

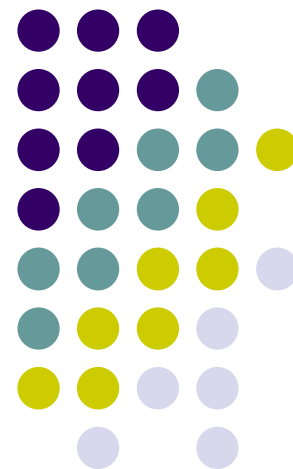


# 程序设计

毛莺池

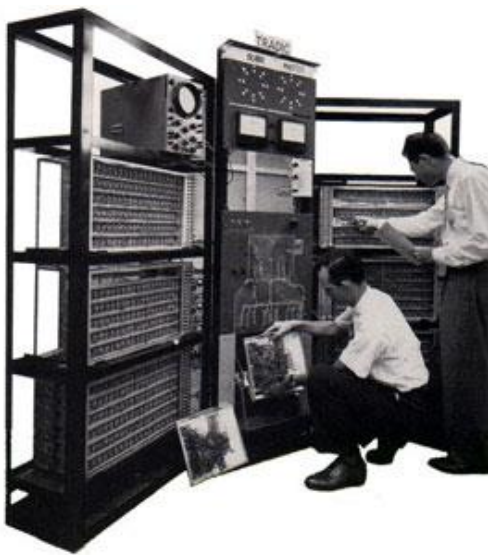
[yingchimao@hhu.edu.cn](mailto:yingchimao@hhu.edu.cn)

河海大学计算机与信息学院  
勤学楼4209

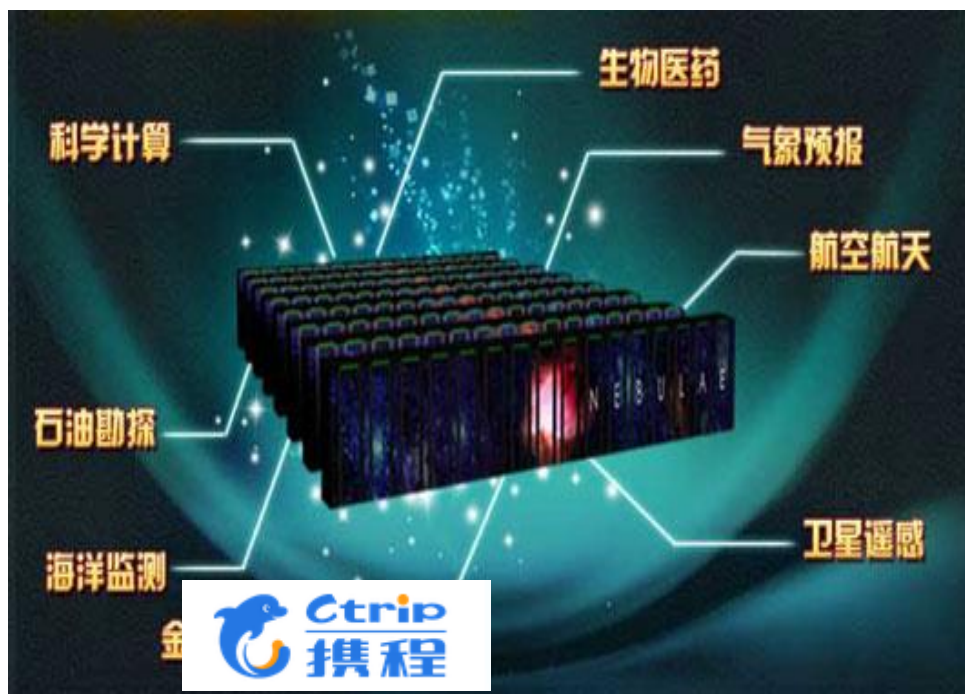
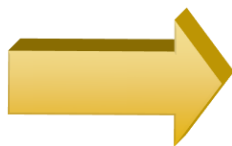


# 问题1

□ 计算机是什么？



数值计算

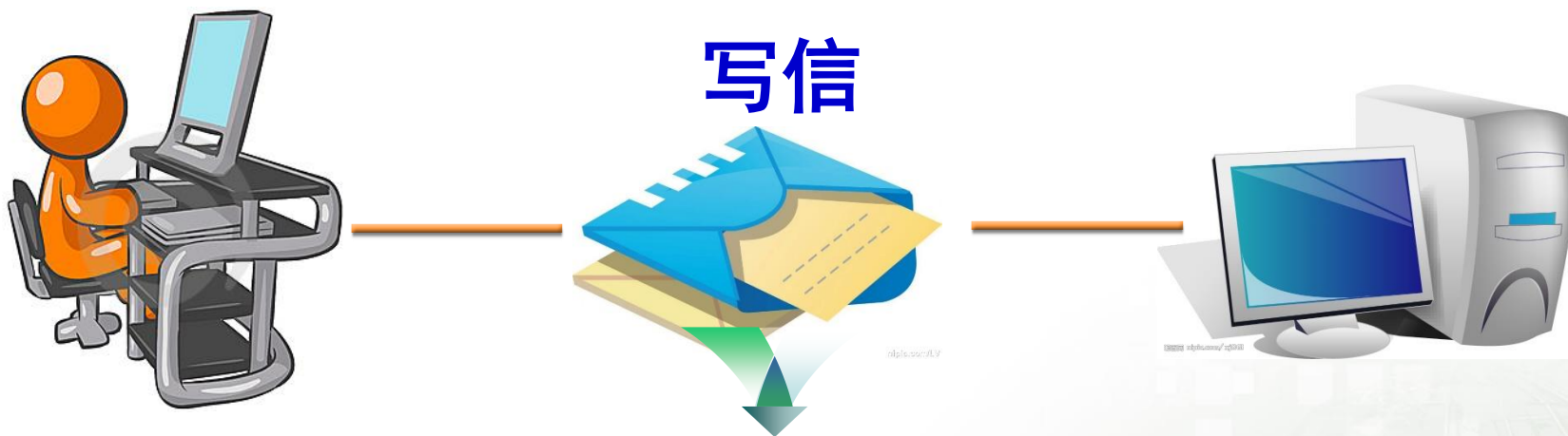


Apple Store

□ 计算机是**人类能力最强、智慧最高的工具**。

# 问题2

□ 人如何使用计算机？



- “谁” — 操作的对象（定义）
- “做” — 操作的过程（执行处理序列）
- “什么” — 操作的结果（输出）

写信的过程：程序设计

写信的工具：程序设计语言

表达能力与易使用

# 问题3

□ Java是什么？

□ 程序设计语言

□ 面向对象思想

□ 表达能力强

□ 开发平台



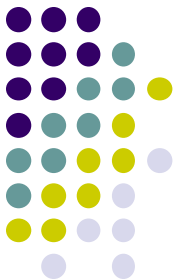
主要面向计算机专业学生





# 课程简介

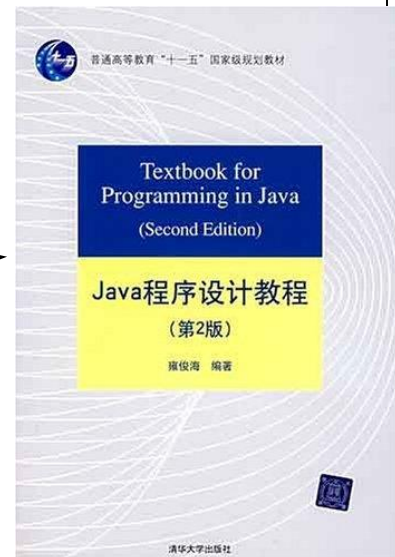
- **Java语言程序设计**
  - Java基础知识
  - 面向对象的方法
- **先修课程**
  - 一门程序设计语言（**C/C++**、**、**、**、**）
- **课程安排：48+16**
  - 讲课**48**学时，上机**16**学时



## ■ 教材

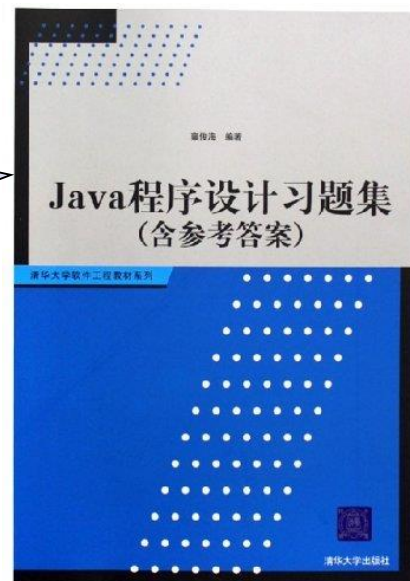
### 《Java程序设计教程》（第2版）

作者：雍俊海  
清华大学出版社



### 《Java程序设计习题集》

作者：雍俊海  
清华大学出版社



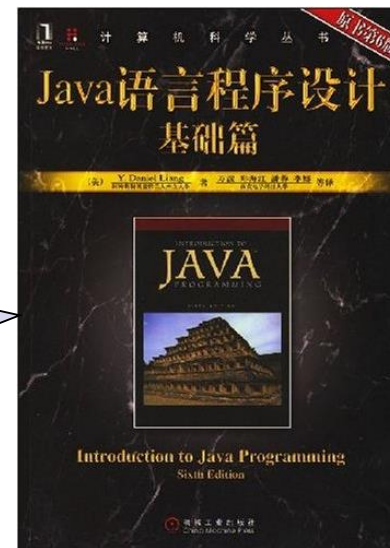




## ■ 参考书目

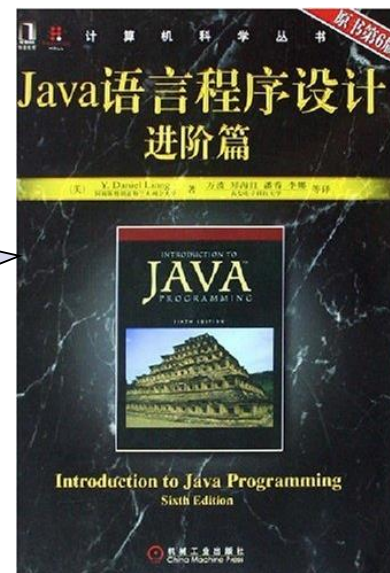
### Java语言程序设计:基础篇(原书第6版) [ Introduction to Java Programming ]

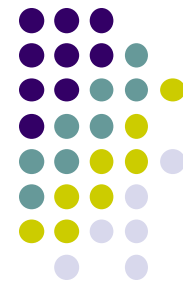
作者: Y.Daniel Liang  
译者: 万波 郑海红 潘蓉 李娜  
机械工业出版社



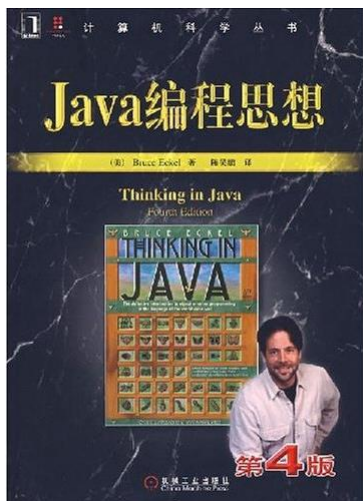
### Java语言程序设计:进阶篇(原书第6版) [ Introduction to Java Programming ]

作者: Y.Daniel Liang  
译者: 万波 郑海红 潘蓉 李娜  
机械工业出版社



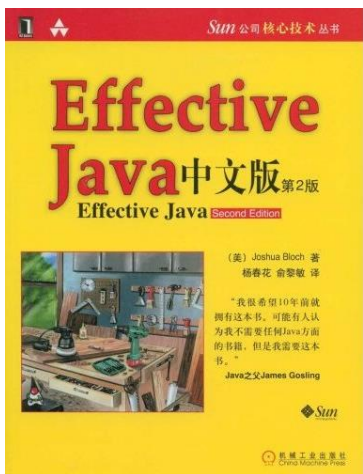


## ■ 进一步学习



### Java编程思想 [ Thinking in Java (TIJ) ]

作者: **Bruce Eckel**  
译者: 陈昊鹏  
机械工业出版社



### Effective Java(第2版)

作者: **Joshua Bloch**  
译者: 杨春花 俞黎敏  
机械工业出版社





## ■ 网络资源

- **Java论坛:** <http://www.matrix.org.cn/forum.shtml>
- **Java中文世界论坛:** <http://www.chinajavaworld.com/index.jspa>
- **CSDN论坛:** <http://community.csdn.net/>
- .....
- **Java主页:** <http://www.oracle.com/technetwork/java/index.html>
- **Java官方教程:** <http://download.oracle.com/javase/tutorial/>
- .....
- [www.google.com](http://www.google.com) / [www.google.com.hk](http://www.google.com.hk)
- [www.baidu.com](http://www.baidu.com)

# Java程序设计



- Java 语言的思路
- Java 程序设计基础（结构化程序设计）
- Java面向对象程序设计
- Java编程

# 第一章 Java概述



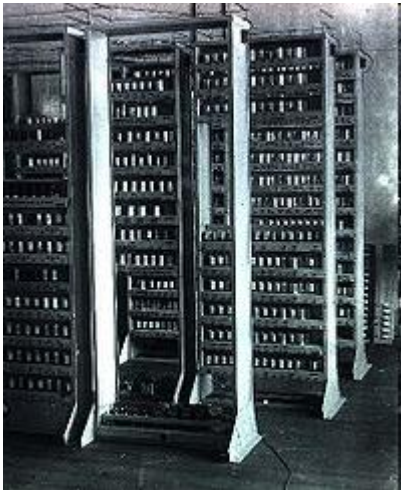
## ■ 学习目标

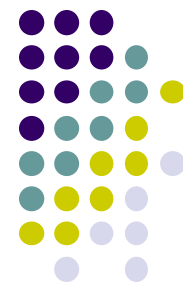
- 计算机的基本组成、程序和操作系统
- Java简介
- 第一个例子：Hello World
- 创建、编译、运行Java程序
- 理解Java运行环境
- 学习Java程序的基本语法
- 在控制台和对话框中显示输出

