The hf-tikz package*

Claudio Fiandrino†

March 1, 2013

Abstract

This package provides a way to *highlight* formulas in both documents and presentations thanks to Ti*k*Z. The idea comes out from this question on TeX.StackExchange and it is based on the `tikzmark` macro from Andrew Stacey and Peter Grill.

Contents

1 Introduction and requirements

The aim of the package is to provide a simple way to highlight formulas. This is not the first package that tries to accomplish this task, but, rather than empheq, hf-tikz provides

*This document corresponds to hf-tikz v0.3, dated 2013/01/13.

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

also a way to highlight formulas overlay-aware inside a presentation, not only in standard documents. Moreover, in contrast with empheq, hf-tikz even allows to highlight just a part of an equation.

The package uses Ti*k*Z and it is based on the `tikzmark` macro from Andrew Stacey and Peter Grill (see as reference this answer and this question): among the numerous versions present on TeX.SX, the reference one implemented is taken from this answer. Indeed, as explained later, the concept of *extendible markers* helps in customizing the box dimension.

The packages loaded by hf-tikz are:

- Ti*k*Z and the libraries `shadings` and `decorations.markings` (this library is not always loaded, see subsection 3.6);
- xparse;
- etoolbox.

2 The usage

2.1 The basic commands

Formulas can be highlighted by means of the insertion of delimiters before and after the part to be highlighted. Two compilation runs are always necessary: the first one to compute the position of the markers (also called delimiters) and the second one to place the box.

`\tikzmarkin` The starting delimiter should be introduced with the `\tikzmarkin` macro: it has a different syntax upon being in `beamer` mode or not as it will be pointed out in subsection 3.1.

`\tikzmarkend` The end delimiter should be introduced by means of the `\tikzmarkend` macro: despite `\tikzmarkin`, this macro keeps the same syntax also in `beamer` mode.

An example of the basic usage is:

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

which produces:

$$x+\tikzmarkin{a}y\tikzmarkend{a}=400$$

Notice that delimiter labels, also called `marker-ids`, should characterize *uniquely* the part highlighted therefore reuse the same label name twice will lead to undesired results. Along this documentation there are examples showing how it is possible to give label names.

In presence of fractions, sums, integrals and other operators, the standard command is not suitable. Consider the following example:

```
\[\tikzmarkin{a-1}x+\dfrac{z}{y}=400\tikzmarkend{a-1}\]
```

It leads to:

$$\tikzmarkin{a-1}x+\dfrac{z}{y}=400\tikzmarkend{a-1}$$

In this case, the user must specify manually which are the *shift-offsets* that delimits the box:

```
\begin{equation}
```

```
\tikzmarkin{right delim frac}(0.1,-0.4)(-0.1,0.5)
x+\dfrac{z}{y}=400
\tikzmarkend{right delim frac}
\end{equation}
```

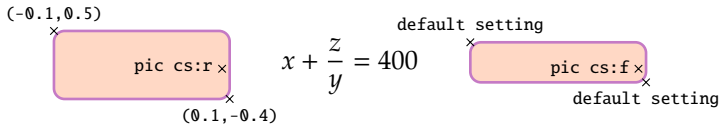
and this fixes the problem:

$$x+\frac{z}{y}=400 \tag{1}$$

The *shift-offsets* should be introduced following this syntax:

```
\tikzmarkin{marker-id}(below right offset)(above left offset)
```

Here is an image that explains the differences between the default setting and the *shift-offsets* manually inserted for the previous example:



Manual shifts allow to customize dimensions on the base of user's needs: they should be introduced inside round braces as coordinate points. Coordinates, indeed, introduce more degree of freedom from the user's point of view while other solutions are more restrictive. Markers, therefore, are *extensible*. Notice that is is not possible to use markers separately, but they should be declared in pair.

