Java 语言程序设计(连载一)

第一篇　Java 导论

杨乔林　(中科院计算所)

在这个关于 Java 语言、编程及应用的连载中，我们依次就 Java 系统的概貌、Java 语言规范、Java 语言编程、Java 语言 API 、Java 类库以及 Java 虚拟机等内容，逐一作详细的介绍。在 Java 词法、语法的叙述部分，我们将与 C/C++ 作比较；在 Java API 和类库的讲解过程中，将结合例子，介绍它们的使用方法。目前，SUN 公司已在 Solaris、Windows 95 等环境上，提供了 Java 的运行环境，本连载使用的是 Win 95 下的 Java 环境，所有的例子，都在它上面编译、调试通过。Web 浏览器使用的是 Win 95 环境下的 Netscape Navigator 2.0 浏览器。

一、Java 简单举例

为了使读者对 Java 编程有个完整的概念，我们首先介绍两类不同的 Java 程序的开发与运行过程。利用 Java 语言，可以开发两种运行方式不同的程序：一种是 Java Applet(小应用程序)，这是嵌入 HTML 文本的 Java 小程序，另一种便是一般的普通应用程序。

1. Java Applet 的开发与运行

在下面的例子中我们编制了一个简单的 Web 页，在上面用 Java Applet 写出 "Welcome to Java" 一行字，其步骤如下：

. 建立保存 Java Applet 和 HTML 文本的目录，例如，Javapage。

. 编写 Java Applet，假定其文件名为 Welcome.java，其内容为：

```
// 定义包含系统类库
import java.awt.Graphics;
import java.applet.Applet;
// 定义类 welcome
// extends 指定 welcome 父类为 Applet
public class welcome extends Applet{
// 定义方法 paint
public void paint( Graphics gx ){
// 引用类 Graphics 方法
gx.drawString("Welcome to java",40,20);
}
}
```

这里，像 C++ 一样，// 引出的是 Java 的注释语句。

. 调用 Java 编译程序，将 Java 文件编译成类文件。其命令行如：

```
javac welcome.java
```

在 welcome.java 没有错误的情况下，java 编译器，生成 welcome.class，这便是与平台无关的 Java 字节代码。

. 编写 Web 页即编写 HTML 文本，并将 Java Applet——welcome.class 嵌入 文本中。这里，我们假定这个 HTML 文件名为 welcome.html。文本内容为：

```
<! - - A Java's HTML Example - - > <! - - this is a comment - - >
<HTML>
<HEAD>
<TITLE> Java's HTML Example</TITLE>
<! - - this is HTML title - - >
</HEAD> <! - - The end of HEAD - - >
<BODY>
< APPLET CODE = welcome.class WIDTH = 200 HEIGHT = 20 >
<! - - This is applet tag - - >
</APPLET> <! - - The end of applet tag - - >
</BODY> <! - - The end of the body - - >
</HTML> <! - - The end of HTML file - - >
```

从上面文本可以看出，在 HTML 中加入 < Applet code = welcome.

class> 标示出对 Java Applet 的调用；<!.....> 是对 HTML 语言加 以说明的注释语句。

. 启动 Web 浏览器，例如 Netscape Navigator 2.0 浏览器装载 welcome.html，在显示屏上便出现：

Welcome to Java

在一般情况下，HTML 页与 Java Applet 都是在

Web 服务器上编制、调试完成的。在本地的情况下，使用文档 URL(Uniform Resource Locator)：

file:C:\
javapage\welcome.html

对于 Netscape Navigator 2.0，在 Win95 的环境中，使用鼠标，也可以将 welcome.html 拖入 Netscape 的 open 图标中，以完成装载任务。

在 Internet 场合，用户从客户机上启动 Netscape Navigator 2.0 等 Web 浏览器，在指定 URL 后，通过网络请求，从 Web 服务器传来下载，在 HTML 处理过程中，遇到 < Applet 或 < App 及随后的讯息，便请求下载 java 的 class。

在加载 java 字节代码后，浏览器利用内嵌的 java 解释器，运行嵌入的 java Applet。

2. Java 应用程序的开发

Java 的应用程序与 Java 的 Applet 的运行方式稍有不同，前者不能嵌入 HTML 文本，而是独立地通过 Java 解释器，以解释方式运行，或利用 Java 及时编译器，将 Java 字节代码及时地编译为平台机器码，象其它语言开发的应用程序一样地运行。开发一个 Java 应用程序，其主要步骤如下：

