

Medientechnik

Übung

Scheinkriterien

- In **jedem** Blatt mindestens 50% der Punkte
- **Ein** Freischuss erlaubt, das bedeutet
 - unterpunktet
 - gar nicht bearbeitet
- erfolgreiche Teilnahme an allen 3 Praktika

Abgabeformat

- Format
 - An Namenskonventionen halten
 - Java: in Zukunft nur src-Ordner und lauffähiges jar abgeben
 - Ab sofort: 1 Punkt für richtiges Format 😊



- UniWorX
 - Upload wird jedes Mal überschrieben
 - Immer ALLES hochladen

Bewertungskriterien

- Programmieraufgaben:
 - Programm läuft
 - Funktionalität umgesetzt
 - Guter Programmierstil & Kommentare

Korrektur

- Korrekturzusammenfassung.txt in eurer Lösung im UniWorX:
 - Name des Korrektors/in
 - Punktzahl
 - Anmerkungen
- Anmerkungen auch im Source-Code
⇒ unbedingt anschauen

Anmerkungen zur Java-Aufgabe

- + teilweise sehr gut gelöst
- + neue Funktionalität getestet
- Abgabe nicht lauffähiger Versionen
- teilweise Probleme mit Java
- Links:
 - OpenBook: “Java ist auch eine Insel”
<http://openbook.galileocomputing.de/javainsel8/>
 - Wikibook „Java Standard“: http://de.wikibooks.org/wiki/Java_Standard
 - Java API <http://java.sun.com/reference/api/index.html>

Beratung zu Programmieraufgaben & Praktika

- <https://wiki.medien.ifi.lmu.de/Main/MedientechnikSS09>
- 17-18Uhr: Programmierberatung im CIP-Pool
- 18-20Uhr: Beratung zum Praktikum im Medienlabor

MedientechnikSS09 < Main < Medieninformatik-Wiki

Search Popular Favorites Work

collaborate with **TWiki**
design provided by TWIKI.NET

Jump Search English

Main Edit Attach

Log In or Register

Main Web
Create New Topic
Index
Search
Changes
Notifications
RSS Feed
Statistics
Preferences

Webs
HCILab
Main
Sandbox
TWiki

You are here: Medieninformatik-Wiki > Main Web > MedientechnikSS09 (05 May 2009, RichardAtterer)

Vorlesung Medientechnik Sommersemester 2009

Foto-Übungen (Gruppen schon festgelegt)

Diese Tabelle dient nur als Referenz für die Zuordnung Gruppe<->Fototermin. Bitte melden Sie sich für eine passende Gruppe über [UniWorx](#) an.

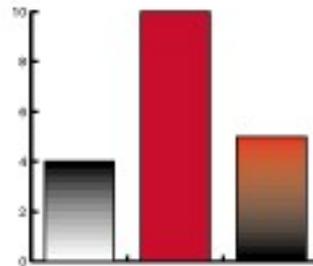
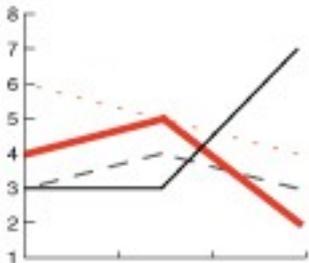
Tag	9 - 12 Uhr	Betreuer	12-15 Uhr	Betreuer	15-18 Uhr	Betreuer	Medienlabor (17-20 Uhr)
11.05.08	Gruppe 1	RaphaelWimmer , RichardAtterer	Gruppe 2	MartinHommer	Gruppe 3	MartinHommer	
12.05.08	Gruppe 4	EduardVodicka	Gruppe 5	RobertKowalski	Gruppe 6	RobertKowalski	RenataWilli
13.05.08	Gruppe 7	RenataWilli	Gruppe 8	MatthiasHoyer	Gruppe 9	MatthiasHoyer	MartinHommer
14.05.08	Gruppe 10	MatthiasHoyer	Gruppe 11	RichardAtterer	Gruppe 12	MatthiasHoyer	RobertKowalski
15.05.08	VL	VL	VL	VL	Gruppe 13	RenataWilli	SonjaRuemelin
18.05.08	Gruppe 14	RenataWilli	Gruppe 15	RobertKowalski	Gruppe 16	RobertKowalski	MatthiasHoyer
19.05.08	Gruppe 17	EduardVodicka	Gruppe 18	MartinHommer	Gruppe 19	SonjaRuemelin	EduardVodicka
20.05.08	Gruppe 20	RenataWilli	Gruppe 21	SonjaRuemelin	Gruppe 22	SonjaRuemelin	MatthiasHoyer
21.05.08		Chri-	sti		Himmel-	fahrt	
22.05.08	VL	VL	VL	VL	Gruppe 23	EduardVodicka	n.n.
25.05.08	Gruppe 24	MatthiasHoyer	Gruppe 25	RobertKowalski	Gruppe 26	RobertKowalski	MatthiasHoyer

Heute

- Java2D:
 - Überblick
 - Farben, Formen und Füllungen
- => nächstes ÜB: Bilder (+ Bildmanipulation)
- Mausinteraktion

Java2D

- ◆ Draw lines, rectangles and any other geometric shape.
- ◆ Fill those shapes with solid colors or gradients and textures.
- ◆ Draw text with options for fine control over the font and rendering process.
- ◆ Draw images, optionally applying filtering operations.
- ◆ Apply operations such as compositing and transforming during any of the above rendering operations.



