The \texttt{longfbox} package

Breakable boxes with standard CSS attributes.

Daan Leijen
2015-12-05

Contents

1. Introduction .................................................. 1
2. Overview ..................................................... 2
3. Styling with CSS .............................................. 4
   3.1. Borders ................................................. 5
   3.1.1. Border style ....................................... 5
   3.1.2. Border width ..................................... 7
   3.1.3. Border color ..................................... 7
   3.1.4. Border corner radius ............................. 9
   3.2. Alignment ............................................. 10
   3.3. Background ........................................... 14
   3.4. Padding and margins ................................. 15
   3.5. Breaking content .................................... 16
   3.6. Height and width .................................... 17
   3.7. Predefined styles .................................. 17
4. Advanced topics ............................................. 18
   4.1. Hooks and rendering ................................. 18
   4.2. Internal attributes .................................. 20
   4.3. Customizing border dashes and dots ............. 20

1. Introduction

The \texttt{longfbox} package provides framed boxes that can be customized using standard CSS attributes. It was written to support precise rendering of Madoko documents in \LaTeX. Notable features are:

- Specify the boxes using standard CSS attributes like \texttt{border-style=dashed} or \texttt{border-top-left-radius=10pt}. Almost all of the CSS 2.0 attributes with regard to borders, background, padding, and margins are supported.
- Fast and portable: only uses the standard \LaTeX \texttt{picture} environment for drawing and does not depend on loading big packages like \texttt{tikz} or
\texttt{pstricks}, which makes running \LaTeX\ much faster.

- Supports breakable boxes that span multiple columns or pages. This package builds on the \texttt{longbox} package to break boxes over multiple pages.
- Much care has been put in precise rendering with proper baseline alignment and no spurious extra whitespace.
- In contrast to larger packages like \texttt{mdframed} or \texttt{tcolorbox}, the \texttt{longfbox} package does not have more extensive features like frame titles, middle lines, skins, etc. The focus of this package is on rendering boxes well with full CSS support where every corner and side can be styled separately with good looking transitions, where we only depend on the standard \texttt{picture} environment.

2. Overview

There are two ways to create a framed box, the environment \texttt{longfbox} and the command \texttt{lfbox}.

\begin{verbatim}
\lfbox[\langle options \rangle]{\langle content \rangle}
\end{verbatim}

The \texttt{\langle options \rangle} are optional and specify CSS attributes. Just like the regular \texttt{\fbox} command, the \texttt{\lfbox} command sets the content in a horizontal box and cannot break the content over multiple lines (but it can contain other boxes like a \texttt{\parbox} or \texttt{minipage} environment.). The default width of an \texttt{\lfbox} is the natural width of its content. This corresponds to a CSS inline element.

Here is an \texttt{\lfbox{\texttt{\langle inline \rangle}}} box, just like an \texttt{\fbox{fbox}}.

\begin{verbatim}
Here is an \texttt{\fbox{\langle inline \rangle}} box, just like an \texttt{\fbox{}}}.
\end{verbatim}

\begin{verbatim}
begin{longfbox}[\langle options \rangle]{\langle content \rangle}end{longfbox}
\end{verbatim}

The \texttt{\begin{longfbox}[\langle options \rangle]{\langle content \rangle}end{longfbox}} environment sets the content in a long vertical box and can break content over multiple columns or pages. The default width is the current \texttt{\linewidth}. This corresponds to a CSS block element.
The `longfbox` can contain much longer content and will by default be as wide as the current line width.

\begin{longfbox}
\end{longfbox}

The `longfbox` can contain much longer content and will by default be as wide as the current line width.

\newfboxstyle{⟨name⟩}{⟨options⟩}

Defines a new style that can be used to specify commonly used options. For example, the package defines:

\newfboxstyle{tight}{padding=0pt,margin=0pt,baseline-skip=false}

The new style `tight` can now be used to render a tight box:

Here is a `\lfbox[tight]` box.

Here is a `tight` box.

