

The fancytooltips package*

Robert Marik
marik@mendelu.cz

May 9, 2008

1 Introduction

The package `fancytooltips` is a package for L^AT_EX. The pdf can be created by pdf_latex or by latex + dvips + AdobeDistiller¹ + Adobe Acrobat². It allows to create tooltips in a similar way like `cooltooltips` package, but the tooltip is a page from another PDF file. In this way you can use mathematics, pictures and animations in your tooltips. The resulting PDF file can be used also with free Adobe Reader.

The tooltips are activated by clicking the active area on the screen and deactivated after closing page or by moving mouse outside the link. You can try the links [here](#) (Einstein's formula) and also [here](#) (animation – numbers from 1 to 6). You have to use the free Adobe Reader or nonfree Adobe Acrobat to see the effect (xpdf, evince and others fail to work with JavaScripts). For more examples how the presentation may look like see the `example.pdf` and `example-min.pdf` files in the `examples` subdirectory.

The buttons are created using `eforms.sty` which is a part of AcroTeX bundle.

2 Usage

2.1 The file with tooltips

The file with tooltips is an ordinary pdf file, one tooltip per page, tooltips should be in the top right corner at the page, in a colored box and the rest of the page should be transparent. If you consider to use `movetips` option (see below), then every page should have the dimensions equal to the dimensions of the colored box with tooltip³. We also provide simple cross referencing mechanism to refer to the tooltips. If the pdf file is created by L^AT_EX, you can define keywords to refer to the pages using `\keytip` command. Sim-

`\keytip`

*This document corresponds to `fancytooltips` v1.3a, dated 2007/07/28.

¹not free ps2pdf

²not free Adobe Reader.

³Look at the files `tooltipy.tex` and `tooltipy.pdf` from `examples` subdirectory for a simple example how to meet this condition under pdfL^AT_EX

ply put `\usepackage[createtips]{fancytooltips}` into preamble and write `\keytip{<foo>}` in document. This writes information about keyword `<foo>` and the pagenumber into file `fancytips.tex`. If you use option `mouseover`, the tooltips are activated by mouseover action.

2.2 The file with presentation – pdf \LaTeX users

In the file with presentation, the user is responsible

- input either `color` or `xcolor` package in the preamble
- \LaTeX the file two times (we write some macros into `aux` file).

This is not comfortable for the user, but everybody uses different set of packages and from this reason, this part is left to the user. (And among others, the `color` or `xcolor` package is probably inputted by the package which is used to build the presentation.)

filename option To input the tooltips from file `<foo.pdf>` call the package with `filename` option: `\usepackage[filename=foo]{fancytooltips}`.

movetips option By default, tooltip appears in the top right corner of the page (use View–PageLayout–Single Page in your Adobe Reader, please). If the option `movetips` is used, then tooltip appears close to the mouse pointer. More precisely, tooltip appears with left down corner at the mouse position, if there is enough place. If not, tooltip appears with right down corner at the mouse position. Finally, the tooltip is shifted down to fit the page, if necessary⁴.

\tooltip The user can put the tooltip into her or his presentation using the command `\tooltip{<stuff>}{<keyword-or-pagenumber>}` where `<stuff>` is the printed text in `<tooltipcolor>` color and `<keyword-or-pagenumber>` is either the pagenumber of the

\TooltipExtratext tooltip in the external file or the keyword defined by `\keytip` command. The printed text `<stuff>` is followed by `\TooltipExtratext` command. The default value is small blue soap in a box with zero dimensions, as you have seen in the second paragraph of this documentation. There is a package option `noextratext` which defines `\TooltipExtratext` to be empty.

noextratext option The user can put a series (animation) of tooltips into the presentation by using `\tooltipanim{<stuff>}{<start>}{<end>}` command, where `<start>` and `<end>` are

\tooltipanim keywords defined by `\keytip` command or page numbers. The delay between two frames is `\delayinterval` milliseconds. The default value is 200, you can change it by command `\def\delayinterval{100}`.

\delayinterval The file `example.tex` from `exmaples` subdirectory shows, how to redefine these macros to gain different behavior, see the demo file `example.pdf`.