From version 0.3 it is also possible to exploit a key-interface to set the *shift-offsets*; for example, the previous example, could be done in another manner:

```
\begin{equation}
\tikzmarkin[below right offset={0.1,-0.4},above left offset={-0.1,0.5}]
 {right delim frac 2}
x+\dfrac{z}{y}=400
\tikzmarkend{right delim frac 2}
\end{equation}
```

leads to:

$$x+\frac{z}{y}=400 \tag{2}$$

The list of keys available to customize the *shift-offsets* are:

- `left` (`initial: -0.1/-0.075`): this key sets the left offset (the second value is active when the `fill` option is active);
- `right` (`initial: 0.1/0.075`): this key sets the right offset (the second value is active when the `fill` option is active);
- `above` (`initial: 0.35`): this key sets the above offset;
- `below` (`initial: -0.18`): this key sets the below offset;

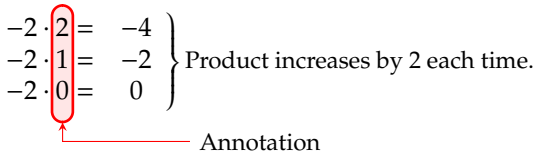
- `below right` (`initial: 0.1/0.075,0.35`): this key sets contemporarely the below and the right offsets;
- `above left` (`initial: -0.1/-0.075,0.35`): this optikeyon sets contemporarely the above and the left offsets.

All the keys, not only those ones dedicated to the *shift-offsets*, should be introduced in the first optional argument if `beamer` option is disables or in the second argument in the other case. In this way, the keys have a local application; if, instead, they are set by means of `\tikzset{}` they are applied to the whole document. For example:

```
\tikzset{above left offset={-0.1,0.325},below right offset={0.1,-0.4}}
```

2.2 An advanced example

This example shows how to insert an annotation aligned with a sentence: it requires the `calc` library from Ti*k*Z. The way in which colors are set is explained in subsection 3.2.



The code is:

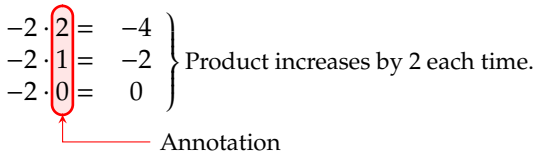
```
\begin{equation*}
\left.\begin{array}{cc}
 -2\cdot \tikzmarkin{col}(0.05,-0.3)(-0.05,0.4)2=& -4 \\
 -2\cdot 1=& -2 \\
 -2\cdot 0\tikzmarkend{col}=& 0
\end{array}\right\} \text{\small Product increases by 2 each time.}
\end{equation*}

% To insert the annotation
\begin{tikzpicture}[remember picture,overlay]
 % adjust the shift from "col" to move the position of the annotation
 \coordinate (col-aa) at ($(col)+(1.825,-1.8)$);
 \node[align=left,right] at (col-aa) {\small{Annotation}};
 \path[-stealth,red,draw] (col-aa) -| ($(col)+(0.14,-1.55)$);
\end{tikzpicture}
```

The message here is that, when something is highlighted, the `marker-id` could be used to subsequently add elements on the image, i.e. the annotation.

From version 0.3 there is a more simpler manner to make annotations: for this the option `markings` should be enabled.

Look the previous example done in this way (the annotation is put a bit close to the highlighted area intentionally):



The code:

```
\begin{equation*}
\left.\begin{array}{cc}
 -2\cdot \tikzmarkin[mark at=0.93]{col 1}(0.05,-0.2)(-0.05,0.4)2=& -4 \\
 -2\cdot 1=& -2 \\
 -2\cdot 0\tikzmarkend{col 1}=& 0
\end{array}\right\} \text{\small Product increases by 2 each time.}
\end{equation*}

\begin{tikzpicture}[remember picture,overlay]
 \coordinate (col-aa) at ($(col 1)+(1.3,-1.8)$);
 \node[align=left,right] at (col-aa) {\small{Annotation}};
 \path[-stealth,red,draw,use marker id] (col-aa) -| (0,0);
\end{tikzpicture}
```

Thus, it is sufficient to `mark at` a given position the rectangle delimiting the highlighted area and then access this coordinate by means of `use marker id`. Further details are provided in subsection 3.6.