Using 2D Graphics API to display complex charts



Image



Blur



Sharpen

Graphics2D - Klasse

- API siehe

<http://java.sun.com/j2se/1.4.2/docs/api/java/awt/Graphics2D.html>

- Im java.awt-Paket

- Methoden:

	abstract void	rotate (double theta, double x, double y) Concatenates the current Graphics2D Transform with a translated rotation transform.	
	abstract void	scale (double sx, double sy) Concatenates the current Graphics2D Transform with a scaling transformation S.	
abstract void	clip (Shape s) Intersects the current Clip with the interior of the specified Shape and sets the Clip to the result.	abstract void	setBackground (Color color) Sets the background color for the Graphics2D context.
abstract void	draw (Shape s) Strokes the outline of a Shape using the settings of the current Graphics2D context.	abstract void	setComposite (Composite comp) Sets the Composite for the Graphics2D context.
void	draw3DRect (int x, int y, int width, int height, boolean raised) Draws a 3-D highlighted outline of the specified rectangle.	abstract void	setPaint (Paint paint) Sets the Paint attribute for the Graphics2D context.
abstract void	drawGlyphVector (GlyphVector g, float x, float y) Renders the text of the specified GlyphVector using the Graphics2D context's rendering attributes.	abstract void	setRenderingHint (RenderingHints.Key hintKey, Object hintValue) Sets the value of a single preference for the rendering algorithms.
abstract void	drawImage (BufferedImage img, BufferedImageOp op, int x, int y) Renders a BufferedImage that is filtered with a BufferedImageOp .	abstract void	setRenderingHints (Map hints) Replaces the values of all preferences for the rendering algorithms with the specified Map.
abstract boolean	drawImage (Image img, AffineTransform xform, ImageObserver obs) Renders an image, applying a transform from image space into user space before drawing.	abstract void	setStroke (Stroke s) Sets the Stroke for the Graphics2D context.
abstract void	drawRenderableImage (RenderableImage img, AffineTransform xform) Renders a RenderableImage , applying a transform from image space into user space before drawing.	abstract void	setTransform (AffineTransform Tx) Overwrites the Transform in the Graphics2D context.
abstract void	drawRenderedImage (RenderedImage img, AffineTransform xform) Renders a RenderedImage , applying a transform from image space into user space before drawing.	abstract void	shear (double shx, double shy) Concatenates the current Graphics2D Transform with a shearing transform.
abstract void	drawString (AttributedCharacterIterator iterator, float x, float y) Renders the text of the specified iterator, using the Graphics2D context's current Paint.	abstract void	transform (AffineTransform Tx) Composes an AffineTransform object with the Transform in this Graphics2D context.
abstract void	drawString (AttributedCharacterIterator iterator, int x, int y) Renders the text of the specified iterator, using the Graphics2D context's current Paint.	abstract void	translate (double tx, double ty) Concatenates the current Graphics2D Transform with a translation transform.
abstract void	drawString (String s, float x, float y) Renders the text specified by the specified String, using the current text attribute state in the Graphics2D context.	abstract void	translate (int x, int y) Translates the origin of the Graphics2D context to the point (x, y) in the current coordinate system.
abstract void	drawString (String str, int x, int y) Renders the text of the specified String, using the current text attribute state in the Graphics2D context.		
abstract void	fill (Shape s) Fills the interior of a Shape using the settings of the Graphics2D context.		

Primitive

Übergang von Java 1.1 zu Java 2:

Graphics

`drawXxx`
(`drawLine`, `drawRect`,
`drawArc`, etc.)

`fillXxx`
(`fillRect`, `fillArc`,
etc.)

Graphics2D

`draw(Shape s)`
`fill(Shape s)`

Shape ist Oberklasse von z.B.:

`Arc2D`, `Ellipse2D`,
`Rectangle2D`,
`RoundRectangle2D`

(jeweils `Xxx.Float` und
`Xxx.Double`)

Text: `drawString(String s, float x, float y)`

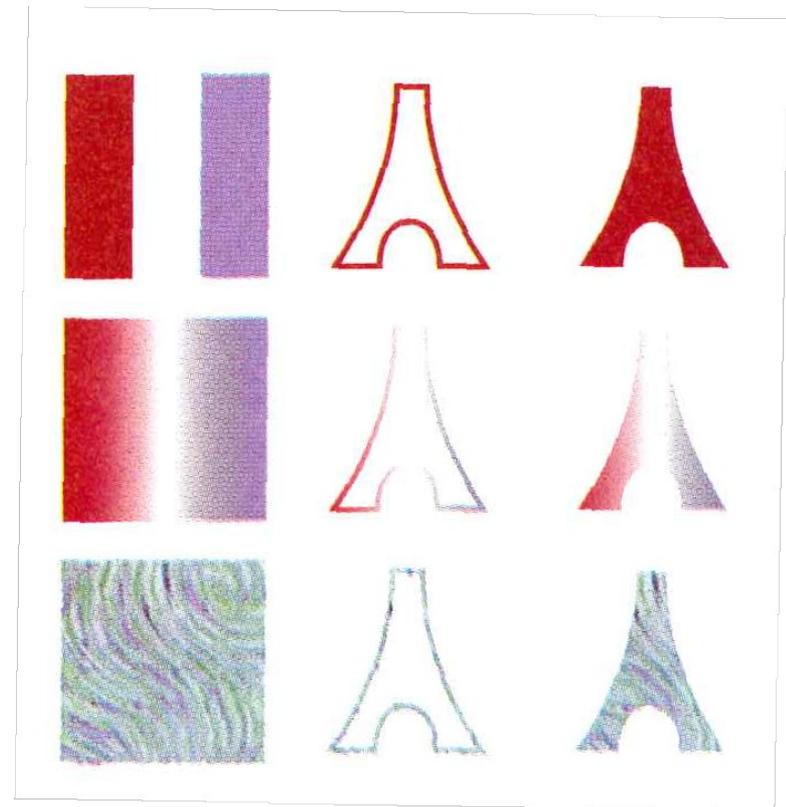
Farben und Füllungen

Setzen von Füllvarianten:

```
setPaint(Paint p)
```

Paint ist Oberklasse von:

Color	(Farbe)
GradientPaint	(Gradient)
TexturePaint	(Bild)



paint(Graphics g) - Methode

- Klasse Component (und viele Unterklassen wie Buttons, Labels etc.) im Paket java.awt bietet eine paint(Graphics g) – Methode an
- Diese kann in Unterklassen überschrieben werden um das Graphics-Objekt zu manipulieren

java.awt

Class Component

[java.lang.Object](#)

|

+--java.awt.Component

All Implemented Interfaces:

[ImageObserver](#), [MenuContainer](#), [Serializable](#)

Direct Known Subclasses:

[Button](#), [Canvas](#), [Checkbox](#), [Choice](#), [Container](#), [Label](#), [List](#), [Scrollbar](#), [TextComponent](#)

void	paint(Graphics g) Paints this component.
void	paintAll(Graphics g) Paints this component and all of its subcomponents.

