

```

1 import java.util.ArrayList;
2
3 public class Example {
4     public static void main(String[] args) {
5         System.out.println("Hallo_Welt");
6         if(args==null)
7             System.out.println("wau");
8     }
9 }

```

```

1 \begin{java}
2 |info|import java.util.ArrayList;|info|
3
4 public class Example {
5     public static void main(String[] args) {
6         System.out.|err|println|err|("Hallo Welt");
7         if(|warn|args==null|warn|)
8             System.out.println("wau");
9     }
10 }
11 \end{java}

```

Bemerkung 0.1 – Kompilierung

Dieses Dokument wurde kompiliert durch:

`pdflatex` test-1st.tex

1 Einfache Benutzung zum Code-Highlighting, etlicher Sprachen

Java:

```

1 public static void main(String[] args) {
2     int i = 42*3;
3     System.out.println("Hallo_Welt");
4 }

```

C++:

```

1 #include <iostream>
2 public static int main(int argc, char** argv) {
3     std::cout << "Hallo_Welt" << std::endl;

```

```

4      std::cout << "Die Antwort ist: " << 21*2+14-(2*7) << endl;
5  }

```

Die Existenz einer Sprache kann dank `\LIB LILLYxLIST` leicht geprüft werden:

```

1  \isLanguageLoaded{java/lJava} % → 'TRUE' (\true)
2  \isLanguageLoaded{super/lSuper} % → 'FALSE' (\false)

```

Bemerkung 1.1 – Notation

Soll dies auf nur 'java' geändert werden?

Alle geladenen Sprachen finden sich in der Liste `RegisteredLanguages`.

2 Besonderheiten

Durch die Syntax `'<Keyword>:'` können Sonderzeichen und andere tolle Features in listings eingeführt werden, so wurden die oberen Pfeile durch `'yields:'` erzeugt. Weiter interessant sind vielleicht noch `'lan:'` (left angle: `'<'`) und `'ran:'` (right angle: `'>'`)

3 Befehle

Für jede Sprache wie 'Java' existiert ein entsprechend kleingeschriebenes Environment¹ was das Highlighting übernimmt (hier: 'java'), soll die Sprache ohne Firlefanz gesetzt werden, so existiert `lstplain`:

```

1  \begin{lstplain}[language=lJava]
2  public static void main(String[] args) {
3      int i = 42*3;
4      System.out.println("Hallo Welt");
5  }
6  \end{lstplain}

```

Ergibt:

```

public static void main(String[] args) {
    int i = 42*3;
    System.out.println("Hallo Welt");
}

```

Für das inline-Code-Highlighting, existieren für jede Sprache `c<Sprache>` (code) und `b<Sprache>` (blank):

¹C++ ist 'cpp'

```
1 \cpython{print('hallo')}, \bpython{print('hallo')}
```

```
print("hallo") , print('hallo')
```

Ersterer kann aufgrund der Implementation des Hintegrund nicht umgebrochen werden, Listings versucht sich allerdings in einem normalen Maße der bad-paragraph (überlauf Problematik) anzunehmen. Letzterer kann ganz normal umgebrochen werden. Hier ein Beispiel:

```
print('hallo_welt, _na_wie_geht_es_dir?_dies_ist_sehr_langer_
Text, _ein_bisschen_unangenehm_oder?_aber_irgendwie_muss_ich_die_Zeile_sprengen_tut_mir_
wirklich_leid.')
```

Dahingegeben:

```
print('hallo_welt, _na_wie_geht_es_dir?_dies_ist_sehr_langer_
Text, _ein_bisschen_unangenehm_oder?_aber_irgendwie_muss_ich_
die_Zeile_sprengen_tut_mir_wirklich_leid.')
```

Weiter existiert noch '**i<Sprache>**', diesem Befehl kann eine Datei übergeben werden der diese dann in der jeweiligen Sprache hervorhebt. Beispiel mit diesem Dokument:

```
1 \ilatex{test-1st.tex}
```

Da es sich hierbei auch um das Dokument handelt, findet sich das Ergebnis auf der nächsten Seite.

