

```

1 import ij.ImagePlus;
2 import ij.plugin.filter.PlugInFilter;
3 import ij.process.ImageProcessor;
4
5 public class MyInverter_ implements PlugInFilter {
6
7     public int setup(String arg, ImagePlus img) {
8         return DOES_8G; // this plugin accepts 8-bit grayscale images
9     }
10
11     public void run(ImageProcessor ip) {
12         int w = ip.getWidth();
13         int h = ip.getHeight();
14
15         for (int u = 0; u < w; u++) {
16             for (int v = 0; v < h; v++) {
17                 int p = ip.getPixel(u,v);
18                 ip.putPixel(u,v,255-p);
19             }
20         }
21     }
22
23 }

```

**Programm 3.1**

ImageJ-Plugin zum Invertieren von 8-Bit-Grauwertbildern (File MyInverter\_.java).

**Die setup()-Methode**

Vor der eigentlichen Ausführung des Plugin, also vor dem Aufruf der `run()`-Methode, wird die `setup()`-Methode vom ImageJ-Kernsystem aufgerufen, um Informationen über das Plugin zu erhalten. In unserem Beispiel wird nur der Wert `DOES_8G` (eine statische `int`-Konstante in der Klasse `PlugInFilter`) zurückgegeben, was anzeigt, dass dieses Plugin 8-Bit-Grauwertbilder (8G) verarbeiten kann. Die Parameter `arg` und `img` der `setup()`-Methode werden in diesem Beispiel nicht benutzt (s. auch Aufg. 3.4).

**Die run()-Methode**

Wie bereits erwähnt, wird der `run()`-Methode ein Objekt `ip` vom Typ `ImageProcessor` übergeben, in dem das zu bearbeitende Bild und zugehörige Informationen enthalten sind. Zunächst werden durch Anwendung der Methoden `getWidth()` und `getHeight()` auf `ip` die Dimensionen des Bilds abgefragt. Dann werden alle Bildkoordinaten in zwei geschachtelten `for`-Schleifen mit den Zählvariablen `u` und `v` horizontal bzw. vertikal durchlaufen. Für den eigentlichen Zugriff auf die Bilddaten werden zwei weitere Methoden der Klasse `ImageProcessor` verwendet:

```
int getPixel (int x, int y)
```

Liefert den Wert des Bildelements an der Position  $(x, y)$ .