Grundlagen der Informatik für Ingenieure I

2 Java: Java-Einführung

2.1 Java - Entwicklungsgeschichte

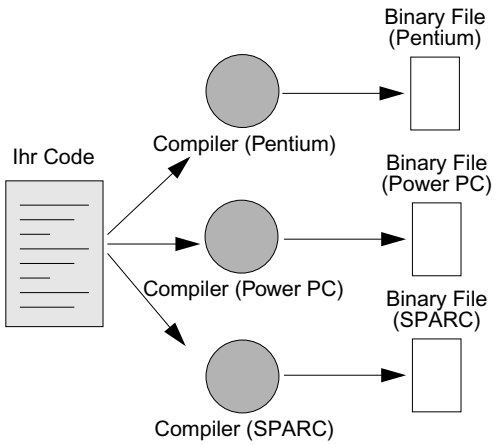
- **Java wurde zu Beginn der 90er Jahre von Sun Microsystems entwickelt.**
- **Ziele von Java:**
  - einfach
  - objektorientiert
  - plattformunabhängig
  - sicher
- **Anwendungsbereiche:**
  - sog. *“Embedded Systems”* - insbesondere für den *Consumer*-Bereich.
  - später: Internetanwendungen

2.2 Java - Entwicklungsgeschichte

- **Gründe für den Erfolg:**
  - Der Einstieg zum Erfolg war die Entwicklung eines Java-Browsers für das Internet (WWW) - ***HotJava***.
  - Der endgültige Durchbruch gelang dadurch, daß die Fa. Netscape in ihren WWW-Browser die ***Java Virtual Machine (JVM)*** integrierte.
  - Hierdurch stand die ***Java-VM*** auf allen gängigen Rechnerplattformen zur Verfügung; ein erstes Ziel - die Plattformunabhängigkeit - war erreicht.
- **Nachteil:**
  - Wie bereits erwähnt haben die Eigenschaften “objektorientiert”, “plattformunabhängig” und “sicher” (letztere beiden Eigenschaften realisiert u. a. durch **interpretative Abarbeitung**) auch ihren Performancepreis.
  - Rechenzeiten sind erheblich länger als bei herkömmlichen Sprachen
  - durch moderne leistungsfähige Prozessoren teilweise kompensiert

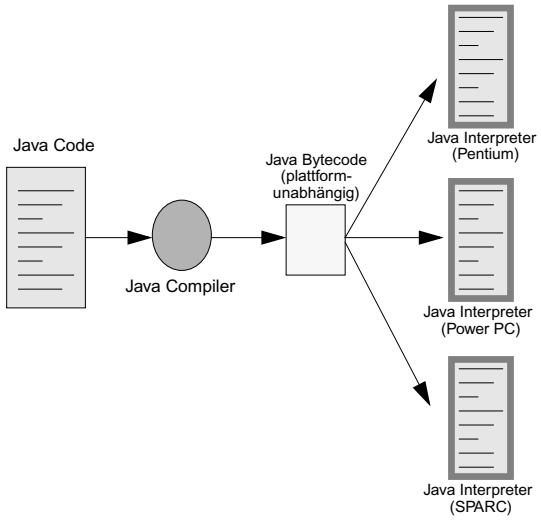
2.3 Java - Eigenschaften

- Java ist **plattformunabhängig**, durch die Bereitstellung einer Virtuellen Maschine (JVM), die den übersetzten Javacode (sog. ***Bytecode***) interpretativ abarbeitet. Die JVM steht auf allen gängigen Rechensystemen zur Verfügung.
  - die traditionelle Methode:



2.3 Java - Eigenschaften

- die Java-VM-Methode



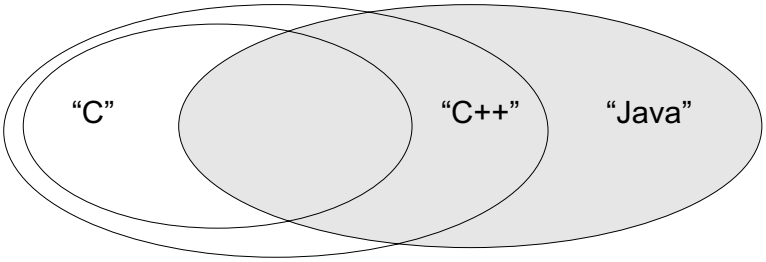
2.3 Java - Eigenschaften

2.3 Java - Eigenschaften

- Java ist **objektorientiert**
  - Viele Eigenschaften von Java stammen aus der Programmiersprache C++
  - C++ ist eine objektorientierte Erweiterung der Sprache C,
  - C ist die Implementierungssprache von UNIX.
  - C entstand Anfang der 70er Jahre ursprünglich als hardwarenahe Systemprogrammiersprache und ist weit verbreitet.
  - Weitere Konzepte von existierenden objektorientierten Sprachen wurden in Java übernommen.
  - die Objektorientierung wird in Java erzwungen. Sie ist nicht quasi optional wie bei C++.

2.3 Java - Eigenschaften

◆ C, C++, Java



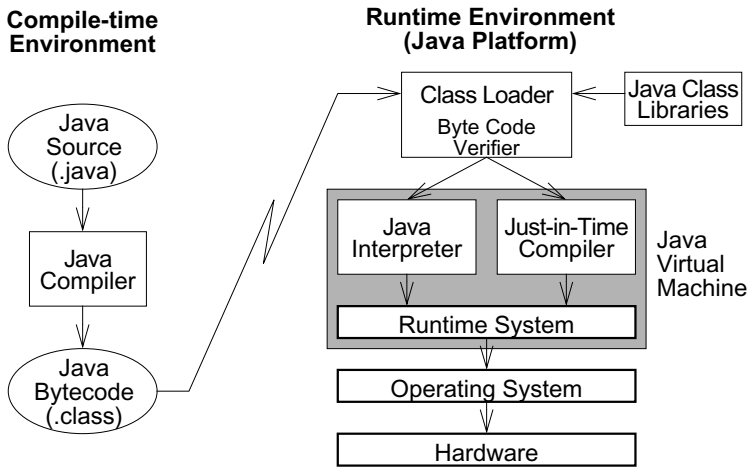
- Java ist **sicher**, **einfach** und **robust.**
  - Auf die Sicherheitsaspekte wird später eingegangen.
  - ob die beiden anderen Aussagen zutreffen, werden Sie im Laufe des Semesters selber beurteilen können.

2.4 Java - Entwicklungsumgebung

- Als Entwicklungsumgebung verwenden wir Sun’s ***JDK*** (***Java Development Kit***), mit
  - allen notwendigen Klassen- (Programm-) Bibliotheken
  - dem Java-Compiler: ***javac***
  - der Java-VM zum Ablauf von Java-Applicationen[1]: ***java***
  - einem Applet-Viewer zum Test von Java-Applets: ***appletviewer***
- Wir verwenden den Browser ***“netscape”*** für Java-Applets, die in ***HTML***-Seiten eingebettet sind.
- Wir verwenden Rechnersysteme unter dem Unix-Betriebssystem.
- Soweit notwendig werden wir im Anschluß an dieses Kapitel und in der 1. Übung alle wichtige Eigenschaften des Systems kennenlernen, insbesondere den Umgang mit dem Dateisystem und den Texteditoren.

1. alle Ihnen unbekannten Termini werden in den nächsten Abschnitten erläutert.

2.4 Java - Entwicklungsumgebung



2.5 Applications und Applets

- Eine **Java-Applikation**
  - ist ein Java-Programm, das keinen ***Webbrowser*** benötigt.
  - Es läuft in einer eigenen Laufzeitumgebung ab, wie Programme anderer Sprachen auch, z. B. in C oder Fortran.
  - Der ***Bytecode*** wird von der ***Java-VM*** interpretativ abgearbeitet.
  - Um den Perfomanceverlust in Grenzen zu halten
    - hat man ***Just-in-Time*** - Compiler ***(JIT)*** entwickelt
    - ***JIT*** - Compiler übersetzen den Bytecode in Maschinencode.
    - Das bringt deutliche Leistungssteigerungen.