The `\lfbox` and `longfbox` environment can replace many commands in \LaTeX through the rich CSS interface. In particular:

- `framed`, `\framebox`, `\fbox`: through the `text-align` and `width` attributes.
- `minipage`, `\parbox`: through the `width`, `text-align`, `vertical-align` and `baseline` attributes.
- `\makebox`: through `border-style=none` and the `text-align` and `width` attributes.
- `\raisebox`: through the `raise` attribute.
- `\colorbox`, `\fcolorbox`: through the `background-color` attribute.
- `\doublebox`, `\ovalbox`: through the `double` border style, and the `border-radius` attribute.
- `\shadowbox`: soon :-)

\fboxset{⟨options⟩}

Options can be set for the current scope through this declaration. For example, to make all borders rounded and red by default, use:
Figure 1. Attributes of the \texttt{longfbox} are modeled after the corresponding CSS attributes. Borders are drawn using the standard \texttt{picture} environment.

\begin{verbatim}
\fboxset{rounded,border-color=red}\
Here is a \texttt{lfbox}\{rounded\} box.
Here is a \underline{rounded} box.
\end{verbatim}

3. Styling with CSS

In the options you can specify standard CSS attributes to style your boxes. Figure 1 shows the basic CSS attributes that determine how a box is rendered. If you have used CSS boxes before specifying a box in LATEX is straightforward. Here is an example with an \texttt{lfbox}:

\begin{verbatim}
Here is a
\lfbox[
border-width=0.8pt,
border-left-color=red,
border-style=dotted,
padding={0.2ex,0.4ex}, %top\&bottom, right\&left]
\{fancy\} box.
Here is a \underline{fancy} box.
\end{verbatim}
Here is another example using the \texttt{longfbox} environment:

\begin{longfbox}[
  margin-right=6em,
  padding=1.5em,
  background-color=floralwhite,
  background-clip=padding-box,
  border-width=2pt,
  border-radius=15pt,
  border-top-left-radius=30pt,
  border-left-width=8pt,
  border-left-color=teal,
  border-right-style=double,
]
  A \texttt{longfbox} example. A longfbox can contain much longer content and will by default be as wide as the current line width.
\end{longfbox}

3.1. Borders

The \texttt{border} options determine how the frame of the box is rendered.

3.1.1. Border style

Each border can have an individual style:

\begin{verbatim}
  border-style=⟨style sides⟩
  border-top-style=⟨style⟩ (=solid)
  border-right-style=⟨style⟩ (=solid)
  border-bottom-style=⟨style⟩ (=solid)
  border-left-style=⟨style⟩ (=solid)
\end{verbatim}

where all CSS styles are supported:

\begin{verbatim}
  ⟨style⟩ = none | hidden | solid | dotted | dashed | double
\end{verbatim}
The <style sides> value can take 1 to 4 style arguments just like in CSS:

\[
<attr\;sides> = <attr>
| {{<top-bottom\;attr>, <left-right\;attr>}}
| {{<top-attr>, <left-right\;attr>, <bottom-attr>}}
| {{<top-attr>, <right-attr>, <bottom-attr>, <left-attr>}}
\]

Here are some examples of each CSS style:

\[
\%box{border-style=solid}{\{solid\}},
\%box{border-style=dashed}{\{dashed\}},
\%box{border-style=dotted}{\{dotted\}},
\] 
\[
\%box{border-style=double, border-width=2pt}{\{double\}},
\] 
\[
\%box{border-style={solid, none, dashed, none}}{\{various\}}.
\]

Note, for a dotted border, it is often nicer to use the dotted style (Section 3.7) since it makes the dots a bit larger, e.g.

\texttt{Compare \%box\{dotted\}} versus \%box\{border-style=dotted\}.

\texttt{Compare \ldots versus \ldots}.

