Grundlagen der Informatik für Ingenieure I

6. Applets

6.1 Applikationen - Applets

- Java-Applikationen laufen auf dem Rechner ab, auf dem sie gestartet werden.
  - Die Ladeobjekte liegen im lokalen Netz-(NFS-)Dateisystem.
  - Diesbezüglich unterscheiden sie sich nicht von Programmen, die in C, C++ oder Fortran geschrieben sind.
- Java-Applets laufen “in der Umgebung” eines *“javafähigen” WWW-Browsers* ab.
  - Die Referenz auf ein Applet ist in eine HTML-Seite eingebettet.
  - Daraus ergeben sich u. a. folgende Konsequenzen:
    - Ein Java-Applet kann auf einem beliebigen Web-Server im Internet liegen.
    - Der Webbrowser lädt das Applet vom Webserver und bringt es **lokal** zur Ausführung.

6.2 Applets kreieren

- Kreiert man ein Applet, dann kreiert man eine Subklasse der Klasse ***Applet:***

```
public class MeineKlasse extends java.applet.Applet {
....
}
```

- Die ***Applet-Class*** ist Teil des *java.applet package*
- Die ***Applet-Class*** stellt das nowendige Umfeld bereit, damit ein Browser mithilfe der sog. *Java-Virtual-Maschine(JVM)* ein Java-Applet ablaufen lassen kann.
- Das grafische Umfeld wird vom *Abstract Windowing Toolkit (AWT)* bereitgestellt.
- Die ***AWT-Classes*** sind Teil des ***Java.awt package.***

6.2 Applets kreieren

- **Klassenhierachie:**

java.lang.Object

--> java.awt.Component

--> java.awt.Container

--> java.awt.Panel

--> java.applet.Applet

6.2 Applets kreieren

- Im Gegensatz zur Applikation gibt es bei Applets keine ausgezeichnete Methode *main()*.
- Die Klasse *Applet* stellt eine Anzahl *“activity-methods”* bereit.
- Will man diese Methoden für das eigene Applet nutzen, so muss man sie *“überschreiben” (overriding)*.
- *activity methods:*
  - zur Initialisierung

```
public void init(){
...
}
```

    - Diese Routine wird ähnlich der Methode *main()* nach einem Ladevorgang (oder *reloading)* aufgerufen.
    - Sie dient der Initialisierung des Applets und evtl. der Bereitstellung von übergebenen Parametern, etc.

6.2 Applets kreieren

- *activity methods (cont.)*:
  - Starten

```
public void start() {
...
}
```

    - Nach der Initialisierung wird das Applet gestartet.
    - Dies kann während des Lebenszyklus eines Applets mehrfach vorkommen.
    - z. B. falls ein “Leser” eine Seite verlassen hat und später wieder auf sie zurückkommt.

  - Stoppen

```
public void stop() {
...
}
```

Das Gegenteil von *start();* z. B. beim Verlassen der Seite.

6.2 Applets kreieren

◆ *activity methods(cont)*:

- Zerstören

```
public void destroy() {
...
}
```

  – Diese Methode dient der geordneten Terminierung eines Applets.

  – Sie setzt das Applet selbst in die Lage, seine genutzten Ressourcen an das Ressourcemanagement zurückzugeben.

6.2 Applets kreieren

◆ Ein vollständiges Beispiel:

```
import java.applet.Applet;
import java.awt.Graphics;
import java.awt.Event;

public class SimpleClick extends Applet {
    StringBuffer buffer;

    public void init() {
       buffer = new StringBuffer();
       addItem( "initializing... " );
    }
    public void start() {
       addItem( "starting... " );
    }
    public void stop() {
       addItem( "stopping... " );
    }
    public void destroy() {
       addItem( "preparing for unloading..." );
    }
    void addItem( String newWord ) {
       System.out.println( newWord );
       buffer.append( newWord );
       repaint();
    }
```

