Visualisierung von mathematischen „Objekten“ und „Prozessen“

1. Übersicht Package java.applet
2. Übersicht Package java.awt – Abstract Window Toolkit
3. Übersicht Package java.awt.event – Event Handling
4. Beispiele

1. Übersicht Package java.applet

Die Klasse Applet

- Ein Applet ist ein **Programm**, das in einer Webseite direkt durch einen HTML Eintrag eingebunden werden kann.
- Applets werden **über das Internet geladen und lokal ausgeführt**
- Der Browser stellt eine **Umgebung für Fenster, Grafik** und **Ereignisverwaltung** zur Verfügung
- Applets können nur mit dem Server, von dem sie geladen wurden kommunizieren und sie haben auf dem Host, auf dem der Browser läuft, keine Lese- und Schreibberechtigung (Ausnahme: signierte Applets)
- Der Browser erzeugt nach dem Laden ein Objekt der Applet-Klasse. Für dieses Objekt werden dann die Methoden **init(), start(), stop(), destroy(), paint()** implizit aufgerufen.

Methode: init()

Wird von Browser aufgerufen, nachdem das Applet zum ersten Mal geladen wurde(gilt auch für einen Reload)

Methode: start()

Diese Methode wird unmittelbar nach init() aufgerufen. Wird auch aufgerufen, wenn die Seite vom Browser erneut gelesen wird.

Methode: stop()

Wird vor der Methode destroy() aufgerufen und jedesmal, wenn die Seite verlassen wird.

Methode: destroy()

Wird aufgerufen, bevor das Applet-Objekt gelöscht wird.

Methode: paint()

Diese Methode wird aufgerufen:

- wenn zum ersten Mal etwas gezeichnet werden muss
- wenn ein repaint() Aufruf erfolgt oder wenn ein anderes Fenster über das Applet bewegt wird

Erzeugen einer Applet-Klasse

Eine Applet-Klasse erzeugt man als **Subklasse** der **Klasse Applet**, die im Paket **java.applet** enthalten ist. Diese ist abgeleitet von den Klassen **Component, Container** und **Panel**. Objekte werden über die Methode **add(Component)** eingefügt.

Beispiel:

```
public class Beispielapplet extends java.applet.Applet
{
        public void init(){
          add(new Label("Addition"));
        }
        public void start(){}
        public void stop(){}
        public void destroy(){}
        public void paint(Graphics g){}

}
```

repaint()-Aufruf

- ruft die Methode **update(Graphics g)** auf
  - ***füllt den Hintergund mit der aktuellen Farbe***
  - ***ruft paint(Graphics g) auf***
- Dies führt bei öfterem repaint() zu einem Flackern
- **Lösung:**
  - ***Bild-Objekt erstellen / Graphikbereich des Bildes holen***
  - ***in diesen Graphikbereich zeichnen***
  - ***das Bild in den Graphikbereich des Applets zeichnen***
- Das Füllen mit der Hintergrundfarbe geschieht im Bild und nicht im Applet

Refresh.java

Refresh.html

2. Übersicht Package java.awt (Abstract Window Toolkit)

Beinhaltet alle **Klasse** zur Erstellung von **Benutzerschnitt-stellen** und zum **Zeichnen** von graphischen Elementen und Bildern.

2.1 Komponenten für Benutzerschnittstellen

- TextField, TextArea, Button, Label, Checkbox, usw.

2.2 Die Klasse Graphics

2.3 Die Klasse Frame

2.4 Colors, Fonts und Layouts

2.1 Komponenten

Subklassen der Klasse Component

TextField, TextArea, Button, Label, Checkbox, Scrollbar...

→ können in einem Applet als Objekte eingefügt werden

wichtige vererbte Methoden:

Methode	Bedeutung
setBackground(Color)	Setzt die Hintergrundfarbe
setForeground(Color)	Setzt die Vordergrundfarbe
setFont(Font)	Setzt die Schriftart
setSize(int, int)	Setzt die Größe(Breite, Höhe)
setEnabled(boolean)	Aktiviert / deaktiviert Komponente
setBounds(int, int, int, int)	Setzt die absolute Position / Größe
update(Graphics)	Aktualisiert die Komponente
repaint()	Update -Anforderung

Komponentenbsp\index.html

2.2 Die Klasse Graphics

Jedes **Applet** besitzt seinen **Graphik-Kontext**, bzw. sein zugehöriges **Graphics-Objekt**. Dieses Objekt wird an die **paint()** Methode der Klasse Applet **implizit übergeben**.