The final CSS styles darken sides of the border to give a 3D effect:

\[
\%box{border-style=inset, border-color=red, border-width=3pt}{\{inset\}},
\] 
\[
\%box{border-style=outset, border-color=red, border-width=3pt}{\{outset\}},
\] 
\[
\%box{border-style=groove, border-color=teal, border-width=4pt}{\{\texttt{groove}\}},
\] 
\[
\%box{border-style=ridge, border-color=teal, border-width=4pt}{\{\texttt{ridge}\}}.
\]

The darkness can be controlled using the border-dark-mix attribute (Section 3.1.3).

Beyond CSS, there are also styles for the border top and bottom when a box is broken over multiple pages (see Section 3.5):

\begin{verbatim}
border-break-style=\{style\; break-sides\}
border-break-top-style=\{style\} (=none)
\end{verbatim}
\texttt{border-break-bottom-style=\{style\} (=none)}

where \langle style \ text \ break-sides \rangle \ takes \ one \ or \ two \ arguments:

\[
\langle \text{attr \ break-sides} \rangle = \langle \text{attr} \rangle \\
\text{break-top-attr}, \text{break-bottom-attr}\}
\]

See Figure 2 in Section 3.5 for more information.

3.1.2. Border width

\[
\text{border-width} = \langle \text{width \ sides} \rangle \\
\text{border-top-width} = \langle \text{dimen} \rangle (=\text{\fboxrule}) \\
\text{border-right-width} = \langle \text{dimen} \rangle (=\text{\fboxrule}) \\
\text{border-bottom-width} = \langle \text{dimen} \rangle (=\text{\fboxrule}) \\
\text{border-left-width} = \langle \text{dimen} \rangle (=\text{\fboxrule})
\]

This sets the width of each border. By default the \texttt{\fboxrule} width is used which is normally \texttt{0.4pt}.

\begin{verbatim}
A \fbox[border-width=3pt,border-left-color=red]{thick} border, and
\fbox[border-top-width=0pt,border-bottom-width=1pt]{varied}.
A \cellcolor{red}[thick] border, and \cellcolor{black}[varied]
\end{verbatim}

\[
\text{border-break-width} = \langle \text{width \ break-sides} \rangle \\
\text{border-break-top-width} = \langle \text{dimen} \rangle (=0pt) \\
\text{border-break-bottom-width} = \langle \text{dimen} \rangle (=0pt)
\]

These specify border width around page breaks. See Figure 2 in Section 3.5 for more information.

3.1.3. Border color

\[
\text{border-color} = \langle \text{color \ sides} \rangle \\
\text{border-top-color} = \langle \text{color} \rangle (=\text{black})
\]
Sets the color of each border.

\(\text{⟨color⟩} = \text{⟨color name⟩} \mid \text{⟨xcolor spec⟩} \mid \text{#RRGGBB} \mid {}\)

Besides the regular CSS attributes, colors for borders around a page break are set as:

\[
\begin{align*}
\text{border-break-color} &= \text{⟨color break-sides⟩} \\
\text{border-break-top-color} &= \text{⟨color⟩} (=\text{black}) \\
\text{border-break-bottom-color} &= \text{⟨color⟩} (=\text{black})
\end{align*}
\]

See Section 3.5 for further information.
The `border-dark-mix` is used to make a color darker and is used for the inset, outset, groove and ridge styles. This value is basically appended to the main color of the border. The default takes 70% of the main color and mixes in 30% black.

### 3.1.4. Border corner radius

```
border-radius=(radius corners)
border-top-left-radius=(radius) (=0pt)
border-top-right-radius=(radius) (=0pt)
border-bottom-left-radius=(radius) (=0pt)
border-bottom-right-radius=(radius) (=0pt)
```

The above attributes specify the corner radius to enable rounded corners.

```
(radius)=(dimen) | { (x-radius), (y-radius) }
```

A radius value is either a dimension, or a separate x- and y-radius for elliptical borders. Here are some examples:

```
\lfbox[border-radius=0.5ex]{rounded},
\lfbox[border-radius=100ex, background-color=floralwhite]{\strut elliptical},
\lfbox[border-radius=0.5em, height=1em, width=1em, tight, text-align=center, height-align=middle]{1}.
```

Here is an example of a `longfbox` with an elliptical corner:
There is little or no magic about them, except the ordinary everyday sort which helps them to disappear quietly and quickly when large stupid folk like you and me come blundering along, making a noise like elephants which they can hear a mile off.

