Anforderungen

Um mit Java plattformunabhängige Software erstellen zu können, müssen auch die damit erstellten grafischen Benutzeroberflächen unabhängig vom jeweiligen Betriebssystem sein.

Grafische Benutzeroberflächen erfordern:

- **grafische Primitivoperationen**
  zur Darstellung von Texten, Linien und Polygonen,
  zur Zuweisung von Farben, Schriftarten, Positionen und Größen,
- **grafische Interaktionselemente**
  zur Darstellung von Fenstern, Scrollbalken, Menüs, Textfeldern, Containern und Schaltflächen (Buttons, Checkboxen, Radiobuttons, Dropdownlisten),
- **Modell zur Behandlung von Ereignissen**
  zur Verarbeitung von Tastatureingaben, Mausbewegungen, Maus-Klicks und drag & drop.

Toolkits

Java bietet zwei Toolkits (Bibliotheken, Frameworks) für grafische Benutzeroberflächen an:

- **AWT (*Abstract Windows Toolkit*)**
  Das AWT implementiert den GUI-Standard MOTIF für die Plattformen Windows, MacOS und UNIX. Die Visualisierung wird auf Betriebssystemseite durchgeführt und unterliegt daher spezifischen Einschränkungen (z.B. kann ein Textfeld unter Windows maximal 64K Zeichen aufnehmen, auf anderen Betriebssystemen gibt es diese Grenze nicht). Moderne grafische Elemente wie z.B. Icons auf Schaltflächen werden von AWT nicht unterstützt.
- **Swing / JFC (*Java Foundation Classes*)**
  Aufgrund der Einschränkungen und Nachteile des AWT wurde 1997 ein weiteres Toolkit geschaffen, welches zusätzliche Komponenten enthält: die Swing-Komponenten, weshalb Swing zu einem Synonym für JFC wurde. Swing basiert auf sogenannten *Leichtgewicht-Komponenten* und hat keine Betriebsystem-spezifischen Implementierungen.
  Alle Komponenten werden mit primitiven Zeichenoperationen gemalt (!):
  beispielsweise besteht eine Schaltfläche aus Rechteck + Schatten + Text in der Mitte.
  Dieser Weg ist plattformunabhängiger, aber auch langsamer als bei AWT.

"Hallo Welt" mit AWT

```
java.awt
Class Frame

java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
```


```
import java.awt.Frame;

public class HalloAwt
{
    public static void main(String args[]) {
        Frame f = new Frame("Hallo awt");
        f.setSize(200,150);
        f.setVisible(true);
    }
}
```

"Hallo Welt" mit Swing

```
javax.swing
Class JFrame

java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
```


```
import javax.swing.JFrame;

public class HalloSwing
{
    public static void main(String args[]) {
        JFrame f = new JFrame("Hallo swing");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setSize(200,150);
        f.setVisible(true);
    }
}
```

Grafische Primitivoperationen: Zeichnen mit JPanel

In Swing ist die Methode `paintComponent(…)` der Klasse `JPanel` für das Zeichnen des Fensterinhalts verantwortlich. Um festzulegen, was / wann / wo gezeichnet werden soll, muss diese Methode überschrieben werden.
Die einzelnen Zeichen-Methoden stammen von der Klasse `Graphics`:

`drawLine(x1, y1, x2, y2)`	Line zwischen zwei Punkten
`drawRect(x, y, width, height)`	leeres Rechteck, Rahmenbreite 1px
`fillRect(x, y, width, height)`	mit Vordergrundfarbe gefülltes Rechteck
`drawOval(x, y, width, height)`	leeres Oval, Rahmenbreite 1px

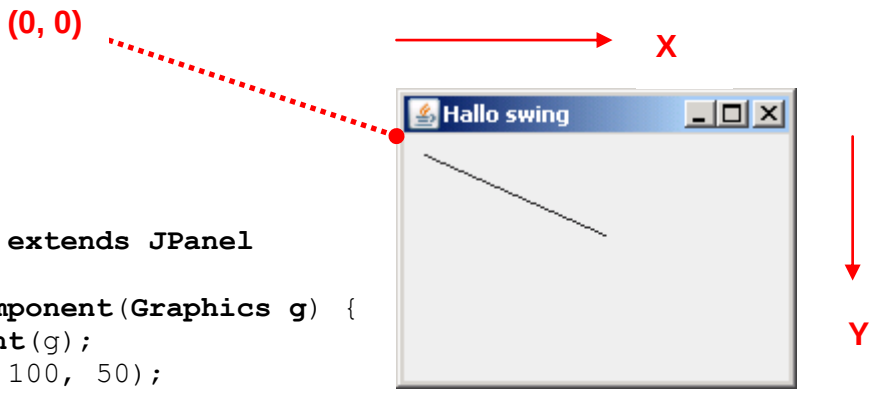
Die Koordinaten können entweder Absolutwerte (in Pixel) sein oder mit den Methoden `getWidth()` bzw. `getHeight()` von der aktuellen (!) Fenster-Breite/-Höhe abgeleitet werden. Da eine Veränderung der Fenster-Lage/-Größe ein automatisches Neu-Zeichnen auslöst, können durch Verwendung dieser Methoden auch Grafiken erstellt werden, die sich automatisch der Fenstergröße anpassen (automatisch *skalieren*).

`drawString(String, x, y)`	Textausgabe, aktuelle Farbe bzw. Zeichensatz
`setFont(Font)`	ändert den Zeichensatz
`setColor(Color)`	ändert die Zeichenfarbe

Liste der im System verfügbaren Zeichensätze:

```
GraphicsEnvironment.getLocalGraphicsEnvironment().getAvailableFontFamilyNames()
```

Beispiel




```
import java.awt.Graphics;
import javax.swing.*;

public class DrawFirstLine extends JPanel
{
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawLine(10, 10, 100, 50);
    }

    public static void main(String args[]) {
        JFrame f = new JFrame("Hallo swing");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setSize(200,150);
        f.add(new DrawFirstLine());
        f.setVisible(true);
    }
}
```