Java基础(Java SE)

继承

本章目标

- 掌握继承的优点和实现
- 掌握子类重写父类方法
- 掌握继承下构造方法的执行过程
- 掌握抽象类和抽象方法的使用
- 掌握final修饰属性、方法和类

初始化块

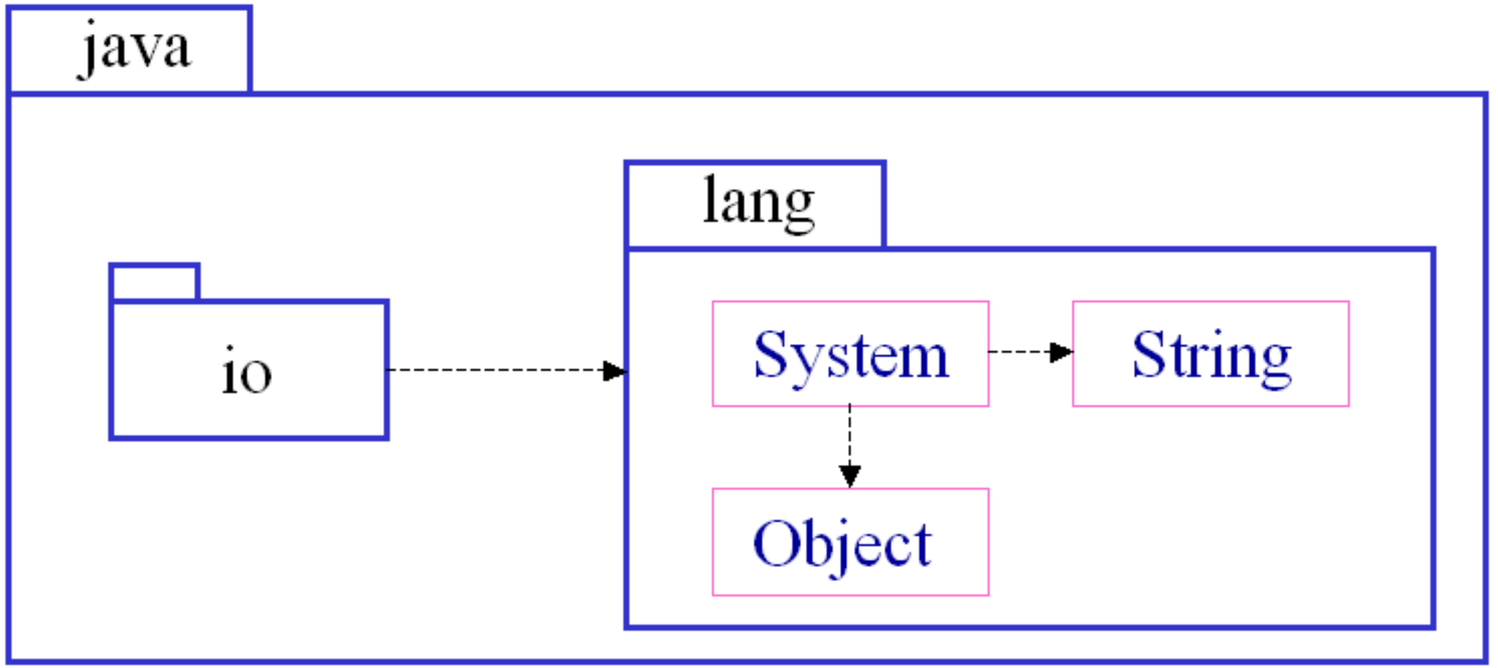
- 前面我们知道两种初始化成员数据的方法
  - 在构造方法中设置值
  - 在声明中设置值
  - 实际上，java还有第三种机制，称为初始化块。在一个类的声明中，可以包含多个代码块。只要构造类的对象时候，这些块就会被执行。