3.2. Alignment

Much care has been taken to ensure proper alignment and preservation of the baseline. In our examples, we use the following definition where we enable show-markers so we can see the baseline and dimensions of the longbox:

\newcommand\alignbox[1][]{%
    \begin{longbox}[width=2em,height=4.25em,show-markers,
        baseline-skip=false,#1]
        foo, \\ bar, \\ gnu.
    \end{longbox}%
}

baseline= bottom | middle | top

The baseline attribute is not a CSS attribute, but we can use it to control the baseline of the content of a box. For example:
Aligning \texttt{\textbackslash alignbox\{baseline=bottom\}}, \texttt{\textbackslash alignbox\{baseline=middle\}}, \texttt{\textbackslash alignbox\{baseline=top\}} done.

Aligning \texttt{\textbackslash alignbox\{ baseline=bottom \}}, \texttt{\textbackslash alignbox\{ baseline=middle \}}, \texttt{\textbackslash alignbox\{ baseline=top \}} done.

Aligning \texttt{foo, bar, gnu.}, \texttt{foo, bar, gnu.}, \texttt{foo, bar, gnu.} done.

The \texttt{vertical-align} property aligns a box:

\texttt{vertical-align} = baseline | bottom | middle | top
| text-bottom | text-top | super | sub

The default \texttt{baseline} attribute aligns the baseline of the box with the baseline of the text. The \texttt{bottom} attribute aligns the bottom of the box with the bottom of the text, \texttt{middle} aligns the middle of the box with the middle of the text, and \texttt{top} aligns the top of the box with the top of the text.

Here are the various attributes in action (compare this to the examples for different baselines):

Aligning \texttt{\textbackslash alignbox\{vertical-align=bottom\}}, \texttt{\textbackslash alignbox\{vertical-align=middle\}}, \texttt{\textbackslash alignbox\{vertical-align=top\}} done.

Aligning \texttt{\textbackslash alignbox\{vertical-align=bottom\}}, \texttt{\textbackslash alignbox\{vertical-align=middle\}}, \texttt{\textbackslash alignbox\{vertical-align=top\}} done.

Finally, the \texttt{super} and \texttt{sub} attributes align the baseline of the box with the baseline of super- and sub-scripts:
Unfortunately, in LATEX we cannot determine the height or depth of the current text line easily so these values are fixed at 0.7\baselineskip and 0.3\baselineskip. The text-bottom and text-top are equal to bottom and top for that reason.

\texttt{raise=(dimen)}

In CSS it is also possible to assign a length to the \texttt{vertical-align} attribute. In LATEX we use the \texttt{raise} attribute instead which raises an aligned box by the assigned value.

\texttt{height-align= top | middle | bottom}

This is another non-CSS attribute (but probably one of the most desired :-)): it specifies the vertical alignment of the content inside the box. This attribute only has an effect for boxes where the \texttt{height} is specified. This attribute keeps the baseline unchanged and does not influence the vertical alignment directly.
Aligning `\alignbox{height-align=bottom}`, `\alignbox{height-align=middle}`, `\alignbox{height-align=top}`, `\alignbox{height-align=middle,baseline=top}`, done.

Aligning foo, bar, gnu. done.

text-align= default | left | center | right | justify

Specifies the alignment of the content horizontally. The default is left for a horizontal `\lfbox` and justify for a `longfbox` environment.

Here is `\lfbox[width=6em,text-align=center]{centered} text.`

And vertical `\alignbox{text-align=center,baseline=middle,width=4em}` too.

Here is centered text.

And vertical foo, bar, gnu. too.