<http://java.sun.com/javase/6/docs/api/java/awt/Component.html>

Graphics2D - Beispiel

```
import java.awt.*;
```

```
import java.awt.geom.*;
```

```
import javax.swing.*;
```

```
public class View extends JFrame {
```

```
    public View() {
```

```
        super("View");
```

```
        this.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
```

```
        this.setSize(600, 600);
```

```
    }
```

```
    public void paint(Graphics g) {
```

```
        Graphics2D g2 = (Graphics2D) g;
```

```
        /* Java2D: */
```

```
        g2.setPaint(Color.BLACK);
```

```
        g2.fill(new Rectangle2D.Float(0, 0, this.getWidth(),  
this.getHeight()));
```

```
    }
```

JFrame erbt von Component

Wir überschreiben die geerbte
paint(...)-Methode, um an das
Graphics-Objekt zu gelangen

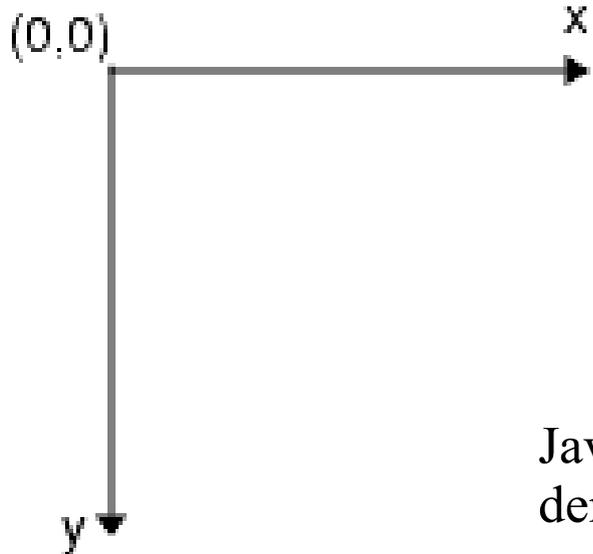
Cast zum Graphics2D-Objekt

Farbe setzen

Rechteck malen

View.java

Etwas Geometrie...

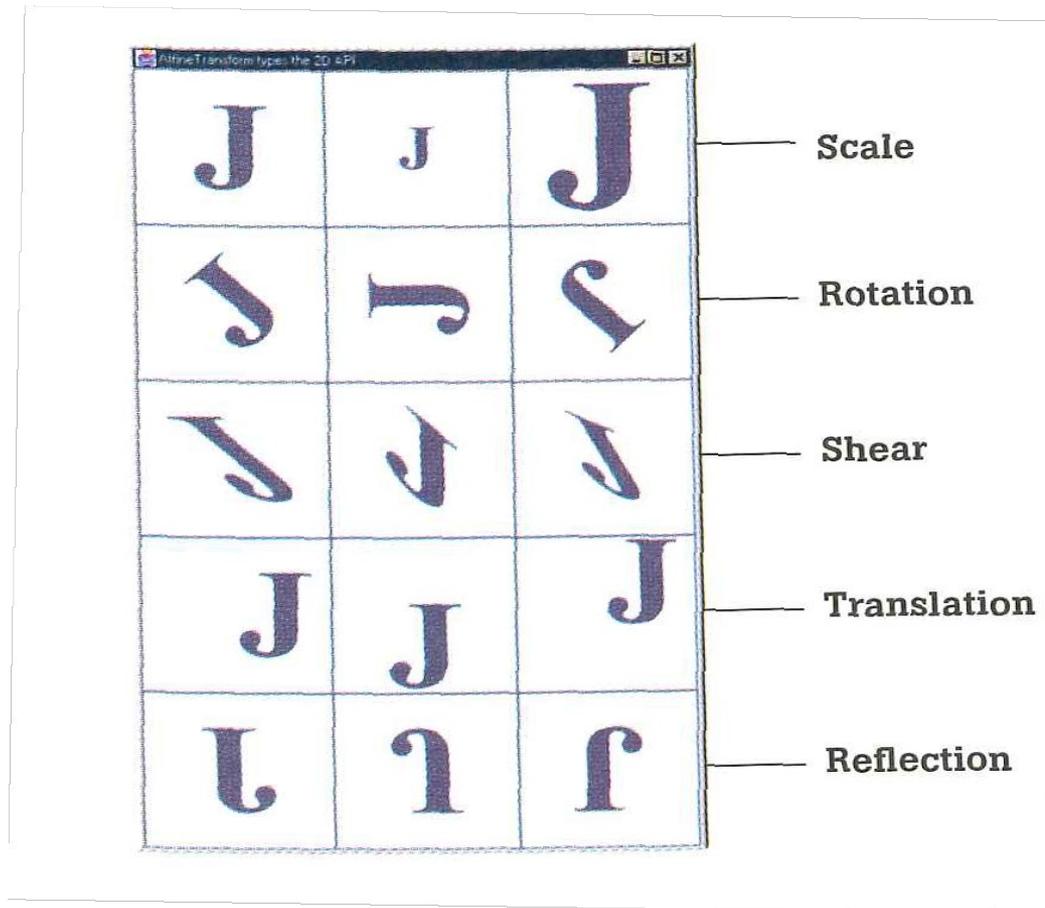


Java2D Koordinatensystem hat den Ursprung in der linken oberen Ecke der jeweiligen Komponente!

