Java 语言与面向对象程序设计

Java 语言与面向对象程序设计 印
清华大学出版社
zmm@cad.zju.edu.cn
教一 3 楼 CAD 实验室

第一课　面向对象软件开发概述
教学目的：集中介绍面向对象软件开发和面向对象程序设计中的基本概念和基本方法，
教学要求：使得对面向对象软件开发方法的体系，原则，基本思想和特点有一定的了解。
学习方法：学习课本知识为主，结合自己曾经学过的 C＋＋等面向对象编程语言的知识.
内容要点：

1．1 面向对象问题求解的提出

面向过程与面向对象技术的关系

面向过程的程序遵循面向过程的问题求解方法。其中心思想是用计算机能够理解的逻辑来描述和表达待解决的问题及其具体的解决过程。数据结构，算法是面向过程问题求解的核心组成。面向对象技术代表了一种全新的程序设计思路和观察，表述，处理问题的方法，与传统的面向过程的开发方法不同，面向对象的程序设计和问题求解力求符合人们日常自然的思维习惯，降低，分解问题的难度和复杂性，提高整个求解过程的可控制性，可监测性和可维护性，从而达到以较小的代价和较高的效率获得较满意效果的目的。

最早的面向对象的软件是 1966 年推出的 Simula I。1980 年提出的 Smalltalk-80 已经是一种比较成熟有效的面向对象的语言了，其后，先后产生了 Lisp,Clascal,Object Pascal, C++等种面向对象的语言。目前使用的最成功的面向对象语言有在 C 语言基础上发展起来的 C++语言和 90 年代新出现的面向对象的编程语言 Java 等。

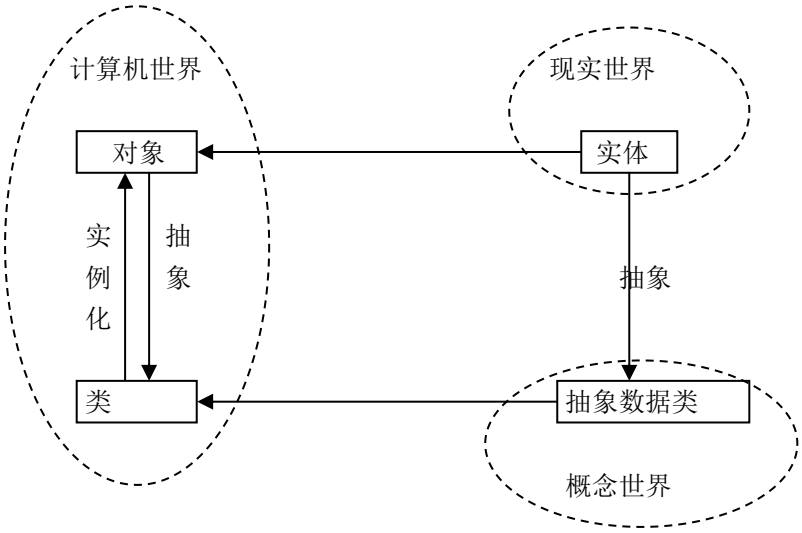
1．2 面向对象问题求解概述

与传统的面向过程的程序设计方法相比，面向对象的程序设计具有如下的优点：

- 对象的数据封装性彻底消除了传统结构方法中数据与操作分离所带来的种种问题，提高了程序的可复用性和可维护性；
- 对象的数据封装性还可以把对象的私有数据和公共数据分离开，保护了私有数据，减少了可能的模块干扰
- 对象作为独立的整体具有良好的自恰性，可以通过自身定义的操作来管理自己；
- 在具有自恰的的同时，通过一定的接口和相应的消息机制与外界相联系
- 继承是面向对象方法中除封装外的另一个重要特性，通过继承可以很方便地实现应用的扩展和已有代码的重复使用

1．3 对象，类与实体

对象的概念是面向对象技术的核心，面向对象技术中的对象就是现实世界中某个具体的物理实体在计算机逻辑中的映射和体现，类是同种对象的集合与抽象。对象，实体与类的关系如下图所示。



