Code Konventionen

Zeilen

- Die Zeilenlänge sollte 120 Zeichen nicht überschreiten.
- Einrückung: 4 Leerzeichen
- Notwendige Zeilenumbrüche sollten nach den folgenden Regeln erfolgen
    - nach Kommata
    - vor Operatoren
    - höherrangige Umbrüche bevorzugen
    - Ergibt sich aus den vorangehenden Regeln eine unübersichtliche Formatierung, so sollte eine Einrückung von 8 Leerzeichen bevorzugt werden.

Kommentare

Einzeiliger Kommentar der Angaben zur speziellen Implementierung macht. Der Kommentar beginnt bei den Kommentarzeichen ('`//`') und endet mit dem jeweiligen Zeilenumbruch.
Mehrere unmittelbar aufeinander folgende einzeilige Kommentare sollten vermieden werden, können jedoch zum Auskommentieren von Quelltextanteilen verwendet werden.

```
// Kommentar

int i = 0; // Kommentar
```

Mehrzeiliger Kommentar der Angaben zur speziellen Implementierung macht. Er kann bei entsprechender Kürze auch einzeilig sein.

```
/* Kommentar */

/**
 * Kommentar
 */
```

Deklaration und Initialisierung

- Deklarationen von Variablen sollten nur eine pro Zeile erfolgen, um sie mit vorangesetzten Kommentaren versehen zu können. Mehrere Deklarationen pro Zeile sind bei gleichem Typ jedoch möglich.

    ```
    int i = 0;
    int j = 34;
    double k, l; // Möglich, aber sollte vermieden werden
    ```

    Methodendeklarationen, -aufrufe und Variablendeklarationen verschiedenen Typs müssen jeweils in einer eigenen Zeile erfolgen.
- Mit Ausnahme von Zählvariablen von `for`-Schleifen sollten lokale Variablen immer am Anfang eines Blockes deklariert und möglichst sofort initialisiert werden. Ein Block ist ein Quelltextbereich, der in geschweifte Klammern `{ ... }` eingeschlossen ist.

Klassen-, Interface- und Methodendeklaration

- Kein Leerzeichen zwischen Methodennamen und der folgenden öffnenden runden Klammer
- Die öffnende geschweifte Klammer eines Blockes sollte in einer neuen Zeile stehen.

- Die schließende geschweifte Klammer eines Blockes sollte in einer neuen Zeile auf Einrückungsebene des zugehörigen Statements erscheinen.

```
void print(int i)
{
    System.out.println(i);
}
```

Statements

- Für jedes Statement sollte eine eigene Zeile verwendet werden.
- Bei Bedingungen und Verzweigungen sollten, wie bei Methoden auch, die geschweiften öffnenden Klammern des Blockes in einer neuen Zeile stehen. Die schließende geschweifte Klammer eröffnet eine neue Zeile auf Einrückungsebene des Statements.

```
for (int i = 0; i < 10; i++)
{
    if (i == 5)
    {
        System.out.println(i);
    }
}
```

Leerzeichen

- ...stehen zwischen Schlüsselwörtern und runden Klammern, jedoch nicht nach Methodennamen.
- ...stehen nach Kommata in Argumentlisten

```
void print(int i, double j)
```

- ...stehen zwischen binären Operatoren und ihren Operanden

```
int i = 5;
```

- ...stehen nicht zwischen unären Increment- und Decrement-Operatoren und deren Operanden

```
a++, --i
```

- ...stehen zwischen den Ausdrücken eines `for`-Statements

```
for (int i = 0; i < 5; i++)
```

- ...stehen nach expliziten Casts

```
float f = 3.14f;
double d = (double) f;
```

Namenskonventionen

Alle Bezeichner sollten grundsätzlich auf Englisch sein, und möglichst intuitiv den Zweck angeben, für den sie stehen. Lediglich nur kurzfristig benötigte Werte, wie bspw. Zählvariablen, können durch Kurzbezeichner repräsentiert werden. Alle Bezeichner müssen aus alphanumerischen Zeichen des ASCII-Zeichensatzs bestehen, dürfen Unterstriche enthalten, jedoch nicht mit einer Ziffer beginnen.

- Klassen- und Interface-Bezeichner sollen mit großem Anfangsbuchstaben in CamelCase[1]-Schreibweise geschrieben werden.

```
class TestConverterGUI
```

- Variablen- und Methoden-Bezeichner sollen mit Ausnahme von Klassenkonstanten mit kleinem Anfangsbuchstaben in CamelCase[1]-Schreibweise geschrieben werden

  ```
  String studentName;
  void printStudentName()
  ```

- Klassenkonstanten (`static final` deklarierte Variablen) werden durchgehend mit Großbuchstaben geschrieben. Werden mehrere Worte verwendet, so werden diese durch Unterstriche getrennt. Magic Numbers werden vermieden.

  ```
  static final int BORDER_WIDTH = 5;
  ```