

# Programlisták készítése a listings csomaggal

Virágh János

2024. október 7.

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

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

## Tartalomjegyzék

1. Bevezetés	2
2. Egyszerű (nem-úsztatott) programlisták	3
3. Úsztatott programlisták	7
4. Új úsztatott környezet definiálása programlisták számára	8

## Ábrák jegyzéke

1. A $\text{\LaTeX}$ forráskód példa ábraként úsztatva	7
2. A mi kis házi oroszánunk	8
3. A Java forráskód példa ábraként úsztatva	8

## Programok jegyzéke

1. A $\text{\LaTeX}$ forráskód példa	9
2. A Java forráskód példa	9
3. A C++ forráskód példa	10

## 1. Bevezetés

Az alábbi példák a listings csomag használatát mutatják be különféle programlisták elkészítésére. Hasznos lehet Tómacs Tibor fóliáiról a vonatkozó részek átolvasása is. Először az `\lstset` parancs segítségével megadjuk a továbbiakhoz szükséges opciókat. Az alkalmazott parancs  $\text{\LaTeX}$  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 törhetőek
    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
    {ú}{\{'H{u}}1
    {ő}{\{'H{o}}1
    {ü}{\{'{u}}1
    {ö}{\{'{o}}1
    {ű}{\{'{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

### $\text{\LaTeX}$ programlista

Az első példában az alapértelmezetteken túl (lásd `lstset` parancs) opcióként a  $\text{\LaTeX}$  nyelvet és a dupla keretezést választottuk.  
A  $\text{\LaTeX}$  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  $\text{\LaTeX}$  kód:

```
1 % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
2 Egy adott betű (vagy szó, mondat, paragrafus, stb.) megjelenítését sok tényező
  ↓ nyező befolyásolhatja:
3 \begin{itemize}
4   \item Melyik betűtípust használjuk (alapértelmezettet,
5     vagy pl. a preambulumban csomaggal megadottat)
6   \item Ennek melyik változatát használjuk (álló, dőlt,
7     döntött, kiskapitális, stb.)
8   \item Milyen betűvastagságot használunk (vékonyított,
9     normál, félkövér, stb.)
10  % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
11  \item Milyen betűméretet használunk (a \LaTeX{} tíz beépített mérete,
12    ↓ vagy egyedi trükkökkel definiált további méret)
\end{itemize}
```

Ha végre is hajtjuk a fenti  $\text{\LaTeX}$  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  $\text{\LaTeX}$  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
7  * WITHOUT
8  * ANY WARRANTY WHATSOEVER. See the file $TEXMACS_PATH/LICENSE for more
9  * details.
10 * If you don't have this file, write to the Free Software Foundation,
11 * Inc.,
12 * 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
13 *****/
14 #include <stdio.h>
15 #include <iostream.h>
16 #define DATA_BEGIN ((char) 2)
17 #define DATA_END   ((char) 5)
18 #define DATA_ESCAPE ((char) 27)
19
20 int
21 main () {
22     cout << DATA_BEGIN << "verbatim:";
23     cout << "An interactive LaTeX->TeXmacs translator\n";
24     cout << "Can also be used outside sessions: select a LaTeX
25     * expression\n";
26     cout << "and press C-F12\n";
27     cout << DATA_END;
28     fflush (stdout);
29
30     while (true) {
31         char buffer[100];
32         cin.getline (buffer, 100, '\n');
33         cout << DATA_BEGIN;
34         cout << "latex:$" << buffer << "$";
35         cout << DATA_END;
36         fflush (stdout);
37     }
38     return 0;
39 }

```

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:

```

1  import numpy as np
2
3  def incmatrix(genl1,genl2):
4      m = len(genl1)
5      n = len(genl2)
6      M = None #to become the incidence matrix
7      VT = np.zeros((n*m,1), int) #dummy variable
8
9      #compute the bitwise xor matrix
10     M1 = bitxormatrix(genl1)
11     M2 = np.triu(bitxormatrix(genl2),1)
12
13     for i in range(m-1):
14         for j in range(i+1, m):
15             [r,c] = np.where(M2 == M1[i,j])
16             for k in range(len(r)):
17                 VT[(i)*n + r[k]] = 1;
18                 VT[(i)*n + c[k]] = 1;
19                 VT[(j)*n + r[k]] = 1;
20                 VT[(j)*n + c[k]] = 1;
21
22     if M is None:
23         M = np.copy(VT)
24     else:
25         M = np.concatenate((M, VT), 1)
26
27     VT = np.zeros((n*m,1), int)
28
29     return M

```

### 3. Úsztatott programlisták

Ilyen állatfajt alpból nem ismer a  $\text{\LaTeX}$ . 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 szűrni. 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.

