The htikz package*

Claudio Fiandrino†

April 28, 2012

Abstract

This package contains some useful commands to *highlight* formulas in documents and presentations thanks to TikZ. The idea comes out from this question and actually the package is just an adaptation of the code provided by Andrew Stacey in this answer.

Contents

1 The usage

`\tikzmarkin`

`\tikzmarkend`

The way in which it is possible to highlight formulas is thanks to the insertion of delimiters before and after the part to be highlighted. To start delimiting the formula you should use the macro `\tikzmarkin` which can behave differently upon being in beamer mode or not. To end delimiting the formula to be highlighed you should use the macro `\tikzmarkend`: despite `\tikzmarkin`, this macro keep the same behavior also in beamer mode. An example of the basic usage is:

```
\[\tikzmarkin{a}x+y=400\tikzmarkend{a}\]
```

*This document corresponds to htikz v0.1, dated 2012/04/25.

†e-mail: claudio dot fiandrino at gmail dot com

which produces:

$$x + y = 400$$

When you wonder to highlight fractions, integrals or sums, the basic commands are not suitable. You should use the *extended* ones:

- `\exttikzmarkin`
- `\exttikzmarkend`

An example of this usage is:

```
\[\exttikzmarkin{r}\dfrac{100}{x}\exttikzmarkend{r}\]
```

2 The options

`beamer` When you call the package:

```
\usepackage[beamer]{htikz}
```

you enter in beamer mode and the `\tikzmarkin` and `\exttikzmarkend` macro allows you to decide when to insert the highlighting. For example:

```
\begin{align}
\tikzmarkin<1->{a1}a_i\tikzmarkend{a1} + b_j = 10 \\
\tikzmarkin<3>{c}c_j + d_j + \tikzmarkin<2>{b}a_i\tikzmarkend{b} >= 30\tikzmarkend{c}
\end{align}
```

`nofill` Using the option `nofill` allows you to simply not see the color in background. When the option is active, you can not change this behaviour inside the document.

`customcolors` This option allows you to customize both the fill and the background color. When using this option, two commands become available:

- `\setfillcolor`
- `\setbordercolor`

They can be use in whatever part of the document allowing a high customization of colors. For example:

```
\setfillcolor{red!10}
\setbordercolor{red}
\[\exttikzmarkin{c}\dfrac{100}{x}\exttikzmarkend{c}\]
```

produces:

$$\frac{100}{x}$$

Then:

```
\setfillcolor{blue!10}
\setbordercolor{blue}
\[\tikzmarkin{x}x+y=400\tikzmarkend{x}\]
```

produces:

$$\boxed{x+y=400}$$

Notice that, once `customcolors` is active, you need to specify colors at least in the preamble.

3 Implementation

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{htikz}[2012/04/25 v0.1 A simple way to highlight formulas parts.]
\RequirePackage{tikz}
\RequirePackage{etoolbox}
```

3.1 Options definition

In this subsection we show the definitions of pre-defined colors and options.

```
%% Colors

% Pre-defined colors
\definecolor{lightbrown}{RGB}{255,218,195}
\definecolor{violet}{RGB}{197,122,195}

\newcommand{\fcol}{lightbrown}
\newcommand{\bcol}{violet}

%% Package option

% Decide whether to fill or not the highlighting
\newbool{fill}
\booltrue{fill}
\DeclareOption{nofill}{\boolfalse{fill}}

% Decide whether to change pre-defined colors
\DeclareOption{customcolors}{
\def\setfillcolor#1{\def\@fillcolor{#1}}
\def\setbordercolor#1{\def\@bordercolor{#1}}

\renewcommand{\fcol}{\@fillcolor}
\renewcommand{\bcol}{\@bordercolor}
}

% Usage inside beamer class
\newbool{beamer}
```

```
\boolfalse{beamer}
\DeclareOption{beamer}{\booltrue{beamer}}

\ProcessOptions
```

3.2 General settings

In this subsection we show the general settings that allow the highlighing.

```
%% Settings

\ifbool{beamer}{%true
   \newcounter{jumping}
   \resetcounteronoverlays{jumping}

   \def\jump@setbb#1#2#3{%
     \@ifundefined{jump@#1@maxbb}{%
       \expandafter\gdef\csname jump@#1@maxbb\endcsname{#3}%
     }{%
       \csname jump@#1@maxbb\endcsname
       \pgf@xa=\pgf@x
       \pgf@ya=\pgf@y
       #3
       \pgfmathsetlength\pgf@x{max(\pgf@x,\pgf@xa)}%
       \pgfmathsetlength\pgf@y{max(\pgf@y,\pgf@ya)}%
       \expandafter\xdef\csname jump@#1@maxbb\endcsname{\noexpand\pgfpoint{\the\pgf@x}{\the\pg
     }
     \@ifundefined{jump@#1@minbb}{%
       \expandafter\gdef\csname jump@#1@minbb\endcsname{#2}%
     }{%
       \csname jump@#1@minbb\endcsname
       \pgf@xa=\pgf@x
       \pgf@ya=\pgf@y
       #2
       \pgfmathsetlength\pgf@x{min(\pgf@x,\pgf@xa)}%
       \pgfmathsetlength\pgf@y{min(\pgf@y,\pgf@ya)}%
       \expandafter\xdef\csname jump@#1@minbb\endcsname{\noexpand\pgfpoint{\the\pgf@x}{\the\pg
     }
   }