2.3 Changes for dvips users

pages option dvips users have to specify option `dvips` in `fancytips` package. They have to use also a `pages` option with the number of pages in the PDF file with tooltips. You

⁴This option works in this way if every page of the file with tooltips has dimensions of the box with tooltip. See the `examples` subdirectory.

have to call the package by something like this:

```
\usepackage[dvips,filename=tooltipy,pages=27]{fancytooltips}
```

You have to `latex` (two times) and `dvips` your file first. This produces `filename.ps` and `Tooltipsdljs.fdf` files. Distill the pdf file into `filename.pdf` and open this file by Adobe Acrobat - this imports macros from `Tooltipsdljs.fdf` file. In Acrobat's JavaScript console (**Ctrl+J**) run (**Ctrl+Enter**) the command `ImportTooltips()`; which is defined for the document and it creates invisible buttons on the first page, imports icons (the file with icons specified as `<filename>` parameter when loading `fancytooltips` must be in working directory) and returns 1. Then save the file under another name.

3 Known problems

The package works only with the last `eforms.sty`, version 2006/10/03 v1.0a. You can download this version from www.arotex.net site. The version on CTAN and in MikTeX repositories is old and this package does not work with this old version.

4 Implementation

```
1 <*package>
2 \RequirePackage{everyshi}
3 \RequirePackage{graphicx}
4 \RequirePackage{xkeyval}
5 \RequirePackage{eso-pic}
6
7 \newif\ifcreatetips\createtipsfalse
8 \DeclareOptionX{createtips}{\createtipstrue}
9
10 \newif\ifTooltip@usepdftex\Tooltip@usepdftexttrue
11 \DeclareOptionX{dvips}{\Tooltip@usepdftextfalse}
12
13 \newif\ifextratext\extratexttrue
14 \DeclareOptionX{noextratext}{\extratextfalse}
15
16 \newif\ifmovetips\movetipsfalse
17 \DeclareOptionX{movetips}{\movetipstrue}
18
19 \newif\ifmouseover\mouseoverfalse
20 \DeclareOptionX{mouseover}{\mouseovertrue}
21
22 \DeclareOptionX{filename}{\xdef\TooltipFilename{#1}}
23 \DeclareOptionX{pages}{\xdef\TooltipPages{#1}}
24
25 \ProcessOptionsX
26
27 \ifx\TooltipFilename\undefined
28 \PackageWarning{fancytooltips}{** The filename with tooltips is not given. **}
```

```

29 \fi
30
31 \ifTooltip@usepdftex
32 \RequirePackage[pdftex]{eforms}
33 \def\TooltipExtratext{\hbox to 0 pt{\smash
34   {\raisebox{0.5em}{\includegraphics[width=0.7em]{%
35     {fancytipmark.pdf}}}\hss}}
36 \else
37 \RequirePackage[dvips]{eforms}
38 \def\TooltipExtratext{\hbox to 0 pt{\smash
39   {\raisebox{0.5em}{\includegraphics[width=0.7em]{%
40     {fancytipmark.eps}}}\hss}}
41 \fi%\ifTooltip@usepdftex
42 \ifextratext\else\let\TooltipExtratext\relax\fi
43
44 \ifcreatetips

```

This part (three lines) is processed if the option `createtips` is used. In the opposite case we process the second part, up to the end of the package.

```

45 \newwrite\tipfile
46 \immediate\openout\tipfile fancytips.tex
47 \def\keytip#1{\write\tipfile{\string\tooltipname{#1}{\arabic{page}}}}
48 \else

```

This part is processed if the option `createtips` is not used. We define macros which put the hidden button with the name `ikona.n` in the background of the page `n`, if one of the commands `\tooltip` or `\tooltipanim` has been used on this page. Javascripts defined by `\tooltip` and `\tooltipanim` commands then unhide this button and show the corresponding picture.

```

49
50 \newdimen\buttontipwidth
51 \newdimen\buttontipheight
52 \AtBeginDocument{
53 \buttontipwidth=\paperwidth
54 \buttontipheight=\paperheight
55 }
56
57 \ifTooltip@usepdftex
58 \def\frametip@{%
59   \pdfstartlink user{%
60     /Subtype /Widget
61     /F 6
62     /T (ikona.\thepage)
63     /FT /Btn
64     /Ff 65536
65     /H /N
66     /BS << /W 1 /S /S >>
67     /MK << /TP 1 /IF <</A[1.0 1.0]/SW /B>> >>
68   }%
69   \vbox to \buttontipheight {\vss\hbox to \buttontipwidth{\hss}}\pdfendlink}
70 \else

