Programlisták készítése a listings csomaggal

Virágh János

2024. október 14.

```
\tableofcontents    % tartalomjegyzék
\listoffigures      % ábrajegyzék
\listofprogramkod   % programjegyzék
```

A fönti parancsokkal legeneráltatjuk a dokumentumunkhoz tartozó különféle jegyzékeket. Figyeljük meg, hogy az alábbi példák közül melyik szerepel a második, és melyik a harmadik jegyzékben.

Tartalomjegyzék

Ábrák jegyzéke

Programok jegyzéke

1. Bevezetés

Az alábbi példák a `listings` csomag használatát mutatják be különféle programlisták elkészítésére. Hasznos lehet Tómács Tibor fóliáiról a vonatkozó részek átolvasása is. Először az `\lstset` parancs segítségével megadjuk a továbbiakhoz szükséges opciókat. Az alkalmazott parancs LaTeX forráskódja:

```
\lstset{% kezdete
    backgroundcolor=\color{Gray!30},       % háttérszín
    basicstyle=\scriptsize\ttfamily,       % az alapértelmezett betűméret, stílus
    keywordstyle=\bfseries\color{blue},    % a kulcsszavak stílusa
    commentstyle=\slshape\color{green},    % a megjegyzések stílusa
    stringstyle=\color{magenta},           % a sztringek stílusa
    breaklines,                            % a hosszú programsorok törhetők
    prebreak=\hbox{$\color{red}\Rdsh\ $},  % hosszú sorok törése előtti jel
    postbreak=\hbox{$\color{red}\Ldsh\ $}, % hosszú sorok törése utáni jel
    breakindent=10pt,                      % hosszú sorok törése utáni behúzás
    tabsize=4,
   %showspaces,                            % szóközök látható megjelenítése
   %showtabs,                              % tabulátor karakterek látható megjelenítése
    frame=none,                            % nincs keretezés, shadowbox, vagy trblTRBL részhalmaza
    numbers=left,                          % sorok számozása balról
    numberstyle=\scriptsize,               % hogy megegyezzen a kód betűméretével
    numbersep=1em,
    inputencoding=utf8,
    extendedchars=true,
    literate=%
    {á}{{\'{a}}}1
    {í}{{\'{i}}}1
    {ű}{{\H{u}}}1
    {ő}{{\H{o}}}1
    {ü}{{\"{u}}}1
    {ö}{{\"{o}}}1
    {ú}{{\'{u}}}1
    {ó}{{\'{o}}}1
    {é}{{\'{e}}}1
    {Á}{{\'{A}}}1
    {Í}{{\'{I}}}1
    {Ű}{{\H{U}}}1
    {Ő}{{\H{O}}}1
    {Ü}{{\"{U}}}1
    {Ö}{{\"{O}}}1
    {Ú}{{\"{U}}}1
    {Ó}{{\'{O}}}1
    {É}{{\'{E}}}1
```

2. Egyszerű (nem-úsztatott) programlisták

LaTeX programlista

Az első példában az alapértelmezetteken túl (lásd `lstset` parancs) opcióként a LaTeX nyelvet és a dupla keretezést választottuk.
A LaTeX forráskód:

```
\begin{lstlisting}[language={[LaTeX]TeX}, frame=TRBL]
% most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
Egy adott betű (vagy szó, mondat, paragrafus, stb.) megjelenítését sok tényező befolyásolhatja:
\begin{itemize}
\item Melyik betűtípust használjuk (alapértelmezettet,
vagy pl. a preambulumban csomaggal megadottat)
\item Ennek melyik változatát használjuk (álló, dőlt,
döntött, kiskapitális,stb.)
\item Milyen betűvastagságot használunk (vékonyított,
normál, félkövér, stb.)
% most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
\item Milyen betűméretet használunk (a \LaTeX{} tíz beépített mérete, vagy egyedi trükkökkel definiált
\end{itemize}
\end{lstlisting}
```

A szépen megformázott LaTeX kód:

```latex
% most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
Egy adott betű (vagy szó, mondat, paragrafus, stb.) megjelenítését sok tényező befolyásolhatja:
\begin{itemize}
    \item Melyik betűtípust használjuk (alapértelmezettet,
        vagy pl. a preambulumban csomaggal megadottat)
    \item Ennek melyik változatát használjuk (álló, dőlt,
        döntött, kiskapitális,stb.)
    \item Milyen betűvastagságot használunk (vékonyított,
        normál, félkövér, stb.)
    % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
    \item Milyen betűméretet használunk (a \LaTeX{} tíz beépített mérete, vagy egyedi trükkökkel definiált további méret)
\end{itemize}
```

Ha végre is hajtjuk a fönti LaTeX kódot, ezt látjuk a kiszedett szövegben:

Egy adott betű (vagy szó, mondat, paragrafus, stb.) megjelenítését sok tényező befolyásolhatja:

- Melyik betűtípust használjuk (alapértelmezettet, vagy pl. a preambulumban csomaggal megadottat)
- Ennek melyik változatát használjuk (álló, dőlt, döntött, kiskapitális,stb.)
- Milyen betűvastagságot használunk (vékonyított, normál, félkövér, stb.)
- Milyen betűméretet használunk (a LaTeX tíz beépített mérete, vagy egyedi trükkökkel definiált további méret)