6.2 Applets kreieren

◆ Ein vollständiges Beispiel (cont.):

```
    public void paint( Graphics g ) {

      //Draw a Rectangle around the applet's display area.
        g.drawRect( 0, 0, getSize().width - 1, getSize().height - 1 );

      //Draw the current string inside the rectangle.
        g.drawString( buffer.toString(), 5, 15 );
    }

    public boolean mouseDown( Event event, int x, int y ) {
       addItem( "click!... " );
       return true;
    }
}
```

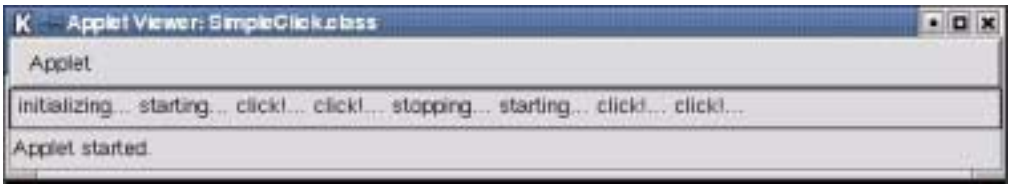

6.2 Applets kreieren

◆ Ein vollständiges Beispiel (HTML-File):

```
<html>
<head>
<title>SimpleClick</title>
</head>
<body>
<p>Output Simple Click:</p>
<applet code="SimpleClick.class" width=600 height=25>
</applet>
</body>
</html>
```

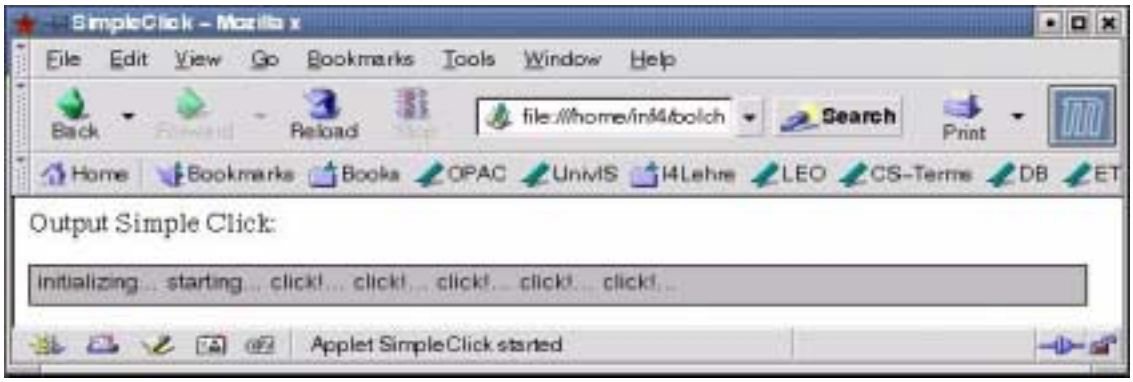
6.2 Applets kreieren

- Ergebnis mit Appletviewer:



6.2 Applets kreieren

- Ergebnis mit Browser (Mozilla):



6.3 Painting

◆ *Painting*

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

- Diese Methode wird immer dann benutzt, wenn etwas auf dem Bildschirm auszugeben ist, z. B.
  - wenn das Fenster aus dem Hintergrund wieder in den Vordergrund kommt oder verschoben wird,
  - das Applet selbst Daten ausgeben will, etc.
- Als Parameter wird eine Objekt der Klasse *Graphics* übergeben, das bereits erzeugt wurde.
- Wir müssen sicherstellen, dass die Definition dieser Klasse, die Teil des *java.awt packages* ist, auch dem Compiler zu Verfügung steht. Dies geschieht mit dem Import-Statement:

```
import java.awt.Graphics;
```


6.3 Painting

◆ Ein Beispiel (HelloAgain!):

```
import java.awt.Graphics;
import java.awt.Font;
import java.awt.Color;

public class HelloAgainApplet extends java.applet.Applet {

  Font f = new Font( "TimesRoman", Font.BOLD, 36 );

  public void paint( Graphics g ) {
    g.setFont( f );
    g.setColor( Color.red );
    setBackground( Color.green );
    g.drawString( "Hello again!", 5, 40 );
  }
}
```