# 什么是计算机？



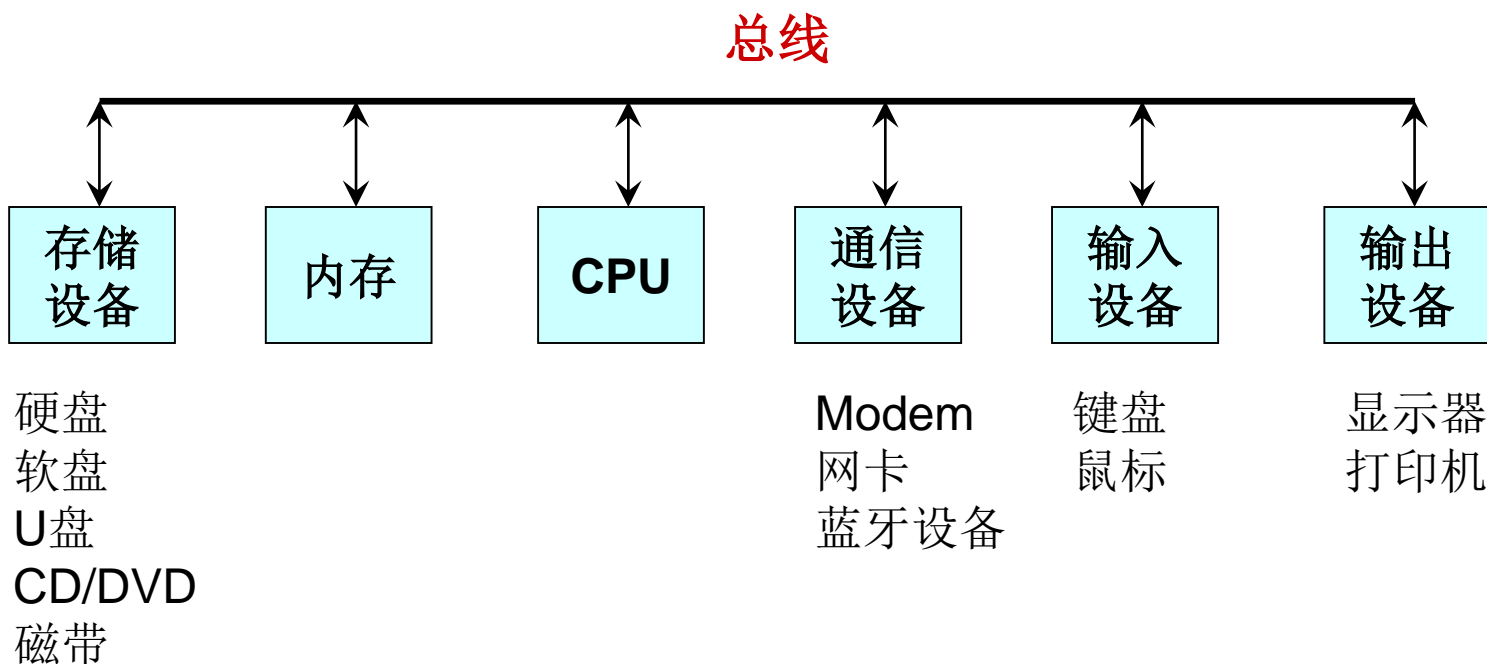
- A **computer** is a machine that manipulates data according to a set of instructions.  
( <http://en.wikipedia.org/wiki/Computer> )
- 计算机是一种能够按照事先存储的程序，自动、高速地进行大量数值计算和各种信息处理的现代化智能电子设备。  
( <http://baike.baidu.com/view/3314.htm> )





## ■ 计算机一般包括：

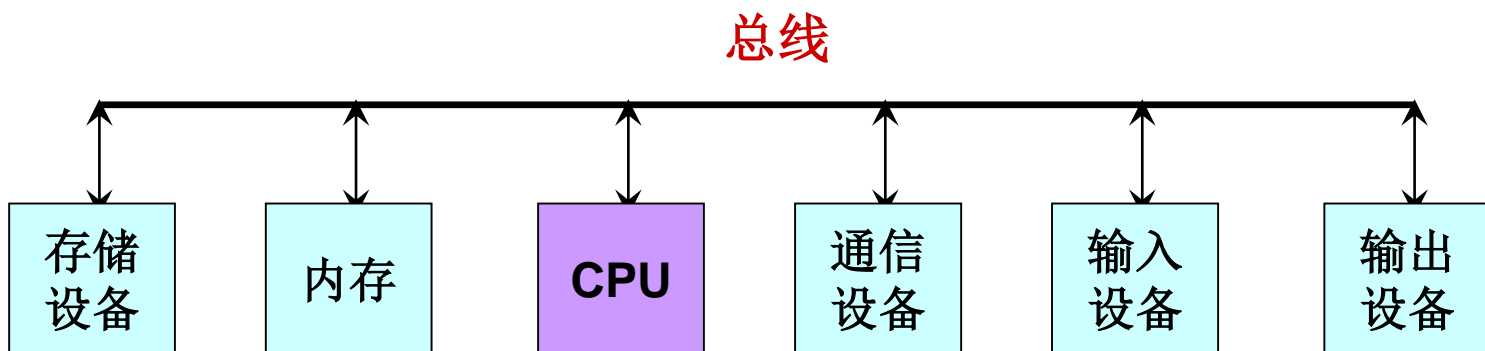
- CPU
- 内存
- 存储设备（硬盘、软盘、光盘）
- 输入输出设备（显示器、打印机、键盘、鼠标、网卡）
- 通信设备



# CPU



- **CPU**（**Central Processing Unit**, 中央处理器）是计算机的大脑，它从内存中提取指令并执行之
- **CPU的速度**



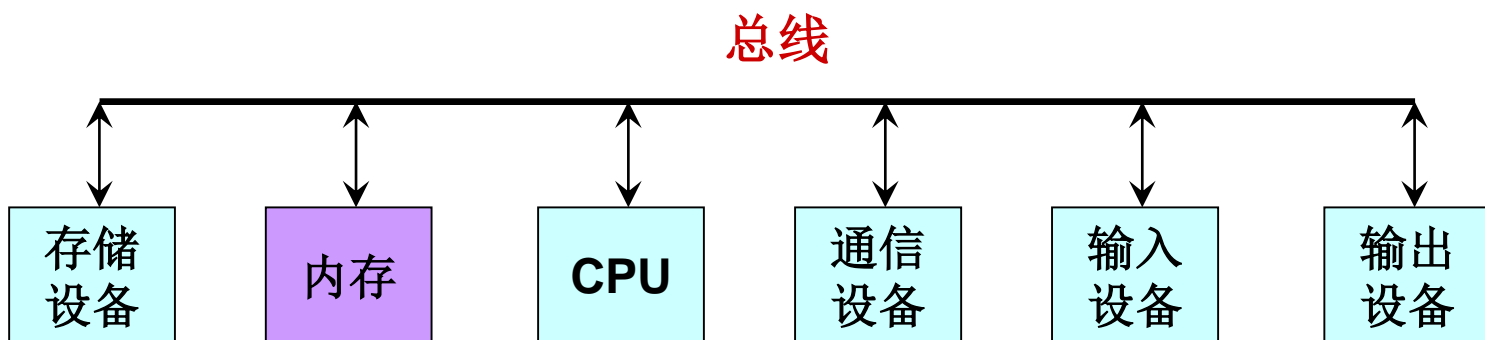


# 内存



## ■ 内存（Memory）

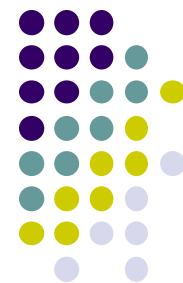
- 用来存数据和CPU执行的程序指令
- 内存单元是由**字节**（byte）构成的有序序列，每个字节由8个**比特**（bit）组成
- 程序执行前必须将它和它的数据装入内存
- 内存的易丢失性



# 数据如何存储？

- 各种类型的数据，比如数字、字符和字符串，都编码为一个位的序列（位是指二进制数：0或1）
- 因为计算机的数字设备有两种平稳的状态，习惯上记作0（zero）和1（one），所以计算机用0和1进行存储。
- 程序员不需要关心数据的编码和解码，系统根据编码表自动执行的
- 编码表多种多样，例如，在流行的ASCII编码中，字符‘J’用一个字节01001010表示。小数字，比如3，可以用单个字节存储。
- 如果计算机需要存储用单个字节放不下的大数，就使用相邻的多个字节。
- 两个数据不能共享或分割同一个字节，字节是最小的存储单位。

地址	内容	
2000	01001010	字符 J
2001	01100001	字符 a
2002	01110110	字符 v
2003	01100001	字符 a
2004	00000011	数字 3

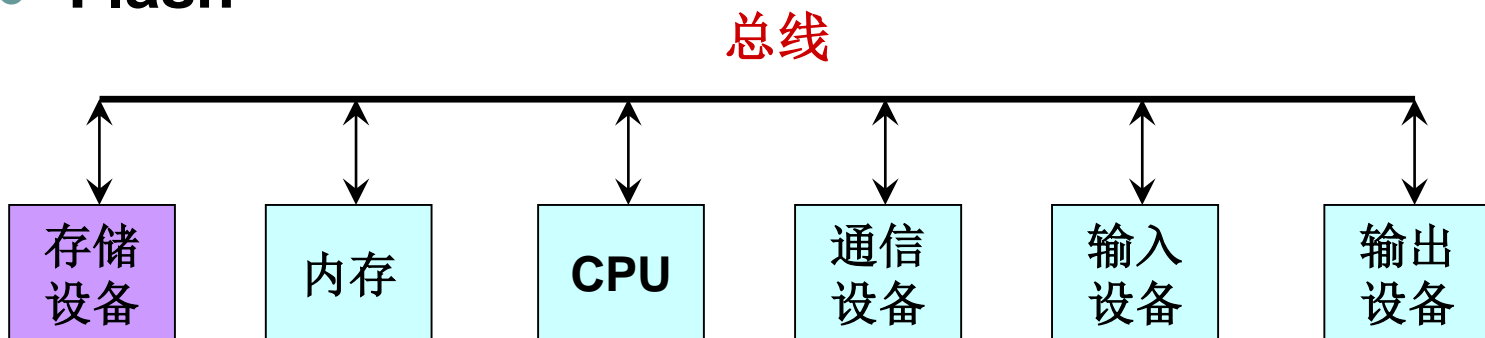


# 存储设备



- 内存是不能长久保存数据的，断电时信息就会丢失。数据和程序都永久地存放在存储设备上，当计算机确实使用它们时再装入内存

- 磁盘
- 光盘
- 磁带
- Flash



# 输入、输出设备

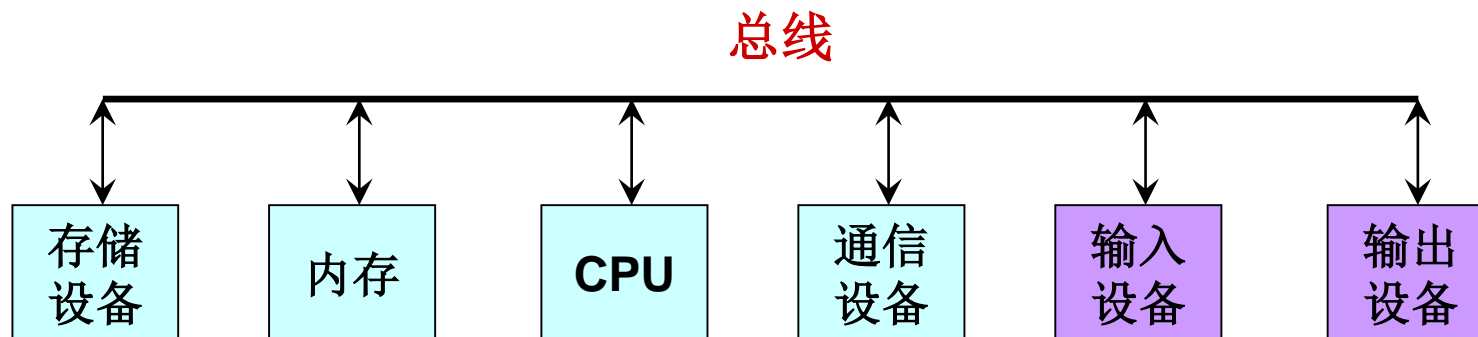


## ■ 输入设备

- 键盘、鼠标、语音、摄像头

## ■ 输出设备

- 显示器、打印机



# 通信设备

- 网卡（Ethernet、WiFi、3G/4G...）
- 普通调制解调器（Modem）
- ADSL Modem
- Cable Modem



# 程序



- 计算机程序是发给计算机的指令
- 你通过程序告诉计算机该做什么。没有程序，计算机就是一个空机器。
- 计算机不能理解人类的语言，所以需要使用计算机语言和计算机进行交流
- 程序使用编程语言书写





## ■ 机器语言

机器语言

汇编语言

高级语言

机器语言（Machine Language）是植入各台计算机的  
**原始指令集**。

二进制代码形式

用机器语言编写程序是一件**单调乏味**的事情，而且所编的**程序非常难读、难改**。

例如：两数相加，可能必须写成如下的二进制形式：  
1101101010011010



## ■ 汇编语言

机器语言

汇编语言

高级语言

汇编语言（Assembly Language）降低了编程的难度。

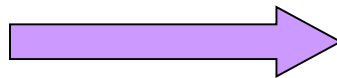
计算机不懂得汇编语言，需要使用汇编器（assembler）的程序，**将汇编语言程序转换为机器指令**。

例如，两数相加，用汇编代码所编写的指令形式如下：  
ADDF3 R1, R2, R3

汇编语言文件

ADDF3 R1 R2 R3

Assembler



机器代码文件

1101101010011010



## ■ 高级语言

机器语言

汇编语言

高级语言

高级语言（High-Level Language）很像英语，易于学习和编写程序。

例如，

两数相加： $R3 = R2 + R1;$

计算半径为5的圆面积： $area=5 * 5 * 3.14;$



## ■ 流行的高级语言

- COBOL (Common Business Oriented Language)
- FORTRAN (FORMula TRANslation)
- BASIC (Beginner All-purpose Symbolic Instructional Code)
- Pascal (以Blaise Pascal命名)
- Ada (以Ada Lovelace命名)
- **C**
- Visual Basic (Microsoft公司开发的类似Basic的可视化语言)
- **C++** (以C语言为基础的一种面向对象程序设计语言)
- **C#** (微软公司开发的语言)
- **Java** (本课程使用的语言)
- Python、Perl、PHP、JavaScript、Ruby、Lisp、TCL/TK 、 、 、
- Delphi (Borland公司开发的类似Pascal的可视化语言)

作为计算机专业的学生，必须至少精通一种编程语言，了解并能欣赏其他类型语言的风格，并具备在短时间学习一门新语言并使用的能力。

# Java 与 C/C++



- C/C++曾被认为是计算机领域最重要的程序语言，但Internet的兴起，Java的出现打破了这一局面。Internet在现代信息系统中的地位使Java成为互联网时代的核心语言之一。
- 微软为了对抗Java，于2002年推出C#编程语言和.NET平台。两虎相争十余年，Java以其开放的特点占据优势，但是C#也打下自己的江山，两者各安其位。