Java programlista

Az alábbi példában opcióként a Java nyelvet és az árnyékolt keretezést választottuk.
A forrásprogram:

```
\begin{lstlisting}[language=Java, frame=shadowbox]
class HelloWorldApp {
    public static void main(String[] args) {
        //Display the string
        System.out.println("Hello World!");
    }
}
\end
\end{lstlisting}
```

A megformázott programkód:

```
1 class HelloWorldApp {
2     public static void main(String[] args) {
3         //Display the string
4         System.out.println("Hello␣World!");
5     }
6 }
7 \end
```

Az alábbiakban egy C++ demo programot látunk.
A forráskód:

```
%\lstset{language=C++}
\begin{lstlisting}[language=C++]
/*****************************************************************
* MODULE     : substitute.cpp
* DESCRIPTION: Converts LaTeX to TeXmacs
* COPYRIGHT  : (C) 2003  Joris van der Hoeven
*****************************************************************
* This software falls under the GNU general public license and comes WITHOUT
* ANY WARRANTY WHATSOEVER. See the file $TEXMACS_PATH/LICENSE for more details.
* If you don't have this file, write to the Free Software Foundation, Inc.,
* 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
*****************************************************************/
#include <stdio.h>
#include <iostream.h>
#define DATA_BEGIN   ((char) 2)
#define DATA_END     ((char) 5)
#define DATA_ESCAPE  ((char) 27)

int
main () {
cout << DATA_BEGIN << "verbatim:";
cout << "An interactive LaTeX -> TeXmacs translator\n";
```

```
cout << "Can also be used outside sessions: select a LaTeX expression\n";
cout << "and press C-F12\n";
cout << DATA_END;
fflush (stdout);

while (true) {
char buffer[100];
cin.getline (buffer, 100, '\n');
cout << DATA_BEGIN;
cout << "latex:$" << buffer << "$";
cout << DATA_END;
fflush (stdout);
}
return 0;
}
\end{lstlisting}
```

A megformázott példa:

```
1  /*****************************************************************
2  * MODULE     : substitute.cpp
3  * DESCRIPTION: Converts LaTeX to TeXmacs
4  * COPYRIGHT  : (C) 2003  Joris van der Hoeven
5  *****************************************************************
6  * This software falls under the GNU general public license and comes
     WITHOUT
7  * ANY WARRANTY WHATSOEVER. See the file $TEXMACS_PATH/LICENSE for more
     details.
8  * If you don't have this file, write to the Free Software Foundation,
     Inc.,
9  * 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
10 *****************************************************************/
11 #include <stdio.h>
12 #include <iostream.h>
13 #define DATA_BEGIN   ((char) 2)
14 #define DATA_END     ((char) 5)
15 #define DATA_ESCAPE  ((char) 27)
16
17 int
18 main () {
19     cout << DATA_BEGIN << "verbatim:";
20     cout << "An interactive LaTeX -> TeXmacs translator\n";
21     cout << "Can also be used outside sessions: select a LaTeX
         expression\n";
22     cout << "and press C-F12\n";
23     cout << DATA_END;
24     fflush (stdout);
25
26     while (true) {
27         char buffer[100];
28         cin.getline (buffer, 100, '\n');
29         cout << DATA_BEGIN;
30         cout << "latex:$" << buffer << "$";
31         cout << DATA_END;
32         fflush (stdout);
33     }
34     return 0;
35 }
```

Az alábbiakban egy Python demo programot látunk.
A forráskód:

```
\shadowbox{
\begin{lstlisting}[language=Python]
import numpy as np

def incmatrix(genl1,genl2):
m = len(genl1)
n = len(genl2)
M = None #to become the incidence matrix
VT = np.zeros((n*m,1), int)  #dummy variable

#compute the bitwise xor matrix
M1 = bitxormatrix(genl1)
M2 = np.triu(bitxormatrix(genl2),1)

for i in range(m-1):
for j in range(i+1, m):
[r,c] = np.where(M2 == M1[i,j])
for k in range(len(r)):
VT[(i)*n + r[k]] = 1;
VT[(i)*n + c[k]] = 1;
VT[(j)*n + r[k]] = 1;
VT[(j)*n + c[k]] = 1;

if M is None:
M = np.copy(VT)
else:
M = np.concatenate((M, VT), 1)

VT = np.zeros((n*m,1), int)

return M
\end{lstlisting}

}
```

A megformázott kód:

```
        import numpy as np

        def incmatrix(genl1,genl2):
        m = len(genl1)
        n = len(genl2)
        M = None #to become the incidence matrix
        VT = np.zeros((n*m,1), int)  #dummy variable

        #compute the bitwise xor matrix
        M1 = bitxormatrix(genl1)
        M2 = np.triu(bitxormatrix(genl2),1)

        for i in range(m-1):
        for j in range(i+1, m):
        [r,c] = np.where(M2 == M1[i,j])
        for k in range(len(r)):
        VT[(i)*n + r[k]] = 1;
        VT[(i)*n + c[k]] = 1;
        VT[(j)*n + r[k]] = 1;
        VT[(j)*n + c[k]] = 1;

        if M is None:
        M = np.copy(VT)
        else:
        M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

        return M
```

