

# Zpravodaj Československého sdružení uživatelů TeXu

---

Vít Zýka

Používáme pdfTeX V: aktuální pozice sazby

*Zpravodaj Československého sdružení uživatelů TeXu*, Vol. 17 (2007), No. 2, 67–72

Persistent URL: <http://dml.cz/dmlcz/150032>

## 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>

Jedním z posledních dosud v tomto seriálu nepublikovaných rozšíření, o které pdfT<sub>E</sub>X obohatil T<sub>E</sub>X, je zjištění aktuální pozice sazby. Ta může být později využita pro sazbu jiných objektů na takto zapamatovaném místě. Tento článek přináší popis souvisejících nových primitiv a ukázkou jejich použití. V závěru uděláme tečku za tímto seriálem.

## Motivace

T<sub>E</sub>X při sazbě strany dokumentu postupuje sekvenčně. Klade jeden box (znak, čáru, mezeru) ke druhému a posouvá přitom aktuální bod sazby. Pokud potřebujeme klást více boxů přes sebe (např. barevné pozadí, vodoznak), musíme z aktuálního bodu box „vysunout“ do požadované polohy a zajistit, aby se nám přitom aktuální bod sazby neposunul. Tento způsob relativního umístění objektu vůči jednomu bodu sazby není omezující, ale je poněkud nešikovný.

Na makrojazykem T<sub>E</sub>Xu nepřeležitelnou bariéru narazíme, pokud chceme umístit objekt v závislosti na dvou a více nezávislých bodech sazby. Příkladem je podbarvení či orámování odstavců v hlavním vertikálním seznamu tak, aby se zachovala vertikální pružnost a možnost stránkového zlomu, kreslení diagramů se šípkami mezi buňkami umístěnými v nepravidelné mřížce či šípky od jednoho slova v odstavci ke druhému. K tomu bychom potřebovali si místo sazby zapamatovat k pozdějšímu vykreslení závislého objektu. Dosud se tato potřeba řešila přes PostScriptové operátory. pdfT<sub>E</sub>X nám však nabízí přímočařejší a universálnější řešení – pomocí nových primitiv.

## Nová primitiva

Pro práci s aktuální pozicí sazby slouží tři nová primitiva. Prvním je

```
\pdfsavepos
```

Slouží k uložení značky do zpracovávaného seznamu. Po zformátování strany, během akce `\shipout`, se tato značka zpracuje – uloží se do paměti absolutní pozice tohoto bodu sazby vzhledem k levému dolnímu rohu stránky (médiu). Tato  $(x, y)$  pozice se pak dá přečíst primitivou

```
\pdflastxpos
```

```
\pdflastypos
```

Každý z nich vrací celočíselnou hodnotu reprezentující vzdálenost v jednotce `sp` (přesnost `TEXu`).

Protože `\pdfsavepos` muselo být zpracováno až při povelu `\shipout`, je třeba získané hodnoty uložit do souboru pomocí `\write` a načíst a použít je až při dalším průchodu `TEXu`.

Protože použití není zcela přímočaré, ukážeme si jej na příkladě. Spojíme v něm čarou dvě označená slova v odstavci.

## Příklad: čára mezi slovy v odstavci

Naším cílem bude *nakreslit* čáru z místa A do místa B. Podmínkou bude, aby obě místa ležela na téže straně, i když třeba uvnitř odstavce. Každé místo označíme pomocí značky `\posMark{A}`, kde A je zvolené jméno této značky. Dvě místa tímto způsobem definovaná a pojmenovaná pak využijeme k nakreslení *čáry* mezi nimi, jak ukazuje tento odstavec. Ukažme si část jeho zdrojového kódu:

```
1 Naším cílem bude {\it nakreslit\}\posMark{A} čáru z~místa A ...
2 ...
3 k~nakreslení \posMark{B}{\it čáry\} mezi nimi, jak ukazuje ...
```