在用面向对象的软件方法解决现实世界的问题时，首先将物理存在的实体抽象成概念世界中的抽象数据类型，这个抽象数据类型里面包括了实体中与需要解决的问题相关的数据和属性然后再用面向对象的工具，如 JAVA 语言，将这个抽象数据类型用计算机逻辑表达出来，即构造计算机能够理解和处理的类，最后将类实例化就得到了现实世界实体的面向对象的映射——对象，在程序中对对象进行操作，就可以模拟现实世界中的实体上的问题并解决之。

1．4 对象的属性与相互关系

1．4．1 对象的属性

状态和行为是对象的主要属性

状态是对象的静态属性，如对象（电视机）具有状态（种类，品牌，外观，大小等）

行为是对象的操作，如对象（电视机）可以具有操作（打开，关闭，调整音量等）

1．4．2 对象的关系

对象间可能存在的关系有三种：包含，继承和关联

当对象 A 是对象 B 的属性时，称对象 B 包含对象 A。如电视机与显像管是包含关系，电视机包含显像管。

当对象 A 是对象 B 的特例时，称对象 A 继承了对象 B。如彩色电视机是电视机的特例，彩色电视机对象继承了电视机对象，电视机是父亲，彩色电视机是儿子。

当对象 A 的引用是对象 B 的属性是时，称对象 A 和对象 B 之间是关联关系。所谓对象的引用是指对象的名称，地址，句柄等可以获取或操纵该对象的途径。例如，每台电视机都对应一个生产厂商，如果把生产厂商抽象成对象，则电视机对象应该记录自己的生产厂商，电视机对象与生产厂商对象是关联关系。

1．5 面向对象的软件开发过程

1．5．1 面向对象的分析（OOA）

这里介绍较广泛的 Coad&Yourdon 的面向对象分析（OOA）模型，它有 5 个层次：

- 对象－类层
- 静态属性层
- 服务层
- 结构层
- 主题层

每个层次描述需求模型的一个方面，设计完上述 5 个层次，就得到了完整的 OOA 模型。

1．5．2 面向对象的设计（OOD）

分析阶段（OOA）要明确开发的软件系统是“干什么”的，设计阶段（OOD 模型）则是明确这个软件系统将“怎么做”。面向对象的设计就是 OOA 模型再加入界面管理，任务管理和数据管理四个部分

1．5．3 面向对象的实现

面向对象的实现就是具体的编码阶段，其主要任务包括：

- 选择一种合适的面向对象的编程语言
- 用选定的语言编码实现详细设计步骤所得的公式，图形，说明和规则等对软件系统各对象类的详尽描述。
- 将编写好的各个类代码模块根据类的相互关系集成
- 利用开发人员提供的测试样例和用户提供的测试样例分别检验编码完成的各个模块和整个软件系统

1．6 面向对象程序设计方法的优点

- 可重用性: 指一个软件项目中开发的模块，能够不仅限于在这个项目中使用，而且可以重复地使用在其他项目中，从而在多个不同的系统中发挥伤用。
- 可扩展性：指应用软件能够很方便，容易地进行扩充和修改。
- 可管理性：面向对象的开发方法采用内涵比过程和函数丰富，复杂得多的类作为系统的部件，使整个项目的组织更加合理，方便。

小结

这一课概述了面向对象软件开发的基础知识，包括面向对象问题求解的提出和面向对象问题求解的基本过程。

第二课　Java 概述

2．1 第一个 Java Application 程序

根据结构组成和运行环境的不同，Java 程序可以分为两类：Java Application 和 Java Applet.

两者区别是：Java　Application 是完整的程序，需要独立的解释器来解释运行，而 Java Applet 则是嵌在 HTML 编写的 Web 页面中的非独立程序，由 Web 浏览器内部包含的 Java 解释

器来解释运行。

2．1．1 源程序编辑

JAVA 源程序是以.java 为后缀的简单的文本文件，可以用各种 JAVA 集成开发环境中的源代码编辑器来编写，也可以用其他文本编辑工具（如记事本）。参见以下 Java Application 的例子：