```

For dvips users we use the macros from eqxerquiz.sty package.

```

71 \def\everyeqIcon#1{\def\every@eqIcon{#1}}
72 \def\every@eqIcon{}
73 \newcommand\eqIconFTT[4] []
74 {%
75   \push@@Button{#1}{#2}{#3}{#4}{\eq@setButtonProps\eq@Button@driver}%
76   {\eqIconDefaults\every@ButtonField\every@eqIcon}%
77 }
78 \def\eqIconDefaults
79 {%
80   \rawPDF{\S{}}\mkIns{/TP 1 /IF<</A[1.0 1.0]/SW/B>>}\R{0}
81   \CA{}\RC{}\AC{}\BC{}\BG{}\H{B}
82   \textColor{0 g}\Ff{\FfReadOnly}
83 }
84 \def\frametip@{\eqIconFTT[\BC{}\BG{}\F{\FHHidden}]}%
85   {\ikona.\thepage}{\paperwidth}{\paperheight}}%
86 \fi%\ifTooltip@usepdftex
87
88 \def\frametip{%
89   \expandafter\ifx \csname TooltipPage\thepage\endcsname\relax
90   \else
91     \setbox0=\hbox{\frametip@}%
92     \hbox{\raise \dp0 \box0}
93   \fi}%
94 \AddToShipoutPicture{\hbox to 0 pt{\frametip\hss}}

```

In the macros `\tooltip` and `\tooltipanim` we print the text into box with zero dimensions and then we build a button which covers this text and has an associated JavaScript action. The important part is the `\PushButton` macro. You can adjust these macros or write similar macros which do what you need. For some examples see the file `example.tex` from the `examples` directory.

```

95 \definecolor{tooltipcolor}{rgb}{0,0,1}
96 \def\TooltipPage#1{\expandafter\gdef\csname TooltipPage#1\endcsname{#1}}
97 \ifmouseover
98 \def\tooltip#1#2{%
99   \write\@auxout{\noexpand\TooltipPage{\thepage}}%
100   \checkTipNumber{#2}\edef\TipNumber{\FindTipNumber{#2}}%
101   \leavevmode
102   \setbox0=\hbox{{\color{tooltipcolor}{#1}}}\hbox to 0 pt{{\copy0\TooltipExtratext\hss}}%
103   \pushButton[\BC{}\BG{}\S{}\AA{\AAMouseExit{\JS{CloseTooltips();}}
104   \AAMouseEnter{\JS{this.getField("ikona."+this.pageNum+1)).hidden=false;
105     try {app.clearInterval(animace);}catch (e) {}
106     \ifmovetips nastav(\TipNumber);\fi
107     zobraz(\TipNumber);
108   }}}]
109   {\TooltipField}{\wd0}{\ht0}}
110 \else
111 \def\tooltip#1#2{%
112   \write\@auxout{\noexpand\TooltipPage{\thepage}}%
113   \checkTipNumber{#2}\edef\TipNumber{\FindTipNumber{#2}}%

```

```

114 \leavevmode
115 \setbox0=\hbox{{\color{tooltipcolor}{#1}}}\hbox to 0 pt{{\copy0\TooltipExtratext\hss}}%
116 \pushButton[\BC{}\BG{}\S{}\AA{\AAMouseExit{\JS{CloseTooltips();}}}]
117 \A{\JS{this.getField("ikona."+(this.pageNum+1)).hidden=false;
118     try {app.clearInterval(animace);}catch (e) {}
119     \ifmovetips nastav(\TipNumber);\fi
120     zobraz(\TipNumber);
121 }}]
122 {TooltipField}{\wd0}{\ht0}}
123 \fi
124 \def\delayinterval{200}
125 \ifmouseover
126 \def\tooltipanim#1#2#3{%
127     \write\@auxout{\noexpand\TooltipPage{\thepage}}%
128     \checkTipNumber{#2}\edef\TipNumberA{\FindTipNumber{#2}}%
129     \checkTipNumber{#3}\edef\TipNumberB{\FindTipNumber{#3}}%
130     \leavevmode
131     \setbox0=\hbox{{\color{tooltipcolor}{#1}}}\hbox to 0 pt{{\copy0\TooltipExtratext\hss}}%
132     \pushButton[\BC{}\BG{}\S{}\AA{\AAMouseExit{\JS{CloseTooltips();}}}]
133     \AAMouseEnter{\JS{
134         try {app.clearInterval(animace);}catch (e) {}
135         var cislo=\TipNumberA;
136         \ifmovetips nastav(\TipNumberA);\fi
137         function animuj()
138         {
139             if (cislo<\TipNumberB) cislo=cislo+1;
140             this.getField('ikona.'+(this.pageNum+1)).buttonSetIcon(this.getField("animtiph."+cislo)
141         };
142         this.getField('ikona.'+(this.pageNum+1)).buttonSetIcon(this.getField("animtiph."+TipNum
143         this.getField("ikona."+(this.pageNum+1)).hidden=false;
144         animace=app.setInterval('animuj();', \delayinterval);
145     }}}
146 ]{TooltipField}{\wd0}{\ht0}}
147 \else
148 \def\tooltipanim#1#2#3{%
149     \write\@auxout{\noexpand\TooltipPage{\thepage}}%
150     \checkTipNumber{#2}\edef\TipNumberA{\FindTipNumber{#2}}%
151     \checkTipNumber{#3}\edef\TipNumberB{\FindTipNumber{#3}}%
152     \leavevmode
153     \setbox0=\hbox{{\color{tooltipcolor}{#1}}}\hbox to 0 pt{{\copy0\TooltipExtratext\hss}}%
154     \pushButton[\BC{}\BG{}\S{}\AA{\AAMouseExit{\JS{CloseTooltips();}}}]
155     \A{\JS{
156         try {app.clearInterval(animace);}catch (e) {}
157         var cislo=\TipNumberA;
158         \ifmovetips nastav(\TipNumberA);\fi
159         function animuj()
160         {
161             if (cislo<\TipNumberB) cislo=cislo+1;
162             this.getField('ikona.'+(this.pageNum+1)).buttonSetIcon(this.getField("animtiph."+cislo)
163         };