4 Dokumentcode

```
1 \documentclass{article}
2
3 \usepackage{LILLYxLISTINGS}
4 \usepackage{LILLYxCONTROLLERxBOX}
5
6 \begin{document}
7 \begin{java}
8 |info|import java.util.ArrayList;|info|
9
10 public class Example {
11     public static void main(String[] args) {
12         System.out.|err|PrintLn|err|("Hallo Welt");
13         if(|warn|args==null|warn|)
14             System.out.println("wau");
15     }
16 }
17 \end{java}
18 \begin{latex}
19 \begin{java}
20 |info|import java.util.ArrayList;|info|
21
22 public class Example {
23     public static void main(String[] args) {
24         System.out.|err|PrintLn|err|("Hallo Welt");
25         if(|warn|args==null|warn|)
26             System.out.println("wau");
27     }
28 }
29 \end{java}
30 \end{latex}
31
32 \begin{bemerkung}[Kompilierung]
33     Dieses Dokument wurde kompiliert durch: \begin{center}
34         \bbash{pdflatex test-1st.tex}
35     \end{center}
36 \end{bemerkung}
37
38 \section{Einfache Benutzung zum Code-Highlighting, etlicher
39         Sprachen}
40
41 Java:
42 \begin{java}
43 public static void main(String[] args) {
44     int i = 42*3;
```

```

43     System.out.println("Hallo Welt");
44 }
45 \end{java}
46 C++:
47 \begin{cpp}
48 #include <iostream>
49 public static int main(int argc, char** argv) {
50     std::cout << "Hallo Welt" << std::endl;
51     std::cout << "Die Antwort ist: " << 21*2+14-(2*7) << \>
        std::endl;
52 }
53 \end{cpp}
54 Die Existenz einer Sprache kann dank \LILLYxNOTExLibrary{\>
    LILLYxLIST} leicht geprüft werden:
55 \begin{latex}
56 \isLanguageLoaded{java/lJava} % → 'TRUE' (\true)
57 \isLanguageLoaded{super/lSuper} % → 'FALSE' (\false)
58 \end{latex}
59 \begin{bemerkung}[Notation]
60     Soll dies auf nur '\verb|java|' geändert werden?
61 \end{bemerkung}
62 Alle geladenen Sprachen finden sich in der Liste \verb|\>
    RegisteredLanguages|.
63 \section{Besonderheiten}
64 Durch die Syntax '\verb|<Keyword>:|' können Sonderzeichen \>
    und andere tolle Features in listings eingeführt werden\>
    , so wurden die oberen Pfeile durch '\verb|→|' erzeugt.\>
    Weiter interessant sind vielleicht noch '\verb|⟨|' (\>
    left angle: '\bjava{⟨}') und '\verb|⟩|' (right angle: '\>
    \bjava{⟩}')
```

```

65
66 \section{Befehle}
67 Für jede Sprache wie '\verb|Java|' existiert ein \>
    entsprechend kleingeschriebenes Environment\footnote{C\>
    ++ ist 'cpp'} was das Highlighting übernimmt (hier: '\>
    \verb|java|'), soll die Sprache ohne Firlefanz gesetzt \>
    werden, so existiert \verb|lstplain|: \begin{latex}
68 \begin{lstplain}[language=lJava]
69 public static void main(String[] args) {
70     int i = 42*3;
71     System.out.println("Hallo Welt");
72 }
73 \end{lstplain}
74 \end{latex}
75 Ergibt:
76 \begin{lstplain}[language=lJava]
```

```

77 public static void main(String[] args) {
78     int i = 42*3;
79     System.out.println("Hallo Welt");
80 }
81 \end{lstplain}
82 Für das inline-Code-Highlighting, existieren für jede
    Sprache \verb|c<Sprache>| (code) und \verb|b<Sprache>|
    (blank):
83 \begin{latex}
84 \cpython{print('hallo')}, \bpython{print('hallo')}
85 \end{latex}
86 \cpython{print("hallo")}, \bpython{print('hallo')}\newline
87 Ersterer kann aufgrund der Implementation des Hintgrund
    nicht umgebrochen werden, Listings versucht sich
    allerdings in einem normalen Maße der bad-paragraph (ü-
    berlauf Problematik) anzunehmen. Letzterer kann ganz
    normal umgebrochen werden. Hier ein Beispiel:\newline
88 \cpython{print('hallo welt, na wie geht es dir? dies ist
    sehr langer Text, ein bisschen unangenehm oder? aber
    irgendwie muss ich die Zeile sprengen tut mir wirklich
    leid.')} \newline
89 Dahingegeben:\newline
90 \bpython{print('hallo welt, na wie geht es dir? dies ist
    sehr langer Text, ein bisschen unangenehm oder? aber
    irgendwie muss ich die Zeile sprengen tut mir wirklich
    leid.')} \medskip \newline
91 Weiter existiert noch '\verb|i<Sprache>|', diesem Befehl
    kann eine Datei übergeben werden der diese dann in der
    jeweiligen Sprache hervorhebt. Beispiel mit diesem
    Dokument:
92 \begin{latex}
93 \ilatex{test-1st.tex}
94 \end{latex}
95 Da es sich hierbei auch um das Dokument handelt, findet
    sich das Ergebnis auf der nächsten Seite.
96 \clearpage
97 \section{Dokumentcode}
98 \ilatex{test-1st.tex}
99 \end{document}

```