Eigene Codesprache für lstlisting

Alexander Ben Nasrallah

19. April 2015

Zusammenfassung

Wir wollen uns hier anschauen, wie man eine eigene Codesprache für *lstlisting* definiert.

1 Eine eigene Sprache

Wir wollen in diesem Abschnitt ausprobieren, wie man einege eigene, *neue* Codesprache für lstlisting definiert, ohne dabei eine bereits existierende zu verändern.

Mit `\lstlistoflistings` generiert man ein Verzeichnis aller listings mit Caption:

Codestellen

Codestelle 1: Command line

```
cd $HOME
# Comment
echo "Hallo Welt!"
```

Hier ein vergleichs c-Code, um zu sehen, welche Einstellungen durch `\lstset` beeinflusst wurden:

Codestelle 2: c-Code mit Standardeinstellungen

```
// Comment
printf("Hallo Welt!")
```

Für weiter Informationen siehe auch [2], [3] oder [1].

Literatur

[1] Jobst Hoffmann Carsten Heinz Brooks Moses und Jobst Hoffmann. *The Listings Package*. Version 1.5e. 6. Sep. 2014.

[2] *ctan listings page*. URL: https://www.ctan.org/pkg/listings (besucht am 17. 04. 2015).

[3] *Wikibooks: Source Code Listings*. URL: http://en.wikibooks.org/wiki/LaTeX/Source_Code_Listings (besucht am 17. 04. 2015).

2 Sprachen erweitern

In diesem Abschnitt wollen wir zwei identische Codeblöcke vergleichen. Der Codeblock enthält C- und LaTeXanweisungen. Wir wollen nun neue Schlüsselwörter für die *einzelnen* Sprachen definieren.

```
#include <stdio.h>

int main()
{
    printf("Hallo Welt\n");
    return 0;
}

class Hallo {
  public static void main( String[] args ) {
    System.out.println("Hallo Welt!");
  }
}
```

```
#include <stdio.h>

int main()
{
    printf("Hallo Welt\n");
    return 0;
}

class Hallo {
  public static void main( String[] args ) {
    System.out.println("Hallo Welt!");
  }
}
```

Wird die Sprache optional neu gesetzt, scheint die keyword-Liste überschrieben zu werden.

```
#include <stdio.h>

int main()
```

```
{
    printf("Hallo Welt\n");
    return 0;
}

class Hallo {
  public static void main( String[] args ) {
    System.out.println("Hallo Welt!");
  }
}
```

Mit einer Erweiterung einer Sprachdefinition kann man die keyword-Liste erweitern und die Sprache in der Liste optionaler Argumente setzen. Hier haben wir Java um den Dialekt *my* erweitert.

```
#include <stdio.h>

int main()
{
    printf("Hallo Welt\n");
    return 0;
}

class Hallo {
  public static void main( String[] args ) {
    System.out.println("Hallo Welt!");
  }
}
```

Hier noch einmal ohne Dialekt.

```
#include <stdio.h>

int main()
{
    printf("Hallo Welt\n");
    return 0;
}

class Hallo {
  public static void main( String[] args ) {
    System.out.println("Hallo Welt!");
  }
}
```

3 Fazit

Man sollte mit Spracherweiterungen und Dialekten arbeiten anstatt alles in einen `\lstlet`-Block zu packen.