Nejprve si připravíme pomocná makra.<sup>1</sup> Pro vykreslení čáry z jednoho bodu do druhého využijeme PDF či PostScriptové operátory. Čára bude šedá a široká 2 pt. Pomocí makra `\ifpdf` ze stejnojmenného balíku zaručíme, aby makro fungovalo jak při výstupu do PDF, tak i do DVI:

```
4 \ifpdf
5   \def\Line#1,#2--#3,#4{% 1,2=start x y; 3,4=stop x y <jednotka bp>
6     \pdfliteral page {0.7 G 2 w #1 #2 m #3 #4 l S }}
7 \else
8   \def\Line#1,#2--#3,#4{%
9     \special{" 0.7 setgray 2 setlinewidth #1 #2 moveto
10      #3 #4 lineto stroke }}
11 \fi
```

Dále budeme potřebovat makro, které odřízne jednotku *pt* z výpisu registru typu `dimen`:

```
12 {\catcode'\p=12 \catcode'\t=12 \gdef\removePT#1pt{#1}}
```

A poslední přípravnou akcí bude převod čísla v jednotkách *sp* do jednotek *bp*, ve kterých se vyjadřují vzdálenosti v PDF a PostScriptu:

---

<sup>1</sup>Pro povel, které přímo nesouvisí s tematikou tohoto článku, využijeme L<sup>A</sup>T<sub>E</sub>Xové balíky či makra `eso-pic`, `ifpdf`, `afterpage` a `\InputIfFileExist`, aby článek zůstal přehledný.

```

13 \def\defBPfromSP#1#2{% 1=identifikator 2=cislo <sp>
14   \bgroup
15   \dimen0=#2sp
16   \dimen0=.013837\dimen0
17   \dimen0=72\dimen0
18   \expandafter\xdef\csname#1\endcsname{\expandafter\removePT\the\dimen0 }%
19   \egroup}

```

Nyní přistupme k hlavnímu tématu – uložení pozice sazby. Napsali jsme již, že pozice je známa až v okamžiku provedení `\shipout`, a že si ji tedy musíme uložit do pomocného souboru. Nazveme jej podle hlavního souboru, ale s koncovkou `.pos`. Následující makra otevřou na začátku dokumentu tento soubor pro zápis a na jeho konci jej zavřou:

```

20 \newwrite\posHandle   \def\posFile{\jobname.pos }
21
22 \def\posOpen{\openout\posHandle=\posFile}
23 \def\posClose{\closeout\posHandle}
24
25 \AtBeginDocument{\posLoad\posOpen}
26 \AtEndDocument{\posClose}

```

Zápis pozice sazby do souboru obstará uživatelské makro `\posMark{jmeno}`. V něm použijeme všechna tři nová primitiva:

```

27 \def\posMark#1{% 1=jmeno
28   \pdfsavepos
29   \write\posHandle{%
30     \string\posDef\string{#1}\string}%
31     \string{\the\pdflastxpos\string}\string{\the\pdflastypos\string}}

```

Po prvním zpracování dokumentu se nám vytvoří soubor s obsahem:

```

32 \posDef{A}{9507532}{10913469}
33 \posDef{B}{23416299}{8599377}

```

Soubor načteme makrem `\posLoad` umístěným na řádku 25 před otevřením `.pos` souboru makrem `\posOpen`:

```

34 \def\posLoad{\InputIfFileExists{\posFile}{-}{-}}

```

Soubor `.pos` čteme jen tehdy, když soubor existuje, což bude až ve druhém a dalším průchodu `TEX`em. Aby načtení neskončilo chybou, musíme nadefinovat

následující makro `\posDef{jmeno}{x-pos}{y-pos}`. Jeho úkolem bude vytvořit dvě makra `\pos-x-sp-jmeno` a `\pos-y-sp-jmeno` obsahující náležité číselné hodnoty v jednotce *sp* a dvě `\pos-x-bp-jmeno`, `\pos-y-bp-jmeno` v jednotce *bp*:

```

35 \def\posDef#1#2#3{% 1=jmeno 2=x-pos 3=y-pos
36   \expandafter\def\csname pos-x-sp-#1\endcsname{#2}%
37   \posDefXbp{#1}%
38   \expandafter\def\csname pos-y-sp-#1\endcsname{#3}%
39   \posDefYbp{#1}}