## ■ 编译源代码

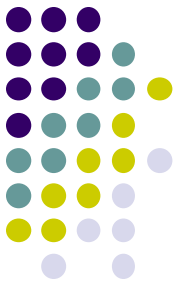
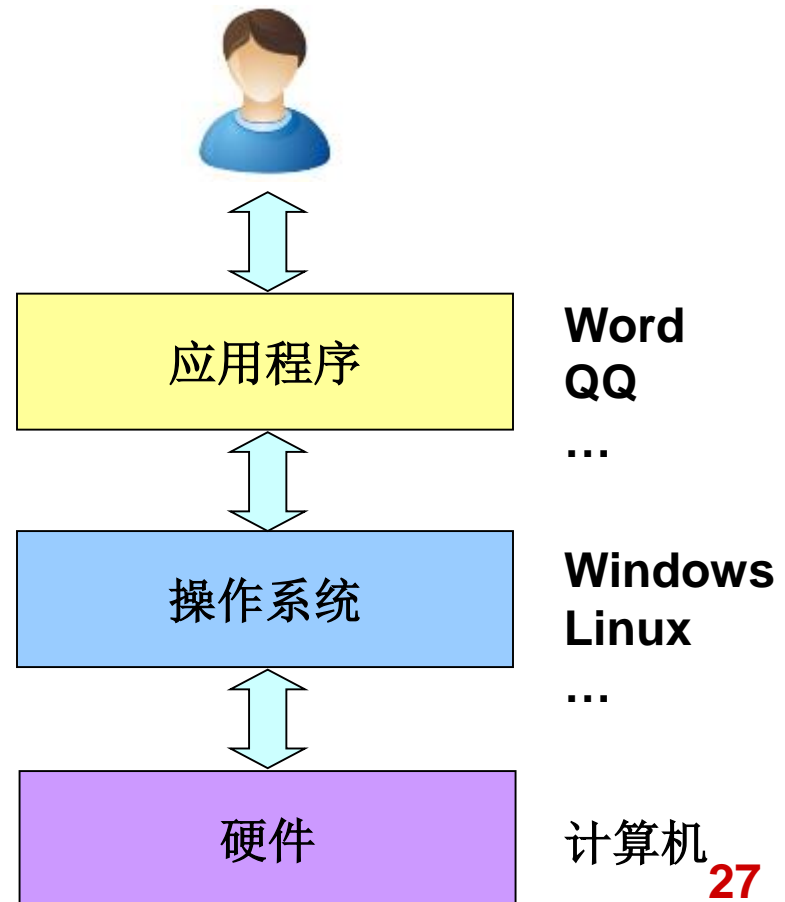
- 用高级语言编写的程序称为**源程序**（Source Program / Source Code）
- 由于计算机不懂得源程序，所以使用一种称为**编译器**（compiler）的程序将源程序翻译成机器语言的程序，这种机器语言程序称为**目标程序**（object program）
- 目标程序与其它支持库代码进行链接，从而构成可执行文件，可执行文件就可以在计算机上运行





# 操作系统

- 操作系统（Operating System, OS）是控制其他程序运行，管理系统资源并为用户提供操作界面的系统软件的集合。
  - Windows（Windows 8/7、Vista、XP、2000、NT 、 、 、 ）
  - Linux
  - Unix
  - Mac OS
- 应用程序运行在操作系统之上



# 第一章 概述



§ 1.1 Java语言简介

§ 1.2 Java语言实现机制

§ 1.3 Java开发环境

§ 1.4 Java程序

# § 1.1 Java语言简介--发展历史



## ■ Java的历史

- 1991年，Sun公司着手开发一个机顶盒项目：Green消费类电子产品（如电视控制盒，电子翻译器）
- 在开发时，为了适应不同的硬件平台，需要一种与平台无关的通用语言。这种与平台无关的通用语言被命名为OAK（橡树），设计者是James Gosling。后来因为商标问题，改为Java（一种产自印尼爪哇的咖啡）
- 1994年，机顶盒开发完成，但推销失败。
  - 开发人员想到，OAK语言可以应用到浏览器中，可以使得浏览器不但可以显示静态页面，而且可以执行程序，同用户互动。
  - 开发人员用Java开发了一个的浏览器，称为HotJava



Java之父：  
James Gosling



## ■ Java 的 Logo

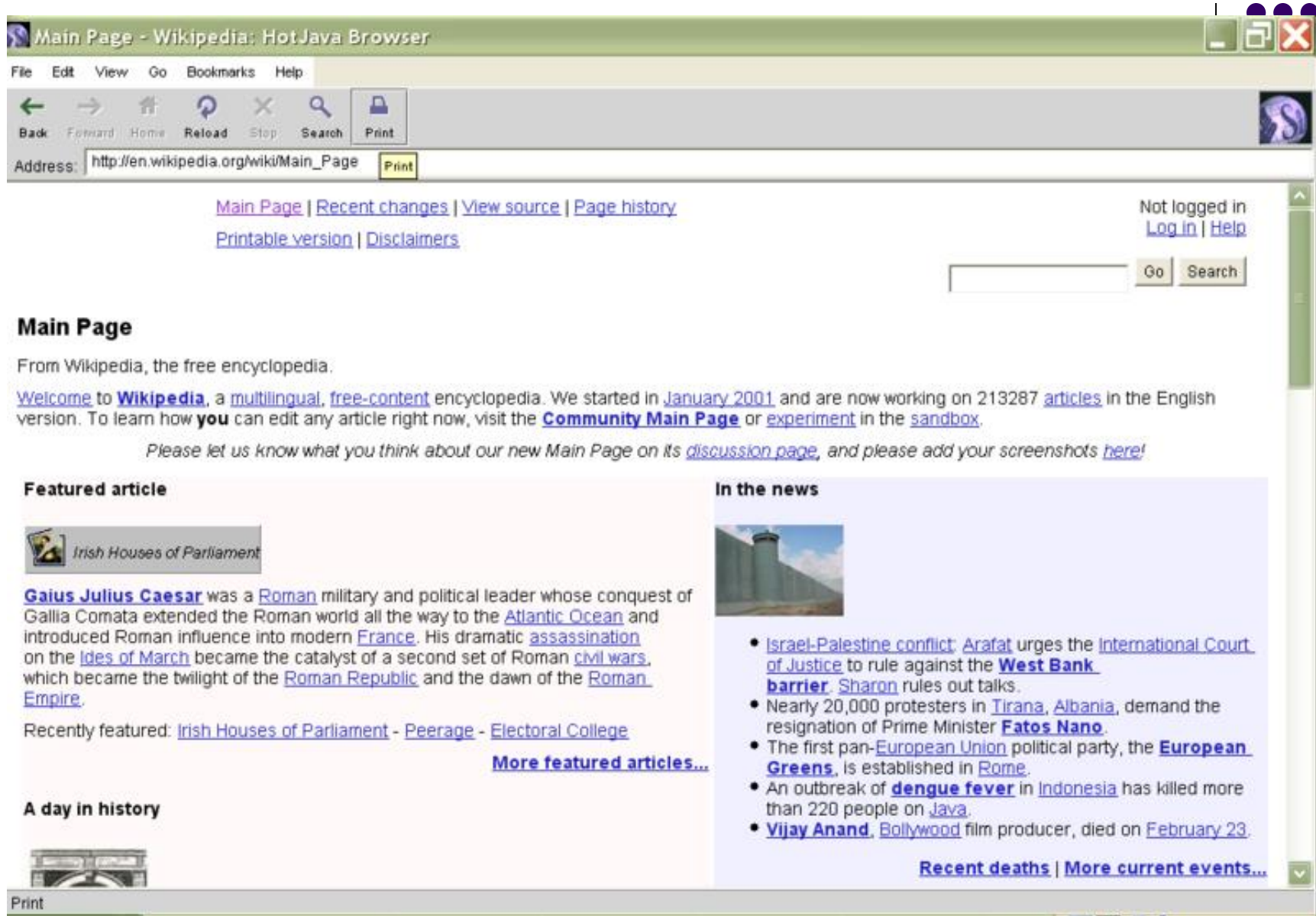


## ■ Java 的吉祥物



Java的吉祥物: Duke

<https://duke.dev.java.net/images/index.html>





## ■ 用来创建具有动态效果的网页

- 1995年5月23日，James Gosling带着Java语言和HotJava浏览器参加技术交流会，结果大获成功
- Netscape决定在1996年1月的Netscape浏览器中，捆绑Java
- Java许可证
  - Netscape、Oracle、Microsoft ...

## ■ 现在主要被用于以下应用领域:

- 开发大规模的企业应用（Java EE）与Web应用（多使用Struts Spring Hibernate框架）
- 开发移动通信设备（Java ME 和 Android）





- 初始，Java还只是一个语言，要想开发复杂的应用程序，必须要有一个强大的开发库：JDK（Java Development Kit）
  - 1996.1.23: JDK 1.0
  - 1997.2.19: JDK 1.1
  - 1998.12.8: J2SE 1.2（Java version 2）
    - Java被分为J2SE、J2EE、J2ME
  - 2000.5.8: J2SE 1.3
  - 2002.2.6: J2SE 1.4
  - 2004.9.30: J2SE 5.0 (1.5)
  - 2006.12.11: J2SE 6.0 (1.6)
    - Current: J2SE6.0 Update 14 (2009.5.28)
  - 2009.4.20: Oracle收购SUN，从此Java变成Oracle的了
  - 2011.7.28: J2SE 7.0(1.7)
  - 参见: [http://en.wikipedia.org/wiki/Java\\_version\\_history](http://en.wikipedia.org/wiki/Java_version_history)



## ■ Java的三个分支

### ● J2SE: Java 2 Standard Edition

- 开发运行于PC和工作站上的普通应用
- 编写, 部署和运行Java应用程序和APPLET: JDK(Java Development Kit), Java 2 SDK

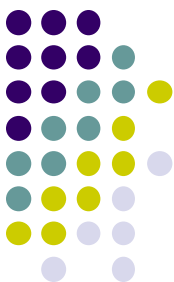
### ● J2EE: Java 2 Enterprise Edition

- 开发多层、Client-Server、面向企业的应用
- JSP, Servlet, EJB, JTS(Java Transaction Service), Java mail, JMS(Java Message Service)等多项技术混合体。用于开发分布式的, 服务器端的多层结构的应用系统 (如电子商务)

### ● J2ME: Java 2 Micro Edition

- 用于嵌入式设备 (如移动设备等) , CPU、内存受限制
- 主要开发电子产品, 如移动电话, 数字机顶盒, 汽车导航系统

### ● 本课程讲述J2SE



# ■ Java的现状

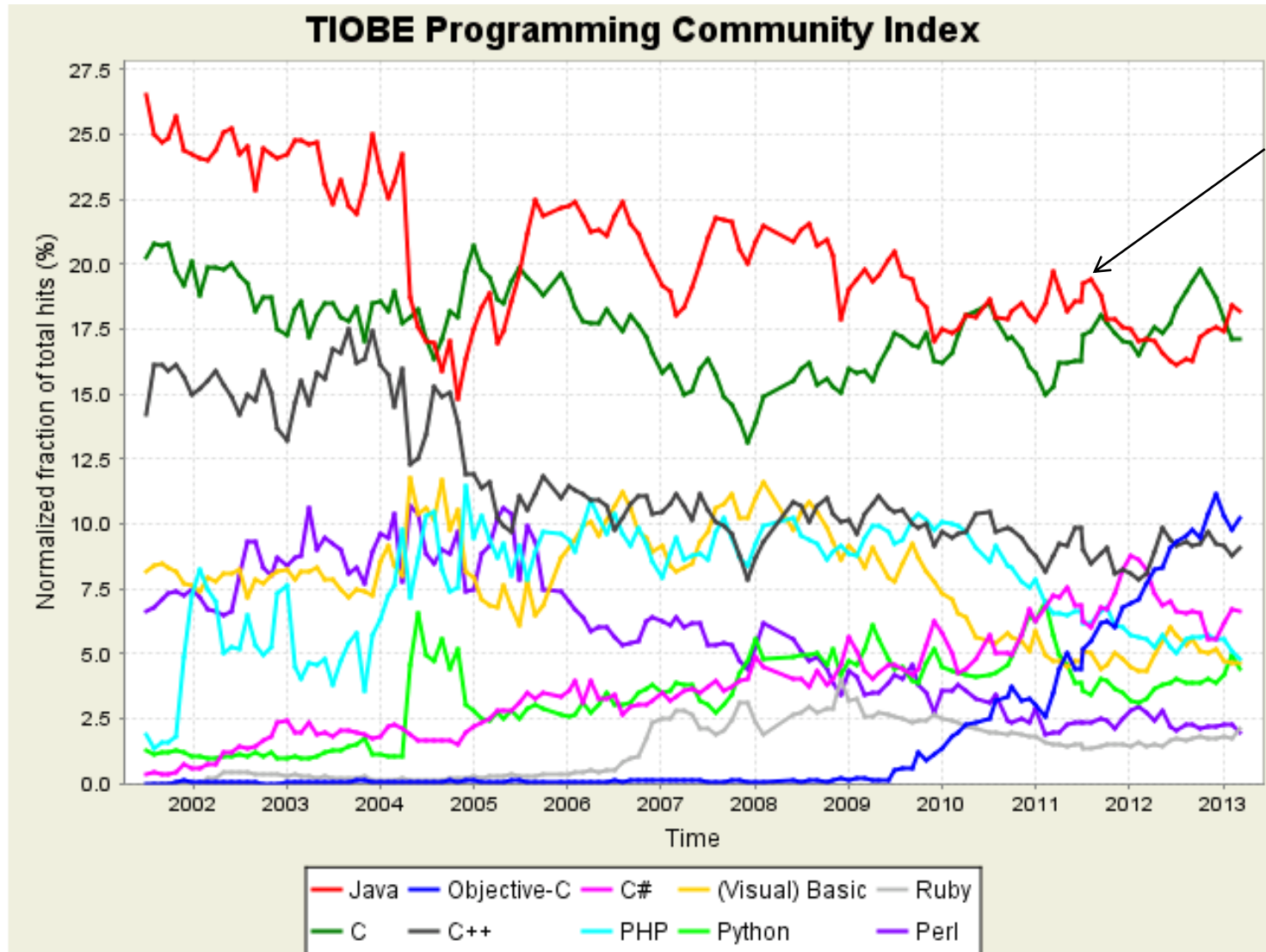
## ● 开发语言排行（截止2013年3月）

<http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>

	Position Mar 2013	Position Mar 2012	Delta in Position	Programming Language	Ratings Mar 2013	Delta Mar 2012	Status
	1	1	=	Java	18.156%	+1.05%	A
	2	2	=	C	17.141%	+0.05%	A
	3	5	↑↑	Objective-C	10.230%	+2.49%	A
	4	4	=	C++	9.115%	+1.07%	A
	5	3	↓↓	C#	6.597%	-1.65%	A
	6	6	=	PHP	4.809%	-0.75%	A
	7	7	=	(Visual) Basic	4.607%	+0.24%	A
	8	9	↑	Python	4.388%	+1.10%	A
	9	13	↑↑↑↑	Ruby	2.150%	+0.74%	A
	10	10	=	Perl	1.959%	-0.74%	A
	11	8	↓↓↓	JavaScript	1.370%	-2.02%	A
	12	48	↑↑↑↑↑↑↑↑	Bash	1.009%	+0.78%	A-
	13	15	↑↑	Lisp	0.942%	+0.02%	A
	14	12	↓↓	PL/SQL	0.921%	-0.50%	A-
	15	11	↓↓↓	Delphi/Object Pascal	0.889%	-0.84%	A
	16	16	=	Visual Basic .NET	0.888%	+0.10%	A
	17	14	↓↓↓	Transact-SQL	0.836%	-0.09%	A-
	18	17	↓	Pascal	0.697%	-0.07%	A-
	19	21	↑↑	Lua	0.697%	+0.17%	B
	20	26	↑↑↑↑↑	Assembly	0.633%	+0.21%	B



## ● 占有率趋势图



Java



- **Java是目前最流行的编程语言**
- **涵盖服务器、桌面、移动平台（如Android）**
  - Java更适合于服务器端开发
  - 基于Java 2 的Web开发，是目前Java开发的主流
  - 在手机开发中，占有重要地位（J2ME）
  - Android采用Java作为开发语言（虽然由于版权问题，Google回避了这一点）
  - .....