. 编写 Java 应用程序，这里，我们假定文件命名为 welcomeapp.Java，

其文件内容为：

```
\\定义类 WelcomeApp
class welcomeApp{
\\ 定义类方法 main()
public static void main(String args[]){
\\ 引用系统定义方法
System.out.println("Welcome to Java!");
}
}
```

注意，在 Java 应用程序中，必须要定义方法 main()，这点是与 JavaApplet 不同的。与其 C++ 一样，Java 可以使用 \\ 作为注释行的开始，也可以利用 *.....*\ 引入跨行的注释段落。

. 编译 — 使用 Java 编译程序，编译 Java 程序。其命令为：

javac welcomeapp.java

在没有错误的场合，生成包含 Java 类的字节代码文件—welcmeApp.class。

注意，这个字节代码文件是根据 Java 类的名字命名的，而不是依据被编译的 Java 文件命名。

. Java 应用程序的解释运行，Java 解释运行的命令是：

java welcomeApp

运行结果是在屏幕上印出 Welcome to Java! 一行字。

在前面的命令语句中，我们假定在环境参数 PATH 中，已经包含了 Java 运行系统的路径，例如：C:\Java\bin。

二、Java 语言的特点

1. 适应网络信息和软件开发的需要

World Wide Web 由于采用了 TCP/IP、HTTP 和 HTML 等平台无关的协议，方便了异构服务器、工作站的联结，而使 WWW 在全世界范围迅速流行起来。作为 Web 页制作的 HTLM 语言只是精工于超文本的链接，但没有程序语言的编程功能。用 Java 编制的 Applet 嵌入 HTLM 文本后，由于 Java 字节代码的与平台无关性、实施了对 HTML 的无缝联接，使 Web 页面有了真正的编程手段。Web 浏览器不再只是一种信息检索、浏览工具，而在音频、视频、动画、三维交互环境、企业计算等方面都能充分发挥作用的工具。

对于一种面象对象的软件开发系统，是否能快速开发一个应用程序，除了语言的功能除外，是否带有丰富的类库和例程也是一个重要的因素。Java 系统为了方便用户在网络通讯方面的编程，提供了两个相关的类库：java.browser 和 java.net 类库。在前一个类库中，系统提供了建造浏览器所需要的各种部件；而在第二个类库中，系统提供了各种网络规程、网络编程 API 和网络编程相关的类及其方法。例如，引用 java.net.URL 的方法，便可以将你的程序与某一网络地址相联结，从中下载你所需要的文件。

在网络软件的开发中，进行双向通讯的程序对(一个称为服务程序，另一个称为客户程序)的开发，也是客户机/服务器系统中，软件开发的基本任务。利用 Java 类库 Java.net 所提供的 socket 和 ServerSocket 便可很快做到这点：引用库中的这两个类，学会如何使用它们读写方法，便可实现二个程序在网络中的双向通讯。

2. Java 的平台无关特性

Java 虚拟机(Java Virtual Machine) 为 Java 语言的运行，定义了一个通用的、与具体软件/硬件平台无关的虚拟计算机。对于一般的程序设计语言，例如 C、C++

等，其数据类型、字长等，随其实现的平台密切相关。例如，对于整数，就有16位、32位之分；浮点数也有不同的表示格式等等。其次，对于一般的语言，编译后的代码就是运行平台的机器码，对于不同的芯片或不同的操作系统，这些机器代码，没有丝毫兼容性可言。

在Java系统中，当Java程序编译时，编译器生成的并不是运行平台的机器代码，而是称之为"字节代码"(Bytecode)的Java虚拟机的指令代码。由于Java虚拟机的定义与平台无关，因而字节代码也是一种具有良好移植性的平台无关的中间格式代码程序。

在Java程序运行时，利用平台上的Java解释程序，也就是Java虚拟机的仿真运行程序，对字节代码实施解释性运行。

在网络的环境下，用户从Web页服务器上，下载包含有Java Applet的Web页，不管用户使用的什么平台(例如是Win3.1、Win95、UNIX、Solaris或是Mac OS)，只要Web浏览器支持Java，都能正确运行。