6.3 Painting

◆ Painting

- Dieses Applet gibt einige Buchstaben auf den Bildschirm aus.
- Es ist daher notwendig, eine eigene *paint()*-Methode zu implementieren, die die *default*-Methode überschreibt.
- Der Zustand des *Graphics-Objects*, das *paint()* als Parameter übergeben wird, beschreibt den Graphikstatus des Applets, wie z. B.
  - *colors*
  - *fonts*
- Mit den Anweisungen

```
g.setFont( f );
g.setColor( Color.red );
```

wird dem *Graphics-Object* ein neuer Fontsatz und
ein *Color-Object,* welches die Farbe ‘rot’ repräsentiert, zugewiesen.

6.3 Painting

◆ Painting

- Die *drawString()-method* gibt den *String* im 1. Parameter aus, gemäß *“f”* und *“color.red”,* an Position x = 5(points) und y = 40(points), beschrieben durch den 2. bzw. 3. Parameter.

6.4 Applet - HTML - Umgebung

◆ Der Aufruf eines Java-Applets wird, wie wir bereits wissen, in HTML-Seiten eingebettet. Dafür stellt HTML das ***Applet-Tag*** bereit.

- Beispiel (HelloAgain!):

```
<html>
<head>
<title>This page has an applet on it</title>
</head>
<body>
<h2>Hello Again!</h2>
<p>My second Java applet says:
<br>
<applet code="HelloAgainApplet.class" width=200 heigth=50>
</applet>
</body>
</html>
```


- Ergebnis mit Appletviewer:

- Ergebnis mit Browser (Mozilla):

6.4 Applet - HTML - Umgebung

◆ Dem *Applet-Tag* können verschiedene Attribute mitgegeben werden:

- **code=**`filename.class`
  - Name der Datei, die den Code des Applets enthält.
  - **code** wird verwendet, wenn sich HTML-Datei und Applet-Datei in der gleichen Directory befinden.
  - Ist dies nicht der Fall, gibt man an, wo diese Datei zu finden ist.
  - Hierzu verwendet man: **codebase**
    mit einem relativen Pfadnamen oder einer URL:

```
<applet code="filename.class" codebase="../applets"
    width=120 heigth=100></applet>
```

oder

```
<applet code="filename.class" codebase=
"http://www4informatik.uni-erlangen.de/j-applets"
    width=120 heigth=100></applet>
```


6.4 Applet - HTML - Umgebung

◆ Dem Applet-Tag werden verschiedene Attribute mitgegeben (cont.):

- **width, height**
  Gibt die Größe der Applet-Box an; die Maßeinheit ist *"pixel"*.
- **align**
  left, center, right, top, texttop, middle, absmiddle, baseline, bottom oder **absbottom**
- **href** (*Link*)
  Der Text aus der angegebenen Datei - hier der Sourcecode des Java-Applets - wird beim Anklicken des *Links* dargestellt.

6.4 Applet - HTML - Umgebung

◆ **align**-Beispiel:

```
<html>
<head>
<title>This page has an applet on it, aligned left</title>
</head>
<body>
<h2>Hello Again (Align)</h2>
<p>
<applet code="HelloAgainApplet.class"
width=200 height=50 align=left></applet>
To the left of this paragraph is an applet. It's a
simple, unassuming applet, in which a small string is
printed in red type, set in 36 point Times bold.
<p>
In the next part of the page, we demonstrate how
under certain conditions, styrofoam peanuts can be
used as a healthy snack.
<p>
<a href="HelloAgainApplet.java">The Source</a>
</body>
</html>
```