Just like with `\makebox` we can use a narrow width to let the text overlay to the left or right:

`\lfbox[width=1.7em,text-align=right]{bummer, too long}.`
3.3. Background

\begin{verbatim}
background-color=color \{=\{} \nlbackground-clip=border-box | padding-box | content-box
\end{verbatim}

Specifies the background color. The background-clip attribute determines if the background color spans all the way out to the border (default), just the padding box, or only the content box itself.

Going beyond CSS, we can also specify the colors for the background at the border and padding separately.

\begin{verbatim}
A \lfbox[rounded, background-color=teal!30, background-padding-color=floralwhite, background-border-color=gray!50, border-style=double, border-width=3pt] too colorful box.
\end{verbatim}
### 3.4. Padding and margins

\begin{longfbox}[width=6.8em]
Here is a longer paragraph with a \lfbox[margin={1em,0.5em,1em},padding=1ex,show-markers]{box} inside with margins.
\end{longfbox}

The padding is the space between the content and the border of a box. By default the padding is equal to the \fboxsep value.

\begin{longfbox}[width=6.8em]
Here is a longer paragraph with a \lfbox{box} inside with margins.
\end{longfbox}

The margin is the transparent area outside the borders. In contrast to the CSS attribute, the margins do not overlap. Please use the standard \addvspace command for adding overlapping top- and bottom-margins.

\begin{longfbox}[width=6.8em]
Here is a longer paragraph with a \lfbox{box} inside with margins.
\end{longfbox}

padding\{padding sides\}
padding-top\{dimen\} (=fboxsep)
padding-right\{dimen\} (=fboxsep)
padding-bottom\{dimen\} (=fboxsep)
padding-left\{dimen\} (=fboxsep)

margin\{margin sides\}
margin-top\{dimen\} (=0pt)
margin-right\{dimen\} (=0pt)
margin-bottom\{dimen\} (=0pt)
margin-left\{dimen\} (=0pt)

padding-break-top\{dimen\} (=0.5em)
padding-break-bottom\{dimen\} (=0.5em)
margin-break-top\{dimen\} (=0pt)
margin-break-bottom\{dimen\} (=0pt)
“Halt!” cried Gandalf, who appeared suddenly, and stood alone, with arms uplifted, between the advancing dwarves and the ranks awaiting them. “Halt!” he called in a voice like thunder, and his staff blazed forth with a flash like the lightning. “Dread has come upon you all! Alas! it has come more swiftly than I guessed.”

Figure 2. A box that has been broken in three parts. The border style and the padding and margin of each break can be set separately. Use the `breakable` option to make boxes breakable over page boundaries. In the above example we also used the `breakat={75pt}` option to break at specific heights, and the `background-clip=content-box` to limit the background color to the content only.

These attributes determine the margin and padding around a page break. By default, there is some padding set around a break such that the vertical borders of a box ‘stick out’ a bit giving the reader a hint that the box continues at the next page (see Figure 2).

3.5. Breaking content

```latex
breakable\langle bool \rangle \ (=false)
```

When the option `breakable` is present, a `longfbox` environment can be broken at page boundaries. All the borders, margins, padding, etc. are preserved over the page break. Moreover, the border, padding, and margin for each break at the top and bottom can be set separately, as shown in in Figure 2.

```latex
extra-split\langle dimen \rangle \ (=1.5\baselineskip)
```

(Advanced) This option determines the minimal height of a box that is broken.
If the height would be less the box is not broken and moved to the next page. This prevents for example to have just the top border and padding on one page followed by content on the next page.

breakat={⟨dimen1⟩,...,⟨dimen_n⟩}

(Advanced) This option lets you break a box manually at specific heights. The breakable option must be set for this to work. Usually a box is broken to fit the space available on the page, but when breakat is given, the box breaks at ⟨dimen1⟩. After the break, the first dimension is removed from the list and the box is broken again at ⟨dimen2⟩ until only one element (⟨dimen_n⟩) is left. After that the box is repeatedly broken at ⟨dimen_n⟩ until all content is processed. For example, in Figure 2 we used breakat={75pt} to break the box at three 75pt heights. Note that when a box breaks, it chooses the largest height that is less than 75pt where the content can be broken nicely, i.e. a box never breaks in the middle of a line for example.

