

`\overbar{a_1} \overbar{a_2}`

<code>\overbar{a_1} \overbar{a_2}</code>	$\overline{a_1}\overline{a_2}$
<code>\overline{a_1} \overline{a_2}</code>	$\overline{a_1a_2}$

```

1 public class Example {
2     public static void main(String[] args) {
3         System.out.println(42*3+7-8^42);
4         System.out.println("42*3+7-8^42");
5         int x = 23, z=42;
6         // int y = 42,q=13;
7     }
8     Hallo3 Ich bin eine 4 und f4nt4st1sch und dies ist eine s3,
9         hr 14ng3 z3113
10 }

```

Hi

```

1 \documentclass{article}
2
3 %\usepackage{LILLYxLISTINGS}
4 \usepackage{LILLYxLISTINGSxADVANCED}
5
6 \usepackage{LILLYxCONTROLLERxBOX}
7
8 \begin{document}

```

```
import java.util.ArrayList;
```

```

public class Example {
    public static void main(String[] args) {
        System.out.println("Hallo_Welt");
        if(args==null)
            System.out.println("wau");
    }
}

```

```

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 }

```

Listing 1: Hi

```
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:

pdf<sub>l</sub>at<sub>e</sub>x test-lst.tex

## 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) << std::
5     endl;
6 }
```

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

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

## Bemerkung 0.2 – Notation

Soll dies auf nur 'java' geändert werden?

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

## 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: `'>'`)

## Befehle

Für jede Sprache wie 'Java' existiert ein entsprechend kleingeschriebenes Environment<sup>(a)</sup> 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}
```

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

---

<sup>(a)</sup>C++ ist 'cpp'

```
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):

```
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_
angenehm_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}
```

Hey: [ ] Code for inline default:

```
public static void main (String[] args)
```

Code for inline blank: `public static void main (String[] args)`

Code for inline input:

```
1 \errorcontextlines 10000
2 \documentclass[debug]{Lilly}
3
4 %\usepackage{LILLYxLISTINGS}
5 \usepackage{LILLYxLISTINGSxADVANCED}
```

```
1 public static void main (String[] args){
2
3 }
4 %
```

