border.tex

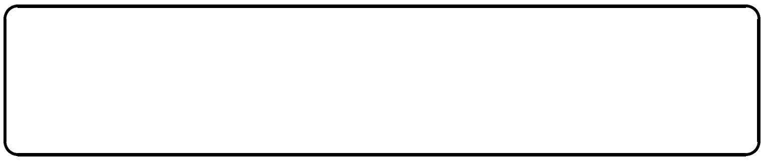
Macros to typeset borders.
Version: 1.0 (21 October 1991) (doc: 23 Oct 1991)

The macros described in this manual section are used to typeset borders, such as the border around this paragraph. These borders are built from different "characters": in this example from quarter circles for the corners and straight line segments for the edges, but, given an appropriate font, other borders can also be typeset. (Below we explain how a border font must be designed in order to be useful with these macros.) Before any of the macros described below can be used, you must `\input` border.tex.*

There are several ways of typesetting borders, depending on the characters that form the border. Macros such as `\border` create a vbox of specified height and width (the depth is always zero), containing only the border. Typesetting text within a border is done by creating a box with the text and superimposing it on the box with the border, as explained later. The types of border we will now describe differ in the way they compose characters to build a border; the choice of the characters is determined by another macro (`\borderelm`). The border around the previous paragraph was typeset with `\border`. It is built from eight different characters: '╭', '─', and '╮' for the top; '│' and '│' for the center; and '╰', '─', and '╯' for the bottom. `\border` takes two arguments, the height and the width of the border.

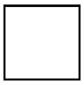
```
\border{2cm}{10cm}
```

creates a border like this:

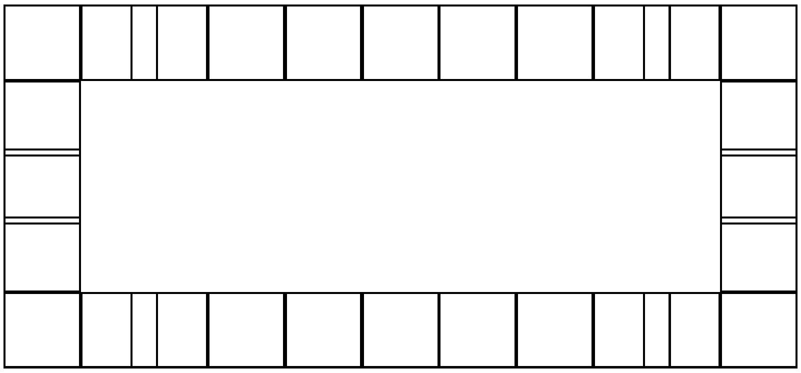


It is a vbox with height 2 cm, depth 0, and width 10 cm.

In order to create a border of the exact specified dimensions, some characters may need to overlap. Suppose we built a border from the following character, a 1 cm by 1 cm box:



The following is the border created by `\border{4.8cm}{10.5cm}`:



As you see, some of the boxes overlap, in order to get the specified dimensions. This overlap is always between the edge elements, there is never overlap with the corner elements (unless the size is too small). In

* Border.tex also works with LaTeX.

fact, the overlap is always with the edge elements that are closest to the corners. (`\border` always sets at least one edge element for each edge.)

`\border` works fine if the edge elements can overlap, which basically restricts them to be "rule like." There is an alternative to `\border`, called `\xborder`. It is used in the same way as `\border`, but the exact size is obtained by putting extra space between the elements, rather than overlapping them. Suppose we build a border with the same corner elements as before, but with the edge built from "bullets" rather than straight line segments. The top element consists of a bullet centered in a 1 cm wide box. Here are the top lines obtained from `\border{...}{4.5cm}` and `\xborder{...}{4.5cm}`, respectively.

In case of `\xborder`, the proper vertical size is also obtained by spacing.