The breakat can also be used to break a box in a multicols environment.

3.6. Height and width

height=⟨dimen⟩
width=⟨dimen⟩

Specifies the height or width of the inner content box, see Figure 1. By default, the width of horizontal \lfbox is the natural width of its content. In contrast, the default width of a vertical longfbox environment is the current \linewidth. The default height is always the natural height of the content. When the height is fixed, you can use height-align to align the content in the available area.

outer-height=⟨dimen⟩
outer-width=⟨dimen⟩

For convenience, there are also these non-CSS attributes that let you specify the width and height of the entire box including the padding, border, and margins. Internally translated to the height or width before processing the box.

3.7. Predefined styles

tight={padding=0pt,margin=0pt,padding-break=0pt}
rounded={border-radius=1ex}
Some predefined convenient styles. **tight** puts the box tightly around the content. The **rounded** style uses a border radius that looks nice relative to the font size, and **dotted** uses a slightly larger border width which looks better with dots.

A \(\texttt{lfbox[tight]{tight}}, \texttt{lfbox[dotted]{dotted}}, \texttt{lfbox[rounded]{rounded}}\) box.

4. Advanced topics

4.1. Hooks and rendering

**render**=default | plain | picture

There are two ways to render a box border: using just plain \LaTeX{} and the \LaTeX{} picture environment. The plain renderer just uses \texttt{\textbackslash hrule} etc. and can only render simple boxes with solid or dashed styles without rounded corners. The default will pick the plain renderer if a box is simple, and use the picture renderer otherwise. This is mostly to make the rendering more efficient but there should be no other difference.

\texttt{\textbackslash lfbox[insert-before=\{"bgroup\textitshape\}], insert-after=\{\textbackslash egroup\}}\texttt{\{quoted\}} box.

A “quoted” box.

**render-insert-before**={value}
**render-insert-after**={value}

These are hooks to insert content just before or after the content box.
These are hooks to insert content just before or after the border-box is rendered.

\newcommand\makecircle{\setlength\unitlength{1pt}\begin{picture}(0,0)% make it occupy no space \color{red}\put(0,0){\circle{5}}\end{picture}}
\lfbox[render-insert-before=\makecircle]{funny}.

\newcommand\crossout{\color{red}\linethickness{1pt}\roundcap\moveto(0,0)\lineto(\optionunit{/fbox/@border-box-width},\optionunit{/fbox/@border-box-height})\strokepath}
\lfbox[rounded,picture-insert-after=\crossout]{crossed out}.

\newcommand\crossout{\color{red}\linethickness{1pt}\roundcap\moveto(0,0)\lineto(\optionunit{/fbox/@border-box-width},\optionunit{/fbox/@border-box-height})\strokepath}
\lfbox[rounded,picture-insert-after=\crossout]{crossed out}.

These are hooks to insert content just before or after the border-box is rendered.

\newcommand\makecircle{\setlength\unitlength{1pt}\begin{picture}(0,0)% make it occupy no space \color{red}\put(0,0){\circle{5}}\end{picture}}
\lfbox[render-insert-before=\makecircle]{funny}.

\newcommand\crossout{\color{red}\linethickness{1pt}\roundcap\moveto(0,0)\lineto(\optionunit{/fbox/@border-box-width},\optionunit{/fbox/@border-box-height})\strokepath}
\lfbox[rounded,picture-insert-after=\crossout]{crossed out}.

\newcommand\crossout{\color{red}\linethickness{1pt}\roundcap\moveto(0,0)\lineto(\optionunit{/fbox/@border-box-width},\optionunit{/fbox/@border-box-height})\strokepath}
\lfbox[rounded,picture-insert-after=\crossout]{crossed out}.