Beinhaltet grundlegende Graphikmethoden:

Methode	Bedeutung
drawLine(int, int, int, int)	Zeichnet eine Linie von (x1, y1) nach (x2, y2)
drawPolyline(int [], int[], int)	Zeichnet einen Streckenzug
drawRect(int, int, int, int)	Zeichnet ein Rechteck (x, y, Breite, Höhe)
draw3DRect(int, int, int, int, boolean)	Zeichnet ein Rechteck mit 3dim Rand
drawOval(int, int, int, int)	Zeichnet ein Oval (x, y, Breite, Höhe)
drawPolygon(int [], int [], int)	Zeichnet ein Polygon

Zum Zeichnen **gefüllter** Bereiche sind die Methoden **fillRect, fillOval** etc. implementiert

- Punkte: Methode drawLine - identischer Anfangs- und Endpunkt
- Kreise: Methode drawOval - identische Breite und Höhe

weitere Methoden:

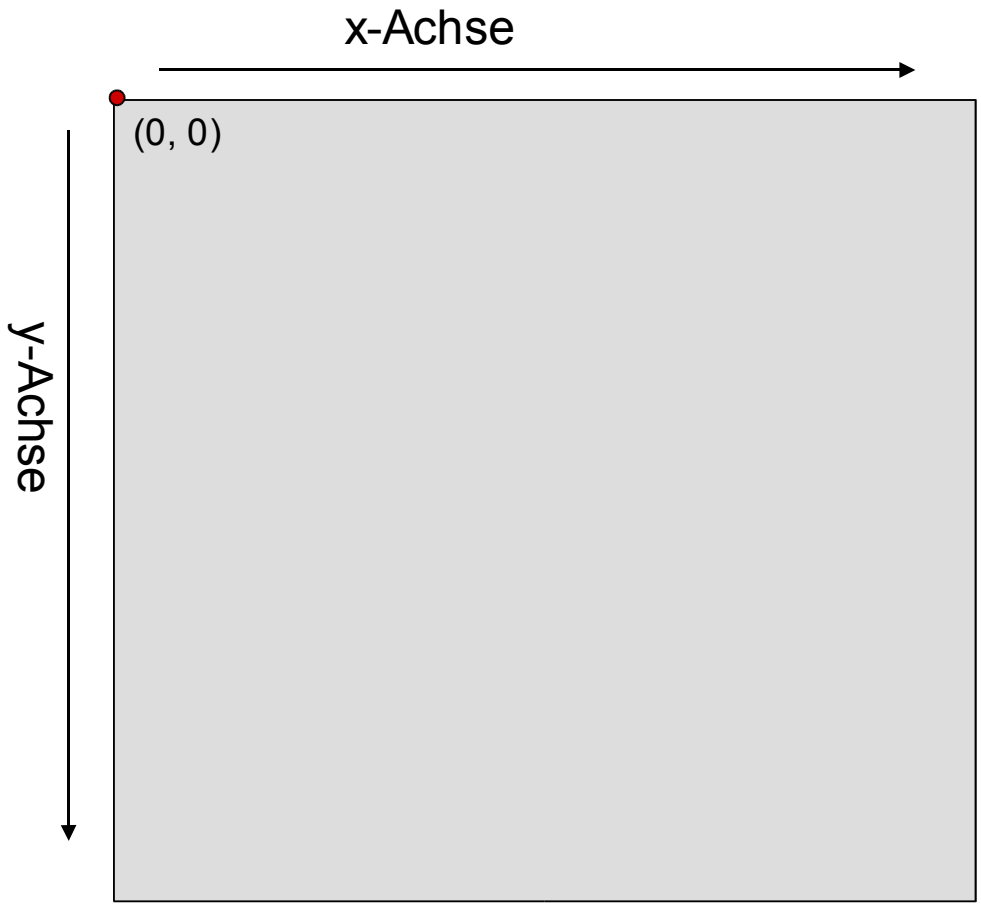
Methode	Bedeutung
create(int, int, int, int)	Liefert ein neues Graphics-Objekt zurück, das auf dem alten basiert (x, y, Breite, Höhe)
setClip(int, int, int, int)	Setzt neue Ränder für das Graphics-Objekt
setColor(Color)	Setzt die Farbe, mit der gezeichnet wird
translate(int, int)	Verschiebt den Ursprung nach (x,y)

