Underlining (and more) with soulpos

Javier Bezos

2021-10-15

This package just combines soul with the `savepos` mechanism provided by the pdftex engine, so that you can create (almost) arbitrary underlining and similar “decorations”, including rules, leaders and even pictures (pgf, pstricks, etc.). Unlike soul underlines, which are built by repeating small elements, here each chunk of text to be underlined is a single element.

The main drawback is obvious – since it relies on `\pdfsavepos` two passes are necessary. Further, to prevent exhausting the hash table, the auxiliary file containing the information about each underline is read only when needed, which may impact performance negatively.[1]

Internally soul knows to some point where a break happens, and this information may be used to set diferent styles depending on the position.

This version (1.0) does almost no checking (e.g., to warn about the need for a new run), which is left for a later release.

The package soulutf8 is loaded if it (or soul) has not been loaded before (note the UTF-8 encoding is not necessary for soulutf8 to work).

1 Usage

Underlining macros are defined much like in soul.

```
\ulposdef{<name>}[<options>]{<commands>}
```

Defines an underline as `<commands>`, which is placed in a box of width zero with either `\llap` or `\rlap`, as explained below. Typically, `<commands>` will contain a rule or leaders. If the text spans more than one line, then there will be several chunks to be undelined (one per line).

You can use the following macros in `<commands>`.

```
\ulwidth
```

The width of the text block to be underlined. A basic underlining is:

```
\ulposdef{\uline}{\rule[-.8ex]{\ulwidth}{.5pt}}
```

```
\ifulstarttype{<type>}{<true>}{<false>}
\ifulendtype{<type>}{<true>}{<false>}
```

These tests can be used in `<commands>`, to set different underlining styles depending on where the current chunk begins or ends. Here `<type>` is: 0 if the very start or end of the underlined text, 1 if a space, 2 if a discretionary hyphen and 3 if an explicit hyphen. See an example below.

```
\ulstarttype    \ulendtype
```

Macros storing the values described above, so that you can use LaTeX conditionals (or TeX ones). So, to check if the end is an hyphen, test `\ulendtype>1`.

Valid keys/values in `<options>` are:

`xoffset` This key provides a simple way to fix an unpleasant effect found in many programs when colouring the text background – the colour starts and ends just at the edges of the first and last characters. Just set the offset to a value larger than 0pt, as for example .1 em. This value is added (×2) to `\ulwidth`, but no space is added to the text (which can be done with the `gap` key). Of course, you can still do finer adjustments in the definition of the underline, as shown in the samples below. You can use `xoffset-start` and `xoffset-end` to set the corresponding values separately.

`gap` It is equivalent to the outer space in `\sodef`. This value is *not* added to `\ulwidth`. You can use `gap-start` and `gap-end` to set the corresponding values separately.

---

[1] Actually, two files are created, with extensions `upa` and `upb`.

`hyphens` Sometimes excluding the hyphen from the underlined text could make sense. Default is `hyphens=include` but you can set it to `hyphens=exclude`.

`overdraw` By default underlines are drawn before the text is typeset (with `\rlap`), so that they are placed behind. However, it can be drawn after (with `\llap`), on top the text, with `overdraw` or, equivalently, `overdraw=true` (default is `overdraw=false`).

`\ulpostolerance`

The current algorithm is based on changes of the y coordinate of savepos and therefore presumes a regular baseline. Any increasing or decreasing of y is considered a new chunk, but you can give a certain tolerance with, for example:

```
\renewcommand{\ulpostolerance}{12}
```

2 Future work

- Warnings.
- In short documents, providing an option for using the aux file.
- Redefinable macros.
- `offset`, including edges at line breaks.
- `.upb` is generated at the end of the run, so it might be used by another program to generate the decorations. As of 1.0. however, its syntax is likely to change and therefore unsupported.
- Better manual.
- Improved performance.
- Fixing some issues in multicolumn text (it does not work if there are just two lines, one at the left and the other at the right).
- Some predefined "decorations".

3 Examples

```
\ulposdef{\ulpgfA}{%
  \raisebox{-.75ex}{%
    \begin{tikzpicture}%
    \clip (0,-1pt) rectangle (\ulwidth,1pt);
    \draw[
      color=black!40,
      line width=.7pt,
      decorate,
      decoration=
        {random steps,
         segment length=1.5mm,
         amplitude=.5pt}]
    (0,0) -- +(\ulwidth+3pt,0);
    \end{tikzpicture}}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.

```
\ulposdef{\ulpgfB}{%
  \raisebox{-.75ex}{%
    \begin{tikzpicture}%
    \clip (0,-1pt) rectangle (\ulwidth,1pt);
    \draw[color=black!40,
      line width=.7pt,
      decorate,
      decoration=
        {snake,
         amplitude=.3pt,
         segment length=1mm,}]
     (0,0) -- +(\ulwidth+3pt,0);
     \end{tikzpicture}}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.

```
\ulposdef{\ulpgfC}[xoffset=.15em]{%
  \ifulstarttype{0}%
    {\def\arr{|}}%
    {\def\arr{<}}%
  \ifulendtype{0}%
    {\edef\arr{\arr-|}}%
    {\edef\arr{\arr->}}%
  \raisebox{-.7ex}{%
    \tikz
    \draw[\arr,color=black!40,
          line width=1pt]
    (0,0) -- +(\ulwidth-1pt,0);}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.

```
\ulposdef{\ulbgdD}{%
  \mbox{%
    \color{black!30}%
    \rule[-.8ex]{\ulwidth}{13pt}}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.

```
\ulposdef{\ulbgdE}[xoffset=.1em]{%
  \mbox{%
    \color{black!30}%
    \rule[-.8ex]{\ulwidth}{13pt}}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.

```
\ulposdef{\uldash}{%
  \makebox[\ulwidth]{%
    \color{blue}%
    \xleaders\hbox to.27em
      {\hss\rule[-.8ex]{.18em}{.5pt}\hss}%
    \hfill}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.

```
\ulposdef{\uldot}{%
 \mbox{%
   \raisebox{-.85ex}{%
     \xleaders\hbox to.2em
       {\hss\footnotesize.\hss}\hskip\ulwidth}}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.

```
\ulposdef{\ulflag}{%
  \mbox{%
    \color{red}\rule[-.85ex]{.25\ulwidth}{1.5pt}%
    \color{yellow}\rule[-.85ex]{.5\ulwidth}{1.5pt}%
    \color{red}\rule[-.85ex]{.25\ulwidth}{1.5pt}}}
```

A single word. Now a few words. Longer: this text spans several lines, so that you can see the behaviour of soulpos where there are line breaks.