# 第一章 概述



§ 1.1 Java语言简介

§ 1.2 Java语言实现机制

§ 1.3 Java开发环境

§ 1.4 Java程序

## § 1.2 Java语言简介—实现机制：Java虚拟机



### ■ Java怎么读？

- <http://dictionary.reference.com/search?q=Java>

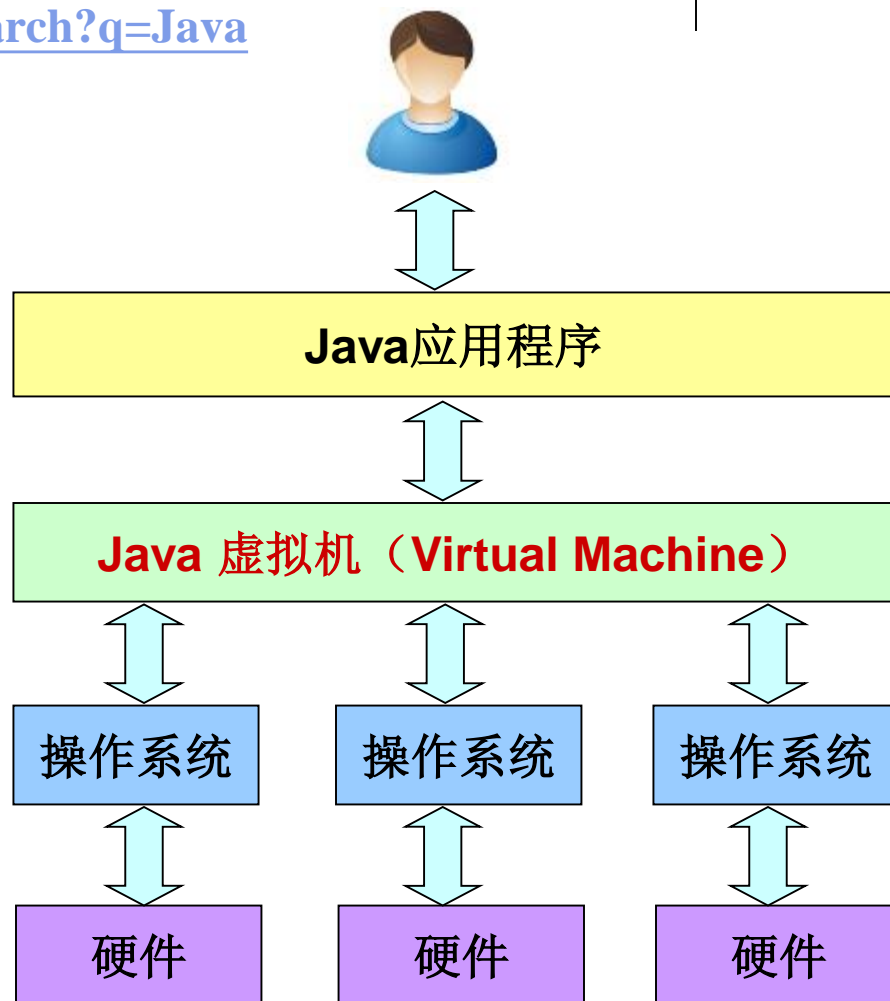
### ■ Java是什么？

- Java是一种编程语言
- Java更是一个平台
- **Write Once, Run Everywhere**

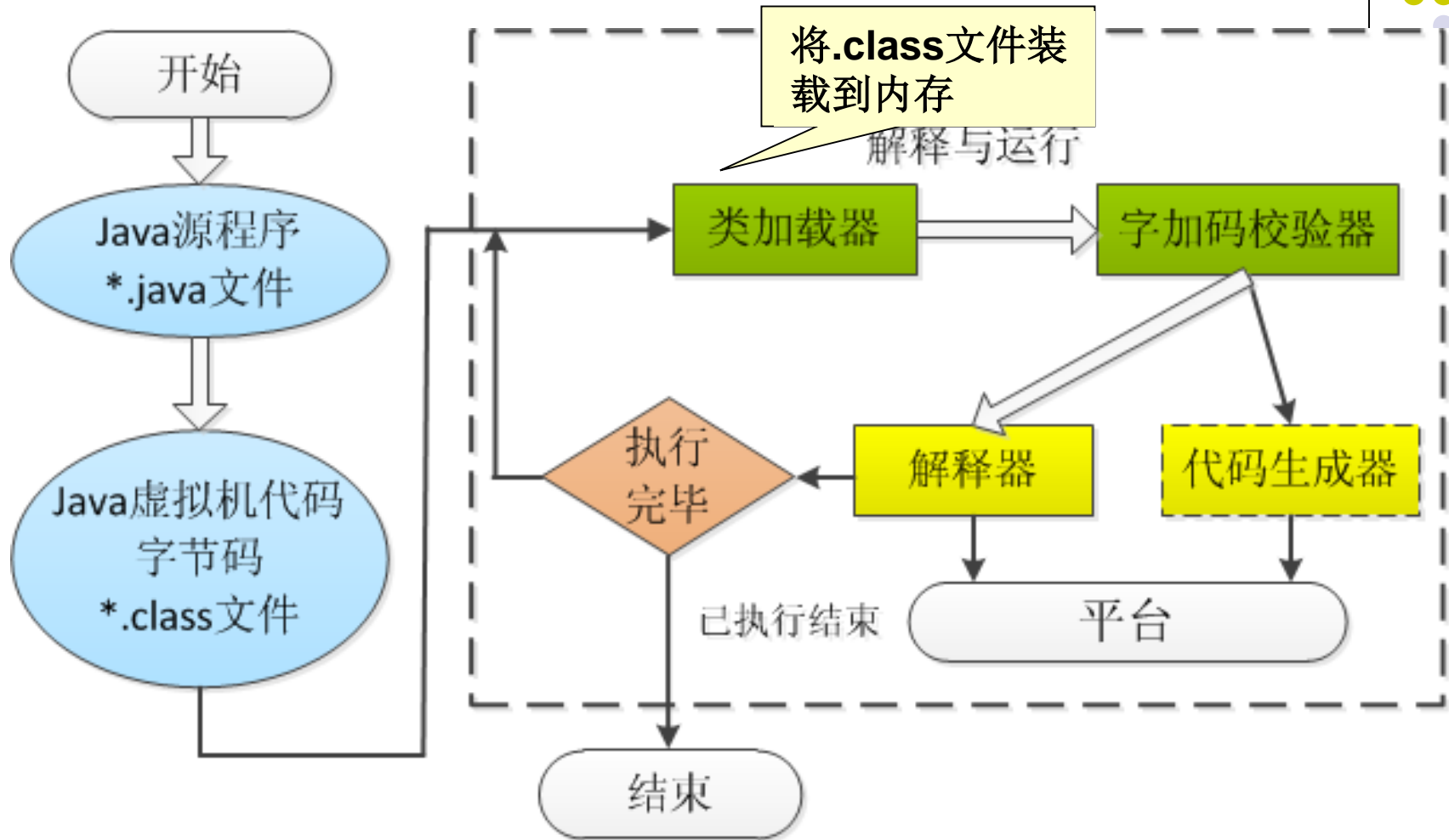
### ■ Java是一个“混合类型”的编程语言，跨平台，运行在JVM中，源代码是标准的文本文件。

### ■ Java虚拟机

- Java Virtual Machine(JVM)
- 运行Java程序
- 屏蔽底层（操作系统、硬件）的差异性



## § 1.2 Java语言简介—实现机制：Java虚拟机



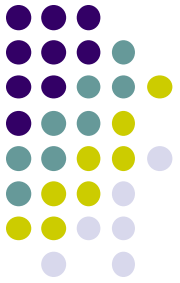
# Java虚拟机的工作流程



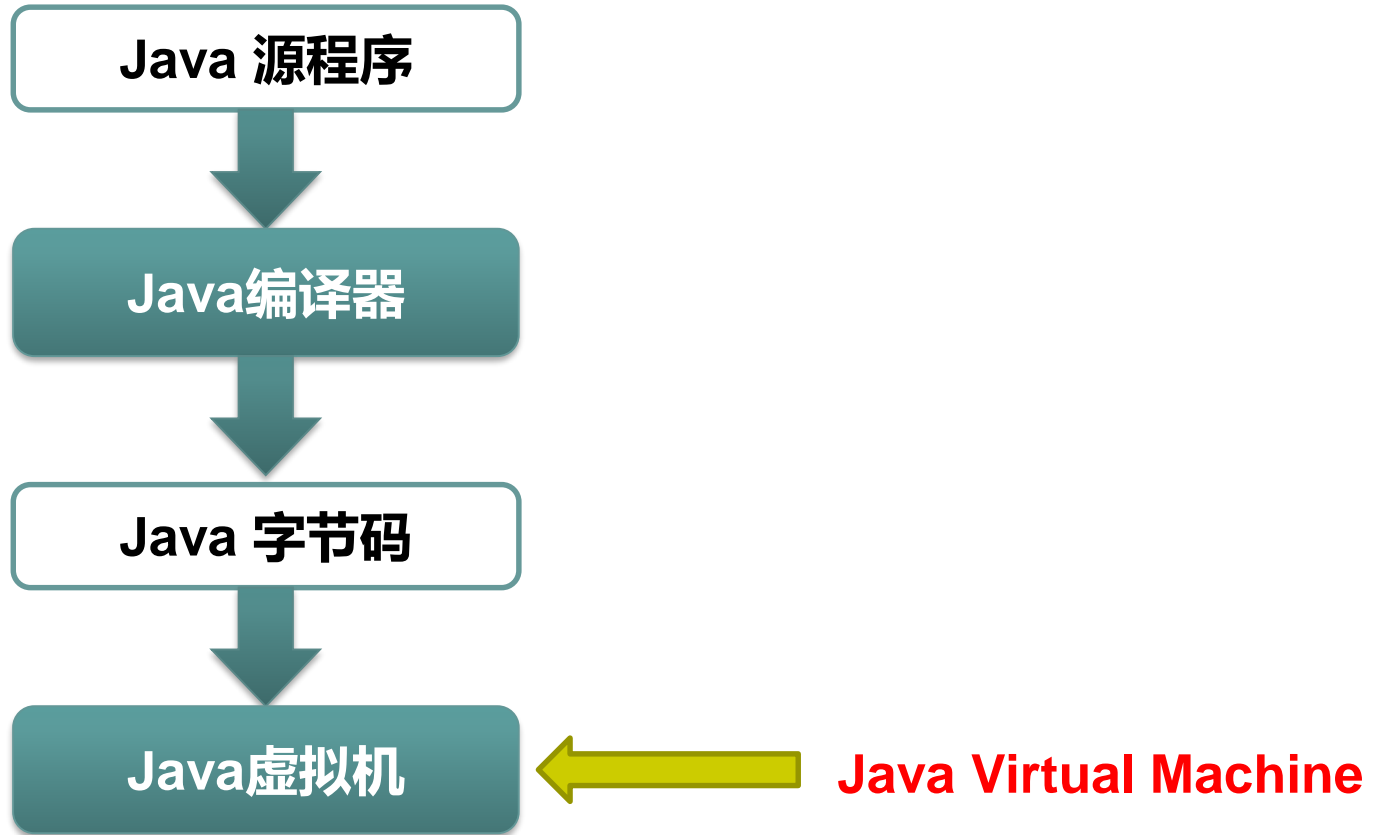


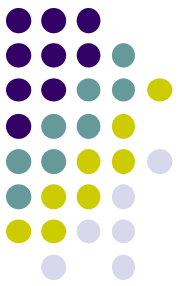
## ■ 编译Java代码

- 使用Java，可以只写一次程序并将它编译成字节码（bytecode）。
- 字节码可以在任何装有Java虚拟机（Java Virtual Machine, JVM）的计算机上运行。
- **Java虚拟机是解释Java字节码的软件**  
**Write Once, Run Everywhere**
- 在HelloWorld例子中，编译生成的HelloWorld.class文件是字节码格式文件，可以被Java虚拟机解释执行



## § 1.2 Java语言简介—实现机制





## § 1.2 Java语言简介—实现机制

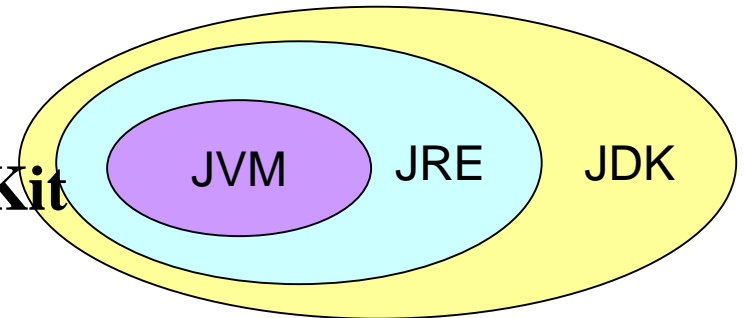
### ■ JDK和JRE

- **JRE: Java Runtime Environment**

- Java运行时环境，包含了运行Java程序的所需要的所有组件，如JVM、动态库等。
- 没有JRM，不可能在Windows等操作系统上运行Java程序。

- **JDK: Java SE Development Kit**

- Java开发工具包，包含
  - JDK = JRE+相关的开发工具
  - 如需要开发Java程序，必须安装JDK
  - 只需要运行编译好的Java程序，安装JRE即可。
- 本课程使用JDK