程序 2－1 MyJavaApplication.java

```
        程序源代码                                              语句说明

import java.io.*;                                  //用 import 语句加载已定义好的包 java.io.*，
                                                   //类似于 C 中的#include，要在屏幕上进行输出
                                                       //的程序都要加载该包
public class MyJavaApplication                     // 主类 MyJavaApplication 的头说明
{
    public static void main(String args[])        // 程序入口，主类的主函数 main
    {
        System.out.println(“Hello, Java World!”);  //标准输出函数，在屏幕上打印
                                                   //  “Hello, Java World!”
    }
}
```

解释要点：

- 类体中的类的成员包括域和方法
- 主类必须说明成 public class
- main 方法的说明必须是： public static void main(String args[])
- System 是系统内部定义的一个系统对象；out 是 System 对象中的一个域，也是一个对象；println 是 out 对象的一个方法

2．1．2 字节码的编译生成

程序编辑完成之后，接着要做的步骤是：

```
        命令行                          说明
javac MyJavaApplication.java        //生成字节码
```

解释要点：

- 调用 JDK 软件包中的 Java 编译器程序 javac.exe 后编译所得的目标码称为字节码
- 源代码中定义了几个类，编译结果就生成几个字节码文件: 类名.class
- 含有 main 方法的类就称为主类，类名就是 Java 源文件名

2．1．3 字节码的解释与运行

Java 字节码不能直接运行在一般的操作系统平台上，而必须运行在一个称为“Java 虚拟机”

的操作系统之外的软件平台上，在运行 Java 程序时，首先应该启动这个虚拟机，然后由它来负责解释执行 Java 的字节码：

命令行	说明
java MyJavaApplication	//解释执行字节码

2．2 第一个 Java Applet 程序

对于 Java Applet 程序，字节码文件必须嵌入到另一种语言 HTML 的文件中并由负责解释 HTML 文件的 WWW 浏览器充当其解释器。

2.2.1 源程序的编辑与编译

参见以下 Java Applet 的例子：

程序 2－2 MyJavaApplet.java ：

```
        程序源代码                                                    语句说明
import java.awt.Graphics;                          //将 java.awt 包中的类 Graphics 引入，这样//以
                                                     下的程序中可以引用该类及其中的
                                                       //函数 g.drawString，用于画图和写文字。
import java.applet.Applet;                         //是 Applet 程序都要引入 java.applet 包中的//系
                                                   统类 Applet
public class MyJavaApplet extends Applet           //Applet 程序的主类必须是 Applet 的子类，//这
                                                   里 extends 是表示继承的关键字，//MyJavaApplet
                                                   是 Applet 的儿子
{
    public void paint(Graphics g)                  //用于在窗口中画图和写字的系统函数，
                                                     //只能在图形界面中可用，故 Application 程//
                                                     序不可用。
    {
        g.drawString("Hello, Java Applet World!", 10, 20);   //在窗口中显示文字"Hello, Java
                                                              //Applet World!"
    }
}
```

解释要点：

- Java Applet 中不需要 main 方法，要求程序中有且必须有一个类是系统类 Applet 的子类
- paint 方法表示它将在 WWW 所显示的 Web 页面需要重画时(窗口移动或放缩时)被浏览器自动调用并执行。