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

## Dokumentcode

```
1 \errorcontextlines 10000
2 \documentclass[debug]{Lilly}
3
4 %\usepackage{LILLYxLISTINGS}
5 \usepackage{LILLYxLISTINGSxADVANCED}
6 %\usepackage{LILLYxEMBLEMS}
7 \usepackage{LILLYxCONTROLLERxBOX}
8 \usepackage{LILLYxMATH}
9 \usepackage{LILLYxTABLES}
10
11 \DeclareDocumentCommand{\Hallo}{}{}{
12   Hi
13 }
14 % \foreach \x in {Hi,Hu,Ho}{
15 % \edef\cur{\noexpand\NewLstEnv{\x}}%
16 % \cur%
17 % }
18 \begin{document}
19 % \blatex[morekeywords={ [5]{sqrt}}]{\sqrt[3]{42}}
20 \blatex[morekeywords={ [5]{overbar}}]{\overbar{a_1}\ \overbar{a_2}\ }
21 \begin{center}
22   \begin{tabular}{!{\VRule[1pt]}@{\hspace{1em}}l@{\hspace{1em}}|@{\hspace{1em}}c@{\hspace{1em}}!{\VRule[1pt]}}
23     \specialrule{1pt}{0pt}{0pt}
24     {\blatex[morekeywords={ [5]{overbar}}]{\overbar{a_1}\ \overbar{a_2}}} & \(\overbar{a_1}\overbar{a_2}\)\hline
25     {\blatex[morekeywords={ [5]{overline}}]{\overline{a_1}\ \overline{a_2}}} & \(\overline{a_1}\overline{a_2}\)\hline
26     \specialrule{1pt}{0pt}{0pt}
27   \end{tabular}
28 \end{center}
29 \begin{java}
30 public class Example {
31   public static void main(String[] args) {
32     System.out.println(42*3+7-8^42);
33     System.out.println("42*3+7-8^42");
34     int x = 23, z=42;
35     // int y = 42,q=13;
36   }
37   Hallo3 Ich bin eine 4 und f4nt4st1sch und dies ist eine s3hr l4ng
38     3 z3113
39 }
40 \end{java}
41 % \begin{Hi}
```

```

42 % Hallo Mama
43 % \end{Hi}
44 % \begin{Hu}
45
46 % \end{Hu}
47 \Hallo
48 \begin{lstlisting}[language=LLatex]
49 \documentclass{article}
50
51 %\usepackage{LILLYxLISTINGS}
52 \usepackage{LILLYxLISTINGSxADVANCED}
53
54 \usepackage{LILLYxCONTROLLERxBOX}
55
56 \begin{document}
57 \end{lstlisting}
58
59 \begin{plainjava}
60 |info|import java.util.ArrayList;|info|
61
62 public class Example {
63     public static void main(String[] args) {
64         System.out.|err|PrintLn|err|("Hallo Welt");
65         if(|warn|args==null|warn|)
66             System.out.println("wau");
67     }
68 }
69 \end{plainjava}
70
71 \begin{sjava}
72 |info|import java.util.ArrayList;|info|
73
74 public class Example {
75     public static void main(String[] args) {
76         System.out.|err|PrintLn|err|("Hallo Welt");
77         if(|warn|args==null|warn|)
78             System.out.println("wau");
79     }
80 }
81 \end{sjava}
82
83 \begin{java}[caption={Hi}]
84 |info|import java.util.ArrayList;|info|
85
86 public class Example {
87     public static void main(String[] args) {
88         System.out.|err|PrintLn|err|("Hallo Welt");
89         if(|warn|args==null|warn|)

```

```

90         System.out.println("wau");
91     }
92 }
93 \end{java}
94 \begin{latex}
95 \begin{java}
96 |info|import java.util.ArrayList;|info|
97
98 public class Example {
99     public static void main(String[] args) {
100         System.out.|err|PrintLn|err|("Hallo Welt");
101         if(|warn|args==null|warn|)
102             System.out.println("wau");
103     }
104 }
105 \end{java}
106 \end{latex}
107
108 \begin{bemerkung}[Kompilierung]
109     Dieses Dokument wurde kompiliert durch: \begin{center}
110         \bbash{pdflatex test-1st.tex}
111     \end{center}
112 \end{bemerkung}
113
114 \section{Einfache Benutzung zum Code-Highlighting, etlicher Sprachen}
115
116 Java:
117 \begin{java}
118 public static void main(String[] args) {
119     int i = 42*3;
120     System.out.println("Hallo Welt");
121 }
122 \end{java}
123 C++:
124 \begin{cpp}
125 #include <iostream>
126 public static int main(int argc, char** argv) {
127     std::cout << "Hallo Welt" << std::endl;
128     std::cout << "Die Antwort ist: " << 21*2+14-(2*7) << std::endl;
129 }
130 \end{cpp}
131 Die Existenz einer Sprache kann dank \LILLYxNOTExLibrary{\LILLYxLIST}
132 leicht geprüft werden:
133 \begin{latex}
134 \isLanguageLoaded{java/lJava} % → 'TRUE' (\true)
135 \isLanguageLoaded{super/lSuper} % → 'FALSE' (\false)
136 \end{latex}
137 \begin{bemerkung}[Notation]