3 The options

3.1 The `beamer` mode

`beamer` The call:

```
\usepackage[beamer]{hf-tikz}
```

let the package to enter in `beamer` mode and the `\tikzmarkin` macro is *overlay-aware*, that is overlay specifications could be introduced as first argument. 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}
```

Examples in which the overlay specifications could be introduced are:

- a single number: <1>;
- multiple numbers separated by commas and delimited by braces: <{1,2,3}>;
- a single number followed by a dash: <1->.

3.2 Customize colors

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

- \hfsetfillcolor
- \hfsetbordercolor

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

```
\hfsetfillcolor{red!10}
\hfsetbordercolor{red}
\[
\tikzmarkin{z}(0.2,-0.4)(-0.2,0.6)
 \dfrac{100}{x}
\tikzmarkend{z}
\]
```

produces:

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

Then:

```
\hfsetfillcolor{blue!10}
\hfsetbordercolor{blue}
\[\tikzmarkin{z1}x+y=400\tikzmarkend{z1}\]
```

gives:

$$x + y = 400$$

From version 0.3 it is also possible to customize the fill and the border color by means of the keys:

- set fill color (initial: fancybrown): this key sets the fill color;
- set border color (initial: fancyviolet): this key sets the border color.

An example:

```
\[
\tikzmarkin[set fill color=green!50!lime!30,set border color=green!40!black]{z-a}(0.2,-0.4)(-0.2,0.6)
 \dfrac{100}{x}
\tikzmarkend{z-a}
\]
```

The result:

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

Notice that:

- the definition of colors could be also done via `\tikzset` and the application is global

```
\tikzset{set fill color=orange!30,set border color=orange}
```

- global definitions, done via `\tikzset` or `\hfsetfillcolor` and `\hfsetbordercolor` are always overridden by local ones; that is:

```
\tikzset{set fill color=orange!30,set border color=orange}
\[
\tikzmarkin[set fill color=green!50!lime!30,set border color=green!40!black]{label}(0.2,-0.4)(-0.2,0.6)
 \dfrac{100}{x}
\tikzmarkend{z-a}
\]
```

still gives:

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

3.3 Using shadings

shade

The option shade activates the possibility of introducing shaded backgrounds besides the fill color currently set. Available shadings are:

- vertical shading;
- horizontal shading;
- radial shading.

Example with vertical shading

Code:

```
\[
\tikzmarkin[top color=white, bottom color=blue!20]{vshade}
 x+y=400
\tikzmarkend{vshade}
\]
```

Result:

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

Example with horizontal shading

Code:

```
\[
\tikzmarkin[left color=white, right color=blue!20]{hoshade}
 x+y=400
\tikzmarkend{hoshade}
\]
```

Result:

$$x + y = 400$$

Example with radial shading

Code:

```
\[
\tikzmarkin[outer color=white, inner color=blue!20]{rshade}
 x+y=400
\tikzmarkend{rshade}
\]
```

Result:

$$x + y = 400$$

3.4 Avoid the background color

`nofill` Using the `nofill` option allows to simply not introduce the background color. When the option is active, you can not change this behaviour inside the document. Another option to remove the background color, is to set the fill color by means of `\hfsetfillcolor` with the same color of the page.

3.5 Disable rounded corners

`norndcorners` To disable the rounded corners, there are actually two ways. The first one, which is general, is the option `norndcorners`: as the other options it should be provided when loading the package.

There is also a second way, which actually disables the rounded corners locally; you should proceed using the `disable rounded corners` key set to `true`.

For example:

```
\[
\tikzmarkin[disable rounded corners=true]{mark 1}
 x+y=400
\tikzmarkend{mark 1}
\]
```

The result:

$$x + y = 400$$

3.6 The `markings` option

`markings` Loading the package with the `markings` option allows to mark some positions on the rectangle delimiting the highlighted area. This is done by setting the key `mark at=<pos>` where `<pos>=[0,1]`; the position could be later on access with `use marker id=<id-num>` where `<id-num>` is the progressive identifier of the positions previously marked.

For example:

$$x + y = 400$$

is realized by means of:

```
\tikzset{set fill color=orange!30,set border color=orange}
\[
\tikzmarkin[show markers,mark at=0,]{marker 1}
 x+y=400
\tikzmarkend{marker 1}
\tikz[remember picture,overlay]\draw[use marker id,blue,xscale=-1](0,0)arc(270:90:1.5mm);
\]
```

Actually, it is possible to mark more than one point:

$$x + y = 400$$

by referring them in order as they have been marked:

```
\[
\tikzmarkin[show markers,mark at=0,mark at=0.55]{marker 2}
 x+y=400
\tikzmarkend{marker 2}
\tikz[remember picture,overlay]\draw[use marker id=1,blue,xscale=-1](0,0)arc(270:90:1.5mm);
\tikz[remember picture,overlay]\draw[use marker id=2,blue](0,0)arc(270:90:1.5mm);
\]
```

The markers are visible when the `show markers` key is present, but for annotations it is preferable to not show them (by defaul they are not); moreover, markes can be customized:

- `marker size` (`initial: 1pt`): this key sets the radius of the marker;
- `marker color` (`initial: blue`): this key sets the color of the marker.

The option necessitates of the `decorations.markings` from Ti*k*Z: this library, however, is not loaded always, but just in case the `markings` option is active.

4 Implementation

```
\RequirePackage{tikz}
\usetikzlibrary{shadings}
\RequirePackage{xparse}
\RequirePackage{etoolbox}
```

This warning is arised at first compilation run, to inform that a second run is necessary to get the final result. The code used as base is taken from this answer in TeX.SX.

```
\AtEndDocument{%
\let\oldpgfsyspdfmark\pgfsyspdfmark
\def\pgfsyspdfmark#1#2#3{%
  \expandafter\let\expandafter\tmp\csname pgf@sys@pdf@mark@pos@#1\endcsname
  \oldpgfsyspdfmark{#1}{#2}{#3}%
  \expandafter\ifx\csname pgf@sys@pdf@mark@pos@#1\endcsname\tmp\else
  \let\oldsavepointas\savepointas
  \def\savepointas##1##2{%
    \immediate\write\@auxout{hf-TikZ Warning: Mark '##1' changed. Rerun to get mark in right position.}%
  }
  \fi
}}
```

4.1 Options definition

In this subsection the definitions of pre-defined colors and options are shown.

```
%% Colors

% Pre-defined colors
\definecolor{fancybrown}{RGB}{255,216,197}
\definecolor{fancyviolet}{RGB}{197,122,197}

\newcommand{\fcol}{fancybrown}
\newcommand{\bcol}{fancyviolet}

%% Package option

\newbool{fill}
\booltrue{fill}
\DeclareOption{nofill}{\boolfalse{fill}}

\DeclareOption{customcolors}{
\def\hfsetfillcolor#1{\renewcommand{\fcol}{#1}}
\def\hfsetbordercolor#1{\renewcommand{\bcol}{#1}}
\pgfkeys{/tikz/.cd,
  set fill color/.code={\renewcommand{\fcol}{#1}},
  set border color/.code={\renewcommand{\bcol}{#1}}
 }
}

\newbool{shade}
\boolfalse{shade}
```

```
\DeclareOption{shade}{\booltrue{shade}}

\newbool{beamer}
\boolfalse{beamer}
\DeclareOption{beamer}{\booltrue{beamer}}

\newbool{norndcorners}
\boolfalse{norndcorners}
\DeclareOption{norndcorners}{\booltrue{norndcorners}}

\newbool{markings}
\boolfalse{markings}
\DeclareOption{markings}{\booltrue{markings}}

\ProcessOptions
```

The keys definition to locally disable rounded corners.

```
\pgfkeys{/tikz/.cd,%
      not use rounded corners/.is choice,%
      not use rounded corners/true/.style={rounded corners=0pt},%
      not use rounded corners/false/.style={rounded corners},%
}%

\tikzset{disable rounded corners/.estyle={%
      not use rounded corners=#1,%
      },%
      disable rounded corners/.default=false,%
}%

```

The keys definition of the offsets: the initial values change according to the presence of `fill` option to compatibility reasons.