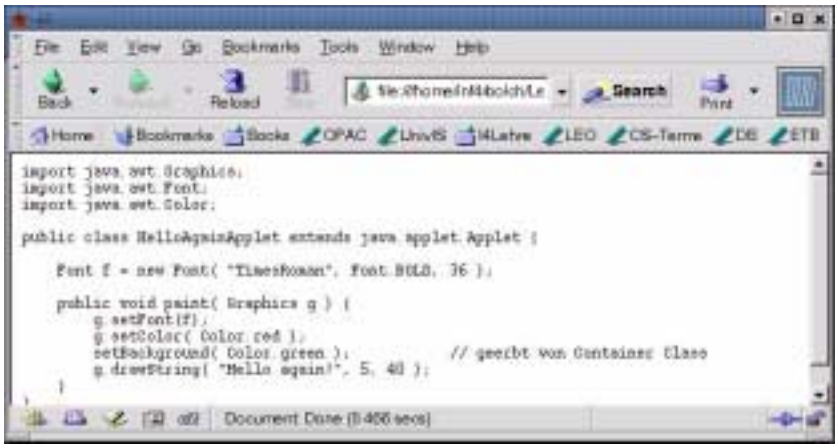

6.4 Applet - HTML - Umgebung

6.4 Applet - HTML - Umgebung

• Ergebnis **align**-Beispiel:

6.4 Applet - HTML - Umgebung

- Ergebnis **`align`**-Beispiel (Source):



6.4 Applet - HTML - Umgebung

◆ Dem Applet-Tag werden verschiedene Attribute mitgegeben (cont.):

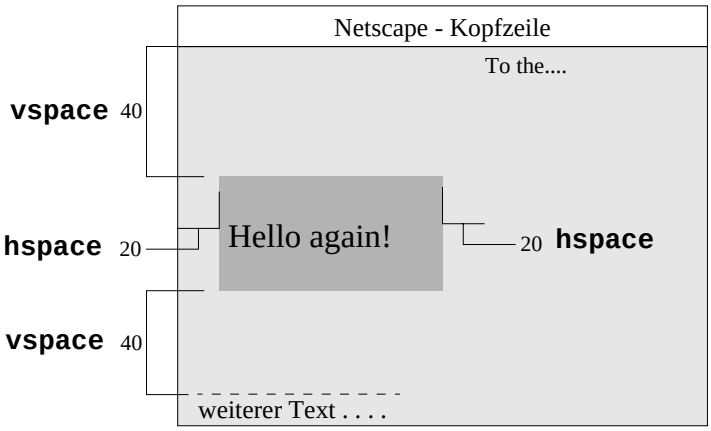
- **`hspace, vvspace`**
  Abstand in *Pixel* zum “umfließenden Text”; Beispiel (HelloAgain!):

```
<html>
<head>
<title>This page has an applet on it, with space around it</title>
</head>
<body>
<h2>Hello Again (Space)</h2>
<p><applet code="HelloAgainApplet.class" width=200 height=50
align=left vspace=40 hspace=20></applet>
To the left of this paragraph is an applet. It's a
simple, unassuming applet, in which a small string is
printed in red type, set in 36 point Times bold.
<p>
In the next part of the page, we demonstrate how
under certain conditions peanuts can be
used as a healthy snack.
<p>
<a href="HelloAgainApplet.java">The Source</a>
</body>
</html>
```

6.4 Applet - HTML - Umgebung

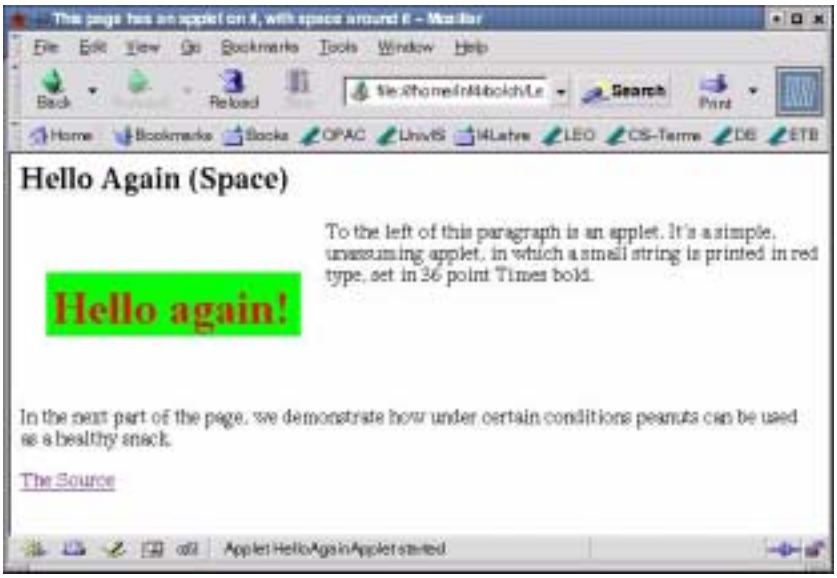
◆ Dem Applet-Tag werden verschiedene Attribute mitgegeben (cont.):