2.2.2 代码嵌入

程序 2－2 是 Java Applet 程序，用命令：

```
javac MyJavaApplet.java
```

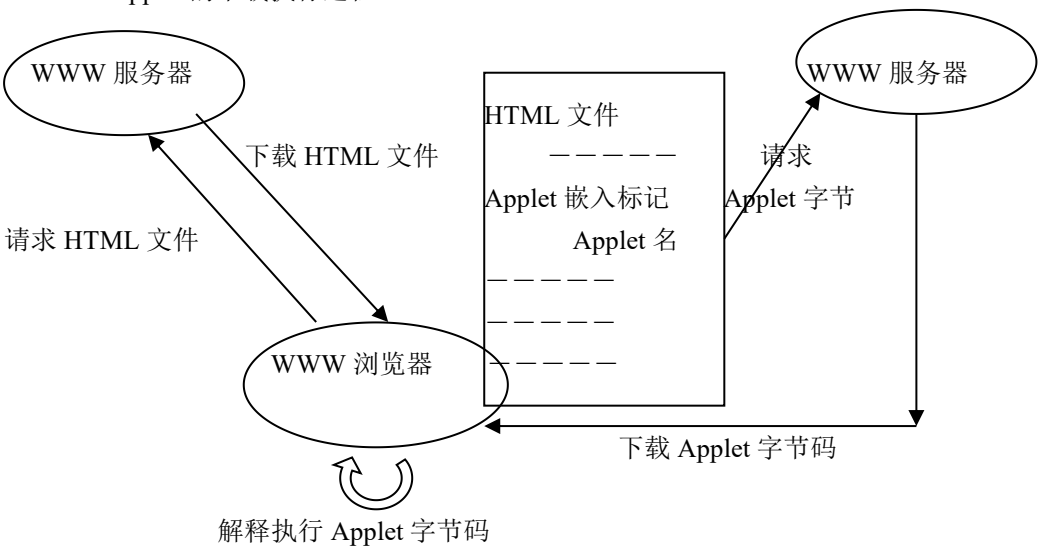
生成字节码后，必须将其字节码嵌入到 HTML 文件中，它可以嵌入在如下的 HTML 文件中：

程序 2－3　AppletInclude.html:

程序源代码	语句说明
<HTML>	//HTML 文件开始
<BODY>	
<APPLET CODE= “MyJavaApplet.class”　HEIGHT=200 WIDTH=300>	//嵌入字节码文件 MyJavaApplet.class //在 Web 页中占用的高度为 200，宽度//为 300
</APPLET>	
</BODY>	
</HTML>	//HTML 文件结束

2.2.3 Applet 的运行

Java Applet 的下载执行过程：



首先将编译好的字节码文件和编写好的 HTML 文件保存在 Web 服务器的合适路径下，当 WWW 浏览器下载此 HTML 文件并显示时，它会自动下载此 HTML 文件中指定的 Java Applet 字节码，然后调用内置在浏览器中 Java 解释器来解释执行下载到本机的字节码程序。

2．3 图形界面的输入输出

图形用户界面（Graphics User Interface）简称 GUI，是目前大多数应用程序使用的输入输出界面，它在图形模式下工作，操作简便，美观。

2.3.1 Java Applet 图形界面输入输出

Java Applet 程序需要在 WWW 浏览器中运行，而浏览器本身是图形界面的环境，所以 Java Applet 程序只能在图形界面下工作。

程序 2－4　AppletInOut.java

```
        程序源代码                                                    语句说明
import java.applet.* ;
import java.awt.* ;                                          //凡使用图形界面，必须加载 java.awt 包
import java.awt.event.* ;                                    //凡使用图形界面的事件处理进行输入输//出，
                                                               必须加载 java.awt.event 包
public class AppletInOut extends Applet implements ActionListener
                                                             // implements ActionListener 说明该类还要//通
                                                             过图形界面的事件处理进行输入输出
{
    Label prompt;                                            //定义标签对象 prompt
    TextField input, output;                                 //定义两个文本框对象 input, output

    public void init()                                       //init()方法是 Applet 类的系统方法，在浏览器
                                                             //调用 Java Applet 程序时自动执行
    {
        prompt = new Label("Input your name: "); //prompt 初始化置为"Input your name: "
        input = new TextField(6);                            //input 置为宽度 6 的空文本框
        output = new TextField(20);                          //output 置为宽度 20 的空文本框
        add(prompt);                                         //prompt 放入图形界面上去
        add(input);                                          //input 放入图形界面上去
        add(output);                                         //output 放入图形界面上去
        input.addActionListener(this);                       //文本框 input 输入结束打回车，则自动转入
                                                             //事件处理函数 actionPerformed
    }

    public void actionPerformed(ActionEvent e)    //事件处理函数，由 addActionListener 转入
    {
        output.setText(input.getText()+ ", Welcome!");   //在 output 文本框中显示 input 中的文
                                                         //字及", Welcome!"
    }
}
```

解释要点：

- 凡是 Java Applet 程序，必须加载 java.applet 包；凡是使用图形界面，必须加载 java.awt 包；凡是使用图形界面的事件处理，必须加载 java.awt.event 包
- 主类必须是 Applet 类的子类，用“extends Applet”来说明