```

\ifbool{fill}{%
 \pgfkeys{/tikz/.cd,%
      left offset/.initial=-0.1,
      right offset/.initial=0.1,
      above offset/.initial=0.35,
      below offset/.initial=-0.18,
 }
}{
 \pgfkeys{/tikz/.cd,%
      left offset/.initial=-0.075,
      right offset/.initial=0.075,
      above offset/.initial=0.35,
      below offset/.initial=-0.18,
 }
}


```

```
\pgfkeys{/tikz/.cd,%
      left offset/.get=\leftoff,
      left offset/.store in=\leftoff,
      right offset/.get=\rightoff,
      right offset/.store in=\rightoff,
      above offset/.get=\aboveoff,
      above offset/.store in=\aboveoff,
      below offset/.get=\belowoff,
      below offset/.store in=\belowoff,
      below right offset/.initial={\rightoff,\belowoff},
      below right offset/.get=\belowrightoff,
      below right offset/.store in=\belowrightoff,
      above left offset/.initial={\leftoff,\aboveoff},
      above left offset/.get=\aboveleftoff,
      above left offset/.store in=\aboveleftoff,
}%
```

The keys and style definition of the markings: they are active when the `markings` option is present. This is a feature request from Bodo Manthey and the implementation is taken from Jake's answer in TeX.SX: thanks to both.

```
\ifbool{markings}{
  \usetikzlibrary{decorations.markings}
  \newif\ifshowmarkers
  \pgfkeys{/tikz/show markers/.is if=showmarkers}
  \pgfkeys{/tikz/show markers=false}

  \pgfkeys{/tikz/.cd,%
      marker color/.initial=blue,
      marker color/.get=\colmarker,
      marker color/.store in=\colmarker,
      marker size/.initial=1pt,
      marker size/.get=\sizemarker,
      marker size/.store in=\sizemarker,
  }

  \tikzset{
   mark at/.style={
     decoration={
        markings,
        mark=
         at position #1
         with
         {
           \coordinate (marker point-\pgfkeysvalueof{/pgf/decoration/mark info/sequence number}) at (0pt,
           \coordinate (marker unit vector-\pgfkeysvalueof{/pgf/decoration/mark info/sequence number}) at
           \coordinate (marker orthogonal unit vector-\pgfkeysvalueof{/pgf/decoration/mark info/sequence
           \ifshowmarkers% conditional to make them appear just when invoked
             \draw[draw=none,fill=\colmarker,radius=\sizemarker] (0,0) circle ;
           \else
             \relax
```

```
          \fi
        }
    },
    postaction=decorate
  },
  use marker id/.style={
    shift=(marker point-#1),
    x=(marker unit vector-#1),
    y=(marker orthogonal unit vector-#1)
  },
  use marker id/.default=1,
 }
}
```

4.2 General settings

In this subsection the general settings that allow the highlighing are shown.

```
%% 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\pgf@y}}%
   }
   \@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\pgf@y}}%
   }
 }

 \tikzset{%
   remember picture with id/.style={%
```

```
    remember picture,
    overlay,
    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\pgf@picminy}
      },
      \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,
    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,(0pt,0pt)\@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
  }%
}
```

4.3 The highlighting commands

In this subsection the definition of the highlighing commands in beamer mode and not are shown. Thanks to etoolbox it is possible to perform a check on the options active, then the commands are consequently declared.

```
\ifbool{norndcorners}{%true-norndcorners
  \ifbool{beamer}{%true-beamer
   \ifbool{fill}{%true-fill
  \ifbool{shade}{%true-shade
   \NewDocumentCommand{\tikzmarkin}{r<> o m D(){\belowrightoff} D(){\aboveleftoff}}{%
   \IfNoValueTF{#2}{%true-val
      \only<#1>{\tikz[remember picture,overlay]
      \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol]
```

```
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}
   }{%false-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol,#2]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}}
   }
}{%false-shade
 \NewDocumentCommand{\tikzmarkin}{r<> o m D(){\belowrightoff} D(){\aboveleftoff}}{%
  \IfNoValueTF{#2}{%true-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}
   }{%false-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol,#2]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}}
   }
}
 }{%false-fill
\NewDocumentCommand{\tikzmarkin}{r<> o m D(){\belowrightoff} D(){\aboveleftoff}}{%
  \IfNoValueTF{#2}{%true-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,draw=\bcol]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}
   }{%false-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,draw=\bcol,#2]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}}
   }
}
}{%false-beamer
 \ifbool{fill}{%true-fill
\ifbool{shade}{%true-shade
 \NewDocumentCommand{\tikzmarkin}{o m D(){\belowrightoff} D(){\aboveleftoff}}{%
 \IfNoValueTF{#1}{%true-val
    \tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol]
    (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
    ;
 }{%false-val
 \tikz[remember picture,overlay]
 \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol,#1]
 (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
 ;}}
```