```

Převod mezi jednotkami obstaralo makro

```

40 \def\posDefXbp#1{\defBPfromSP{pos-x-bp-#1}{\posGetX{#1}}} % 1=jmeno
41 \def\posDefYbp#1{\defBPfromSP{pos-y-bp-#1}{\posGetY{#1}}} % 1=jmeno

```

Volání takto automaticky vytvořených maker si zjednodušíme definicí

```

42 \def\posGetX#1{% 1=jmeno
43   \expandafter\ifx\csname pos-x-sp-#1\endcsname\relax0
44   \else\csname pos-x-sp-#1\endcsname\fi\relax}
45 \def\posGetY#1{% 1=jmeno
46   \expandafter\ifx\csname pos-y-sp-#1\endcsname\relax0
47   \else\csname pos-y-sp-#1\endcsname\fi\relax}
48 \def\posGetXbp#1{% 1=jmeno
49   \expandafter\ifx\csname pos-x-bp-#1\endcsname\relax0
50   \else\csname pos-x-bp-#1\endcsname\fi\relax}
51 \def\posGetYbp#1{% 1=jmeno
52   \expandafter\ifx\csname pos-y-bp-#1\endcsname\relax0
53   \else\csname pos-y-bp-#1\endcsname\fi\relax}

```

Takto získané hodnoty použijeme pro vykreslení čáry v absolutních souřadnicích. Vhodné místo je při akci `\shipout`, když je nastaven základní souřadnicový systém. Využijeme k tomu balíku `eso-pic`:

```

54 \def\AbsLine#1,#2--#3,#4{% 1,2=start x y; 3,4=stop x y <jednotka bp>
55   \AddToShipoutPicture{\AtPageLowerLeft{\Line#1,#2--#3,#4}}}

```

A konečně, zde je finální uživatelské makro používající pojmenované body:

```

56 \def\AbsLineFromTwoMarks#1#2{% 1=jmeno-A 2=jmeno-B
57   \AbsLine\posGetXbp{#1},\posGetYbp{#1}--\posGetXbp{#2},\posGetYbp{#2}}

```

Takže po našem úvodním odstavci jsme museli uvést ještě tyto dvě řádky, aby se nám vykreslila požadovaná čára:

```
58 \AbsLineFromTwoMarks{A}{B}
59 \afterpage{\ClearShipoutPicture}
```

Druhý řádek zařídí, aby se nám tato čára netiskla na každé další straně.

## Závěr

Tento díl seriálu *Používáme pdf<sub>TEX</sub>* je pravděpodobně poslední. Hlavní témata, o která pdf<sub>TEX</sub> rozvíjí klasický <sub>TEX</sub>, byla popsána a samotný vývoj se přesunul k lua<sub>TEX</sub>u [1].

V této souvislosti je dobré připomenout, že pdf<sub>TEX</sub> vznikl jako diplomová a později disertační práce vietnamského studenta Hàn Thê Thànha na Masarykově universitě v Brně pod vedením Jiřího Zlatušky a Petra Sojky. Později se vývoj přenesl na Martina Schrödera, Hartmuta Henkela, Taco Hoekwatera a Hanse Hagen. Hàn Thê Thành zůstal hlavním koordinátorem.

pdf<sub>TEX</sub> nepřinesl jen nový výstupní formát – PDF – k původnímu DVI, ale též mnohá rozšíření. Jejich použití usnadňují makrobalíky. Pro <sub>La</sub>TEX jmenujme několik nejdůležitějších:

- microtype (h-z algoritmus, prostrkání okrajů sazby, ligatury),
- hyperref (hypertext a formuláře),
- graphicx (PDF, JPG, PNG obrázky, ořez, zvětšení, rotace),
- color (barvy),
- pdfpages (slučování dokumentů, vyřazování),
- movie15 (vkládání videí),
- animate (vytváření PDF animací),
- acrotex (formuláře, interaktivní dotazníky, JavaScript).