3. Úsztatott programlisták

Ilyen állatfajt alapból nem ismer a LaTeX. Bár betehetünk programlistát a beépített úsztatott környezetekbe (`table`, `figure`), ezzel előbb-utóbb gondok lesznek. Az alábbi „programkód-ábra” (`figure` úsztatott környezet) majd „valahol” megjelenik a szövegben – ahová a latex fordító be tudja szúrni. A megfelelő sor ugyan megjelenik az ábrák listájában, de praktikusabb lenne a kódpéldákról külön listát készíteni, ami így csak utólag, a `.lof` fájl „kézi faragásával” volna elérhető.

```
% most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
Egy adott betű (vagy szó, mondat,paragrafus, stb.) megjelenítését sok tényező befolyásolhatja:
\begin{itemize}
    \item Melyik betűtípust használjuk (alapértelmezettet,
        vagy pl. a preambulumban csomaggal megadottat)
    \item Ennek melyik változatát használjuk (álló, dőlt,
        döntött, kiskapitális,stb.)
    \item Milyen betűvastagságot használunk (vékonyított,
        normál, félkövér, stb.)
    % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
    \item Milyen betűméretet használunk (a \LaTeX{} tíz beépített mérete, vagy egyedi trükkökkel definiált további méret)
\end{itemize}
```

1. ábra. A LaTeX forráskód példa ábraként úsztatva

A változatosság kedvéért beillesztünk egy „igazi” ábrát is kedvenc házi oroszlánunkról.

2. ábra. A mi kis házi oroszlánunk

Megadjuk a Java kód úsztatott változatát is.

```
class HelloWorldApp {
    public static void main(String[] args) {
        //Display the string
        System.out.println("Hello World!");
    }
}
\end
```

3. ábra. A Java forráskód példa ábraként úsztatva

4. Új úsztatott környezet definiálása programlisták számára

A `newfloat` csomag segítségével definiálunk egy új programkod nevű úsztatott környezetet. A preambulumba(!) írjuk ezt:

```
\usepackage{caption}
\usepackage{newfloat}
\DeclareFloatingEnvironment[%
name=program,
listname={Programok jegyzéke},
fileext=lop,
placement=hp,
within=,
]
{programkod}
```

Megjegyzés. Ezt a konstrukciót már láttuk korábban a `newfloat-ex.tex` példafájlban, csak akkor az `altt` csomag felhasználásával adtuk meg a forráskódokat.
Ismét beillesztjük a LaTeX forráskód példát, csak most már az új `programkod` úsztatott környezetet használjuk.

```
% most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
Egy adott betű (vagy szó, mondat,paragrafus, stb.) megjelenítését sok tényező befolyásolhatja:
\begin{itemize}
    \item Melyik betűtípust használjuk (alapértelmezettet,
        vagy pl. a preambulumban csomaggal megadottat)
    \item Ennek melyik változatát használjuk (álló, dőlt,
        döntött, kiskapitális,stb.)
    \item Milyen betűvastagságot használunk (vékonyított,
        normál, félkövér, stb.)
    % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
    \item Milyen betűméretet használunk (a \LaTeX{} tíz beépített mérete, vagy egyedi trükkökkel definiált további méret)
\end{itemize}
```

1. program. A LaTeX forráskód példa

A Java program következik az úsztatott `programkod` környezetben.

```
class HelloWorldApp {
    public static void main(String[] args) {
        //Display the string
        System.out.println("Hello World!");
    }
}
\end
```

2. program. A Java forráskód példa

A C++ program következik az úsztatott programkod környezetben.

```
/*
 ******************************************************************
 * MODULE      : substitute.cpp
 * DESCRIPTION: Converts LaTeX to TeXmacs
 * COPYRIGHT   : (C) 2003  Joris van der Hoeven
 ******************************************************************
 * This software falls under the GNU general public license and comes WITHOUT
 * ANY WARRANTY WHATSOEVER. See the file $TEXMACS_PATH/LICENSE for more details.
 * If you don't have this file, write to the Free Software Foundation, Inc.,
 * 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
 ******************************************************************/
#include <stdio.h>
#include <iostream.h>
#define DATA_BEGIN   ((char) 2)
#define DATA_END     ((char) 5)
#define DATA_ESCAPE  ((char) 27)

int
main () {
    cout << DATA_BEGIN << "verbatim:";
    cout << "An interactive LaTeX -> TeXmacs translator\n";
    cout << "Can also be used outside sessions: select a LaTeX expression\n";
    cout << "and press C-F12\n";
    cout << DATA_END;
    fflush (stdout);

    while (true) {
        char buffer[100];
        cin.getline (buffer, 100, '\n');
        cout << DATA_BEGIN;
        cout << "latex:$" << buffer << "$";
        cout << DATA_END;
        fflush (stdout);
    }
    return 0;
}
```

3. program. A C++ forráskód példa