Good-looking line breaks with the `listings` package

Burkart Lingner

April 24, 2011

What to do

If you want to typeset source code in LaTeX you'll probably end up using the `listings` package. The output looks good and you have quite a few options to customize it to your taste. Some of those options deal with how long lines are wrapped. Personally, I like them to look this way:

```
1 if (line.length() > ↙
      ↳ line.max_length())
2 {
3     line.wrap();
4 }
```

Here, automatically wrapped lines are indicated in a number of ways, namely

- pictographs (↙ and ↳),
- indentation of the wrapped lines, and
- a gap in the line numbers.

How to do it

In order to achieve this behavior you have to set a few parameters for the `listings` package:

```
\lstset{prebreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\rhookswarrow}}}
\lstset{postbreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\rcurvearrowse\space}}}
\lstset{breaklines=true, breakatwhitespace=true}
\lstset{numbers=left, numberstyle=\scriptsize}
```

The parameters *prebreak* and *postbreak* are responsible for the pictographs. Unfortunately the complicated construct using **\raisebox** and **\ensuremath** is required to make it work. The **\space** at the end of *postbreak* adds, well, a space between the pictograph and the continued source code.

Setting *breaklines*=true enables automatic line wrapping whereas *breakatwhitespace*=true ensures that line breaks don't occur inside expressions like variable names.

The last line enables line numbering using *numbers*=left and specifies their appearance as small numbers with *numberstyle*=**\scriptsize**.

Indentation is enabled by default and set to 20 pt. If you want to change that, use i.e. *breakindent*=10pt.

Finally just one more thing: The commands for the pictographs are defined in the MnSymbol package, therefore you need to load it by writing **\usepackage**{MnSymbol} in the document's preamble.

Alternative pictographs

If you're unhappy with the pictographs I selected, the next two pages show a few alternatives.

\hookleftarrow and \hookrightarrow

No special packages required but looks different with `MnSymbol` loaded.

```
\lstset{prebreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\hookleftarrow}}}
\lstset{postbreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\hookrightarrow\space}}}
```

yields

```
if (line.length() > ↩
    ↪ line.max_length())
{
    line.wrap();
}
```

\rightharpoondown and \hookrightarrow

No special packages required but looks different with `MnSymbol` loaded.

```
\lstset{prebreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\rightharpoondown}}}
\lstset{postbreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\hookrightarrow\space}}}
```

yields

```
if (line.length() > ⇁
    ↪ line.max_length())
{
    line.wrap();
}
```

\swarrow and \hookrightarrow

No special packages required but looks different with `MnSymbol` loaded.

```
\lstset{prebreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\swarrow}}}
\lstset{postbreak=\raisebox{0ex}[0ex][0ex]
        {\ensuremath{\hookrightarrow\space}}}
```

yields

```
if (line.length() > ↙
    ↪ line.max_length())
{
    line.wrap();
}
```

\Righttorque and \Lefttorque

Requires package `marvosym`.

```
\lstset{prebreak=\Righttorque}
\lstset{postbreak=\Lefttorque}
```

yields

```
if (line.length() > ↻
    ↺ line.max_length())
{
    line.wrap();
}
```