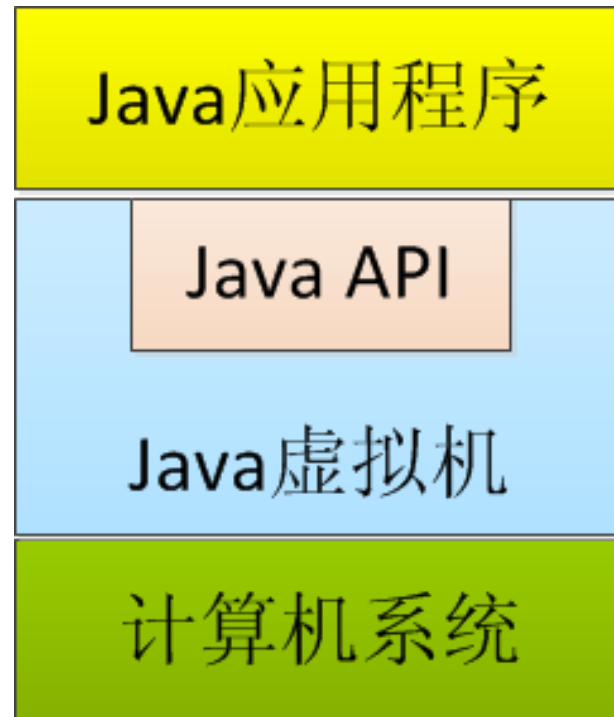




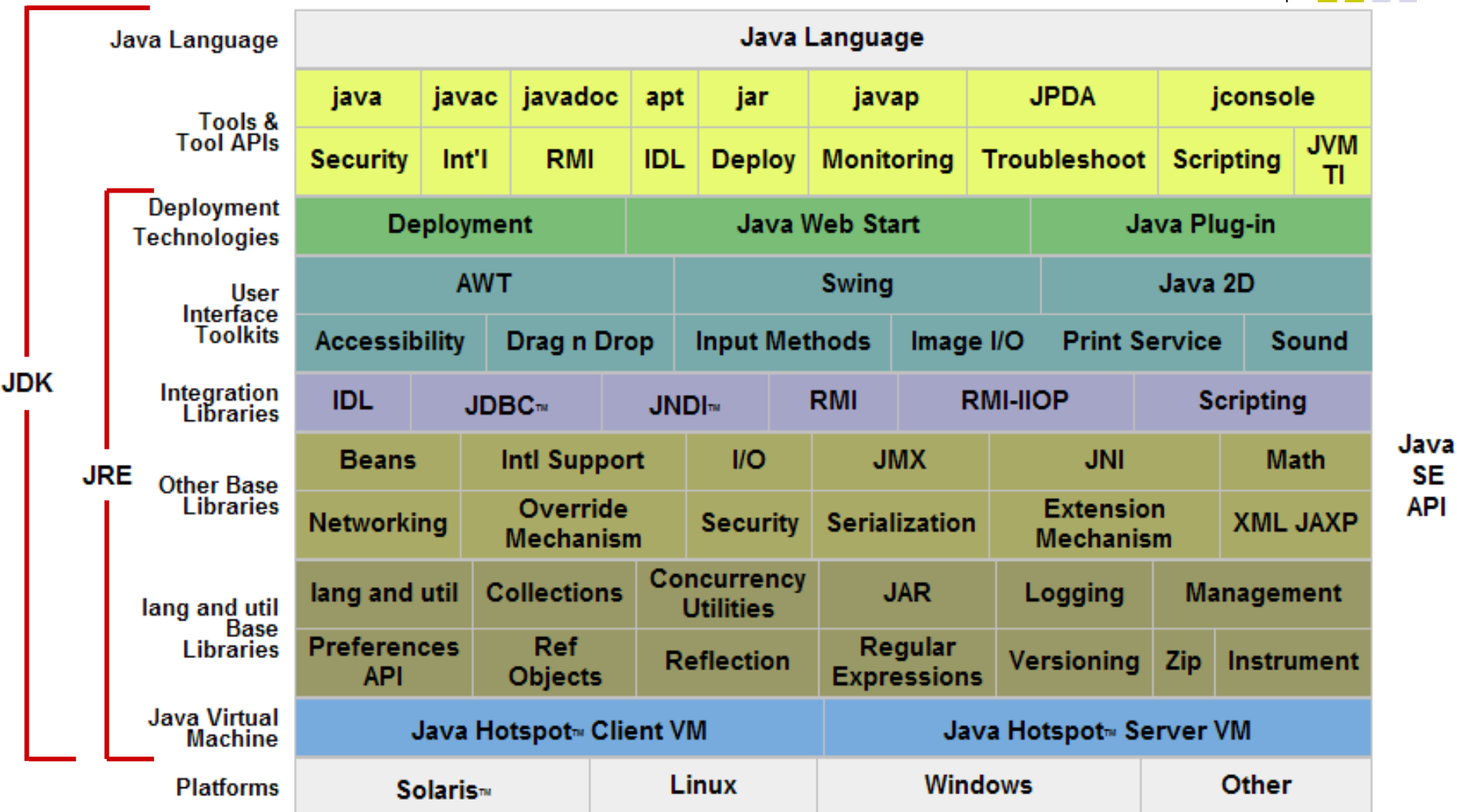
## § 1.2 Java语言简介—实现机制

### ■ Java平台

- **Java APIs (应用程序接口)**
  - 经过编译的，可在程序中使用的Java代码标准库。
- **Java VM (虚拟机)**
  - Java 程序由Java虚拟机程序执行（或解释执行）



# J2SE 全家福



## § 1.2 Java API



- **java.applet:** 提供创建Applet所必须的类，以及Applet与其上下文进行通信的类。
- **java.awt:** 含有创建用户接口、绘图和图像的所有类。
- **java.io:** 提供系统的输入与输出。
- **java.lang:** 提供Java编程语言最基础的类。
- **java.net :** 提供了实现网络应用程序的类。
- **java.sql :** 提供了使用Java访问和处理数据源中数据的类。
- **java.util:** 含有一个集合框架类、事件模型、日期和时间以及一些杂类，如针对jar和zip技术的类。

## § 1.2 Java API



- **java.math:** 提供了用于执行任意精度整数算法和任意精度小数算法的类。
- **java.beans:** 提供了与bean上下文有关的类和接口
- **java.rmi:** 提供了RMI包。
- **java.security:** 提供了用于安全框架的类和接口。
- **java.text:** 提供了与语言无关方式处理的文本、数据、数字和消息的类与接口。
- **javax.swing:** 提供了一套“轻量级”的组件，在所有平台上都可工作。
- **javax.naming:** 提供了访问命名服务的类和接口。创建用户接口、绘图和图象的所有类。

# 第一章 概述



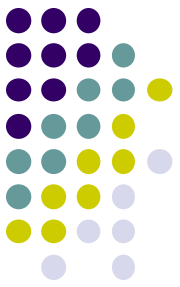
§ 1.1 Java语言简介

§ 1.2 Java语言实现机制

§ 1.3 Java开发环境

§ 1.4 Java程序





## § 1.3 Java开发环境

### ■ 实例

- 在命令行下编译第一个Java程序：HelloWorld

```
C:\Windows\system32\cmd.exe
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Julie>d:

D:\>cd java\demo

D:\Java\Demo>dir
驱动器 D 中的卷没有标签。
卷的序列号是 0C51-1345

D:\Java\Demo 的目录
2014/02/16  17:28    <DIR>        .
2014/02/16  17:28    <DIR>        ..
2014/02/17  00:59                425 HelloWorld.class
2013/12/24  12:35                123 HelloWorld.java
                2 个文件          548 字节
                2 个目录 91,470,192,640 可用字节

D:\Java\Demo>javac HelloWorld.java

D:\Java\Demo>java HelloWorld
HelloWorld!

D:\Java\Demo>
```

# ■ 实例

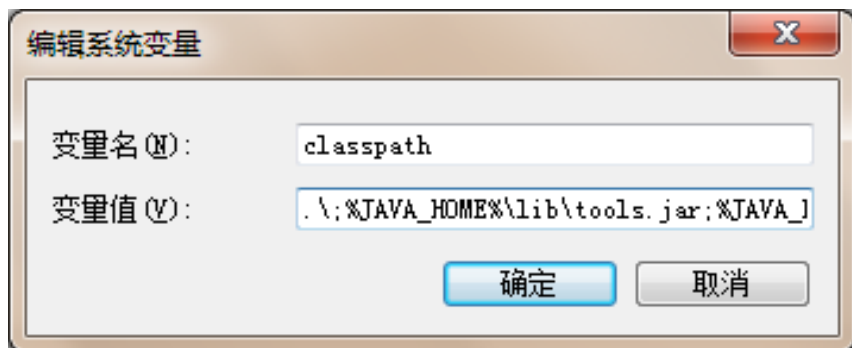
- 环境变量设置



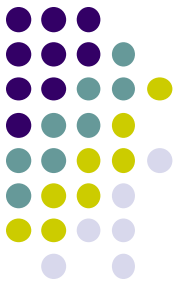
设置JAVA\_HOME环境变量为JDK安装路径  
(依安装版本会有所不同) :  
D:\Java\jdk1.7



系统可以在任何路径下识别java命令:  
.\; %JavaHome%\bin;  
\\%JavaHome%\jre\bin



CLASSPATH为java加载类(class or lib)路径, 只有类在classpath中, java命令才能识别  
.\;%JavaHome%\lib\tools.jar;%JavaHome%\jre\lib\rt.jar



## ■ 注意

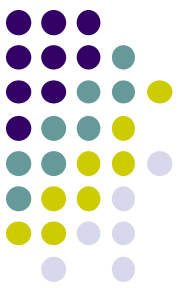
- javac编译, java运行
- 类名和源文件名必须完全一致 (区分大小写)
- 一定要正确地设置环境变量

检测Java环境变量是否安装成功: `java -version`

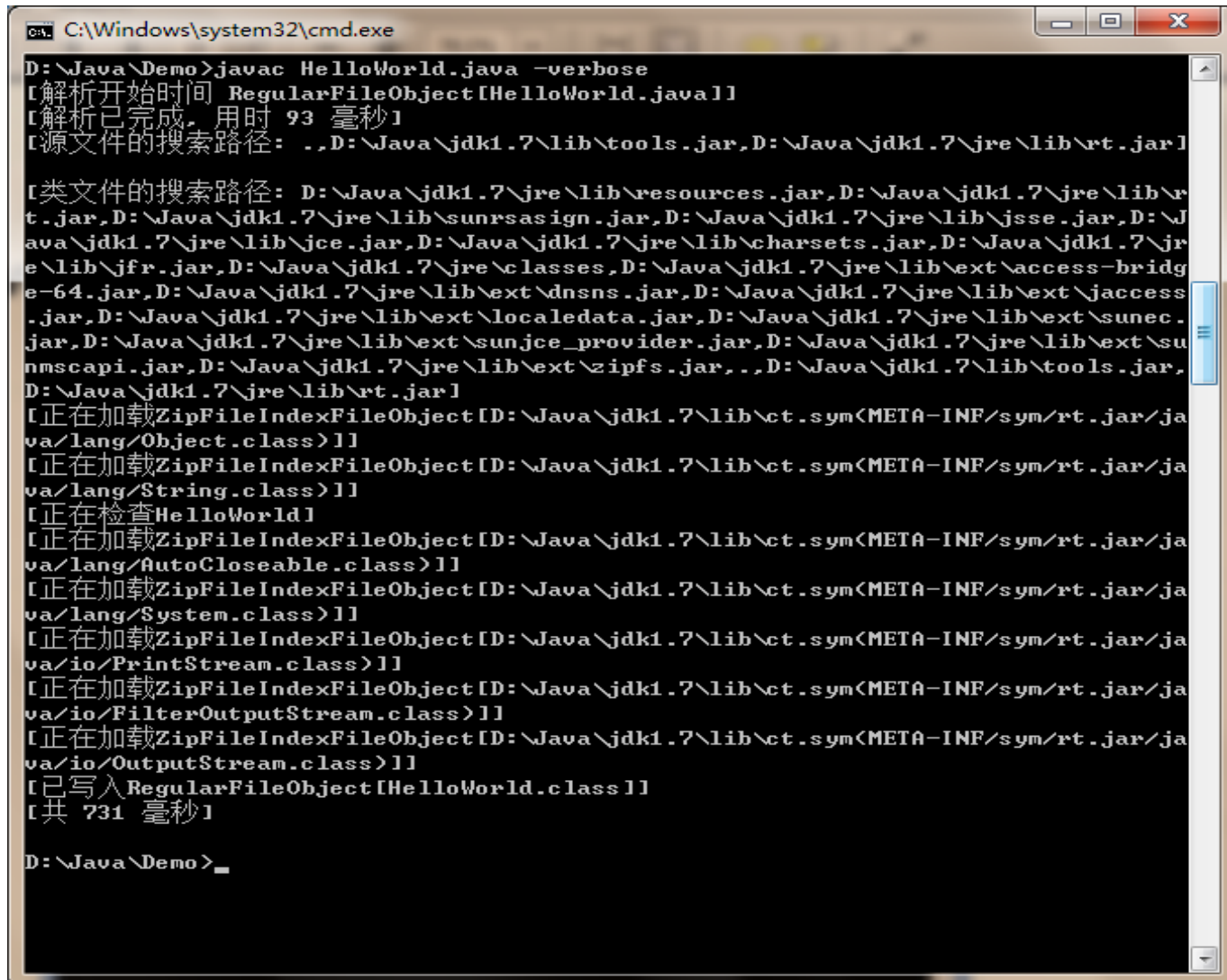
```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Julie>java -version
java version "1.7.0_45"
Java(TM) SE Runtime Environment (build 1.7.0_45-b18)
Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)

C:\Users\Julie>
```



- javac干了些什么事?
- javac HelloWorld.java -verbose



```
C:\Windows\system32\cmd.exe

D:\Java\Demo>javac HelloWorld.java -verbose
[解析开始时间 RegularFileObject[HelloWorld.java]]
[解析已完成, 用时 93 毫秒]
[源文件的搜索路径: ..D:\Java\jdk1.7\lib\tools.jar,D:\Java\jdk1.7\jre\lib\rt.jar]

[类文件的搜索路径: D:\Java\jdk1.7\jre\lib\resources.jar,D:\Java\jdk1.7\jre\lib\rt.jar,D:\Java\jdk1.7\jre\lib\sunrsasign.jar,D:\Java\jdk1.7\jre\lib\jsse.jar,D:\Java\jdk1.7\jre\lib\jce.jar,D:\Java\jdk1.7\jre\lib\charsets.jar,D:\Java\jdk1.7\jre\lib\jfr.jar,D:\Java\jdk1.7\jre\classes,D:\Java\jdk1.7\jre\lib\ext\access-bridge-64.jar,D:\Java\jdk1.7\jre\lib\ext\dnsns.jar,D:\Java\jdk1.7\jre\lib\ext\jaccess.jar,D:\Java\jdk1.7\jre\lib\ext\localedata.jar,D:\Java\jdk1.7\jre\lib\ext\sunec.jar,D:\Java\jdk1.7\jre\lib\ext\sunec_provider.jar,D:\Java\jdk1.7\jre\lib\ext\sunmscapi.jar,D:\Java\jdk1.7\jre\lib\ext\zipfs.jar,.,D:\Java\jdk1.7\lib\tools.jar,D:\Java\jdk1.7\jre\lib\rt.jar]

[正在加载ZipFileIndexFileObject[D:\Java\jdk1.7\lib\ct.sym(META-INF/sym/rt.jar/java/lang/Object.class)]]
[正在加载ZipFileIndexFileObject[D:\Java\jdk1.7\lib\ct.sym(META-INF/sym/rt.jar/java/lang/String.class)]]
[正在检查HelloWorld]
[正在加载ZipFileIndexFileObject[D:\Java\jdk1.7\lib\ct.sym(META-INF/sym/rt.jar/java/lang/AutoCloseable.class)]]
[正在加载ZipFileIndexFileObject[D:\Java\jdk1.7\lib\ct.sym(META-INF/sym/rt.jar/java/lang/System.class)]]
[正在加载ZipFileIndexFileObject[D:\Java\jdk1.7\lib\ct.sym(META-INF/sym/rt.jar/java/io/PrintStream.class)]]
[正在加载ZipFileIndexFileObject[D:\Java\jdk1.7\lib\ct.sym(META-INF/sym/rt.jar/java/io/FilterOutputStream.class)]]
[正在加载ZipFileIndexFileObject[D:\Java\jdk1.7\lib\ct.sym(META-INF/sym/rt.jar/java/io/OutputStream.class)]]
[已写入RegularFileObject[HelloWorld.class]]
[共 731 毫秒]

D:\Java\Demo>
```