```

```

164         this.getField('ikona.'+(this.pageNum+1)).buttonSetIcon(this.getField("animtiph."+TipNumb
165         this.getField("ikona."+(this.pageNum+1)).hidden=false;
166         animace=app.setInterval('animuj();', \delayinterval);
167     }}
168 ]{TooltipField}{\wd0}{\ht0}}
169 \fi

    This code closes tooltip if the page is closed.
170 \ifTooltip@usepdfTeX
171 \def\TooltipPageopencloseJS{ \global\pdfpageattr{%
172     /AA << /O << /S /JavaScript /JS (CloseTooltips();) >> >>}%
173 }
174 \pdfximage{\TooltipFilename.pdf}%
175 \edef\TooltipPages{\the\pdflastximagepages}%
176 \else
177 \def\TooltipPageopencloseJS{
178 \literalps@out{%
179     [ {ThisPage} << /AA <<
180     /O << /S /JavaScript /JS (CloseTooltips();) >>
181     >> >> /PUT pdfmark}}
182 \OpenAction{/S /JavaScript /JS (CloseTooltips();)}
183 \fi%\ifTooltip@usepdfTeX
184 \EveryShipout{\TooltipPageopencloseJS}%
185
186 \ifTooltip@usepdfTeX
187 \begin{insDLJS}[fancyTooltipsLoaded]{Tooltipsdljs}{DLJS for Tooltips}
188     var animace;
189     app.focusRect = false;
190     var fancyTooltipsLoaded = true;
191
192     function CloseTooltips()
193     {
194         try {this.getField("ikona").hidden=true;}catch (e) {}
195         try {app.clearInterval(animace);}catch (e) {}
196     }
197
198     function nastav(cislo)
199     {
200         var f=this.getField("ikona."+(this.pageNum+1));
201         var g=this.getField("animtiph."+cislo);
202         var sourf=f.rect;
203         var sourg=g.rect;
204         if ((mouseX+sourg[2]-sourf[0])<sourf[2])
205             var percX=100*(mouseX-sourf[0])/((sourf[2]-sourf[0])-(sourg[2]-sourg[0]));
206         else
207             var percX=100*(mouseX-sourf[0]-(sourg[2]-sourg[0]))/((sourf[2]-sourf[0])-(sourg[2]-sourg[0]));
208         var percY=100*(mouseY-sourf[3])/((sourf[1]-sourf[3])-(sourg[1]-sourg[3]));
209         if (percX>100) percX=100;
210         if (percY>100) percY=100;
211         if (percX<0) percX=0;