- **hspace, vspace**



6.4 Applet - HTML - Umgebung

- Ergebnis **space**-Beispiel:



6.4 Applet - HTML - Umgebung

◆ Parameterübergabe an Applets:

- Zur Bereitstellung der Parameter gibt es das *HTML-Tag*
  - **param**
    mit den beiden Attributen
    - ***name***; Name des Parameters
    - ***value***; Wert des Parameters

```
<applet code="filename.class" codebase="../applets"
                                        width=120 height=100>
<param name="font" value="TimesRoman">
<param name="size" value="36">
</applet>
```

- Zur Übernahme der Parameter verwendet man in der *init()-method* die Methode *getParameter()* :

```
String theFontName = getParameter( "font" );
int theSize = Integer.parseInt( getParameter( "size" ) );
```


6.4 Applet - HTML - Umgebung

◆ Parameterübergabe an Applets:

- Beispiel (HelloAgain!):

```
import java.awt.Graphics;
import java.awt.Font;
import java.awt.Color;

public class HelloAgainFont extends java.applet.Applet {

        Font f;

        public void init() {
            String theFont = getParameter( "font" );
            int theSize =  Integer.parseInt( getParameter( "size" ) );
            f = new Font( theFont, Font.BOLD, theSize );
        }

        public void paint( Graphics g ) {
            g.setFont(f);
            g.setColor( Color.red );
            setBackground( Color.green );//geerbt von Container Class
            g.drawString( "Hello again!", 10, 50 );
        }
}
```

6.4 Applet - HTML - Umgebung

◆ Parameterübergabe an Applets:

• Beispiel (HTML-File):

```
<html>
<head>
<title>This page has an applet on it</title>
</head>
<body>
<h2>Hello Again (getParameter)</h2>
<p>My Java applet says:
<br>
<br><applet code="HelloAgainFont.class" width=500 height=65>
<param name="font" value="Courier">
<param name="size" value="66">
</applet>
</body>
</html>
```


6.4 Applet - HTML - Umgebung

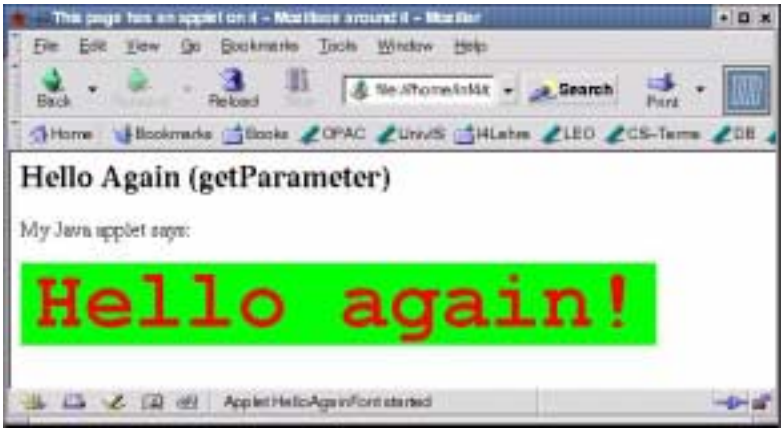
◆ Parameterübergabe an Applets:

• Ergebnis mit Appletviewer:

6.4 Applet - HTML - Umgebung

◆ Parameterübergabe an Applets:

- Beispiel mit Browser (Mozilla):



Notizen