2.5 Applications und Applets

- Ein **Java-Applet**
  - ist ein Java-Programm, dessen Aufruf aus einer ***HTML*** - Seite erfolgt.
  - Dieses Programm benötigt zu seinem Ablauf einen ***WWW-Browser*** mit integrierter Java-VM, wie z. B. ***netscape***.
  - Zum Testen steht auch ein im ***JDK*** enthaltener ***appletviewer*** zur Verfügung

2.6 Ein erstes Programm

- Eine erste **Java-Applikation:**
  - ein Programm, das "Hello World" ausdrucken soll
  - wird mit Hilfe eines Editors (z.B. ***xemacs***) eingegeben.
  - Der Quellcode ***(source)*** wird in eine Datei mit Namen **HelloWorld.java** (allgemein: *classname*.java) gespeichert.
  - Kommando: `xemacs HelloWorld.java &`.

2.6 Ein erstes Programm

- Eine erste **Java-Applikation** (cont)**:**
  - Quellcode:

```
/* Hello World, my first Java Application */

public class HelloWorld {

    public static void main( String args[] ) {
        System.out.println( "Hello World!" );
    }
}
```


2.6 Ein erstes Programm

- Eine erste **Java-Applikation** (cont)**:**
  - Compilieren des Quellcodes mit:

```
javac HelloWorld.java
```

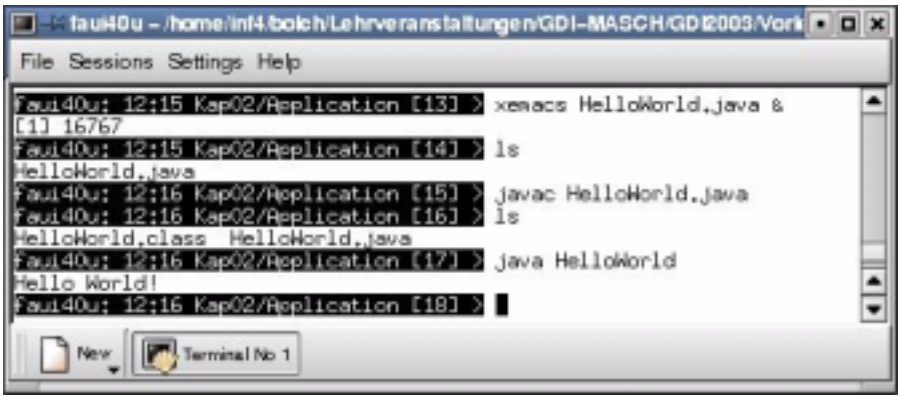
    - Der Compiler hinterlegt den übersetzten ***Source-Code***, also den ***Bytecode***, in eine Datei mit Namen **HelloWorld.class** (allgemein: *classname*.class).
  - Diese Datei wird vom ***Java-Interpreter*** interpretiert mit:

```
java HelloWorld
```

    - die Extension *“class”* wird nicht mit angegeben!
  - Bei Fehlermeldungen: Kontrollieren Sie Namen der Klasse und der Datei. Beide müssen exakt übereinstimmen, auch Groß- und Kleinschreibung!
  - Sollte das einwandfrei sein, suchen Sie nach Tippfehlern im Quellcode; , insbesondere bei den Sonderzeichen “{“, “}”, “;”.

2.6 Ein erstes Programm

- Eine erste **Java-Applikation** (cont)**:**
  - Ergebnis:



2.6 Ein erstes Programm

- Ein erstes **Java-Applet:**
  - wird mit Hilfe eines Editors (z.B. ***xemacs***) eingegeben.
  - Der Quellcode (***source***) wird in eine Datei mit Namen **HelloWorldApplet.java** (allgemein: *classname*.java) gespeichert,
  - Kommando: `xemacs HelloWorldApplet.java &`

2.6 Ein erstes Programm

- Ein erstes **Java-Applet** (cont)**:**

  - Quellcode:

```
/* Hello World Applet, my first Java Applet */

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

public class HelloWorldApplet extends Applet {

    public void paint( Graphics g ) {
       g.drawString( "Hello world!", 5, 25 );
    }
}
```


2.6 Ein erstes Programm

- Ein erstes **Java-Applet** (cont)**:**

  - Compilieren des Quellcodes mit:

```
javac HelloWorldApplet.java
```

    – Der Compiler hinterlegt den übersetzten ***Source-Code***, also den ***Bytecode***, in eine Datei mit Namen **HelloWorldApplet.class** (allgemein: *classname*.class).