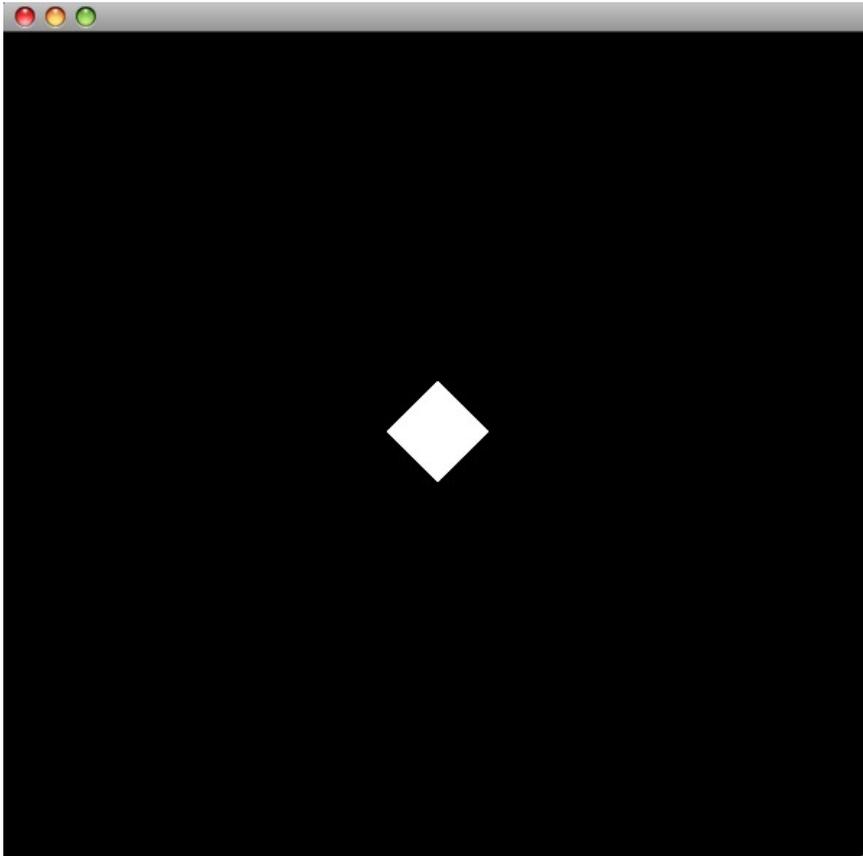
Koordinatensystem kann manipuliert werden, um Transformationen und Verzerrungen zu ermöglichen, d.h. translate/rotate usw. bezieht sich immer auf das Koordinatensystem und ändert dieses

Reihenfolge wichtig!! Aufzeichen hilft!

Transformation von Objekten

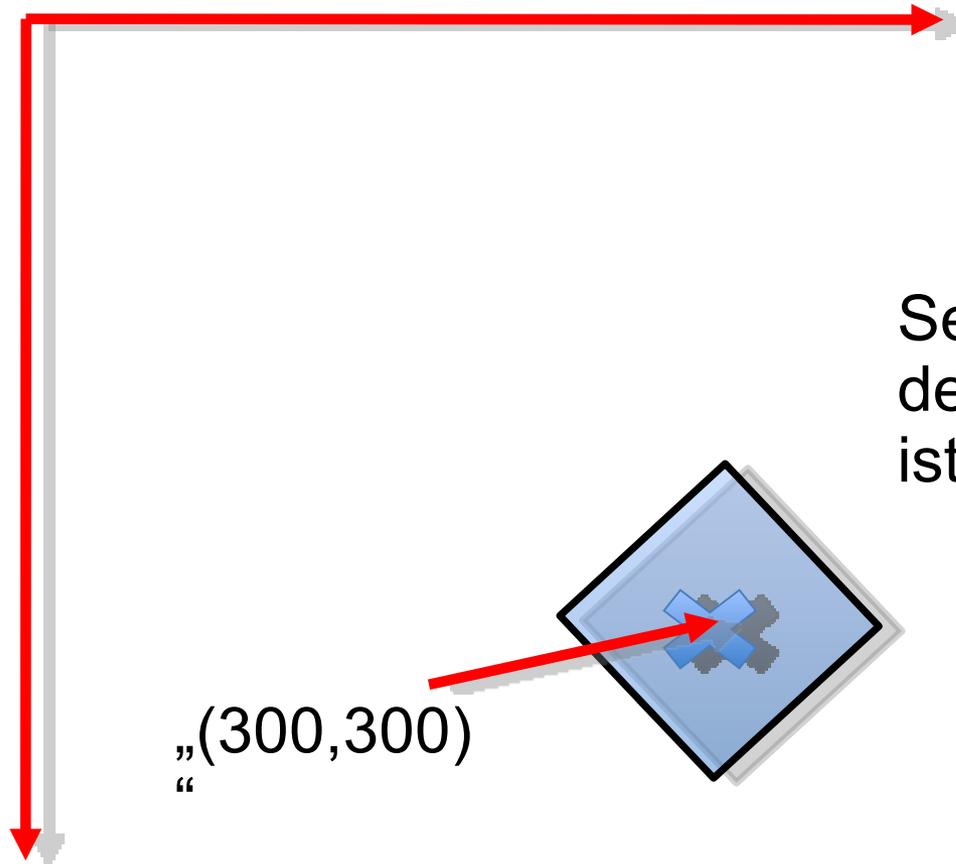


Aufgabe in der Übung



- Quadrat (50 x 50)
 - In der Mitte des Fensters (600 x 600)
 - Um 45° gedreht
- => Was sind die einzelnen Transformationen?

JFrame (600x600)

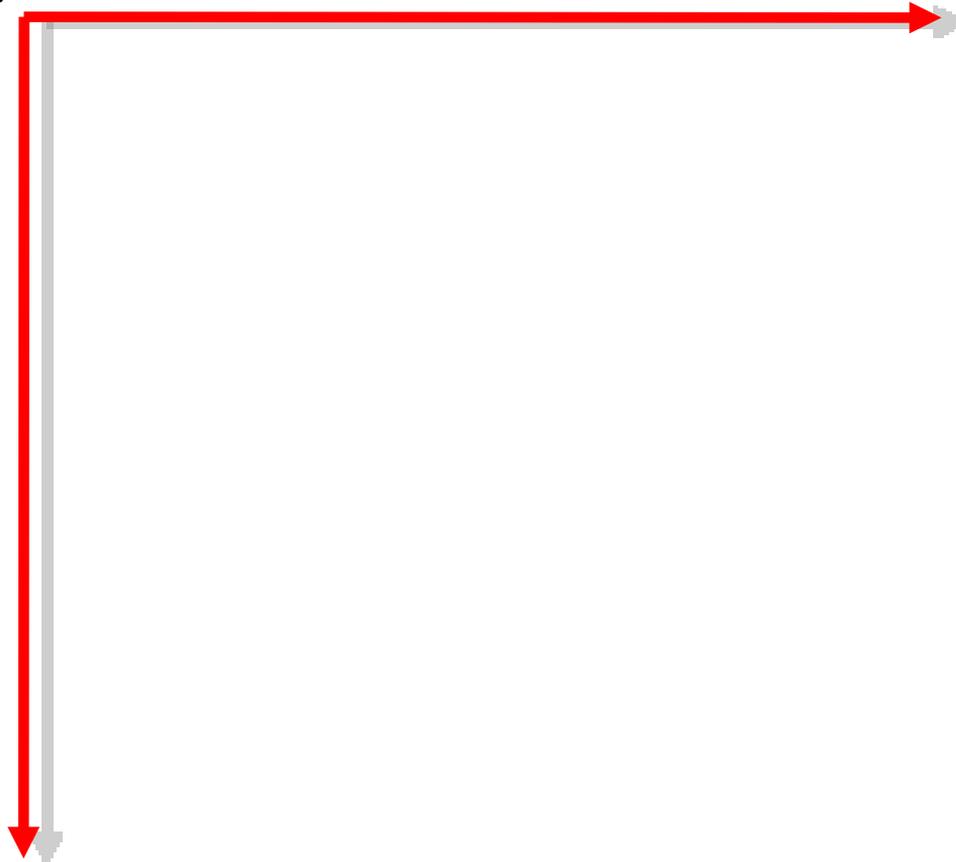


„(300,300)
“

Seitenlänge
des Rechtecks
ist 50x50

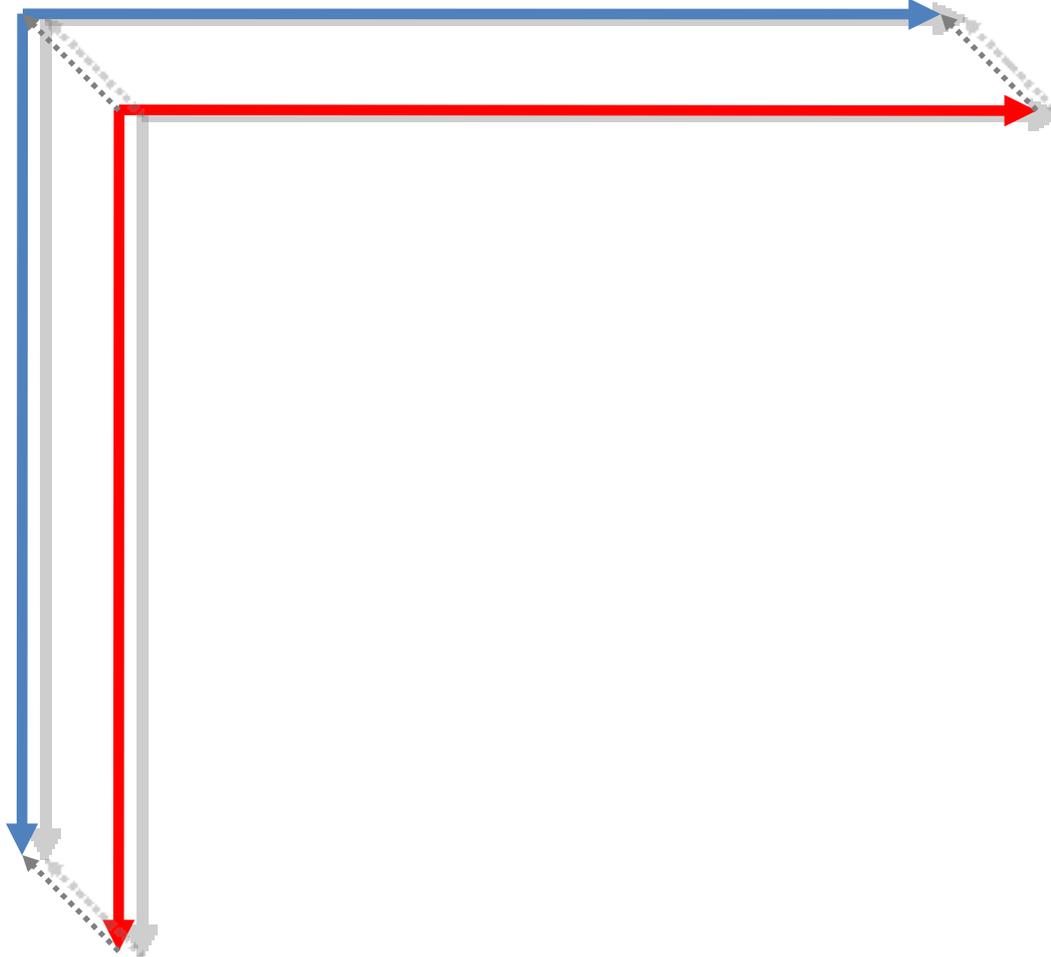
(0,0)