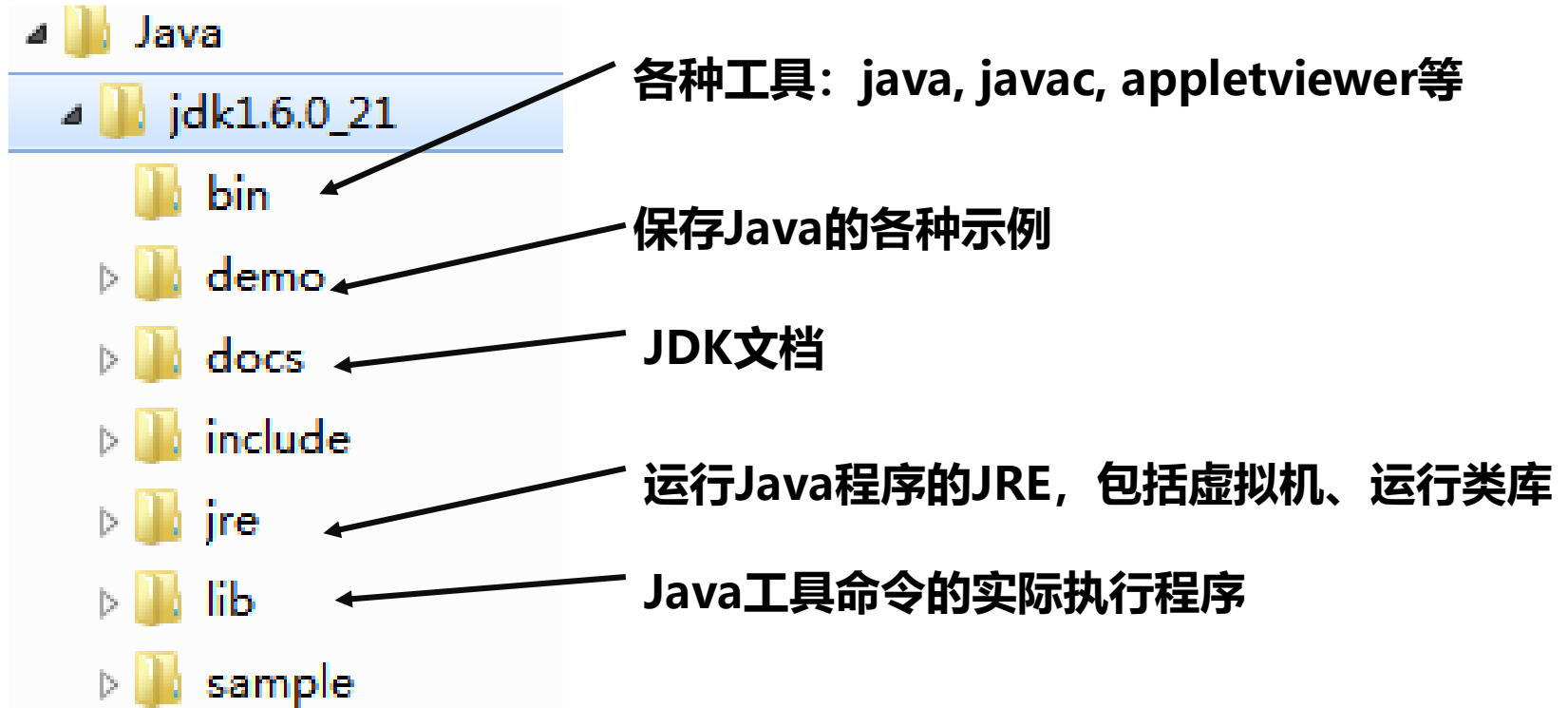


## § 1.3 Java开发环境

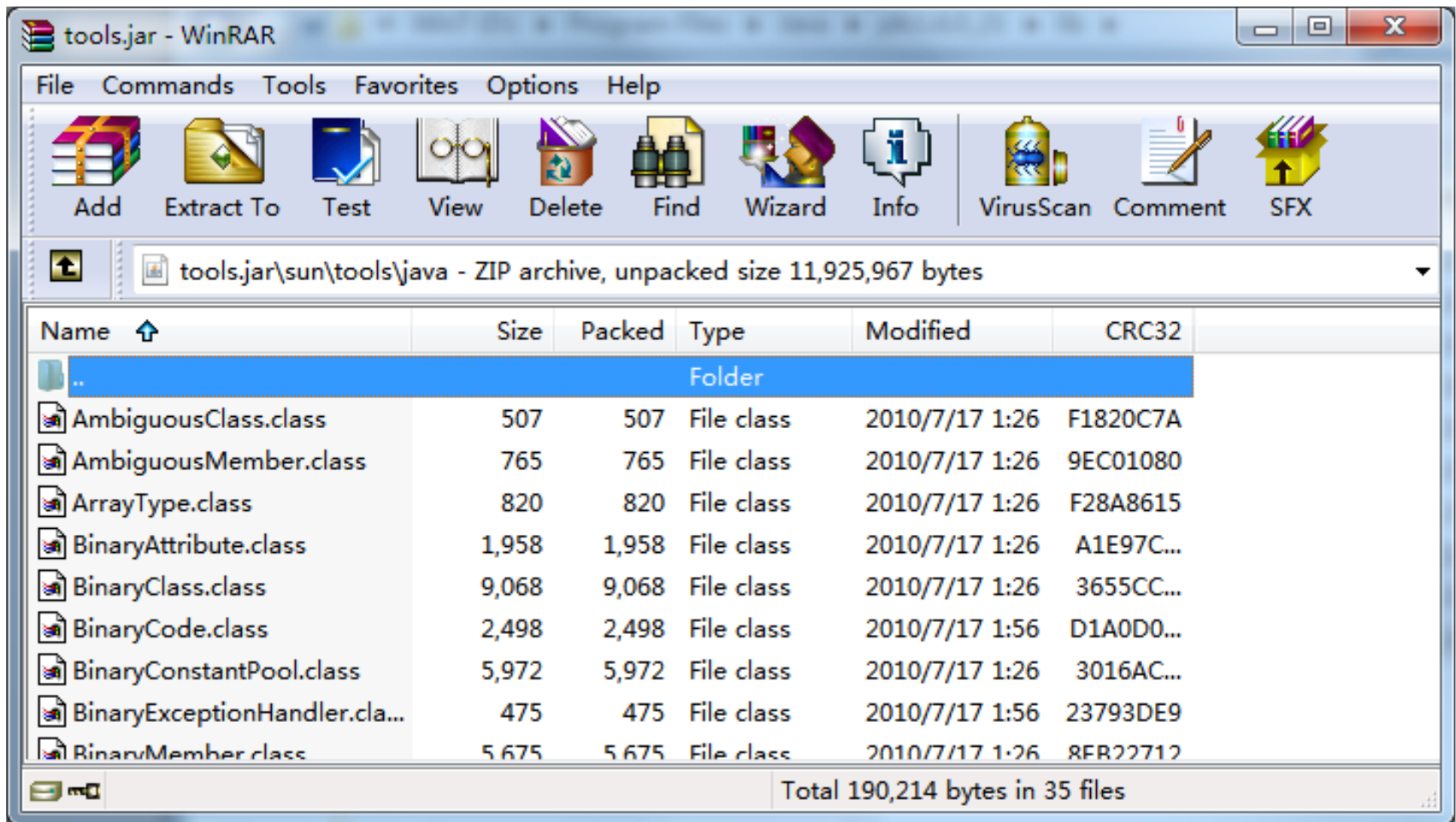
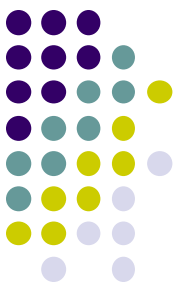
- **Java SDK**——Java **S**oftware **D**evelopment **K**it
  - **Java SE: Java 2 Standard Edition**
  - **Java EE: Java 2 Enterprise Edition**
  - **Java ME: Java 2 Micro Edition**
- **Java SE**——**SDK**的核心部分
  - **开发工具**
    - 编译器
    - 调试器
    - 文档制作工具
  - **运行环境**
    - Java 虚拟机
    - 组成Java 2 平台API的类。
    - 帮助文档
  - **附加库**



# 了解JDK的目录结构



- Java的许多工具命令也是用Java写的，比如运行一个Java程序的“java”命令，它的字节码文件就放在Tools.jar文件中。
- lib目录下的tools.jar



## § 1.3 Java开发环境

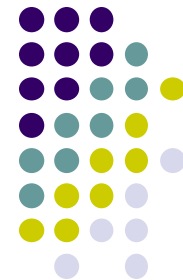


Java开发工具包括

- **Javac:**
  - Java编译器, 用来将java程序编译成 Bytecode。
- **Java:**
  - Java解释器, 执行已经转换成Bytecode的java应用程序。
- **Jdb:**
  - Java调试器, 用来调试java程序。
- **Javap:**
  - 反编译, 将类文件还原回方法和变量。
- **Javadoc:**
  - 文档生成器, 创建HTML文件。
- **Appletviewer:**
  - Applet解释器, 用来解释已经转换成Bytecode的java小应用程序。

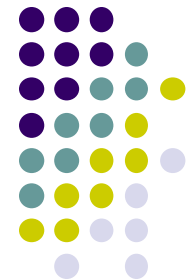


## § 1.3 Java开发环境



- 普通文本工具
  - 记事本 / NotePad++ / UltraEdit/JCreator
- IDE (Integrated Development Environment, 集成开发工具)
  - Eclipse
    - 开源软件
    - <http://www.eclipse.org>
  - NetBeans (最新版: 7.0)
    - 开源软件
    - <http://www.netbeans.org>
  - 本课程推荐用Eclipse/MyEclipse

# 第一章 概述



§ 1.1 Java语言简介

§ 1.2 Java语言实现机制

§ 1.3 Java开发环境

§ 1.4 Java程序

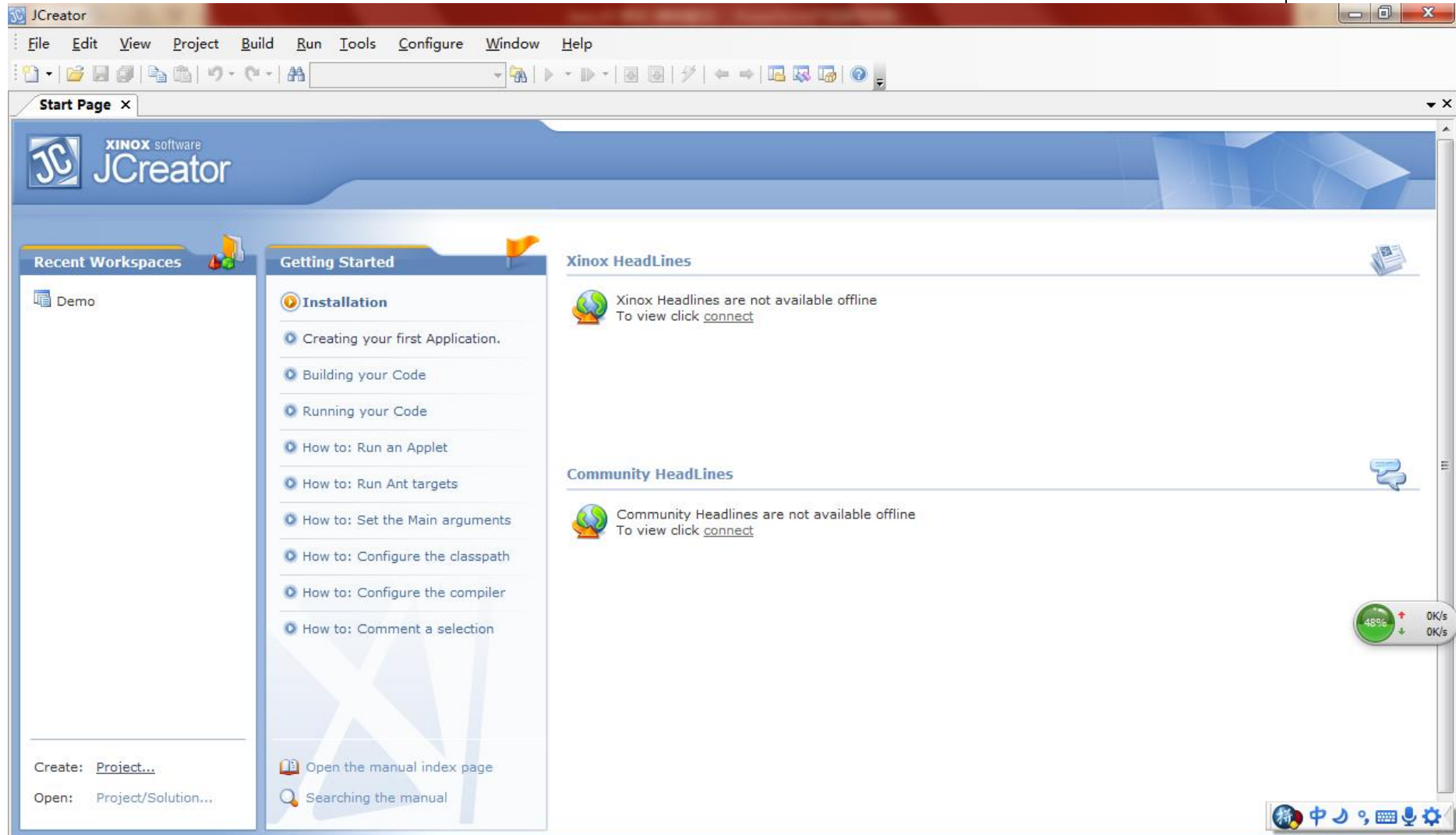
## § 1.4 Java程序



### ■ Java项目类型

- Java可以开发多种类型的项目，每种类型的项目都有特定的应用场景。
- Java Application（Java应用程序）是最简单的一种，是入门的最佳选择。
- 在后继的学习中，会接触到其它的项目类型。

# New Project



# Hello, World!



## ■ 第一个例子

- *The C Programming Language* 一书中，第一个例子就是一段小程序，可以在屏幕上输出 **Hello, World**

```
main()  
{  
    printf("hello, world");  
}
```

- Java版的Hello World程序:

```
// Save to HelloWorld.java  
public class HelloWorld  
{  
    public static void main(String[] args)  
    {  
        System.out.println("Hello, world");  
    }  
}
```



## ■ Hello World 攻略

- 安装JDK，并配置 path 环境变量
- 在命令行中，分别执行：
  - java -version，出现Java版本信息
  - javac -version，出现Java编译器的版本信息

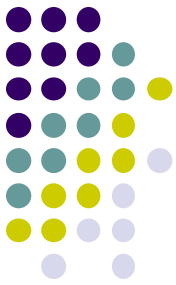


```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Julie>java -version
java version "1.7.0_45"
Java(TM) SE Runtime Environment (build 1.7.0_45-b18)
Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)

C:\Users\Julie>
```

- 使用记事本/UltraEdit/JCreator，输入上面的Hello World代码。  
注意：一个字都不能错，注意大小写！**Java是大小写敏感的！**



The screenshot shows the Notepad++ application window titled '\*new 1 - Notepad++'. The menu bar includes '文件(F)', '编辑(E)', '搜索(S)', '视图(V)', '格式(M)', '语言(L)', '设置(I)', '宏(O)', '运行(R)', 'TextFX', '插件(P)', '窗口(W)', and '?'. The toolbar contains various icons for file operations and editing. The code editor displays the following Java code with syntax highlighting: line 1 is a comment '// Save to HelloWorld.java'; line 2 is 'public class HelloWorld {'; line 3 is 'public static void main(String[] args) {'; line 4 is 'System.out.println("Hello, world");'; line 5 is '}' (closing the main method); line 6 is '}' (closing the class); and line 7 is empty. The status bar at the bottom shows 'Java sou', 'nb char : 150', 'Ln : 7', 'Col : 1', 'Sel : 0', 'UNIX', 'ANSI', and 'INS'.

```
1 // Save to HelloWorld.java
2 public class HelloWorld {
3     public static void main(String[] args) {
4         System.out.println("Hello, world");
5     }
6 }
7
```

Notepad++  
带语法高亮功能（在“语言”菜单中，选择“Java”）即可

The screenshot shows the Windows Notepad application window titled '无标题 - 记事本'. The menu bar includes '文件(F)', '编辑(E)', '格式(O)', '查看(V)', and '帮助(H)'. The code editor displays the same Java code as the Notepad++ window, but without syntax highlighting: line 1 is a comment '// Save to HelloWorld.java'; line 2 is 'public class HelloWorld {'; line 3 is 'public static void main(String[] args) {'; line 4 is 'System.out.println("Hello, world");'; line 5 is '}' (closing the main method); line 6 is '}' (closing the class); and line 7 is empty.

```
// Save to HelloWorld.java
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world");
    }
}
```

记事本



- **然后将文件另存为：D:\Java\Demo\HelloWorld.java**
  - 文件存放到 D:\Java\Demo 目录（当然也可以其他目录）
  - 文件名一定是 HelloWorld.java，注意大小写！
- **编译：打开命令行窗口，编译 HelloWorld.java：**
  - 进入到 D:\Java\Demo 目录，执行：javac HelloWorld.java

```
C:\Windows\system32\cmd.exe
2014/02/16 17:28 <DIR>      ..
2014/02/17 09:28      425 HelloWorld.class
2013/12/24 12:35     123 HelloWorld.java
                2 个文件      548 字节
                2 个目录 90,968,535,040 可用字节

D:\Java\Demo>javac HelloWorld.java

D:\Java\Demo>java HelloWorld
HelloWorld!

D:\Java\Demo>
```

- 没有任何输出，说明已经编译好了。
- 然后可以发现生成了 HelloWorld.class 文件





- **运行:**

- 在 D:\Java\Demo 目录中, 执行: **java HelloWorld**

```
C:\Windows\system32\cmd.exe
2014/02/16 17:28 <DIR>      ..
2014/02/17 09:28          425 HelloWorld.class
2013/12/24 12:35          123 HelloWorld.java
                2 个文件          548 字节
                2 个目录 90,968,535,040 可用字节

D:\Java\Demo>javac HelloWorld.java

D:\Java\Demo>java HelloWorld
HelloWorld!

D:\Java\Demo>
```

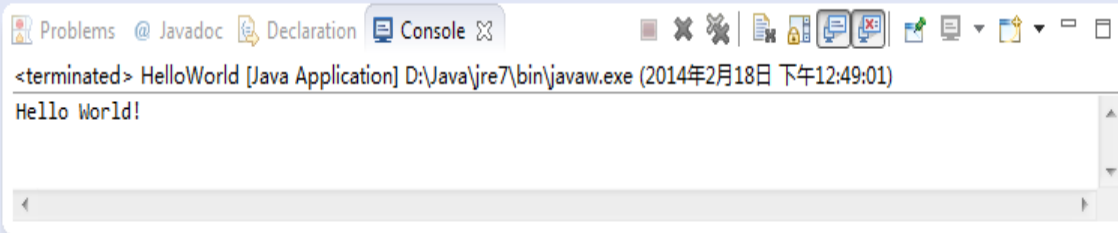
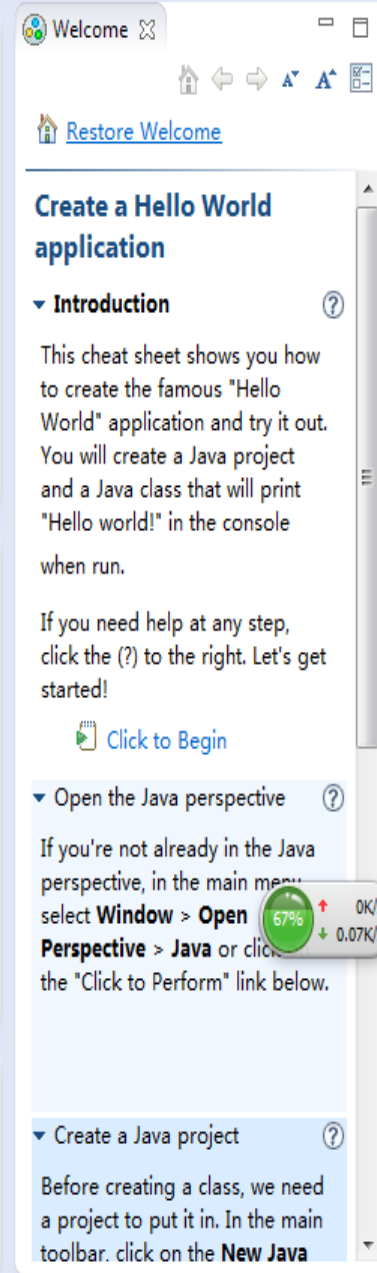
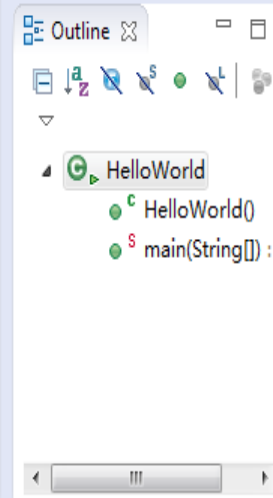
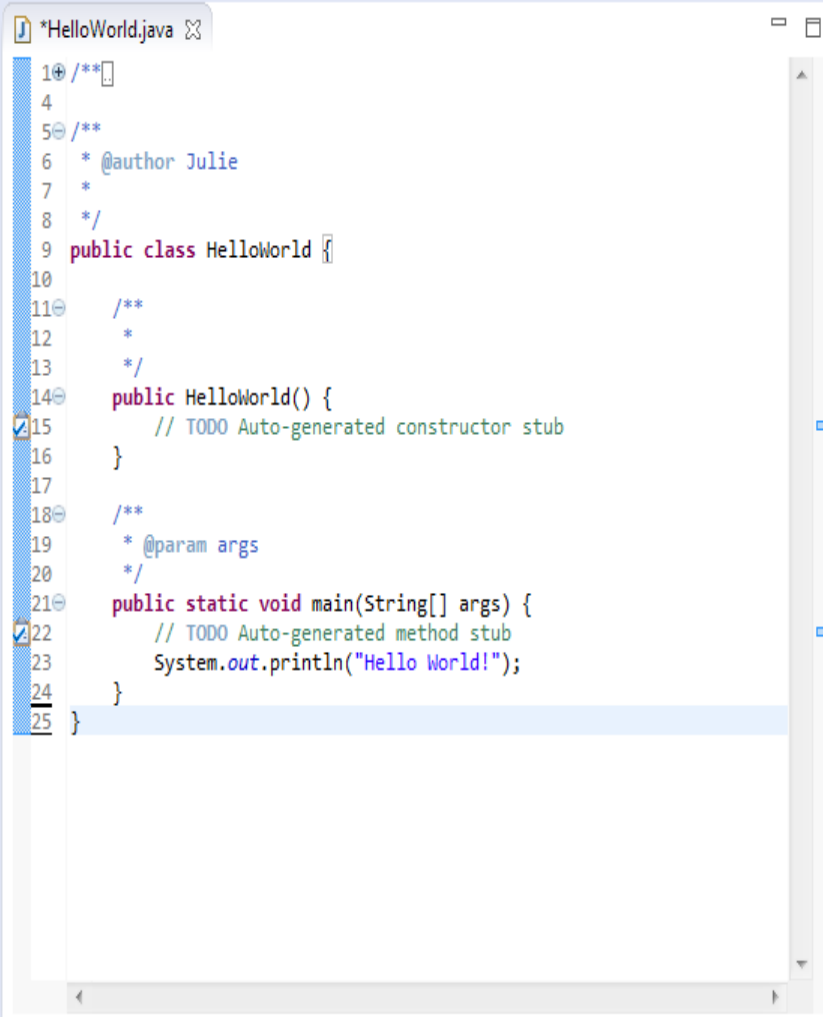
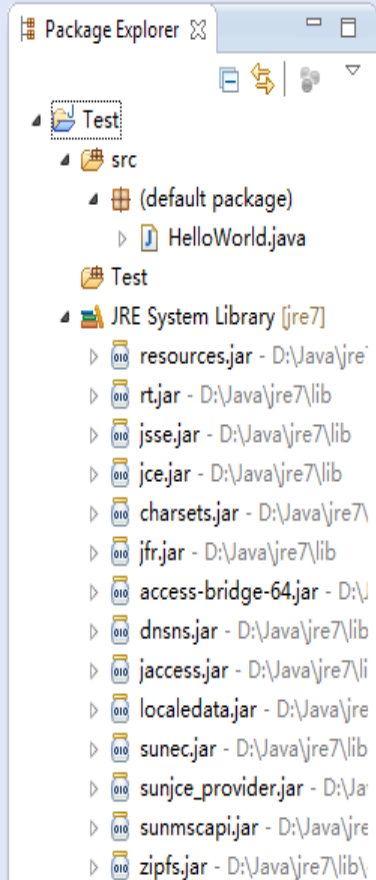
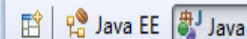
- 可以看到, 输出了 “Hello, world”

**注意:**

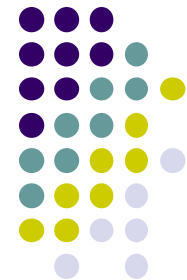
执行时, 不能加文件名后缀, 不能写成 **java HelloWorld.class**



Quick Access

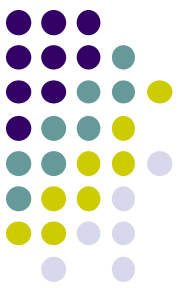


# Java程序剖析



## ■ 一个Java文件，包括：

- 注释
- 保留字
- 修饰符
- 语句
- 块
- 类
- 方法
- main方法



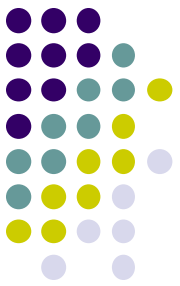
## ■ 注释 (comments)

- 单行注释：用连续的两个斜杠 (//) 引导
- 多行注释：用 /\* 和 \*/ 括住
- 编译器遇到//时，忽略本行//之后的所有内容，遇到/\*时，扫描找到\*/并将/\*与\*/之间的内容忽略

```
// This is single line comment
public class HelloWorld {
    /* This is the main method */
    public static void main(String[] args) {
        System.out.println("Hello, world"); // print hello world
    }
    /* nested comments here
       /*  multiline comments...
    */
}
```

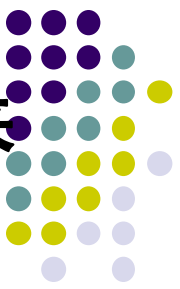
- 此外，Java还有一种特殊的注释方式：

```
/**
    multiline comments here
 */
```



- **保留字** (reserved word、keyword)
  - 对编译器具有特殊意义，在程序中不能用作其他目的的字，如：class、public、static、void 等
  - Java关键字列表(共50个):

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while



- **修饰符**：Java使用被称为修饰符的某些保留字来指定数据、方法和类的属性与使用方式。
  - 修饰符的例子有：`public` 和 `static`
  - 其它修饰符还有 `private`, `final`, `abstract` 和 `protected`
  - 修饰符将在后面章节中详细讨论
- **语句**（Statement）
  - 一条语句表示一个操作或一系列操作
  - 例程中的 `System.out.println("Hello, world");` 就是一条语句
  - 语句都用分号（`;`）结束



## ■ 语句块 (Block)

- 在程序中用花括号将程序的一些成分组合起来，构成一个块 (block)

```
// Save to HelloWorld.java
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world");
    }
}
```

← 方法块      ← 类块



## ■ 类（class）

- 类（class）是Java的基本结构。类是对象的模板和蓝图。
- 为了编写Java程序，必须理解类并能编写和使用它们。
- 类的概念将在后面的章节里专门介绍，现在只要理解程序是使用一个或多个类定义的即可。





## ■ 方法（method）

### ● **System.out.println**是什么？

- System是类（class）
- out是System类中的一个成员变量（对象，object）
- println是out对象的方法（method）：它可以在控制台上打印出一条消息。打印的消息内容就是该方法的参数。字符串参数用圆括号括住：

```
System.out.println("Hello, world");
```

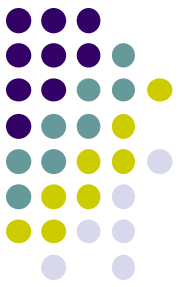
- **本例中的参数是“Hello, world”。可以用不同的参数来调用println方法，从而打印不同的信息**



## ■ main方法

- 运行过程主类必须有一个主方法main(), 作为Java程序运行的入口。
- Java 解释器通过调用main方法执行应用程序
- main方法必须定义为:

```
public static void main(String[] args) {  
    // some statements  
}
```



## ■ 注意：

- 类源文件名=公有类名+ .java
- 如果一个类被声明为public，则它本身所在的源文件名也必须与类名相同，连大小写都不能错！
- 这并非说一个Java源文件中只能写一个类，完全可以写多个类，但其中只能有一个类是“公有（public）”的，并且Java要求文件名也要与之一致。

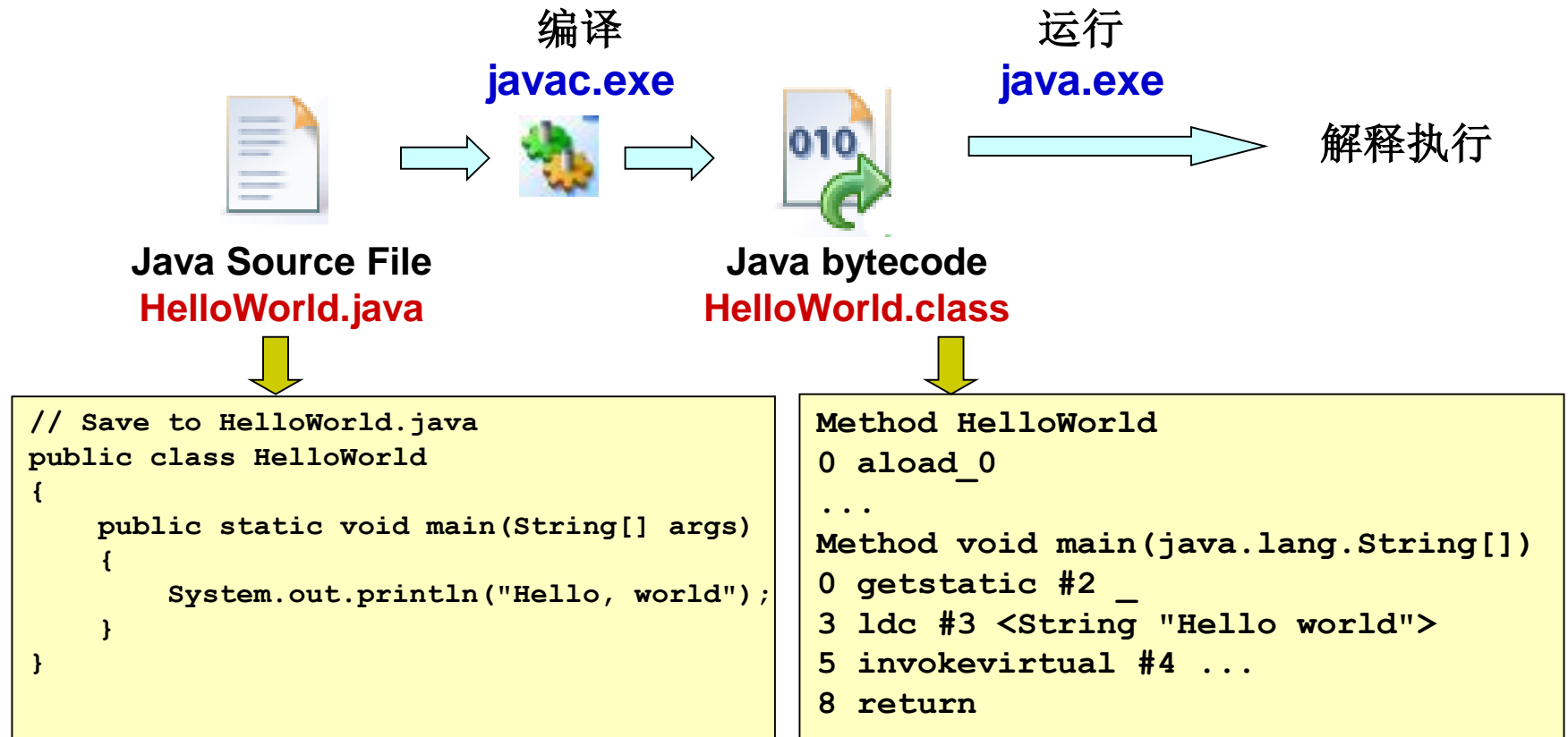


# 一个典型的 Java 程序开发过程

- Java 程序通常经历5个阶段：
  - **Edit**（编辑）  
程序员书写代码并保存到 磁盘上
  - **Compile**（编译）  
编译器生成字节码（bytecodes）
  - **Load**（装入）  
类装载器（Class loader）在内存中保存字节码
  - **Verify**（校验）  
校验器Verifier保证字节码不违反安全规则
  - **Execute**（执行）  
解释器将字节码翻译成机器码



- **javac.exe**: 是编译Java源代码的程序, 在JRE中没有这个程序, 只在JDK中有
- **java.exe**: 是解释执行Java字节码的程序, JRE和JDK中都有



# 用消息对话框显示 Hello, World



- 前一个例子是将**Hello, world**打印到控制台中
- 大多数**Java**应用程序使用窗口或对话框
- 利用**JOptionPane**类中的**showMessageDialog**方法，可以在消息对话框中显示文本消息
  - **JOptionPane**是Java系统中众多的预定义类之一
  - 源代码：

```
// Show "Hello, world" in message dialog
public class HelloWorld2 {
    public static void main(String[] args) {
        javax.swing.JOptionPane.showMessageDialog(null,
            "Hello, world",
            "welcome",
            javax.swing.JOptionPane.INFORMATION_MESSAGE);
    }
}
```

- 将文件保存为**HelloWorld2.java**，并编译、执行之：
  - `javac HelloWorld2.java`
  - `java HelloWorld`



## ■ 运行结果:



```
// Show "Hello, world" in message dialog
public class HelloWorld2 {
    public static void main(String[] args) {
        javax.swing.JOptionPane.showMessageDialog(null,
            "Hello, world",
            "welcome",
            javax.swing.JOptionPane.INFORMATION_MESSAGE);
    }
}
```

## § 1.4 Java程序：Java Applet



- **Java应用是可以独立运行的Java程序**，定义的main() 方法，是程序运行的起始点。
- Applet是Java与WWW结合后的一个重要概念。Applet就是一嵌入HTML文件的Java程序。**Applet不能独立执行**，必须嵌入HTML文件中通过浏览器或 Appletviewer加载执行。



## § 1.4 Java程序: Java Applet



### **JAVA Applet:**

```
import java.awt.Graphics;
import java.applet.Applet;
public class MyApplet extends Applet
{
    public String s;
    public void init()
    {    s=new String("Hello World !");    }
    public void paint(Graphics g)
    {    g.drawString(s,25,25);    }
}
```

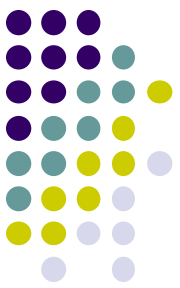
### **HTML:**

```
<applet code= MyApplet.class width=400 height=400>
</applet>
```

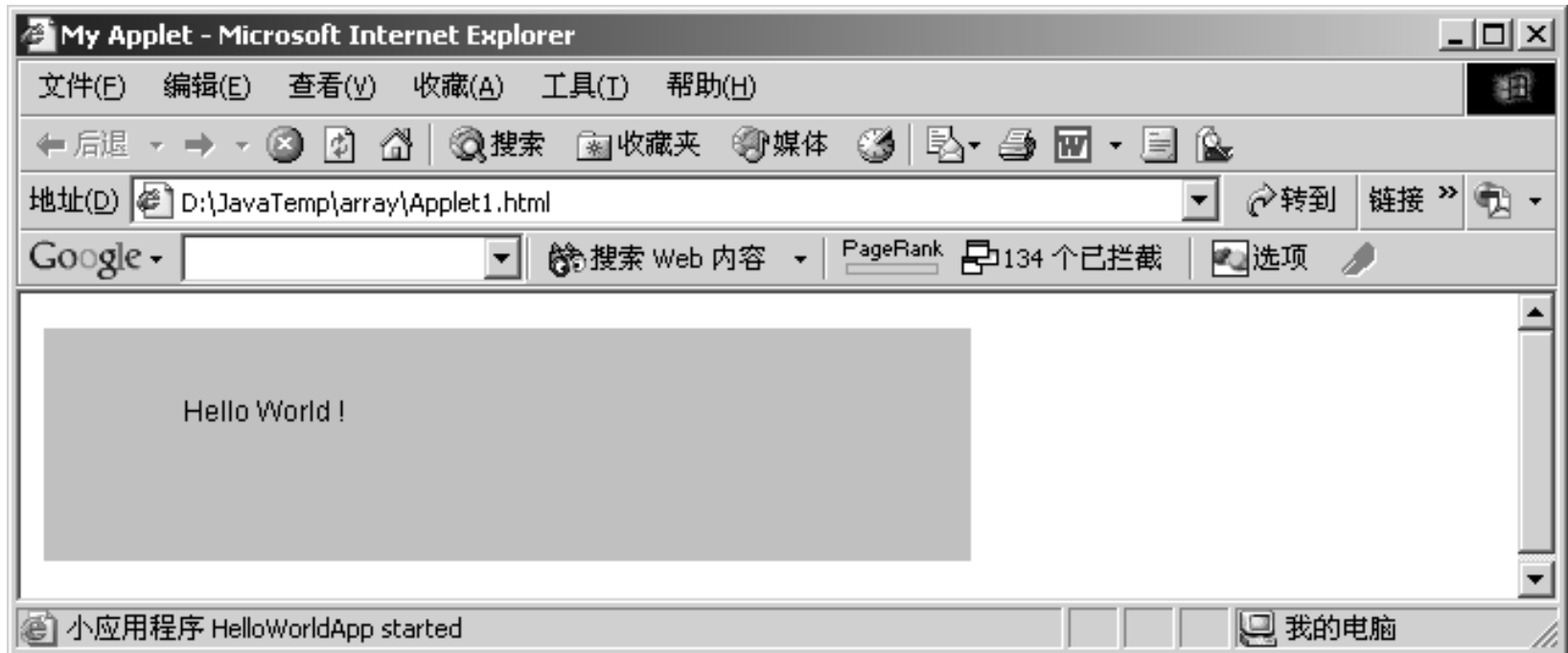
## § 1.4 Java程序： Java Applet



- **Graphics类**
  - 使得applet绘制直线、矩形、椭圆形、字符串等
- **方法init()**
  - 初始化，实现了字符串的创建
- **方法paint() 中**
  - g为Graphics类的对象。调用了Graphics的drawString方法绘制字符串。
  - 此方法执行的结果就是从坐标(60,40)开始绘制出字符串Hello World! 。

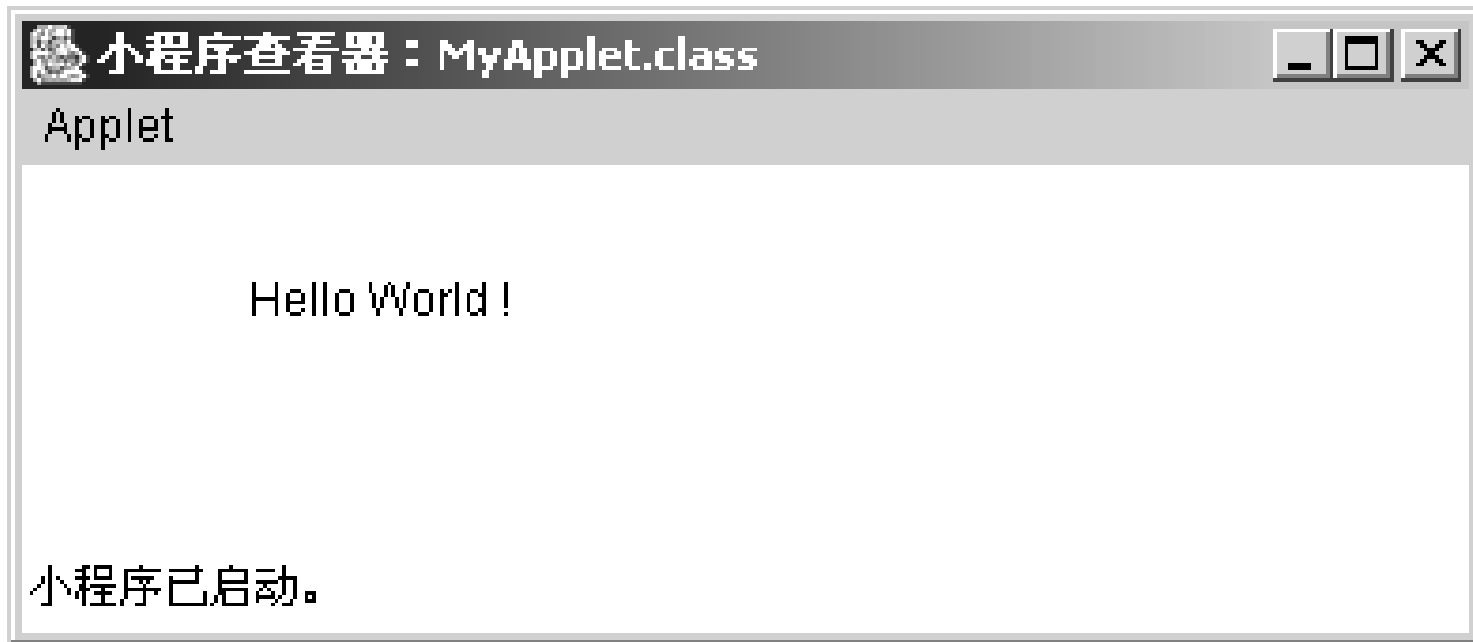


## ■ 用支持Java的浏览器，打开Applet1.html





- 用Java自带的appletviewer浏览
  - 输入：appletviewer Applet1.html



# § 1.4 Java程序： Java Applet



- **编辑：** WelcomeApplet.java
- **编译：** javac WelcomeApplet.java
- **嵌入**

**<HTML>**

**<APPLET CODE="WelcomeApplet.class" >**

**</APPLET>**

**</HTML>**

- **执行**
  - **appletviewer welcome.html;**
  - **浏览器执行**

## § 1.4 Java程序：Java Applet



### ■ Applet——小应用程序

- Applet是Java与WWW结合后的一个重要概念。Applet就是一嵌入HTML文件的Java程序。
- 运行于支持Java的Web浏览器中
- Applet不能独立执行，必须嵌入HTML文件中通过浏览器或 Appletviewer加载执行。
- 浏览器的解释器把字节码转换成和机器匹配的指令，在网页中执行小程序。
- **Applet和Application的差别：**运行环境的不同，小应用程序总是放在Web浏览器的图形用户界面中

## § 1.4 Java程序：Java Applet



- Applet的优点
  - Web 浏览器软件包括很多小应用程序运行所需的功能
- Applet的局限性
  - 在客户端主机的文件系统中读/写受限
  - 不能运行客户端主机的任何程序
  - 仅能在服务器和客户端之间建立联系

## § 1.4 Java程序：Java Applet



- 早期的互联网中，Applet还有一定的应用，比如用它来开发一些网页小游戏，或者作为一个互联网应用的客户端程序，但功能更为强大灵活的Flash挤压了它的生存空间，雪上加霜的是，后来微软推出了Silverlight参与客户端争夺战，而Sun自己也推出了一个JavaFX来取代Applet，2010年以后，移动互联网火爆，在这场正在进行的客户端开发技术大战中，最后的胜者应该是**HTML5**，而Applet基本没戏，因此，客户端Applet可以说是一个被淘汰的技术。
- 但由于Applet绘图比较方便，在Java教学中有时还使用它来编写一些教学示例，不过仅此而已。同学们不必花费太多时间于Applet上，知道怎么回事，并且会运行它即可。



# Java怎么学？



- **Java体系庞大**
  - **Java语言基础 - 本课程要教给大家的**
  - **面向对象基础**
  - **JDK庞大的类库**
  - .....
- **找一本好书**
- **自己敲代码**
- **善用网络资源**
- **循序渐进，在实践中提高**

# 对Java学习者说



- 在学习一门体现了最新的软件开发技术的语言，  
就业前景广阔
- 学Java的关键点是对面向对象理论和思想的理解  
与体会
- 要想学好Java，掌握UML、OOAD（重构、设计  
模式、单元测试.....）、软件工程的基础知识与  
技能也是非常重要的。

# 学好Java的基本途径



- **多上机编程实践**——这是最基本最重要的学习方法
- **多看书上网**——网上可以找到有关Java的大量技术书籍和视频教程
- **寻找机会参与实际项目的开发**——只有通过实战才能成长成为一名优秀的软件工程师

# 入门之后再提高



- 进一步学习**Android**平台，开发智能手机应用
- 进一步学习**Java EE**，开发大规模的企业级应用
- 迈入开源的世界，深入了解与分享**Java**技术的深度应用
- 走理论与实践相结合的路，用**Java**编写软件解决各种实际问题，将迸发出的奇思妙想变为现实。

# Sun公司没有了，Java还有前途吗？



# 本章小结



1. **Java语言的特点（重点掌握平台无关性和面向对象）**
2. **Java虚拟机**
3. **Java Application和Applet的基本程序结构**
4. **Java Application和Applet的开发过程**

# 本章习题



1. 写出**Java**语言的主要特点。
2. 什么是**Java**虚拟机（**JVM**）？具体解释 **Java**虚拟机的工作流程。
3. **Java** 应用和小应用的区别是什么？（从基本程序结构、使用场合、运行方式等几方面说明）
4. 什么是字节码？
5. 什么是**Java API**？
6. **Java** 字节码与其他机器码有何不同？
7. 在计算机编程方面，可移植的含义是什么？
8. 什么是大小写敏感？
9. 在计算机上安装**JDK**、**Eclipse/JCreator**
10. 在计算机上完成**HelloWorld**程序（控制台版、消息对话框版）