení (otázka pár hodin) budete schopni s TEXem pracovat na základní úrovni.

Pak by vám mohl posloužit můj manuálek, podle kterého se lze TEX také naučit (ale už to není jemné, manuál se s nikým nemazlí), ale hlavně vás zavede dále a poslouží jako dobrá reference. Nebude vám ovšem stačit, pokud se rozhodnete jít ještě dál a stát se TEXperty. Pak se nabízí The TEXbook samotného autora TEXu, genia Donalda Knutha, či v češtině publikace kolegy Olšáka.