   \tikzset{%
     remember picture with id/.style={%
       remember picture,
       overlay,
       draw=\bcol,
       save picture id=#1,
     },
     save picture id/.code={%
       \edef\pgf@temp{#1}%
       \immediate\write\pgfutil@auxout{%
```

```
        \noexpand\savepointas{\pgf@temp}{\pgfpictureid}}%
    },
    if picture id/.code args={#1#2#3}{%
      \@ifundefined{save@pt@#1}{%
        \pgfkeysalso{#3}%
      }{
        \pgfkeysalso{#2}%
      }
    },
    onslide/.code args={<#1>#2}{%
      \only<#1>{\pgfkeysalso{#2}}%
    },
    alt/.code args={<#1>#2#3}{%
      \alt<#1>{\pgfkeysalso{#2}}{\pgfkeysalso{#3}}%
    },
    stop jumping/.style={
      execute at end picture={%
        \stepcounter{jumping}%
        \immediate\write\pgfutil@auxout{%
          \noexpand\jump@setbb{\the\value{jumping}}{\noexpand\pgfpoint{\the\pgf@picminx}{\the
        },
        \csname jump@\the\value{jumping}@maxbb\endcsname
        \path (\the\pgf@x,\the\pgf@y);
        \csname jump@\the\value{jumping}@minbb\endcsname
        \path (\the\pgf@x,\the\pgf@y);
      },
    }
  }
}{% false
  \tikzset{%
    remember picture with id/.style={%
      remember picture,
      overlay,
      draw=\bcol,
      save picture id=#1,
    },
    save picture id/.code={%
      \edef\pgf@temp{#1}%
      \immediate\write\pgfutil@auxout{%
        \noexpand\savepointas{\pgf@temp}{\pgfpictureid}}%
    },
    if picture id/.code args={#1#2#3}{%
      \@ifundefined{save@pt@#1}{%
        \pgfkeysalso{#3}%
      }{
        \pgfkeysalso{#2}%
      }
    },
  }
}
```

```
\def\savepointas#1#2{%
  \expandafter\gdef\csname save@pt@#1\endcsname{#2}%
}

\def\tmk@labeldef#1,#2\@nil{%
  \def\tmk@label{#1}%
  \def\tmk@def{#2}%
}

\tikzdeclarecoordinatesystem{pic}{%
  \pgfutil@in@,{#1}%
  \ifpgfutil@in@%
    \tmk@labeldef#1\@nil
  \else
    \tmk@labeldef#1,\pgfpointorigin\@nil
  \fi
  \@ifundefined{save@pt@\tmk@label}{%
    \tikz@scan@one@point\pgfutil@firstofone\tmk@def
  }{%
  \pgfsys@getposition{\csname save@pt@\tmk@label\endcsname}\save@orig@pic%
  \pgfsys@getposition{\pgfpictureid}\save@this@pic%
  \pgf@process{\pgfpointorigin\save@this@pic}%
  \pgf@xa=\pgf@x
  \pgf@ya=\pgf@y
  \pgf@process{\pgfpointorigin\save@orig@pic}%
  \advance\pgf@x by -\pgf@xa
  \advance\pgf@y by -\pgf@ya
  }%
}
```

3.3 The highlighting commands

In this subsection we show the definition of the highlighing commands in `beamer` mode and not. When the `nofill` option is active, there is no definition for the `fill` color.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The highlighting commands

\ifbool{beamer}{%true
   \ifbool{fill}{%true
      \newcommand<>{\tikzmarkin}[1]{%
      \only#2{\tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners,fill=\fcol]
      (pic cs:#1) ++(0.1,-0.18) rectangle (-0.1,0.35)
      ;}}
      \newcommand<>{\exttikzmarkin}[1]{%
      \only#2{\tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners,fill=\fcol]
```

```
      (pic cs:#1) ++(0.1,-0.5) rectangle (-0.1,0.65)
      ;}}
   }{%false
      \newcommand<>{\tikzmarkin}[1]{%
      \only#2{\tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners]
      (pic cs:#1) ++(0.075,-0.18) rectangle (-0.075,0.35)
      ;}}
      \newcommand<>{\exttikzmarkin}[1]{%
      \only#2{\tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners]
      (pic cs:#1) ++(0.1,-0.5) rectangle (-0.1,0.65)
      ;}}
   }
}{%false
      \ifbool{fill}{%true
      \newcommand{\tikzmarkin}[1]{%
      \tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners,fill=\fcol]
      (pic cs:#1) ++(0.1,-0.18) rectangle (-0.1,0.35)
      ;}
      \newcommand{\exttikzmarkin}[1]{%
      \tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners,fill=\fcol]
      (pic cs:#1) ++(0.1,-0.5) rectangle (-0.1,0.65)
      ;}
   }{%false
      \newcommand{\tikzmarkin}[1]{%
      \tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners]
      (pic cs:#1) ++(0.075,-0.18) rectangle (-0.075,0.35)
      ;}
      \newcommand{\extikzmarkin}[1]{%
      \tikz[remember picture with id=#1]
      \draw[line width=1pt,rectangle,rounded corners]
      (pic cs:#1) ++(0.1,-0.5) rectangle (-0.1,0.65)
      ;}
   }

}

\newcommand\tikzmarkend[2][]{%
\tikz[remember picture with id=#2] #1;}

\newcommand\exttikzmarkend[2][]{%
\tikz[remember picture with id=#2] #1;}
```