```

```

136 Soll dies auf nur '\verb|java|' geändert werden?
137 \end{bemerkung}
138 Alle geladenen Sprachen finden sich in der Liste \verb|RegisteredLanguages|.
139 \section{Besonderheiten}
140 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{⟩}')
141
142 \section{Befehle}
143 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|:
144 \begin{latex}
145 \begin{lstplain}[language=Java]
146 public static void main(String[] args) {
147     int i = 42*3;
148     System.out.println("Hallo Welt");
149 }
150 \end{lstplain}
151 \end{latex}
152 \begin{plainlatex}
153 \begin{lstplain}[language=Java]
154 public static void main(String[] args) {
155     int i = 42*3;
156     System.out.println("Hallo Welt");
157 }
158 \end{lstplain}
159 \end{plainlatex}
160 \begin{slatex}
161 \begin{lstplain}[language=Java]
162 public static void main(String[] args) {
163     int i = 42*3;
164     System.out.println("Hallo Welt");
165 }
166 \end{lstplain}
167 \end{slatex}
168 Ergibt:
169 \begin{lstplain}[language=Java]
170 public static void main(String[] args) {
171     int i = 42*3;
172     System.out.println("Hallo Welt");
173 }
174 \end{lstplain}

```



```

175 Für das inline-Code-Highlighting, existieren für jede Sprache \verb|
    c<Sprache>| (code) und \verb|b<Sprache>| (blank):
176 \begin{latex}
177 \cpython{print('hallo')}, \bpython{print('hallo')}
178 \end{latex}
179 \cpython{print("hallo")}, \bpython{print('hallo')}\newline
180 Ersterer kann aufgrund der Implementation des Hintgrund 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:\newline
181 \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
182 Dahingegegen:\newline
183 \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
184 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:
185 \begin{latex}
186 \ilatex{test-1st.tex}
187 \end{latex}
188
189 Hey: [\RegisterLanguage{rubix}{lJava}]
190 % THIS IS WILL FINALLY OPTIONALS
191 Code for inline default: \crubix[morekeywords={args}]{public static
    void main (String[] args)}\
192 Code for inline blank: \brubix{public static void main (String[] args)
    )}\
193 Code for inline input: \irubix[lastline=5]{test-1st.tex}
194 \begin{rubix}
195 public static void main (String[] args){
196
197 }
198 % \end{rubix}
199 % \begin{plainrubix}
200 % Hallo Welt
201 % Hallo Welt
202
203 % Hallo Welt
204 % Hallo Welt
205 % Hallo Welt
206 % Hallo Welt
207 % Hallo Welt
208 % Hallo Welt
209 % Hallo Welt

```

```
210 % Hallo Welt
211 % Hallo Welt
212 % Hallo Welt
213 % Hallo Welt
214 % Hallo Welt
215 % Hallo Welt
216 % Hallo Welt
217 % Hallo Welt
218 % Hallo Welt
219 % Hallo Welt
220 % Hallo Welt
221
222 % Hallo Welt
223 % Hallo Welt
224 % Hallo Welt
225 % Hallo Welt
226 % Hallo Welt
227 % Hallo Welt
228 % Hallo Welt
229 % Hallo Welt
230 % Hallo Welt
231 % Hallo Welt
232 % Hallo Welt
233 % Hallo Welt
234 % Hallo Welt
235 % Hallo Welt
236 % Hallo Welt
237 % Hallo Welt
238
239 % \end{plainrubix}
240
241 % Hey: [\RegisterLanguage{detlef}{lJava}]
242 % Code for inline default: \crubix[morekeywords={args}]{public static
    void main (String[] args)}\
243 % Code for inline blank: \brubix{public static void main (String[]
    args)}\
244 % \begin{rubix}
245 % public static void main (String[] args){
246
247 % }
248 % \end{rubix}
249 % Code for inline default: \cdetlef[morekeywords={args}]{public
    static void main (String[] args)}\
250 % Code for inline blank: \bdetlef{public static void main (String[]
    args)}\
251 % \begin{detlef}
252 % public static void main (String[] args){
253
```

```
254 % }
255 % \end{detlef}
256
257 Da es sich hierbei auch um das Dokument handelt, findet sich das ,
    Ergebnis auf der nächsten Seite.
258 \clearpage
259 \section{Dokumentcode}
260 \ilatex{test-1st.tex}
261 \end{document}
```