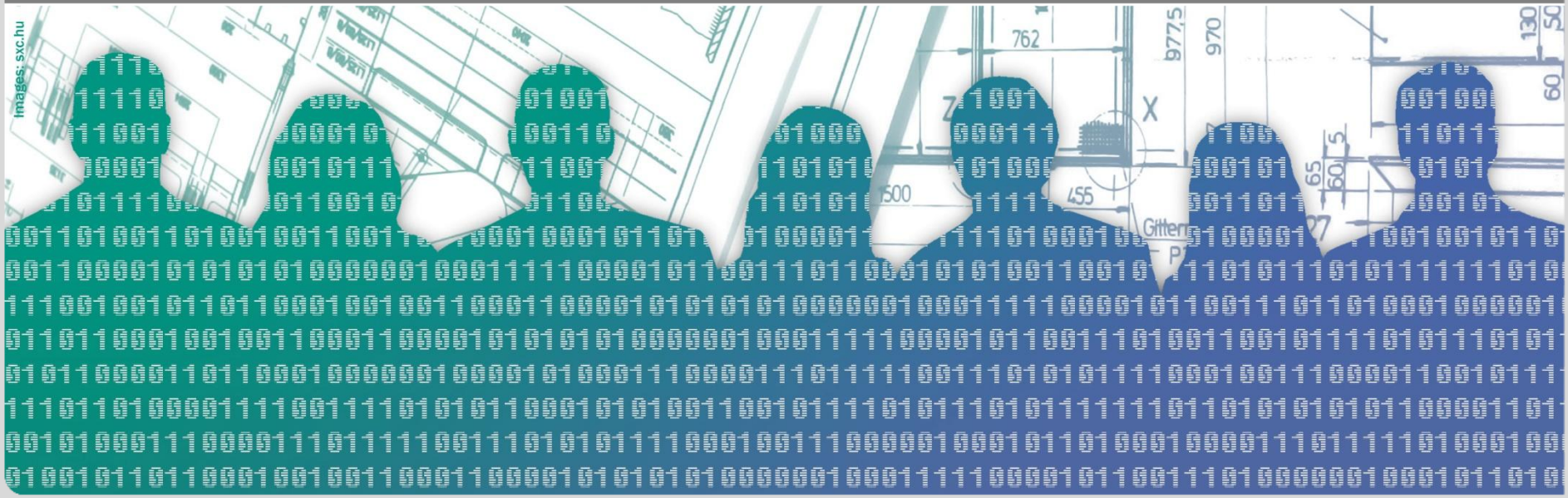
Kapitel X - Zeichnen mit Java2D

SWT I – Sommersemester 2010

Walter F. Tichy, Andreas Höfer, Korbinian Molitorisz

IPD Tichy, Fakultät für Informatik



Literatur

- Informationen zu Java2D finden Sie in der aktuellen Java Dokumentation unter:
  http://java.sun.com/javase/6/docs/api/java/awt/Graphics.html
  und
  http://java.sun.com/javase/6/docs/api/java/awt/Graphics2D.html
- Einen Lehrgang zu Java2D finden Sie unter:
  http://java.sun.com/docs/books/tutorial/2d/index.html

Java2D - Überblick

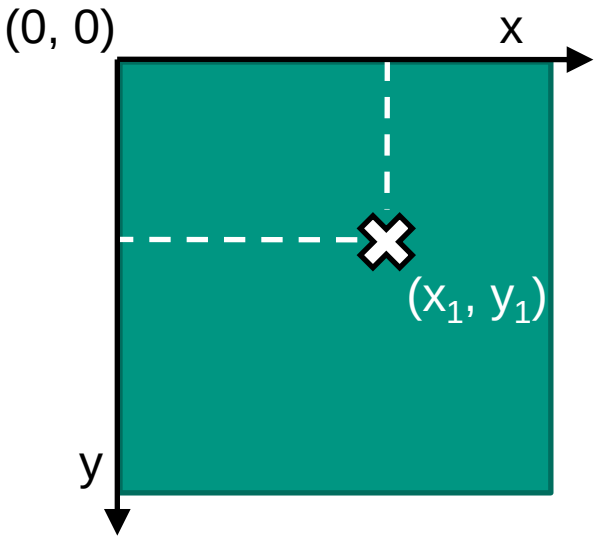
- Die Java2D-Schnittstelle bietet:
  - Ein einheitliches Darstellungsmodell für Bildschirme und Drucker
  - Verschiedene geometrische Primitive (Linien, Rechtecke, Ellipsen, …)
  - Funktionen zur Erkennung von Treffern (bspw. ob sich der Mauszeiger in einer Ellipse e befindet)
  - Ein Modell zum Zusammenfügen von Objekten (inklusive Kontrolle darüber, wie mit überdeckten Objekten umgegangen wird)
  - Möglichkeiten zur Anpassung der Darstellungsqualität der Zeichnungen.