```
  }{%false-shade
   \NewDocumentCommand{\tikzmarkin}{o m D(){\belowrightoff} D(){\aboveleftoff}}{%
   \IfNoValueTF{#1}{%true-val
      \tikz[remember picture,overlay]
      \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol]
      (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
      ;
   }{%false-val
   \tikz[remember picture,overlay]
   \draw[line width=1pt,rectangle,fill=\fcol,draw=\bcol,#1]
   (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
   ;}}
  }
   }{%false-fill
    \NewDocumentCommand{\tikzmarkin}{o m D(){\belowrightoff} D(){\aboveleftoff}}{%
   \IfNoValueTF{#1}{%true-val
      \tikz[remember picture,overlay]
      \draw[line width=1pt,rectangle,draw=\bcol]
      (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
      ;
   }{%false-val
   \tikz[remember picture,overlay]
   \draw[line width=1pt,rectangle,draw=\bcol,#1]
   (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
   ;}}
%   \NewDocumentCommand{\tikzmarkin}{m D(){0.075,-0.18} D(){-0.075,0.35}}{%
%   \tikz[remember picture,overlay]
%   \draw[line width=1pt,rectangle,draw=\bcol]
%   (pic cs:#1) ++(#2) rectangle (#3) node [anchor=base] (#1){}
%   ;}
   }
  }
}{%false-norndcorners
\ifbool{beamer}{%true-beamer
 \ifbool{fill}{%true-fill
\ifbool{shade}{%true-shade
 \NewDocumentCommand{\tikzmarkin}{r<> o m D(){\belowrightoff} D(){\aboveleftoff}}{%
 \IfNoValueTF{#2}{%true-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}
   }{%false-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol,#2]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}}
   }
}{%false-shade
 \NewDocumentCommand{\tikzmarkin}{r<> o m D(){\belowrightoff} D(){\aboveleftoff}}{%
```

```
  \IfNoValueTF{#2}{%true-val
   \only<#1>{\tikz[remember picture,overlay]
   \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol]
   (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
   ;}
  }{%false-val
   \only<#1>{\tikz[remember picture,overlay]
   \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol,#2]
   (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
   ;}}
  }
}
 }{%false-fill
\NewDocumentCommand{\tikzmarkin}{r<> o m D(){\belowrightoff} D(){\aboveleftoff}}{%
 \IfNoValueTF{#2}{%true-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,disable rounded corners,draw=\bcol]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}
   }{%false-val
    \only<#1>{\tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,disable rounded corners,draw=\bcol,#2]
    (pic cs:#3) ++(#4) rectangle (#5) node [anchor=base] (#3){}
    ;}}
  }
 }
}{%false-beamer
 \ifbool{fill}{%true-fill
\ifbool{shade}{%true-shade
 \NewDocumentCommand{\tikzmarkin}{o m D(){\belowrightoff} D(){\aboveleftoff}}{%
 \IfNoValueTF{#1}{%true-val
    \tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol]
    (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
    ;
 }{%false-val
 \tikz[remember picture,overlay]
 \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol,#1]
 (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
 ;}}
}{%false-shade
 \NewDocumentCommand{\tikzmarkin}{o m D(){\belowrightoff} D(){\aboveleftoff}}{%
 \IfNoValueTF{#1}{%true-val
    \tikz[remember picture,overlay]
    \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol]
    (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
    ;
 }{%false-val
 \tikz[remember picture,overlay]
 \draw[line width=1pt,rectangle,disable rounded corners,fill=\fcol,draw=\bcol,#1]
```

```
(pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
;}}
}
}{%false-fill
\NewDocumentCommand{\tikzmarkin}{o m D(){\belowrightoff} D(){\aboveleftoff}}{%
\IfNoValueTF{#1}{%true-val
   \tikz[remember picture,overlay]
   \draw[line width=1pt,rectangle,disable rounded corners,draw=\bcol]
   (pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
   ;
}{%false-val
\tikz[remember picture,overlay]
\draw[line width=1pt,rectangle,disable rounded corners,draw=\bcol,#1]
(pic cs:#2) ++(#3) rectangle (#4) node [anchor=base] (#2){}
;}}
}
}
}

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