V Con<sub>TeX</sub>tu je vše integrováno do jeho jádra, žádné přídatné balíky nejsou potřeba. Uživatelé plain<sub>TeX</sub>u musí studovat referenční manuál PDF [2] a využívat přímo nová primitiva popsaná v manuálu pdf<sub>TeX</sub>u [3]. Sepsal jej Hàn Thê Thành spolu se Sebastianem Rahtzem. Grafickou úpravu zpracoval Hans Hagen. Dnes manuál spravuje Pawel Jackowski.

Úspěch pdf<sub>TeX</sub>u spočíval na 100% zpětné kompatibilitě s <sub>TeX</sub>em, používání mainstreamových formátů pro vstup i výstup, kvalitativních přínosech typografii a testování při praktickém používání. Jeho nástupce bude k úspěchu pravděpodobně potřebovat splnit stejná kritéria. Těžko říci, zda lua<sub>TeX</sub>, ke kterému přesunula své úsilí značná část vývojového týmu pdf<sub>TeX</sub>u, bude stejně úspěšný. Potenciál na to má. Držme vývojářům palce, podpořme je finančně či přiložme ruku k dílu. Je to šance, aby <sub>TeX</sub> obstál i v dalším desetiletí.

## Reference

- [1] Taco Hoekwater and Hans Hagen. *LuaTeX reference manual*, 2007. <http://context.aanhet.net/luatex/snapshot/manual/luatexref-t.pdf>.
- [2] Adobe systems Incorporated. *PDF reference manual, v. 1.7*, 6 edition, 2006. <http://www.adobe.com/devnet/acrobat/>.
- [3] Hàn Thê Thành, Sebastian Rahtz, Hans Hagen, and Hartmut Henkel. *The pdfTeX user manual*, 2007. <http://sarovar.org/docman/view.php/106/66/pdf-tex-s.pdf>.

## Summary: Using pdfTeX V: current typesetting position

One of the extension by which pdfTeX improved TeX is a possibility to obtain a current typesetting position. It can be used for placing objects later. This article brings a description of related primitives and examples of their usage.

Vít Zýka [vit.zyka@seznam.cz](mailto:vit.zyka@seznam.cz)

---

---

# Vkládání JavaScriptů pdfTeXem prakticky

ROBERT MAŘÍK

## Úvod

V poslední době se na CTAN objevuje stále více balíčků, využívajících vkládání JavaScriptů do PDF dokumentů (animate, animfig, cooltooltips, dps, eCards, exerquiz, fancytooltips, jeopardy, jj\_game, pdfanim)

JavaScripty umožňují získat dokumenty, které mají větší možnosti než klasické statické dokumenty, jejich funkčnost se však projeví pouze v prohlížeči Adobe Reader nebo Adobe Acrobat (což je poněkud omezující pro uživatele alternativních programů na čtení PDF).

Cílem tohoto článku je ukázat několik jednoduchých příkladů na vkládání JavaScriptů, které by po příslušném rozšíření umožnily dosáhnout podobných možností, jaké mají výše uvedené balíčky. Budeme se přitom vždy snažit udělat příklady názorné a co nejjednodušší. Článek je doprovázen souborem `testjs.tex`, který obsahuje všechny popisované JavaScripty a některé doplňující komentáře a souborem `testjs.pdf`, který vznikl překladem pdfL<sup>A</sup>TeXem. Tyto soubory jsou