Die Klasse bedient sich des Koordinatensystems des Applets

Methode: translate(int x, int y)

Verschiebt den Ursprung nach x, y

Vorteile für ein kartesisches Koordinatensystem zum Zeichnen einer Funktion

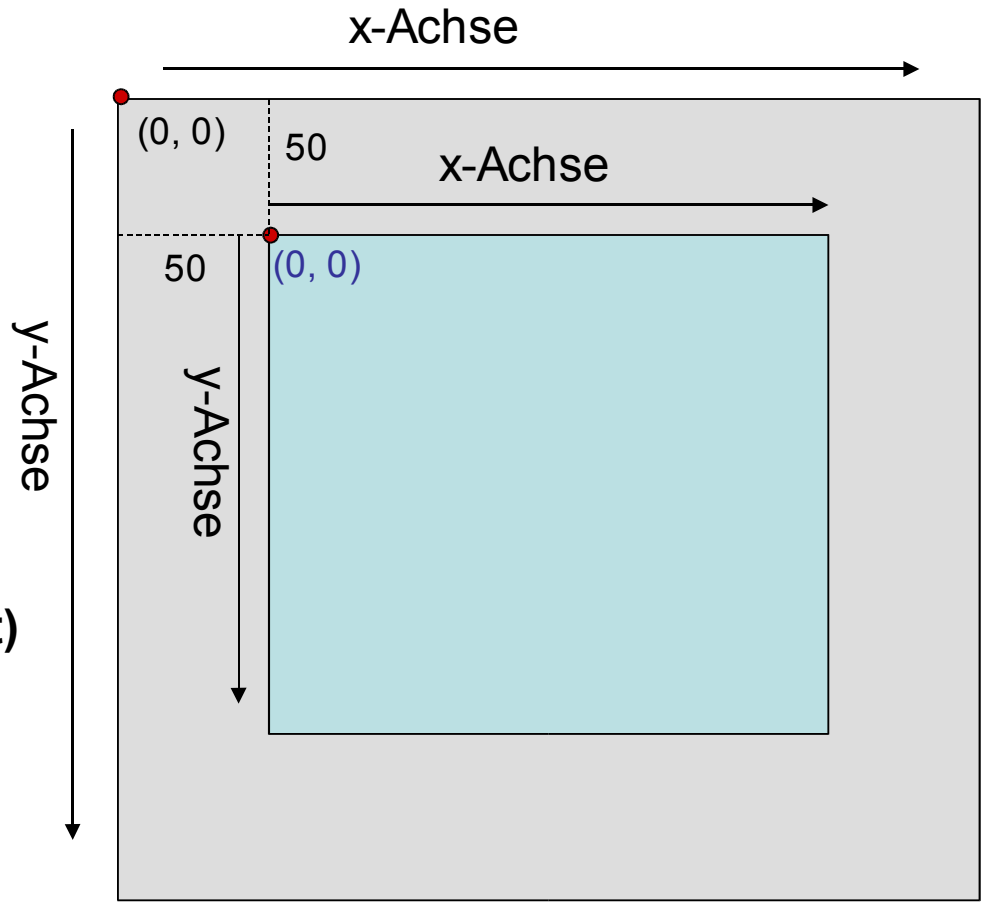


Methode: create(int, int, int, int)

Liefert ein **neues Graphics-Objekt** zurück

Mit **neuem** Ursprung **x,y** und neuer **Breite** und **Höhe**

Methode **translate(int, int)** verschiebt den Ursprung nach x,y



Graphics ksys2 = g.create(50, 50, 200, 200);

Koordinatensystem\Index.htm Koordinatensystem.java

2.2.3 Die Klasse Frame

- **Fenster**-Objekt
- besitzt einen Titel, Rahmen, Menüleiste
- minimier-, maximier- und schließbar
- Titel wird im Konstruktor ***Frame(String)*** übergeben
- Bietet eine **graphische Oberfläche** für Stand-Alone-Anwendungen
- **Ähnliche Entwicklung** wie Applets:
  - gleiche **Komponenten**
  - gleiche **Ereignisverarbeitung**
  - Methoden init(), start(), stop(), destroy() fallen weg, sie gehören zur Applet Klasse
  - Die üblichen Initialisierungen, Listener-Registrierungen werden in den Konstruktor aufgenommen
- Der Frame wird mit ***setVisible(true)*** sichtbar gemacht