3.Java的安全特性

Java语言对Web页的扩展，其精髓是使人们能在网络上发布和传递可执行代码，因而给Web注入了动态和交互的功能。在这种情况下用户经常要从网络上下载已经编译过的Java代码。这些代码是否带有病毒，会不会带有危害系统安全的操作？没有确实的保证，冒然运行这类代码，后果是很危险的。幸好，Java系统在设计和开发过程中，充分注意到安全问题，为它处处把关设防。

首先在语言设计上，Java语言取消了指针类型和数据类型的隐式转换，对内存的访问设置了严格的限制。在java中，由于没有指针类型，代之是真正的数组和串，并对数组的下标实施严格的检查，也不能通过强制转换，"制造"出指针，去访问不该访问的内存区域，或重写其它用户或系统的内存区域，从而为系统提供了一道安全屏障。

其次，Java系统对下载的Java字节代码，实施严格的检查。虽然Java语言规范和Java的编译器，保证了字节代码是不会违反Java规定的安全原则的。但是通过网络，从Web服务器上下载的那个Java字节代码程序是否包含有违反安全性的隐患，是不是一个真正的Java字节代码，是否被病毒感染过的等等。很自然，在字节代码执行前实施检查是十分必要的安全措施。通过字节代码的检验，确保代码段不存在操作数栈的上溢或下溢，字节代码指令参数类型正确无误，类对象访问(包括公共、私有或保护)合理合法等。

Java的下一道安全防线是类装载器(或字节代码装载器)。Java的动态内存分配和类的动态方式的装载，也是Java安全的重要措施。Java的内存分配，不是在编译时决定，而是在运行时，由类装载器实施分配。这样，Java系统完全杜绝了编程人员访问任意内存空间的危险。

类装载器的工作是将使用的各种类分别装载到各自的名字空间。来自本地文件系统的类，装在一个统一的名字空间；而对网络传送过来的类，每个网络源点名分别对应一个独立的名字空间。在程序中，当某个类引用另一个类时，字节装载器材首先从本地类的名字空间寻找，若不存在，再从其它源点的名字空间中查找。这也意味道下载的Java Applet不会写入用户机器的磁盘，也不能访问你的文件系统和任意查看你的内存。

在Java的运行系统中，设有安全管理员(Security Manager)，在发现对系统安全有威胁的操作时，安全管理员使用安全异常处理(Security Exception)，中断该操作的运行，以保障系统的安全运行。

在Java类库中，系统提供了Security Manger类，用户可以通重载它的方法为自己应用程序建立适合自己需要的安全管理员。这里需要说明的是，我们不能为Java Applet按排新的安全管理员，这是因为Java Applet的安全管理完全受控于它的Web浏览器或是Applet Viewer的安全管理。

对于金融、贸易等领域，前述的网络安全措施还是不够的，根据SUN公司的承诺，将来Java版本，还要使用公开密钥以及其他加密技术，确保网络安全的特殊需要。

4. Java的面向对象特性

面想对象技术的关键在封装、继承和动态联编。下面我们就这二个问题论述Java的面向对象特性以及和C++等语言的区别。

在Java语言中，类(class)是一段代码，它包含了对象的状态(数据)和对象的行为(对这些数据的操作)—方法。这就是Java对象的封装特性。和C++不同，Java不再支持独立于类之外的变量和函数，所有的变量和函数一定要封装在某一个类里面，表现出更为彻底的封装特性。

在继承方面，Java和Smalltalk一样，对继承做了简化，只支持单继承，这样增进了语言的稳定性。为了弥补缺乏多继承方面的不足，Java引入接口(Interface)的

概念。接口是方法原型的封装，它可由若干个类(不同的继承链上的类)共享。接口以运行时的较少的系统开销，完成多继承相似的任务。引入接口还有另一个好处是，可以减轻程序员的负担，程序员无需把各种对象的类型、继承关系弄清，也能将接口部分的程序编好。

动态联编的工作，在 Java 中做的更加出色，正如前面我们所介绍的那样，类的真正的联编不是在编译程序中，而是在运行前的类装载程序中进行。对于 Java 的这种工作方式，只要类库中的原型的定义保持兼容，类库升级时，程序无需重新编译，也能正确执行，而对于是 C++ 等语言，类库的升级，程序必须重新编译，才能得到新类库赋予的功能。