```

1  % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
2  Egy adott betű (vagy szó, mondat, paragrafus, stb.) megjelenítését sok tényező
3  ↓ ő befolyásolhatja:
4  \begin{itemize}
5      \item Melyik betűtípust használjuk (alapértelmezett,
6      vagy pl. a preambulumban csomaggal megadottat)
7      \item Ennek melyik változatát használjuk (álló, dőlt,
8      döntött, kiskapitális, stb.)
9      \item Milyen betűvastagságot használunk (vékonyított,
10     normál, félkövér, stb.)
11     % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
12     \item Milyen betűméretet használunk (a \LaTeX{} tíz beépített mérete,
13     ↓ vagy egyedi trükkökkel definiált további méret)
14 \end{itemize}

```

1. ábra. A  $\text{\LaTeX}$  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.

```
1 class HelloWorldApp {  
2     public static void main(String[] args) {  
3         //Display the string  
4         System.out.println("Hello World!");  
5     }  
6 }  
7 \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 L<sup>A</sup>T<sub>E</sub>X forráskód példát, csak most már az új programkod úsztatott környezetet használjuk.

```

1 % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
2 Egy adott betű (vagy szó, mondat, paragrafus, stb.) megjelenítését sok tényez
   ↓ ő befolyásolhatja:
3 \begin{itemize}
4   \item Melyik betűtípust használjuk (alapértelmezett,
5     vagy pl. a preambulumban csomaggal megadottat)
6   \item Ennek melyik változatát használjuk (álló, dőlt,
7     döntött, kiskapitális, stb.)
8   \item Milyen betűvastagságot használunk (vékonyított,
9     normál, félkövér, stb.)
10  % most egy hosszú sor következik, amit automatikusan törni fog a LaTeX:
11  \item Milyen betűméretet használunk (a \LaTeX{} tíz beépített mérete,
   ↓ vagy egyedi trükkökkel definiált további méret)
12 \end{itemize}

```

1. program. A L<sup>A</sup>T<sub>E</sub>X forráskód példa

A Java program következik az úsztatott programkod környezetben.

```

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

```

2. program. A Java forráskód példa

A C++ program következik az úsztatott programkod környezetben.

```
1      /*  
2      ↓ *****  
3      ↓  
4      * MODULE      : substitute.cpp  
5      * DESCRIPTION: Converts LaTeX to TeXmacs  
6      * COPYRIGHT  : (C) 2003 Joris van der Hoeven  
7      * *****  
8      ↓  
9      * This software falls under the GNU general public license and  
10     ↓ comes WITHOUT  
11     * ANY WARRANTY WHATSOEVER. See the file $TEXMACS_PATH/LICENSE  
12     ↓ for more details.  
13     * If you don't have this file, write to the Free Software  
14     ↓ Foundation, Inc.,  
15     * 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.  
16     * *****  
17     ↓ */  
18     #include <stdio.h>  
19     #include <iostream.h>  
20     #define DATA_BEGIN ((char) 2)  
21     #define DATA_END   ((char) 5)  
22     #define DATA_ESCAPE ((char) 27)  
23  
24     int  
25     main () {  
26         cout << DATA_BEGIN << "verbatim:";  
27         cout << "An interactive LaTeX->TeXmacs translator\n";  
28         cout << "Can also be used outside sessions: select a LaTeX  
29         ↓ expression\n";  
30         cout << "and press C-F12\n";  
31         cout << DATA_END;  
32         fflush (stdout);  
33  
34         while (true) {  
35             char buffer[100];  
36             cin.getline (buffer, 100, '\n');  
37             cout << DATA_BEGIN;  
38             cout << "latex:$" << buffer << "$";  
39             cout << DATA_END;  
40             fflush (stdout);  
41         }  
42         return 0;  
43     }
```

3. program. A C++ forráskód példa