[第1章 Java入门 1](#_Toc108626634)

[1.1 Java的地位 1](#_Toc108626635)

[1.2 Java的特点 1](#_Toc108626636)

[1.3 安装JDK 1](#_Toc108626637)

[1.4 Java程序的开发步骤 1](#_Toc108626638)

[1.5 简单的Java应用程序 1](#_Toc108626639)

[1.6 Java反编译 1](#_Toc108626640)

[1.7 编程风格 1](#_Toc108626641)

[1.8 Java之父——James Gosling 1](#_Toc108626642)

[第2章 基本数据类型与数组 1](#_Toc108626643)

[2.1 标识符与关键字 1](#_Toc108626644)

[2.2 基本数据类型 1](#_Toc108626645)

[2.3 类型转换运算 1](#_Toc108626646)

[2.4 输入、输出数据 1](#_Toc108626647)

[2.5 数组 1](#_Toc108626648)

[第3章 运算符、表达式和语句 1](#_Toc108626649)

[3.1 运算符与表达式 1](#_Toc108626650)

[3.2 语句概述 1](#_Toc108626651)

[3.3 if条件分支语句 1](#_Toc108626652)

[3.4 switch开关语句 1](#_Toc108626653)

[3.5 循环语句 1](#_Toc108626654)

[3.6 break和continue语句 1](#_Toc108626655)

[3.7 for语句与数组 1](#_Toc108626656)

[第4章 类与对象 1](#_Toc108626657)

[4.1 编程对象的几个发展阶段 1](#_Toc108626658)

[4.2 类 1](#_Toc108626659)

[4.3 构造方法与对象的创建 1](#_Toc108626660)

[4.4 类与程序的基本结构 1](#_Toc108626661)

[4.5 参数传值 2](#_Toc108626662)

[4.6 对象的组合 2](#_Toc108626663)

[4.7 实例成员与类成员 2](#_Toc108626664)

[4.8 方法重载 2](#_Toc108626665)

[4.9 this关键字 2](#_Toc108626666)

[4.10 包 2](#_Toc108626667)

[4.11 import语句 2](#_Toc108626668)

[4.12 权限访问 2](#_Toc108626669)

[4.13 基本类型的类封装 2](#_Toc108626670)

[4.14 对象数组 2](#_Toc108626671)

[4.15 JRE扩展与jar文件 2](#_Toc108626672)

[4.16 文档生成器 2](#_Toc108626673)

[4.17 应用案例 2](#_Toc108626674)

[第5章 子类与继承 2](#_Toc108626675)

[5.1 子类与父类 2](#_Toc108626676)

[5.2 子类的继承性 2](#_Toc108626677)

[5.3 子类与对象 2](#_Toc108626678)

[5.4 成员变量的隐藏与方法重写 3](#_Toc108626679)

[5.5 super关键字 3](#_Toc108626680)

[5.6 final关键字 3](#_Toc108626681)

[5.7 对象的上转型对象（老虎是动物、狗是动物） 3](#_Toc108626682)

[5.8 继承与多态 3](#_Toc108626683)

[5.9 abstract类和abstract方法 3](#_Toc108626684)

[5.10 面向抽象编程的思想 3](#_Toc108626685)

[5.11 开-闭原则 3](#_Toc108626686)

[5.12 应用举例 4](#_Toc108626687)

[第6章 接口与实现 4](#_Toc108626688)

[6.1 接口 4](#_Toc108626689)

[6.2 实现接口 4](#_Toc108626690)

[6.3 接口的UML图 4](#_Toc108626691)

[6.4 接口回调 4](#_Toc108626692)

[6.5 理解接口 4](#_Toc108626693)

[6.6 接口与多态 4](#_Toc108626694)

[6.7 接口参数 4](#_Toc108626695)

[6.8 Abstract类与接口的比较 4](#_Toc108626696)

[6.9 面向接口编程（\*UML） 4](#_Toc108626697)

[6.10 应用举例（\*UML） 4](#_Toc108626698)

[第7章 内部类与异常 4](#_Toc108626699)

[7.1 内部类 4](#_Toc108626700)

[7.2 匿名类 4](#_Toc108626701)

[7.3 异常类 4](#_Toc108626702)

[7.4 断言 4](#_Toc108626703)

[第8章 常用实用类 4](#_Toc108626704)

[8.1 String类 4](#_Toc108626705)

[8.2 StringTokenizer类 4](#_Toc108626706)

[8.3 Scanner类 5](#_Toc108626707)

[8.4 StringBuffer类 5](#_Toc108626708)

[8.5 Date类与Calender类 5](#_Toc108626709)

[8.6 日期的格式化 5](#_Toc108626710)

[8.7 Math类、BigInteger类和Random类 5](#_Toc108626711)

[8.8 数字格式化 5](#_Toc108626712)

[8.9 Class类与Console类 5](#_Toc108626713)

[8.10 Pattern类与Matcher类 5](#_Toc108626714)

[第9章 组件及事件处理 5](#_Toc108626715)

[9.1 Java Swing概述 5](#_Toc108626716)

[9.2 窗口 5](#_Toc108626717)

[9.3 常用组件与布局 5](#_Toc108626718)

[9.4 处理事件 5](#_Toc108626719)

[9.5 使用MVC结构 5](#_Toc108626720)

[9.6 对话框 5](#_Toc108626721)

[9.7 树组件与表格组件 5](#_Toc108626722)

[9.8 按钮绑定到键盘上（AbstractAction） 5](#_Toc108626723)

[9.9 打印组件 5](#_Toc108626724)

[9.10 发布GUI程序 5](#_Toc108626725)

[第10章 输入、输出流 5](#_Toc108626726)

[10.1 File类 5](#_Toc108626727)

[10.2 文件字节输入流（FileInputStream） 5](#_Toc108626728)

[10.3 文件字节输出流（FileOutputStream） 5](#_Toc108626729)

[10.4 文件字符输入、输出流（FileReader、FileWriter） 6](#_Toc108626730)

[10.5 缓冲流（BufferedReader、BufferedWriter） 6](#_Toc108626731)

[10.6 随机流（RandomAccessFile） 6](#_Toc108626732)

[10.7 数组流 6](#_Toc108626733)

[10.8 数据流（DataInputStream、DataOutputStream） 6](#_Toc108626734)

[10.9 对象流（ObjectInputStream、ObjectOutputStream） 6](#_Toc108626735)

[10.10 序列化与对象克隆 6](#_Toc108626736)

[10.11 使用Scanner解析文件 6](#_Toc108626737)

[10.12 文件对话框（JFileChooser） 6](#_Toc108626738)

[10.13 带进度条的输入流（ProgressMonitorInputStream） 6](#_Toc108626739)

[10.14 文件锁（RandomAccessFile） 6](#_Toc108626740)

[第11章 JDBC与MySQL数据库 6](#_Toc108626741)

[11.1 MySQL数据库管理系统 6](#_Toc108626742)

[11.2 启动MySQL数据库服务器 6](#_Toc108626743)

[11.3 MySQL客户端管理工具 6](#_Toc108626744)

[11.4 JDBC 6](#_Toc108626745)

[11.5 连接数据库 6](#_Toc108626746)

[11.6 查询操作 6](#_Toc108626747)

[11.7 更新、添加与删除工作 6](#_Toc108626748)

[11.8 使用预处理语句 6](#_Toc108626749)

[11.9 通用查询 6](#_Toc108626750)

[11.10 事务 6](#_Toc108626751)

[11.11 连接SQL Server数据库 6](#_Toc108626752)

[11.12 连接Derby数据库 6](#_Toc108626753)

[11.13 应用举例 6](#_Toc108626754)

[第12章 Java多线程机制 6](#_Toc108626755)

[第13章 Java网络编程 6](#_Toc108626756)

[第14章 图形、图像与音频 6](#_Toc108626757)

[第15章 泛型与集合框架 6](#_Toc108626758)

[15.1 泛型 6](#_Toc108626759)

[15.2 链表 6](#_Toc108626760)

[15.3 堆栈 6](#_Toc108626761)

[15.4 散列映射 6](#_Toc108626762)

[15.5 数集 6](#_Toc108626763)

[15.6 树映射 6](#_Toc108626764)

[15.7 自动装箱与拆箱 6](#_Toc108626765)

[附录1 UML图 7](#_Toc108626766)

[1、 类的UML图 7](#_Toc108626767)

[2、 类的关联关系的UML图 7](#_Toc108626768)

[3、 类的依赖关系的UML图 7](#_Toc108626769)

[4、 继承关系的UML图 7](#_Toc108626770)

[5、 开-闭原则的UML图 7](#_Toc108626771)

[6、 接口的UML图 7](#_Toc108626772)

[7、 7](#_Toc108626773)

[附录2 Scanner类的使用 7](#_Toc108626774)

[1、 输入基本型数据 7](#_Toc108626775)

[2、 7](#_Toc108626776)

[附录3 方法重载与方法重写 7](#_Toc108626777)

[1、 方法重载 7](#_Toc108626778)

[2、 方法重写 7](#_Toc108626779)

[附录4 跟包相关的 7](#_Toc108626780)

[附录5 访问权限 7](#_Toc108626781)

[1、 私有变量和私有方法（private） 7](#_Toc108626782)

[2、 共有变量和共有方法（public） 8](#_Toc108626783)

[3、 友好变量和友好方法（无private、public和protected） 8](#_Toc108626784)

[4、 受保护的成员变量和方法（protected） 8](#_Toc108626785)

[5、 public类与友好类 8](#_Toc108626786)

[6、 protected的进一步说明 8](#_Toc108626787)

[附录6 8](#_Toc108626788)

[11.6 查询操作 8](#_Toc108626789)

[11.7 更新、添加与删除工作 8](#_Toc108626790)

[11.8 预处理语句 8](#_Toc108626791)

[11.9通用查询 8](#_Toc108626792)

[11.10事务 8](#_Toc108626793)

[附录7 9](#_Toc108626794)

# Java入门

## Java的地位

## Java的特点

## 安装JDK

## Java程序的开发步骤

## 简单的Java应用程序

## Java反编译

## 编程风格

## Java之父——James Gosling

# 基本数据类型与数组

## 标识符与关键字

### 标识符

### Unicode字符集

### 关键字

## 基本数据类型

### 逻辑类型

### 整数类型

### 字符类型

### 浮点类型

## 类型转换运算

## 输入、输出数据

### 输入基本型数据

### 输出基本型数据

## 数组

# 运算符、表达式和语句

## 运算符与表达式

## 语句概述

## if条件分支语句

## switch开关语句

## 循环语句

## break和continue语句

## for语句与数组

# 类与对象

## 编程对象的几个发展阶段

### 面向机器语言

### 面向对象语言

### 面向对象语言

#### 封装性

#### 继承

#### 多态

## 类

### 类声明

### 类体

### 成员变量

#### 成员变量的类型

#### 成员变量的有效范围

#### 成员变量的编程风格

### 方法

#### 方法头

#### 方法体

#### 区分成员变量与局部变量

#### 局部变量没有默认值

### 需要注意的问题

### 类的UML图

## 构造方法与对象的创建

### 构造方法

#### 默认构造方法与自定义构造方法

#### 构造方法没有类型

### 创建对象

#### 对象的声明

#### 为声明的对象分配变量

#### 创建对象时系统的内存模型

### 使用对象

#### 对象可以操作自己的变量（体现了对象的属性）

#### 对象可以调用类中的方法（体现了对象的行为）

#### 体现封装

### 对象的引用和实体

#### 避免使用空对象

#### 一个重要结论

#### 垃圾收集

## 类与程序的基本结构

## 参数传值

#### 传值机制

#### 基本数据类型参数的传递

#### 引用数据类型参数的传递

#### 可变参数

## 对象的组合

### 组合与复用

### 类的关联关系的UML图

### 类的依赖关系的UML图

## 实例成员与类成员

### 实例变量和类变量的声明

### 实例变量和类变量的区别

#### 不同对象的实例互不相同

#### 所有对象共享类变量

#### 通过类名直接访问类变量

### 实例方法和类方法的定义

### 实例方法和类方法的区别

#### 对象调用实例方法

#### 类名调用类方法

#### 类方法的设计原则

## 方法重载

### 方法重载的语法规则

### 避免重载出现歧义

## this关键字

### 在构造方法中使用this

### 在实例方法中使用this

#### 通常情况下，可以省略变量名字前的“this.”以及static变量前面的“类名.”

#### 但是，当实例成员变量的名字和局部变量的名字相同时，成员变量前面的“this.”或“类名.”就不可以省略

#### 一个实例方法调用另一个方法是可以省略方法名字前面的“this.”或“类名”

## 包

### 包语句

### 有包名的类的存储目录

### 运行有包名的主类

## import语句

### 引入类库中的类

### 引入自定义包中的类

## 权限访问

### 何谓访问权限

### 私有变量和私有方法（private）

### 共有变量和共有方法（public）

### 友好变量和友好方法（无private、public、protected）

### 受保护的成员变量和方法（protected）

### public类与友好类

## 基本类型的类封装

### Double和Float类

### Byte、Short、Integer、Long类

### Character类

## 对象数组

## JRE扩展与jar文件

## 文档生成器

## 应用案例

### 有理数的封装

### 搭建流水线

# 子类与继承

## 子类与父类

### 子类

### 类的树形结构

## 子类的继承性

### 子类和父类在同一包中的继承性

### 子类和父类不在同一包中的继承性

### 继承关系的UML图

### protected的进一步说明

## 子类与对象

### 子类对象的特点

### 关于instanceof运算符

## 成员变量的隐藏与方法重写

### 成员变量的隐藏

#### 子类对象以及子类自己定义的方法，操作与父类同名的成员变量，实际上操作的是指子类重新声明的这个成员变量。

#### 子类继承的方法所操作的成员变量一定是被子类继承或隐藏的成员变量。

#### 子类新定义的方法可以操作子类继承和子类新声明的成员变量，但无法操作子类隐藏的成员变量（需使用super关键字）。

### 方法重写

#### 重写的语法规则

#### 重写的目的

#### 重写的注意事项（\*访问权限）

## super关键字

### 使用super操作被隐藏的成员变量和方法

### 使用super调用父类的构造方法

## final关键字

### final类

### final方法

### 常量

## 对象的上转型对象（老虎是动物、狗是动物）

对象的上转型对象  
（animal）

对象（tiger）

隐藏的变量

继承的变量

新增的变量

继承或重写的方法

新增的方法

## 继承与多态

## abstract类和abstract方法

对于抽象类：

#### 抽象类中可以有抽象方法和非抽象方法，也可以没有抽象方法。普通类中不可以有抽象方法

#### 不能用new运算符来创建抽象类的对象。

#### 如果一个非抽象类是抽象类的子类，那这个类必须重写抽象类的抽象方法。如果一个抽象类是抽象类的子类，那这个类可以重写抽象类的抽象方法，也可以继承抽象类的抽象方法。

#### 可以使用抽象类声明对象，该对象可以成为其子类的上转型对象，该对象可以调用子类重写的方法。

抽象类的作用：

#### 抽象类封装了子类必须要有的行为标准。

#### 抽象类声明的对象可以成为其子类的上转型对象，可以体现子类根据抽象类的行为标准给出的具体行为。

## 面向抽象编程的思想

面向抽象编程的目的是为了应对用户需求的变化，将某个类（如例子中的Pillar类，底部可能是多种形状）中经常因需求变化而需要改动的代码从该类中分离出去。

面向抽象的核心是让类中每种可能的变化对应地交给抽象类的一个子类去负责，从而让该类的设计者不去关系具体实现，避免所设计的类依赖于具体的实现。

面向抽象编程使设计的类容易应对用户需求的变化。

## 开-闭原则

## 应用举例

# 接口与实现

## 接口

### 接口声明

### 接口体

## 实现接口

### 类实现接口

### 重写接口中的方法（\*访问权限）

### 接口的细节说明（\*访问权限、\*包）

## 接口的UML图

## 接口回调

ShowMessage showMessage = new TV()

/\*类TV实现了接口ShowMessage，showMessage变量可以调用类实现的接口方法（这一过程被称为接口回调）\*/

## 理解接口

## 接口与多态

## 接口参数

## Abstract类与接口的比较

## 面向接口编程（\*UML）

## 应用举例（\*UML）

interface Advertisement //制定产品有什么功能的标准

class AdvertisementBoard //制定产品有什么行为的标准

class WhiteCloudCorp implements Advertisement //创建一个具体的产品A，并拥有Advertisement中列举的功能

class BlackLandCorp implements Advertisement //创建一个具体的产品B，并拥有Advertisement中列举的功能

class Example6\_6 //主类，具体地操作类

# 内部类与异常

## 内部类

## 匿名类

## 异常类

## 断言

# 常用实用类

## String类

### 构造String对象

### 字符串的并置

### String类的常用方法

#### public int length()

#### public boolean equals(String s)

#### public boolean startsWith(String s)、public boolean endsWith(String s)

#### public int compareTo(String s)

#### public boolean contains(String s)

#### public int indexOf(String s)、public int lastIndexOf(String s)

#### public String subString(int startpoint)

#### public String trim()

### 字符串与基本数据的相互转换

### 对象的字符串表示

### 字符串与字符数组、字节数组（String.getChars()）

#### 字符串与字符数组（getChars()、toCharArray()）

#### 字符串与字节数组（getBytes()）

#### 字符串的加密算法

### 正则表达式及字符串的替换与分解

#### 正则表达式

#### 字符串的替换（.replaceAll()）

#### 字符序列的分解（.split()）

## StringTokenizer类

## Scanner类

## StringBuffer类

## Date类与Calender类

## 日期的格式化

## Math类、BigInteger类和Random类

## 数字格式化

## Class类与Console类

## Pattern类与Matcher类

# 组件及事件处理

**Component**

**Container**

**JComponent**

**Window**

**JButton**

**JtextField**

**JtextArea**

**JTree**

**Jtable**

**JPanel**

**Frame**

**JFrame**

**JDialogialog**

**Dialog**

## Java Swing概述

## 窗口

### Jframe常用方法

### 菜单条、菜单、菜单项

## 常用组件与布局

### 常用组件

#### JTextField（文本框）

#### JTextArea（文本区）

#### JButton（按钮）

#### JLabel（标签）

#### JCheckBox（复选框）

#### JRadioButton（单选按钮）

#### JComboBox（下拉列表）

#### JPasswordField（密码框）

### 常用容器

#### JPanel面板

#### JTabbedPane选项卡窗格

#### JScrollPane滚动窗格

#### JSplitPane拆分窗格

#### JLayeredPane分层窗格

### 常用布局

#### FlowLayout布局

#### BorderLayout布局

#### CardLayout布局

#### GridLayout布局

#### null布局

#### BoxLayout布局

## 处理事件

### 事件处理模式

### ActionEvent事件

### ItemEvent事件

### DocumentEvent事件

### MouseEvent事件

### 焦点事件

### 键盘事件

## 使用MVC结构

## 对话框

### 消息对话框（showMessageDialog）

### 输入对话框（showInputDialog）

### 确认对话框（showConfirmDialog）

### 颜色对话框（showDialog）

### 自定义对话框（JDialog）

## 树组件与表格组件

### 树组件（DefaultMutableTreeNode）

### 表格组件（JTable）

## 按钮绑定到键盘上（AbstractAction）

## 打印组件

## 发布GUI程序

# 输入、输出流

## File类

## 文件字节输入流（FileInputStream）

## 文件字节输出流（FileOutputStream）

## 文件字符输入、输出流（FileReader、FileWriter）

## 缓冲流（BufferedReader、BufferedWriter）

## 随机流（RandomAccessFile）

## 数组流

### 字节数组流（ByteArrayInputStream、ByteArrayOutputStream）

### 字符数组流（CharArrayReader、CharArrayWriter）

## 数据流（DataInputStream、DataOutputStream）

## 对象流（ObjectInputStream、ObjectOutputStream）

## 序列化与对象克隆

## 使用Scanner解析文件

### 使用默认分隔标记解析文件

### 使用正则表达式作为分隔标记解析文件

## 文件对话框（JFileChooser）

## 带进度条的输入流（ProgressMonitorInputStream）

## 文件锁（RandomAccessFile）

# JDBC与MySQL数据库

## MySQL数据库管理系统

## 启动MySQL数据库服务器

## MySQL客户端管理工具

## JDBC

## 连接数据库

## 查询操作

## 更新、添加与删除工作

## 使用预处理语句

## 通用查询

### 预处理语句的优点

### 使用通配符

## 事务

## 连接SQL Server数据库

## 连接Derby数据库

## 应用举例

### 设计思路

### 具体设计

### 用户程序

# Java多线程机制

# Java网络编程

# 图形、图像与音频

# 泛型与集合框架

## 泛型

### 泛型类声明

### 使用泛型类声明对象

## 链表

### LinkedList<E>泛型类

### 常用方法

### 遍历列表

### 排序与查找

### 洗牌与旋转

## 堆栈

## 散列映射

## 数集

## 树映射

## 自动装箱与拆箱

1. UML图

## 类的UML图

## 类的关联关系的UML图

## 类的依赖关系的UML图

## 继承关系的UML图

## 开-闭原则的UML图

## 接口的UML图

## 

1. Scanner类的使用

## 输入基本型数据

可以用于读入从键盘输入的字符，如：

Scanner reader = new Scanner(System.in);

double x = reader.nextDouble();

还有nextBoolean()，nextByte()，nextShort()，nextInt()，nextLong()，nextFloat()，nextDouble()。

## 

1. 方法重载与方法重写

## 方法重载

一个类中的多个方法**具有相同的名字**，但这些方法的**参数必须不同**：

* 参数的个数不同。
* 参数的个数相同，但参数列表中对应的某个参数的类型不同。

## 方法重写

1. 跟包相关的

5.2.1

5.2.2

1. 访问权限

1、访问权限由高到低的排序是：

public、protected、友好的、private

2、需要特别注意的是：

* 类中的实例方法总是可以操作类中的实例变量和类变量；
* 类方法总是可以操作该类中的类变量。

## 私有变量和私有方法（private）

类B不可以通过类名A来操作和调用类A中的私有类变量和私有类方法。

当类A在类B中创建对象后，如果不希望创建出的对象直接访问类A中的一些成员变量（即通过“.”运算符来操作类A的成员变量），就应当将类A中的这些成员变量的访问权限设置为private。

面向对象编程提倡类要提供操作成员变量的方法。

## 共有变量和共有方法（public）

类B可以通过类名A来操作和调用类A中的共有类变量和共有类方法。

## 友好变量和友好方法（无private、public和protected）

## 受保护的成员变量和方法（protected）

## public类与友好类

## protected的进一步说明



## 11.6 查询操作

### 顺序查询

Connection con = DriverManager.*getConnection*(uri, user, pw); // 建立连接

Statement sql = con.createStatement(); //创建一个SQL语句对象

ResultSet rs = sql.executeQuery(SQL); // 将查询结果存放在一个ResultSet对象中

### 控制游标

// type取值决定滚动方式，concurrency取值决定是否可以用结果更新数据库

Statement sql = con.createStatement(int type, int concurrency);

### 条件与排序查询

## 11.7 更新、添加与删除工作

PreparedStatement sql; // 或者Statement sql;

int ok = sql.executeUpdate(sqlStr)

## 11.8 预处理语句

String sqlStr = "SELECT \* FROM mess WHERE NAME LIKE ? ";

PreparedStatement preSql = con.prepareStatement(sqlStr);

preSql.setString(1, "张%");

ResultSet rs = preSql.executeQuery();

## 11.9通用查询

这一节中的代码，注意rs跳来跳去的方式。

## 11.10事务

con.setAutoCommit(boolean); // 关闭自动提交模式

con.commit(); // 处理事务

con.rollback(); // 回滚事务

|  |  |
| --- | --- |
| 区分 | |
| executeQuery | executeQuery返回的是ResultSet结果集，通常是执行了select操作。 |
| executeUpdate | executeUpdate返回的是int型，表明受影响的行数，通常是执行了insert、update、delete等操作。 |