  - der ***Byte-Code*** des Java-Applets wird aus einem ***HTML-Script*** aufgerufen,
    – das dann durch einen Browser, z.B. ***netscape***,
    – bzw. den ***appletviewer***

    gestartet wird.

2.6 Ein erstes Programm

- Ein erstes **HTML-Script:**
  - wird mit Hilfe eines Editors (z.B. ***xemacs***) eingegeben.
  - Der Quellcode (source) wird in eine Datei mit Namen **HelloWorldApplet.html** (allgemein: *classname*.html) gespeichert,
  - Kommando: `xemacs HelloWorldApplet.html &`

2.6 Ein erstes Programm

- Ein erstes **HTML-Script** (cont):
  - Quellcode:

```
<html>
  <head>
    <title>HelloWorldApplet.html</title>
  </head>
  <body>
    <p>My Java Applet says:</p>

      <applet code="HelloWorldApplet.class" width=100 height=30>
      </applet>

  </body>
</html>
```

2.6 Ein erstes Programm

- Will man das Applet nur **testen**, genügt der im ***JDK*** vorhandene ***appletviewer:***

  `appletviewer` `HelloWorldApplet.html`

(der ***HTML***-Text wird vom ***appletviewer,*** bis auf die für das Applet relevanten Zeilen, ignoriert!)

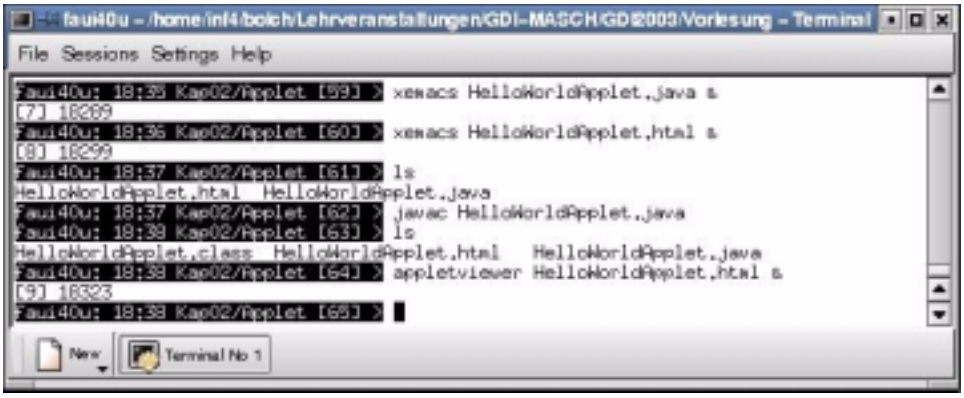
- **Starten des Browsers:**

  z.B. **`netscape`**

- **Starten des Java-Applets:**
  - Wähle aus Menü *File "Open Page"*
  - In dem Pop-Up-Menü klicke *"choose file"* an. Durch Anklicken der *directory*-Namen werden Sie in das *directory* gelangen, in dem Ihre *html*-Datei liegt. Dann diese Datei anklicken und *"OK"* klicken.
  - Als letztes *"Open in Navigator"* anklicken; das Applet wird ausgeführt!

2.6 Ein erstes Programm

- Ergebnis mit **Java-Applet** und HTML-Skript:
  - Kommandozeilen:



2.6 Ein erstes Programm

- Ergebnis mit **Java-Applet** und HTML-Skript (cont):
  - mit Appletviewer:

2.6 Ein erstes Programm

- Ergebnis mit **Java-Applet** und HTML-Skript (cont):
  - mit Browser: