

Übung 5

Applets und Vererbung, Programmierparadigmen

1. Schreiben Sie eine Klasse **public HalloApp**, die sowohl als Applet im Browser laufen, als auch als Standalone-Anwendung ausgeführt werden kann. Wie muss die Deklaration der Klasse aussehen, und welche beiden Methoden müssen Sie dazu mindestens in der Klasse implementieren?

Grundgerüst:

```
import java.applet.*; // Applet-Klasse
import java.awt.*; // Graphics-Klasse u.a.

public class HalloApp extends Applet {

    public static void main(String[] args) {
    }

    public void paint(Graphics g) {
    }

}
```

public static void main(String[] args) ist für den Betrieb als Standalone-Anwendung notwendig, und ***public void paint(Graphics g)*** für den Betrieb im Browser als Applet zuständig.

2. Die Anwendung **HalloApp** soll den Text `Hallo, Welt` innerhalb des Browsers ausgeben (graphisch), sofern sie im Browser als Applet läuft. Wenn der Browser minimiert und anschließend neu aufgebaut wird, soll der Text erneut gezeichnet werden. Welche Funktion verwenden Sie hierfür günstigerweise?

Sie können das Applet mit dieser HTML-Datei testen:

```
<HEAD><TITLE>Übung 4 Applet</TITLE></HEAD>
<BODY>
Wenn im Kasten unten "Hallo, Welt" steht, hat es geklappt:
<CENTER>
<APPLET CODE="HalloApp.class" CODEBASE="." width=400 height=100>
</APPLET>
</CENTER>
</BODY>
```

Zuständig für das (neu-)zeichnen ist die Funktion **paint ()**:

```
public void paint(Graphics g) {
    g.setFont(new Font("SansSerif", Font.BOLD, 64));
    g.drawString("Hallo, Welt", 10, 80);
}
```

3. Wird die Anwendung NICHT im Browser oder Appletviewer, sondern als Standalone-Java-Programm ausgeführt, soll sie ein Fenster öffnen und darin ebenfalls den Text `Hallo, Welt` ausgeben. Versuchen Sie dabei, möglichst viel Code von der vorigen Teilaufgabe wiederzuverwenden, ohne alles „doppelt“ zu schreiben. Welches Programmierparadigma verwenden Sie dabei?

```
public static void main(String[] args) {

    // Fenster erzeugen, 400x100 Pixel
    Frame f = new Frame("HalloApp");
    f.resize(400, 100);
    f.setVisible(true);

    // Da paint() nicht static ist, müssen wir erst mal
    // ein Objekt der Klasse "HalloApp" erzeugen, und können
    // dann daraus paint() aufrufen.
    new HalloApp().paint(f.getGraphics());

}
```

Die Wiederverwendung von **paint ()** lässt auf das prozedurale Programmierparadigma schließen.

4. Überlegen Sie, wie lange das Programm wohl laufen wird, bzw. wie (oder ob) man es in beiden Fällen kontrolliert beenden kann. Vergleichen Sie mit der „Textversion“ (`System.out.println`).

*Obwohl **main ()** und **paint ()** nach Ausführung der darin enthaltenen Befehle beendet sind, läuft das Programm weiter, so lange noch eine Referenz auf irgendein Objekt existiert. Dies betrifft das geöffnete Fenster, bzw. das laufende Applet. So lange diese nicht geschlossen werden, läuft das Programm weiter.*

5. An welchen Stellen verwendet das Programm das objektorientierte Programmierparadigma?

Definitiv beim Erben der Eigenschaften von `Applet`, und bei der Erzeugung des Objektes, aus dem `paint()` in der Standalone-Version aufgerufen wird.