```

```

212     if (percY<0) percY=0;
213     f.buttonAlignX=percX;
214     f.buttonAlignY=percY;
215 }
216
217 function zobraz(cislo)
218 {
219     var f=this.getField("ikona."+this.pageNum+1));
220     var g=this.getField("animtiph."+cislo);
221     f.hidden=false;
222     f.buttonSetIcon(g.buttonGetIcon());
223 }
224 \end{insDLJS}
225 \else
226 \begin{insDLJS}[fancyTooltipsLoaded]{Tooltipsdljs}{DLJS for Tooltips}
227     var animace;
228     app.focusRect = false;
229     var fancyTooltipsLoaded = true;
230
231     function CloseTooltips()
232     {
233         try {this.getField("ikona").hidden=true;}catch (e) {}
234         try {app.clearInterval(animace);}catch (e) {}
235     }
236
237     function ImportTooltips()
238     {
239         console.println("importing pictures");
240         for (var i=1;i<=\TooltipPages;i++)
241         {
242             this.insertPages(this.numPages-1,"\TooltipFilename.pdf",(i-1),(i-1));
243             var rozm=this.getPageBox("Crop",this.numPages-1);
244             this.deletePages(this.numPages-1);
245             var p=this.addField("animtiph."+i,"button",0,rozm);
246             p.buttonPosition=position.iconOnly;
247             p.hidden=true;
248             this.getField("animtiph."+i).buttonImportIcon("\TooltipFilename.pdf",(i-1));
249         }
250         console.println("imported \TooltipPages pictures");
251         return(1);
252     }
253
254     function nastav(cislo)
255     {
256         var f=this.getField("ikona."+this.pageNum+1));
257         var g=this.getField("animtiph."+cislo);
258         var sourf=f.rect;
259         var sourg=g.rect;
260         if ((mouseX+sourg[2]-sourf[0])<sourf[2])
261         var percX=100*(mouseX-sourf[0])/((sourf[2]-sourf[0])-(sourg[2]-sourg[0]));

```



```

262     else
263     var percX=100*(mouseX-sourf[0]-(sourg[2]-sourg[0]))/((sourf[2]-sourf[0])-(sourg[2]-sourg[0])
264     var percY=100*(mouseY-sourf[3]-(sourf[1]-sourf[3])-(sourg[1]-sourg[3]));
265     if (percX>100) percX=100;
266     if (percY>100) percY=100;
267     if (percX<0) percX=0;
268     if (percY<0) percY=0;
269     f.buttonAlignX=percX;
270     f.buttonAlignY=percY;
271 }
272
273 function zobraz(cislo)
274 {
275     var f=this.getField("ikona."+this.pageNum+1));
276     var g=this.getField("animtiph."+cislo);
277     f.hidden=false;
278     f.buttonSetIcon(g.buttonGetIcon());
279 }
280 \end{insDLJS}
281 \fi

```

A cycle is used to create hidden buttons. Each button has associated a page from the file with tooltips as icon. These icons are invoked by JavaScripts defined in \tooltip and \tooltipanim macros.

```

282 \newcount\tooltip@count
283 \ifTooltip@usepdfTeX
284 \newcommand*{\TooltipHidden}{%
285     \count@=0
286     \@whilenum\count@<\TooltipPages \do{%
287         \tooltip@count=\count@
288         \advance \tooltip@count by 1%
289         \bgroup
290         \immediate\pdfximage
291         page \the\tooltip@count{\TooltipFilename.pdf}%
292         \mbox{\leavevmode
293             \vbox to 0 pt{\vss\hbox to 0 pt{\pdfstartlink user{%
294                 /Subtype /Widget
295                 /F 6
296                 /T (animtiph.\the\tooltip@count)
297                 /FT /Btn
298                 /Ff 65536
299                 /H /N
300                 /BS << /W 1 /S /S >>
301                 /MK <<
302                 /TP 1
303                 /I \the\pdflastximage\space 0 R
304                 /IF << /SW /A >>
305                 >>
306             }%
307             \phantom{\pdfrefximage \pdflastximage}%

```

```

308     \pdfendlink\hss}}}%
309     \egroup
310     \advance\count@\@ne}%
311 }
312 \AddToShipoutPicture*{\hbox to 0 pt{\TooltipHidden}}
313 \else
314 \let\TooltipHidden\relax
315 \fi

The keywords for the tooltips can be stored in the file fancytips.tex. The topics
in this file are created by \keytip macro (see the first part of the code).
316 \AtBeginDocument{\IfFileExists{fancytips.tex}{\input{fancytips.tex}
317 \PackageInfo{fancytooltips}{Inputting fancytips.tex.}}}%
318 {\PackageWarning{fancytooltips}{No file fancytips.tex!
319     Your keywords for tooltips will not work!}}}
320
321 \def\tooltipname#1#2{\expandafter\xdef\csname FancyToolTip@#1\endcsname{#2}}
322
323 \def\FindTipNumber#1{\expandafter\ifx \csname FancyToolTip@#1\endcsname\relax
324     #1\else\csname FancyToolTip@#1\endcsname\fi}
325
326 \def\checkTipNumber#1{\expandafter\ifx
327     \csname FancyToolTip@#1\endcsname\relax \PackageWarning{fancytooltips}{No
328     framenumbers assigned to keyword #1. I assume that #1 is the
329     number of the frame.}%
330     \fi}
331
332 \fi
333 \endpackage

```