Java2D – Das Koordinatensystem

- In Java2D werden 2 verschiedene Koordinatenräume eingesetzt:
  - Der **Benutzer-Raum** (engl. *user space*), in welchem die Zeichenobjekte auf der Zeichenfläche platziert werden.
  - Der **Geräte-Raum** (engl. *device space*), welcher den Koordinatenraum des Ausgabegerätes darstellt (bspw. des Bildschirms oder des Druckers)

Java2D – Der Benutzer-Raum

- Die linke obere Ecke der Zeichenfläche besitzt die Koordinate (0,0).
- Je nach Zeichenmethode werden die Koordinaten als Ganz- oder Fließkommazahl angegeben.



Java2D – Grafik-Kontext

- Zum Zeichnen wird in Java ein **Grafik-Kontext** (engl. *graphics context*) verwendet.

- Den Grafik-Kontext erhält man in Java entweder als Parameter (z.B. in der `paint(Graphics g)` Methode) oder kann ihn mittels `getGraphics()` von einer Komponente anfordern.

- Der so erhaltene Grafik-Kontext sollte manuell in einen Java2D-Kontext umgewandelt werden:

```
Graphics2D g = (Graphics2D) getGraphics();
```

Java2D – getGraphics()

- Bei der Verwendung von `getGraphics()` gibt es eine Besonderheit:

  `s.getGraphics()` liefert nur dann ein Objekt zurück, wenn das Steuerelement s bereits in einem schwergewichtigen Container liegt.

Java2D – paint() Methode

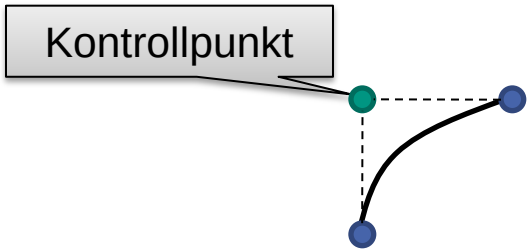
- Jedes Steuerelement zeichnet sich und seine Kindelemente mit Hilfe der `paint()`-Methode.
- Die `paint()`-Methode wird von der zugrunde liegenden Plattform aufgerufen, sobald für sie das Neuzeichnen des Steuerelementes sinnvoll erscheint.
- Es bietet sich daher an, zum Zeichnen in ein Steuerelement s dessen `paint(Graphics g)`-Methode zu überschreiben.

Java2D – Grafische Primitive

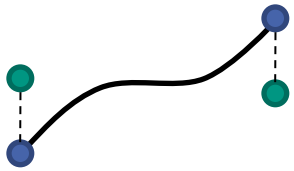
- Java2D bietet verschiedene grafische Primitive an (im Paket `java.awt.geom`):
  - Punkte (Point2D):
    - Hat keine Ausdehnung, keine Farbe und kann nicht dargestellt werden.
  - Linien (Line2D)
  - Rechtecke (Rectangle2D):
    - Bietet Methoden für alle Objekte, welche durch eine rechteckige Zeichen-Box (engl. *bounding box*) beschrieben werden können.

Java2D – Grafische Primitive

- Grafische Primitive (Fortsetzung):
  - Quadratische und Kubische Kurven:
    - Eine quadratische Kurve benötigt zwei Endpunkte und einen Kontrollpunkt.
    - Eine kubische Kurve benötigt zwei Endpunkte und zwei Kontrollpunkte.



Quadratische Kurve



Kubische Kurve

Java2D – Farben

- In Java werden Farben durch die Klasse `Color` repräsentiert.
- Wenn die bereits definierten Farben nicht ausreichen, können beliebige Farben durch Angabe der RGB-Werte (Rot, Grün, Blau) definiert werden.

Java2D – Zeichenfläche leeren

- Muss die Zeichenfläche geleert werden, kann dafür die Methode `g.clearRect(x, y, b, h)` verwendet werden.
- Diese Methode füllt die Zeichenfläche mit der aktuellen Hintergrundfarbe `f`, die über `g.setBackground(f)` gesetzt wurde.

Beispielanwendung: Zeichnen von grafischen Primitiven

```
import java.awt.Color;
import java.awt.Graphics;
import javax.swing.JFrame;

public class Java2DBeispiel extends JFrame {

      public Java2DBeispiel() {
            super("Java2DBeispiel");
            setSize(400, 200);
            setVisible(true);
      }
      public void paint(Graphics g) {
            super.paint(g);
            Graphics2D g2 = (Graphics2D)g;
            g2.setBackground(Color.WHITE);
            g2.clearRect(0, 0, 400, 200);
            g2.setColor(Color.ORANGE);
            g2.fillRect(40, 40, 100, 60);
            g2.setColor(Color.GREEN);
            g2.fillRoundRect(170, 40, 100, 60, 45, 90);
            g2.setColor(Color.BLUE);
            g2.drawOval(40, 105, 100, 60);
            g2.setColor(Color.RED);
            g2.fillOval(170, 105, 100, 60);
      }
      public static void main(String[] args) {
          Java2DBeispiel j2d = new Java2DBeispiel();
      }
}
```

Zeichenfläche leeren

Ausgefülltes Rechteck zeichnen

Ellipse zeichnen