JFrame



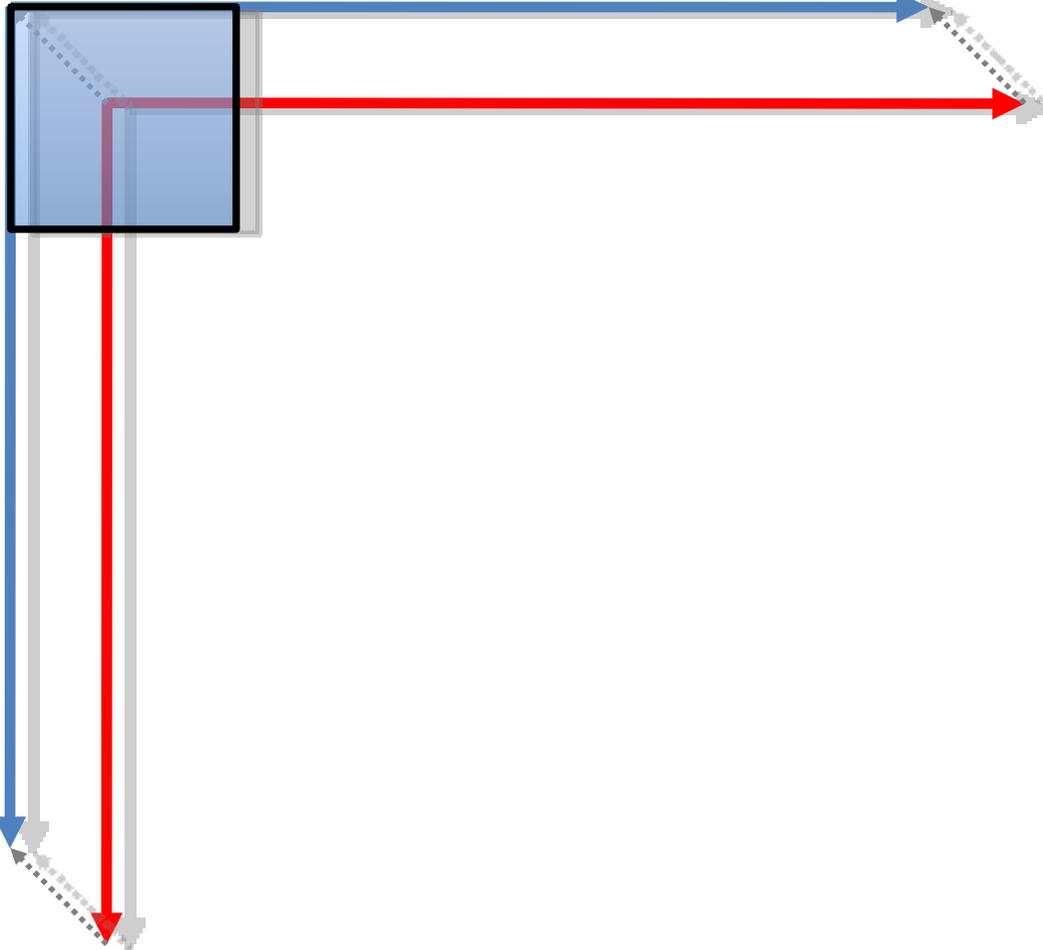
„(-25,-25)“

JFrame



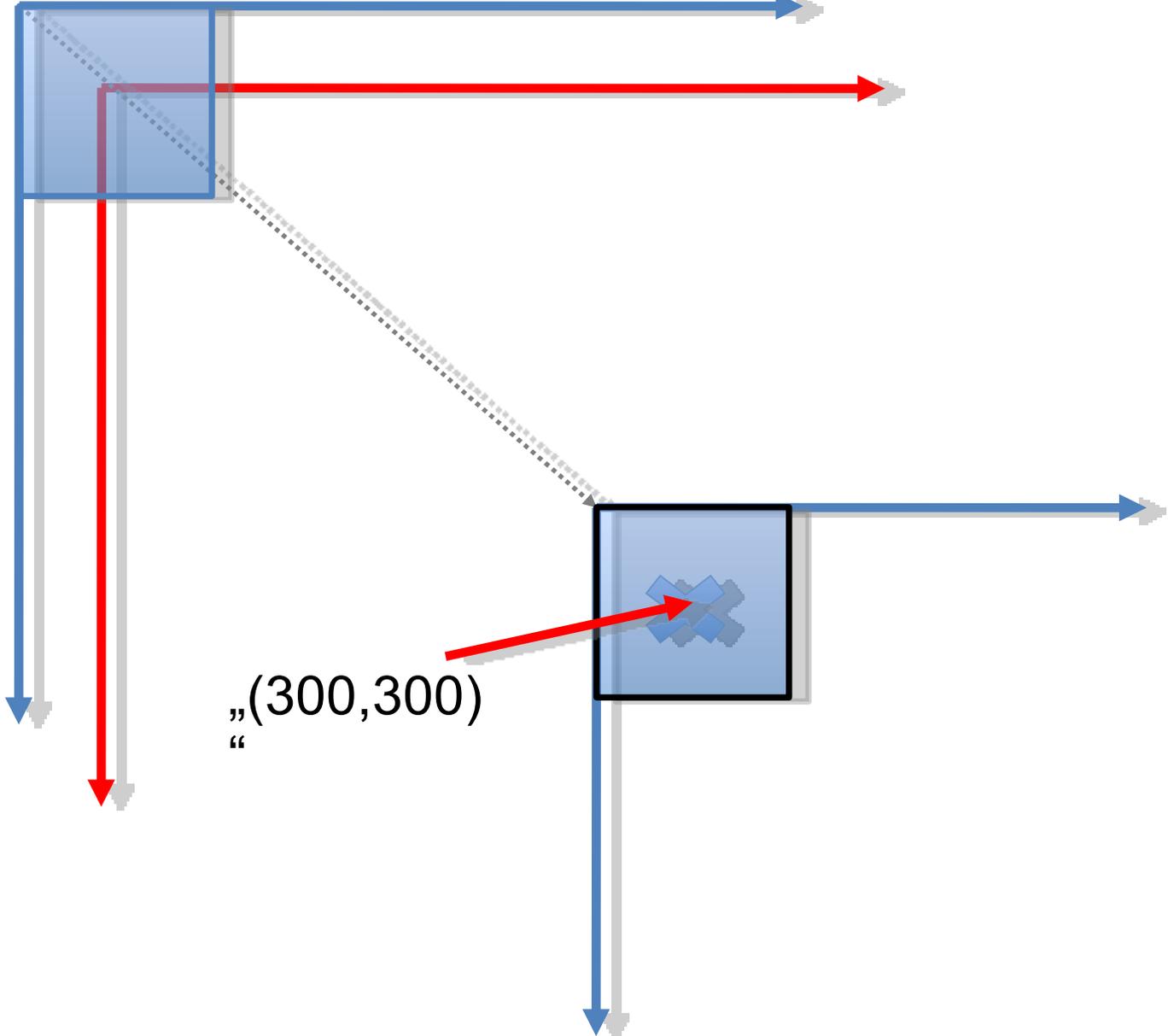
„(-25,-25)“

JFrame



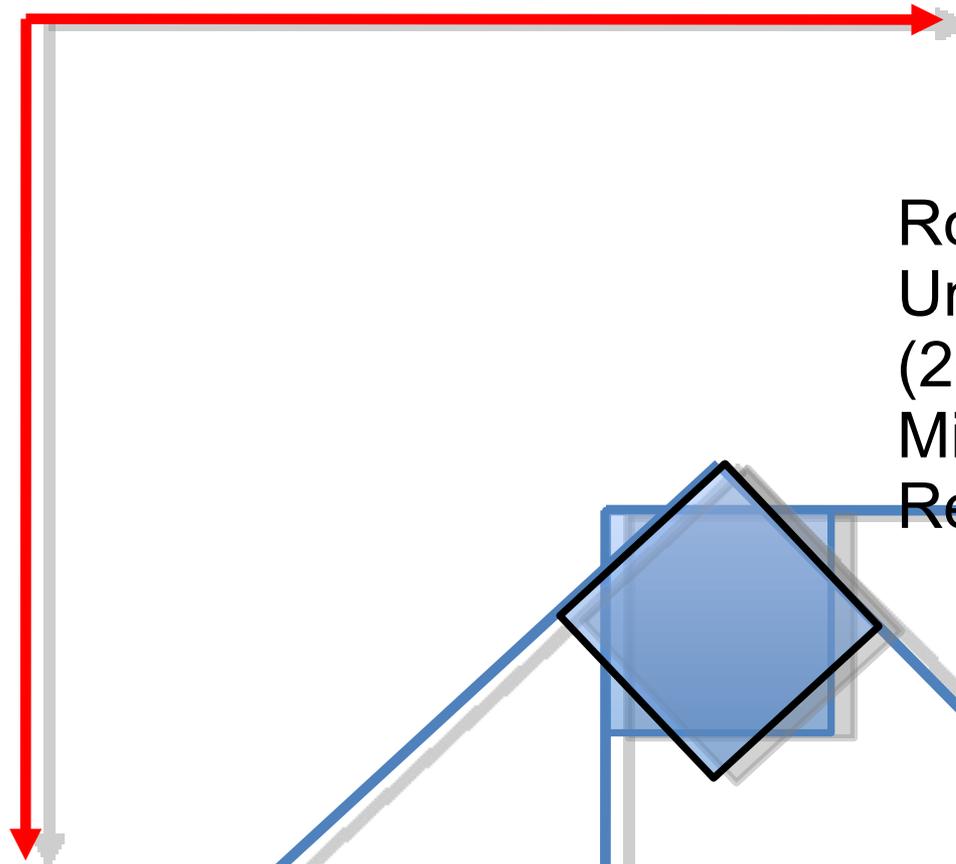
„(-25,-25)“

JFrame

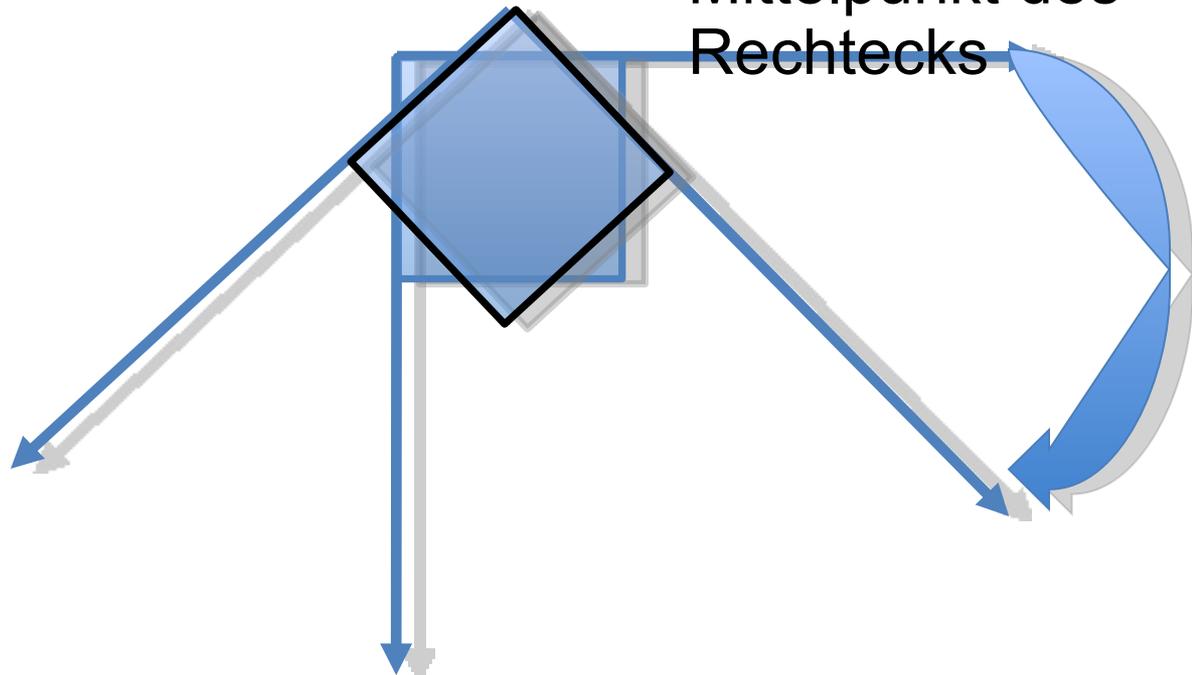


„(300,300)“

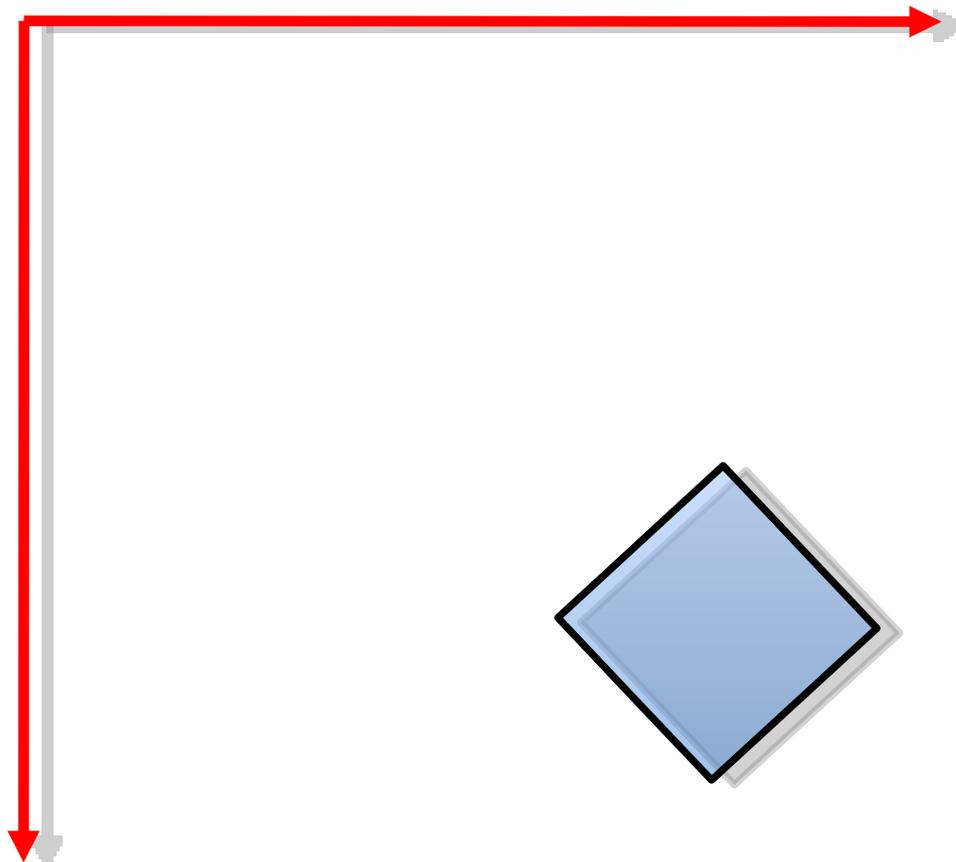
JFrame



Rotation 45°
Um den Punkt
(25,25) =
Mittelpunkt des
Rechtecks



JFrame

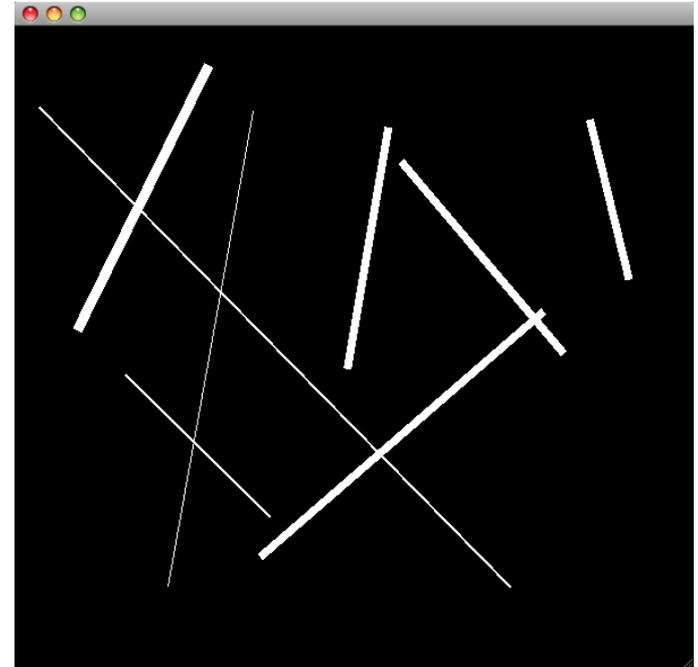


Mausinteraktion

Durch Drücken und Ziehen der Maustaste zeichnet man einen Strich.

Dazu wichtig:

```
JFrame.addMouseListener (some  
MouseListener/ MouseAdapter)
```



MouseListener und MouseEvents

- Folgende Ereignisse kann man als *MouseListener* abfangen:

Method Summary	
void	mouseClicked(MouseEvent e) Invoked when the mouse button has been clicked (pressed and released) on a component.
void	mouseEntered(MouseEvent e) Invoked when the mouse enters a component.
void	mouseExited(MouseEvent e) Invoked when the mouse exits a component.
void	mousePressed(MouseEvent e) Invoked when a mouse button has been pressed on a component.
void	mouseReleased(MouseEvent e) Invoked when a mouse button has been released on a component.