从程序风格来看 Java 保留了大量 C++ 有用的特点，但也大刀阔斧地抛弃了 C++ 如下语言成份：

. 结构　　. 指针　　. 独立函数
. 独立变量　　. 多重继承　　. 操作符重载
. 类型自动转换　　. Goto 语句　　. #define 语句

5. Java **的多线程与内存自动管理**

与其它编程语言比较，支持多线程，又是 Java 语言另一个突出的优点。所谓多线程是指在一个程序中可以同时执行一个以上的线程，例如，在一个程序中同时存在一个复杂的计算线程，一个用户交互线程等等。在这种多线程程序中，任务得到合理的调配，即用户交互的响应不会因为复杂计算而减慢，而复杂计算也不会因为用户的交互而停止进行。特别对于多 CPU 系统多线程程序的开发，特别是有价值的。

在 Java 语言中，提供了 Threadsafe、Synchronized 等修饰词，大大方便了多线程程序设计。Threadsafe 用来修饰变元，使该变元在某一线程使用时，别的线程不能异步地对它进行修改。同样，修饰词 Synchronized 用来修饰方法或代码段，为计算机资源加锁，当某线程访问某一资源时，被同步的方法，或代码段、不能同时地使用这一资源。

自动进行内存垃圾收集和清理是 Java 内存管理的主要特征。在 Java 语言中，类是对象的状态及其行为的模板，类的实例化产生对象、所有实例的对象都安置在内存垃圾收集堆中，只有对象不再使用后，内存垃圾收集清理程序，才将其清除，释放它在原先占用的全部内存。因此，Java 程序员无需象其他语言程序员那样，经常考虑如何使用 malloc()、free() 等函数为自己的程序管理内存。

在 Java 系统中，内存垃圾收集程序是一个后台程序，它以低的优先度进行管理内存。

6. Java **的 GUI (图形用户界面) 程序设计**

在 Apple 的 Mac 图形用户界面推动下，X - window、Windows 3.1、Windows95 等相继出现，图形用户界面成了用户界面的主流。虽然 Java 的用户界面的建立还没有达到可见即可得或 Visual 的水平，但 Java 的 AWT(Abstract Window Toolkit)类库包，提供了图形、图象处理以及图形用户界面所需要的部件。

首先，AWT 提供标准 GUI 构件，例如，Button 、List 、TextArea 、TextField、Checkbox、Choice、Label、Scrollbar 等。用户通过改变参数，将这些 GUI 构件定制化，适合自己的需要。

AWT 除了上述 GUI 构件类外，还提供 Graphics、Image、Event、Font 和 color 等类。基本图元(例如直线、矩形、圆、椭圆、多边形等)的绘制，就是在 GRAPHICS 类中提供。利用 Image 类的方法，进行图象装载、处理及动画的演示。

在 AWT 还包含 GUI 容器类元件，例如：Window、menubar 等。所谓容器就是在它上面，根据用户需要安置基本的 GUI 构件。

由于 Java 的 AWT 类库，全面提供了构筑 GUI 的各种构件，虽然没有 Visual 的 GUI 编辑功能，但是利用 AWT 构造 GUI 还是比较方便的。

7. Java **方法的其他语种编程**

这里 Java 方法其他语种编程是指利用其他语言，例如 C 语言，编写 Java 类的方法，并集成到 Java 程序中。这种利用其他语言编写 Java 方法，在 Java 中称之为 Native Method。

Java 方法其他语种编程，要解决如何从 Java 将对象封装的数据作为参数传给别的语种所编写的方法；如何从这些方法返回数据给对象；Native 方法如何访问 Java 对象，如何建立 Java 对象，如何进行异常处理等等。下面以 C 语言的 Native 方法编程为例，说明其工作步骤：

. 编写 Java 程序
. 编译 Java 源代码，生成 Java 字节代码
. 利用 Javah 处理 Java 类，生成 C 程序的头文件
. 利用 Javah 处理 Java 类，生成 Stubs 文件
. 编写完成 Java 方法的 C 函数
. 将上述几种 C 语言文件一起编译，建立动态可装载的库文件。

Java 的其他特点，例如，Java 的短小精悍、Java 的健壮性以及 Java 的中性的结构等，在以后再作介绍。