These are hooks to insert content just before or after the border-box is rendered.

\newcommand\makecircle{\setlength\unitlength{1pt}\begin{picture}(0,0)% make it occupy no space \color{red}\put(0,0){\circle{5}}\end{picture}}
\lfbox[render-insert-before=\makecircle]{funny}.

\newcommand\crossout{\color{red}\linethickness{1pt}\roundcap\moveto(0,0)\lineto(\optionunit{/fbox/@border-box-width},\optionunit{/fbox/@border-box-height})\strokepath}
\lfbox[rounded,picture-insert-after=\crossout]{crossed out}.

These are hooks to insert content just before or after the border-box is rendered.
4.2. Internal attributes

\texttt{baseline-skip} = \langle \text{bool} \rangle \ (\ = \text{true})

Usually, the library adds a small vertical skip in front of a vertical \texttt{longfbox} such that its content aligns to a \texttt{\baselineskip}. Similarly, it adds such skip just after the box. This generally makes text align better, especially on a double column layout. Setting it to \texttt{false} disables such vertical spacing.

\texttt{show-markers} = \langle \text{bool} \rangle \quad \texttt{marker-color} = \langle \text{color} \rangle \ (\ = \text{gray})
\texttt{marker-width} = \langle \text{dimen} \rangle \ (\ = \text{0.1pt})

If \texttt{show-markers} is \texttt{true} it will show the padding, baseline, margin and border markers using \texttt{marker-color} and \texttt{marker-width}.

\texttt{eject} = \langle \text{value} \rangle \ (\ = \{\text{\protect\vfill}\protect\eject\})

After breaking a box, this command is issued which normally ends the current page. The \texttt{breakat} command sets this to empty so no page break occurs in that case.

\texttt{debug} = \langle \text{bool} \rangle \ (\ = \text{false})
\texttt{verbose} = \langle \text{bool} \rangle \ (\ = \text{false})

Enable debug and verbose log messages.

4.3. Customizing border dashes and dots

The following attributes are used to render dashes and dots for the \texttt{dashed} and \texttt{dotted} border styles.

\texttt{border-dash} = \langle \text{dash sides} \rangle
\texttt{border-top-dash} = \langle \text{dimen} \rangle \ (\ = \text{0.7ex})
\texttt{border-right-dash} = \langle \text{dimen} \rangle \ (\ = \text{0.7ex})
\texttt{border-bottom-dash} = \langle \text{dimen} \rangle \ (\ = \text{0.7ex})
\texttt{border-left-dash} = \langle \text{dimen} \rangle \ (\ = \text{0.7ex})
\texttt{border-break-top-dash} = \langle \text{dimen} \rangle \ (\ = \text{\option{border-top-dash}})
\texttt{border-break-bottom-dash} = \langle \text{dimen} \rangle \ (\ = \text{\option{border-bottom-dash}})
border-dashskip=$(dashskip sides)$
border-top-dashskip=$(dimen) (=0.7\option{border-top-dash})
border-right-dashskip=$(dimen) (=0.7\option{border-right-dash})
border-bottom-dashskip=$(dimen) (=0.7\option{border-bottom-dash})
border-left-dashskip=$(dimen) (=0.7\option{border-left-dash})
border-break-top-dashskip=$(dimen) (=\option{border-top-dashskip})
border-break-bottom-dashskip=$(dimen) (=\option{border-bottom-dashskip})

border-dotskip=$(dotskip sides)$
border-top-dotskip=$(dimen) (=0.7\option{border-top-width})
border-right-dotskip=$(dimen) (=0.7\option{border-right-width})
border-bottom-dotskip=$(dimen) (=0.7\option{border-bottom-width})
border-left-dotskip=$(dimen) (=0.7\option{border-left-width})
border-break-top-dotskip=$(dimen) (=\option{border-top-dotskip})
border-break-bottom-dotskip=$(dimen) (=\option{border-bottom-dotskip})