- Der Übergabeparameter vom Typ *MouseEvent* liefert folgende Informationen:

Method Summary	
int	getButton() Returns which, if any, of the mouse buttons has changed state.
int	getClickCount() Returns the number of mouse clicks associated with this event.
Point	getLocationOnScreen() Returns the absolute x, y position of the event.
static String	getModifiersText(int modifiers) Returns a String describing the modifier keys and mouse buttons that were down during the event, such as "Shift", or "Ctrl+Shift".
Point	getPoint() Returns the x,y position of the event relative to the source component.
int	getX() Returns the horizontal x position of the event relative to the source component.
int	getXOnScreen() Returns the absolute horizontal x position of the event.
int	getY() Returns the vertical y position of the event relative to the source component.
int	getYOnScreen() Returns the absolute vertical y position of the event.
boolean	isPopupTrigger() Returns whether or not this mouse event is the popup menu trigger event for the platform.
String	 paramString() Returns a parameter string identifying this event.
void	translatePoint(int x, int y) Translates the event's coordinates to a new position by adding specified x (horizontal) and y (vertical) offsets.

<http://j>

<http://java.sun.com/javase/6/docs/api/java/awt/event/MouseEvent.html>

Mausinteraktion

```
import java.awt.event.*;
import java.util.*;

public class View(){
    private Point p1;
    private Point p2;

    public View() {
        /* ... */
        this.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent e) {
                p1 = e.getPoint();
            }
            public void mouseReleased(MouseEvent e) {
                p2 = e.getPoint();
                repaint();
            }
        });
    }
}
```

View.java

Mausinteraktion

```
public void paint(Graphics g) {  
    /* ... */  
  
    g2.setPaint(Color.RED);  
  
    if (p1 != null && p2 != null) {  
        g2.setPaint(Color.WHITE);  
        g2.setStroke(new BasicStroke(random.nextInt(10))); //optional  
        g2.drawLine(p1.x, p1.y, p2.x, p2.y);  
    }  
}
```

Aufgaben für die Übung

- Zeichnen von geometrischen Formen
- Implementieren des Zeichen-Fensters, das auf Mausinteraktionen reagiert