Some border characters, however, need to abut in order to form a border. For instance, suppose we want to make a "cloud like" border, where the top corners and top edge elements are all '⌒.' In this case, `\border{...}{1.6cm}` and `\xborder{...}{1.8cm}` give

 and 

Clearly, neither solution works very well. For this case there are two other macros, `\minusborder` and `\plusborder`. They take the same arguments as `\border`, but produce a box of rounded dimensions rather than of the exact size. The border is built by abutting the elements. `\minusborder` rounds the size down, whereas `\plusborder` rounds the size up. `\minusborder{...}{1.8cm}` and `\plusborder{...}{1.8cm}` give

 and 

The above macros all generate empty borders. Typically, one wants to put some text in the border. There is one macro to make that easy, `\setborder`. It is called as follows:

`\setborder[`*width*, *hspace*, *vspace*`]\border{ ... }`

where the ... stands for the text that must be boxed. In this case, `\border` will be used to create the border, but you can also use `\xborder` etc. The border will be *width* wide (or a rounded value); the height of the border is determined by the amount of text. The text itself is first broken in paragraphs and lines, as usual in TeX. The width of the lines is $width - 2 \cdot hspace$. The text is than put in the border with *vspace* space at the top and at the bottom, and *hspace* space at the left and at the right hand sides. For instance, the first paragraph of this manual section was set with

```
\setborder[\hsize,1em,\smallskipamount]\border
{\noindent The macros described ...
 ....
}
```

The *width* and *hspace* must be dimensions; *vspace* can be a dimension or a glue amount (without `\vskip`).

With `\setborder`, the text between braces, `{...}`, is read in the same way as when you write `\vbox {...}`. (You can use `\bgroup` and `\egroup` instead of the braces.) The text is *not* an argument of `\setborder`, which is important if it contains catcode changes. You can change the paragraph shape, or have more than one paragraph of text.

For instance, the text in this paragraph is set with some
extra `\leftskip` and `\rightskip` glue, which takes
care of centering every line. This was done as follows:

```
\setborder[10cm, 1cm, 4mm]\border{
\parindent=0pt \parfillskip=0pt
\leftskip=0pt plus 1cm minus 5mm \rightskip=0pt plus 1cm minus 5mm
```

```
For instance, the text in this paragraph ...
...
}
```

If `\setborder` does not suit your purpose, you can write your own equivalent. Generally, to put text in a border, you first generate a box containing the border of the appropriate size. Then you make a box containing this border box plus the text superimposed on it. Superimposing is done by either "backspacing" using negative kerns, or by making a zero width or height border box, in a similar way as `\llap` makes a zero width box. Usually, you have to use several nested levels of boxing to get the desired result.

We have described the different ways of composing border characters, but not yet how to specify which characters are used. Whenever a border is typeset, it is assumed that a macro `\borderelm#1` exists. This macro is called with as argument a number in the range $0, \ldots, 7$; it should return the appropriate border element, where the elements are numbered as follows:

```
0 1 1 1 1 1 1 1 1 1 2
3                   4
3                   4
5 6 6 6 6 6 6 6 6 6 7
```

`\borderelm` is called within an hbox; it should typeset a character, a box, or an vrule. (Since the macro is expanded within an hbox, you should be careful not to put extra spaces in it; doing that may cause strange gaps in the border.)

A convenient way of writing `\borderelm` is with an `\ifcase` construction. For instance, assume that in font `\qc` the characters `a`, `b`, `c`, etc. correspond to '╭', '—', '╮', etc. Then you could write `\borderelm` as follows:

```
\def\borderelm#1%
   {{\qc \ifcase#1
            a\or b\or c\or d\or e\or f\or g\or h% no extra space
         \fi
   }}
```

In fact, a macro like this is already defined in border.tex. It is called `\roundcorners`, so that you can say `\let\borderelm=\roundcorners` to get the borders with rounded corners. The corner elements come from font `\manfnt`, which corresponds to the external font 'manfnt.' (The corner elements are described on page 389 of The TEXbook.) `\roundcorners` scales, which means that you can say, for instance,

```
\font\manfnt=manfnt scaled 2074
```

to get larger corners — provided you have the appropriately scaled font file (2074 corresponds to `\magstep4`). If you do not want to load the manfnt font (e.g., because your installation does not have it), you should define `\let\manfnt=\relax` before you `\input` border.tex.