- “implements ActionListener”说明该类是动作事件(ActionEvent)的监听者
- 标签(Label)对象和文本框(TextField)的含义
- 方法 init()在浏览器调用 Java Applet 程序时自动执行
- actionPerformed()动作事件的监听者使用这个方法来处理动作事件。

2.3.2 Java Application 图形界面输入输出

与 Java Applet 不同，Java Application 程序没有浏览器提供的现成的图形界面可以直接使用，所以需要首先创建自己的图形界面。

程序 2－5　ApplicationGraphicsInOut.java

程序源代码　　语句说明

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

class FrameInOut extends Frame implements ActionListener   //它是系统类 Frame 的子类，即建//立
                                                            一个窗口
{
    Label prompt;
    TextField input, output;
    Button btn;                                             //定义按钮 btn

    FrameInOut()                                            //与类名同名，又没有返回的函数是
                                                             //构造函数
    {
        super("Graphics Frame of Java Application");   //窗口标题为"Graphics Frame of Java
                                                        // Application"
        prompt = new Label("Input your name: ");

        input = new TextField(6);
        output = new TextField(20);
        btn = new Button("Close");                      //建立按钮，按钮面板上写"Close"

        setLayout(new FlowLayout());                    //布局策略，即流式排放

        add(prompt);
        add(input);
        add(output);
        add(btn);                                       //将按钮放入窗口中

        input.addActionListener(this);
        btn.addActionListener(this);                    //鼠标点击 btn 进入事件处理函数
                                                       //actionPerformed
```

```
        setSize(300,200);                          //设置窗口大小
        show();                                    //显示窗口
    }

    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==input)                   //如果是文本框 input 打回车进入
                                                   //actionPerformed 的，则
            output.setText(input.getText()+ ", Welcome!"); //在 output 中输出
        else                                       //如果是鼠标点击 btn 进入 actionPerformed
                                                   //的，则
        {
            dispose();                             //关闭窗口
            System.exit(0);                        //退出
        }
    }
}

public class ApplicationGraphicsInOut              //它是主类，因为拥有 main 方法的类才是
                                                   //主类，不管它放在程序的前面还是后面
{
    public static void main(String args[])
    {
        new FrameInOut();                          //调用类 FrameInOut 的构造函数
    }
}
```

解释：

- 窗框类 Frame，通过使用窗框对象，Java Application 程序不必借助浏览器也可以实现图形界面下的输入输出功能

2．4 字符界面的输入输出

所有的 Java Applet 程序都是在图形界面的浏览器中运行的，所以只在 Java Application 程序可以实现字符界面中的输入输出。

程序 2－6　ApplicationCharInOut.java

```
        程序源代码                                          语句说明
import java.io.*;                                  //字符界面的输入输出要加载该包
public class ApplicationCharInOut
{
    public static void main(String args[])
    {
        //char c=' ';
```

```
        String s = "";                                  //说明字串 s

        //System.out.print("Enter a character please:");
        System.out.print("Please enter a string : ");

        try{                                            //异常处理
            //c = (char)System.in.read();
            BufferedReader in =
                new BufferedReader(new InputStreamReader(System.in));
                                                        //键盘输入一串字符入缓冲 in
            s = in.readLine();                          //从缓冲 in 入 s
        }catch(IOException e){};                        //异常处理
//      System.out.println("You are entered charcater: " + c);
        System.out.println("You are entered string: " + s);
    }
}
```

解释：

- 所有的 Java Applet 程序都是在图形界面中运行，只有 Java Application 程序可以实现字符界面中的输入输出
- System.in.read()方法将使程序处于阻塞状态，等到用户输入了一个字符并按回车键后再向下执行
- 输入输出类 BufferedReader 和 InputStreamReader

2．5 Java 语言的特点

Java 语言的五大特点：

- 平台无关性
- 面向对象
- 安全稳定
- 支持多线程
- 简单易学

小结

这一课概述了 Java 语言的基础知识，从一个最简单的 Java Application 程序入手，介绍了 Java 程序开发从源程序到字节码编译生成和运行的整个过程。随后又介绍了 Applet 小程序的开发和运行过程，图形用户界面和字符界面下 Java 程序的基本输入输出方法。