Framebsp.java

2.4 Colors, Fonts und Layouts

Die Klasse Color

- Erzeugt ein **Farben-Objekt**, das eine bestimmte Farbdefinition hat
- möglicher Konstruktoraufruf:

```
new Color(int r, int g, int b);      // (0-255)
new Color(int r, int g, int b, int alpha);
```

- Diese Farb-Objekte können als aktuelle Farbe mit *setBackground(Color), setForeground(Color), setColor(Color)* für **Komponenten** und **Graphics-**Objekten verwendet werden
- **13 Farbkonstanten** in der Klasse Color enthalten: black, blue, red, green, usw...

Die Klasse Font

- Erzeugt ein **Font-Objekt** mit einer bestimmten Schriftart
- Konstruktoraufruf:(Name, Stil, Größe)
  new Font("Serif", Font.PLAIN, 20);
- Jedes Java System unterstützt "Serif", "SansSerif" und "Monospaced"
- Stilrichtungen: Font.**PLAIN**, Font.**ITALIC**, Font.**BOLD**
- Schriftgröße wird in Pixel angegeben

Die Layout-Manager

- Sie implementieren alle das Interface **LayoutManager**
- **GridLayout, FlowLayout, BorderLayout**, usw...
- setLayout(LayoutManger) setzt das Layout für das Applet
- **null-Layout**

3. Übersicht Package java.awt.event - Event-Handling

- Jede Komponente kann als Ereignis-Quelle verschiedene Ereignisse erzeugen
- Im Packet java.awt.event stehen zur Ereignisverwaltung ***Listener-Interfaces*** zur Verfügung
- Für jeden ***Ereignistyp*** steht ein entsprechender ***Listener*** zur Verfügung
- Für jeden Listener ist entsprechend eine Methode ***addXYZListener*** in der Komponenten-Klasse deklariert (z.B.: Button → addActionListener)
- Für die Mouse-Ereignisse stehen die Interfaces ***MouseListener*** und ***MouseMotionListener*** zur Verfügung
- Die Interfaces stellen **Methoden** zu Ergeignisverarbeitung bereit, wie z.B. actionPerformed(ActionEvent e)

Komponenten und Events

Komponente	Ereignis
Button	ActionEvent
Checkbox	ItemEvent
Choice	ItemEvent
Scrollbar	AdjustmentEvent
TextField	ActionEvent
Windows	WindowEvent
Component	MouseEvent KeyEvent

Beispiel:

```
....
int x=0;
Button b = new Button("Setze x");
b.addActionListener(new ButtonEvent());
...
public class ButtonEvent implements ActionListener{
        public void actionPerformed(ActionEvent e){
           x=10;
         }
}
```

Eventbsp.html Eventbsp.java

4. Beispiele

Homepage Walter Fendt

- Spiegelung von geometrischen Objekten, z.B. Geraden, Kreise, etc..
- Komplexe Zahlen im kartesischen und polaren Koordinatensystem
- Thaleskreis

3-dim Beispiele

- Geradendarstellung im 3-dimensionalen Raum (Walter Fendt)
- Gerade im 3-dim Raum bestimmen (Mathe Online)
- Ebene im 3-dim Raum bestimmen (Mathe Online)

Eigene Beispiele

- Plotten von Polynomen mit variablen Koeffizienten bis zum 6. Grad
- Darstellung der Ableitung einer Funktion an der Stelle x
- Beispiel einer Kurvendiskussion (Extrema, Wendestellen)

Quellennachweise

- Martin Schader / Lars Schmidt-Thieme
  Java – Einführung in die objektorientierte Programmierung
- Java 2 Standard Edition, Ver. 1.3
  Application Programming Interface Specification
  SUN Microsystems
- Mathe Online Webpage
  http://www.univie.ac.at/future.media/mo/
- Walter Fendt – Applets Mathematik, Phyiks und Astronomie
  http://home.a-city.de/walter.fendt/index.htm

Weitere Links

- ZERO – Mathe Online der Universität Flensburg
  http://www.uni-flensburg.de/mathe/zero/zero.html