Hopefully a variety of border fonts will come available in the future. But even with just the quarter circles you can already vary your borders quite a bit. We already saw a border with half circles before, it is an easy exercise to define a complete border that way. The following definition defines a fancier corner:

```
\setbox0=\hbox{\roundcorners2}
\vbox{\offinterlineskip
      \hbox{\copy0\roundcorners0\copy0}
      \hbox to 3\wd0{\hfil\roundcorners7}
      \hbox to 3\wd0{\hfil\copy0}
     }%
```

It defines the "number 2" (or number 5) corner: ; the other corners can be defined similarly.

If you want, you can omit the corners from `\borderelm` (i.e., write `\borderelm` to return nothing for arguments 0, 2, 5, and 7). But if you do not want any edge elements, you should still put something with width (or height) for the top/bottom (or left/right) edges; for instance, `\hbox to10pt{}` is a valid top element. Border.tex uses several box and dimen registers. Rather than reserving private registers, which would make them unavailable for other purposes, border.tex uses `\box0`, `\box2`, `\box4`, `\dimen0`, `\dimen2`, and `\count2`. Normally, this does not cause problems, since they are always restored to their old value. The only time you should take this into account, is in `\borderelm`: this macro should not depend on any global values of these registers, nor should it make any lasting changes to them.

If you are a font designer, and want to define a font that can be used with border.tex, you should understand something of how `\border` etc. build the borders. A border is built from three horizontal strips. The top and bottom strips consist of a corner element, a number of edge elements, and another corner element. These elements are aligned by their baselines, and the width of the border(-strip) is determined by the width of the bounding boxes. Therefore, the "ink" in the border elements should be close to the outside edges of the bounding boxes; otherwise there will not be a proper relation between the actual size of the border and the size of the visible border. The center strip is built from an edge, white space (glue), and another edge. The amount of white space is chosen so that the width of the center strip is the width of the top strip. The whole border is built by vertically aligning the top strip, a number of center strips, and the bottom strip. This is done by aligning the left hand edges of the horizontal strips. The vertical size of the border is again determined by the vertical size of the bounding boxes. You may notice that the quarter circle elements described on p. 389 of The TeXbook (used in `\roundcorners`), do not fit properly in their bounding boxes for the purpose of setting borders. In fact, `\roundcorners` must go to considerable effort to reposition the bounding boxes. If you can, avoid such problems with any new fonts you design.

There are two more variants of borders. First, to set only one horizontal strip of a border generated by `\border`, you can use `\hborder`. It takes four argument: the width, and the numbers of the left element, middle element, and right element. E.g., `\border{4cm}012` generates the top of the border, whereas `\border{4cm}567` generates the bottom. There are also `\hxborder`, `\hminusborder`, and `\hplusborder`. However, there are no vertical equivalents.

Finally, there are macros `\minusgray` and `\plusgray`. They behave like their border counterparts, except that the center of the border is filled with an extra character (`\borderelm8`) rather than with white space. For use with these macros, `\borderelm` must return characters as follows:

0	1	1	1	1	1	1	1	1	1	2
3	8	8	8	8	8	8	8	8	8	4
3	8	8	8	8	8	8	8	8	8	4
5	6	6	6	6	6	6	6	6	6	7

This is useful if you have characters that contain a "gray" pattern (a "stipple" or "Ben Day" pattern). Although a font with a gray pattern should not be hard to define, to the author's knowledge there is currently no such font in the public domain. These gray macros can also be used with `\setborder`.