初始化块---静态的初始化块

- 首先运行初始化块，然后再运行构造函数
- 还可以静态的初始化块,将代码放在一个块中，并标记关键字static。
- 静态的初始化块对静态属性进行初始化.
- 在类第一次加载时候，会执行静态初始化块
- 静态初始化块只在某个类第一次装入内存时才执行，这就是为什么只有第一次new 才有“静态初始化块”输出，而后面却不执行输出的原因。
- 初始化块在某个对象生成时（被new）执行。

package

- 帮助管理大型的软件系统。解决类的命名冲突问题，提供类的多重类命名空间。
- 包可以包括类和子包。



Java常用包

- 可以使用import关键字引入其他包中的类
- java.lang 基本语言包 为Java语言的基本结构(如字符串类、数组类)提供了基本的类。
- java.util 实用包 提供了一些诸如基本数据结构，哈西表、向量、堆栈之类的实用例程。
- java.io I/O包　提供了标准的输入/输出及文件读写的类。
- java.sql JDBC包　提供了java与数据库交互的类和接口。
- java.net 网络包 为通过诸如telnet、ftp、www之类的协议访问网络提供了例程。

package语句

- package语句作为Java源文件的第一条语句，指明该文件中定义的类所在的包。(若缺省该语句，则指定为无名包)。它的格式为：

```
package pkg1[.pkg2[.pkg3…]];
```

- 每个源文件只能有一个包的声明；
- Java编译器把包对应于文件系统的目录管理，package语句中，用 '.'来指明包(目录)的层次；
- 全路径使用类，解决类名重复的问题。

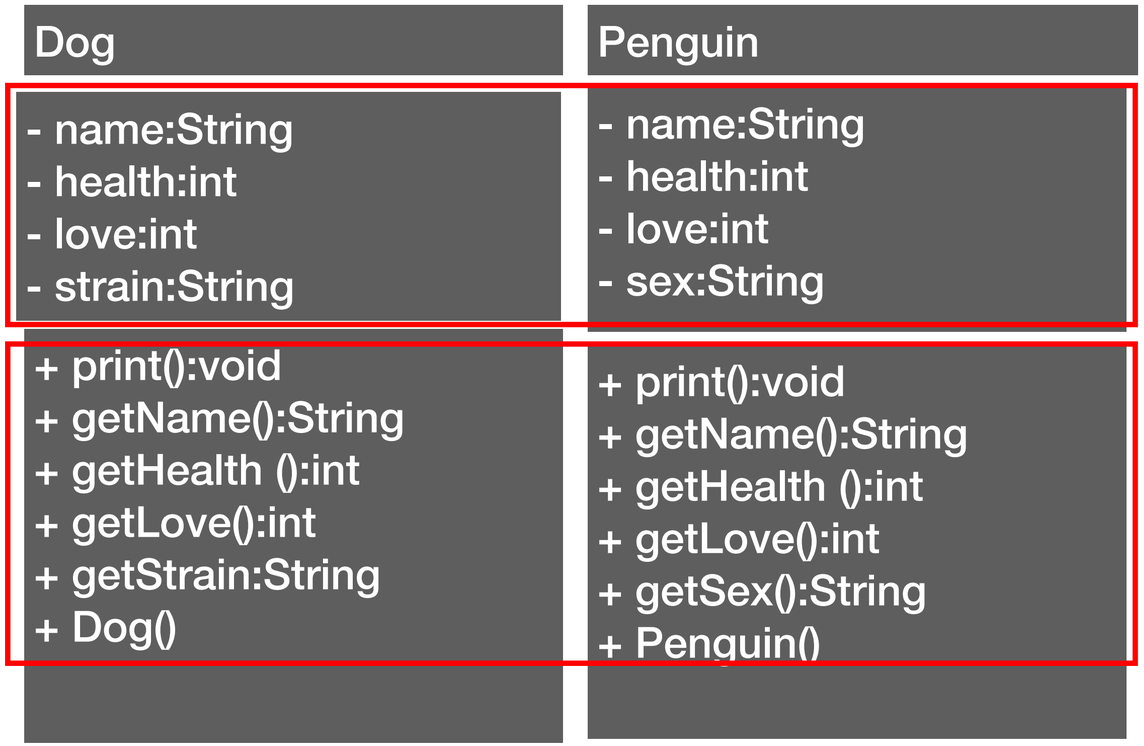
import语句

- 为使用定义在不同包中的Java类，需用import语句来引入所需要的类。告诉编译器所需要的类的路径。
- 语法格式：

```
import package1[.package2…]. (classname |*);
```

为什么使用继承

▷ 这两个类图有什么问题?



将重复代码抽取到父类中

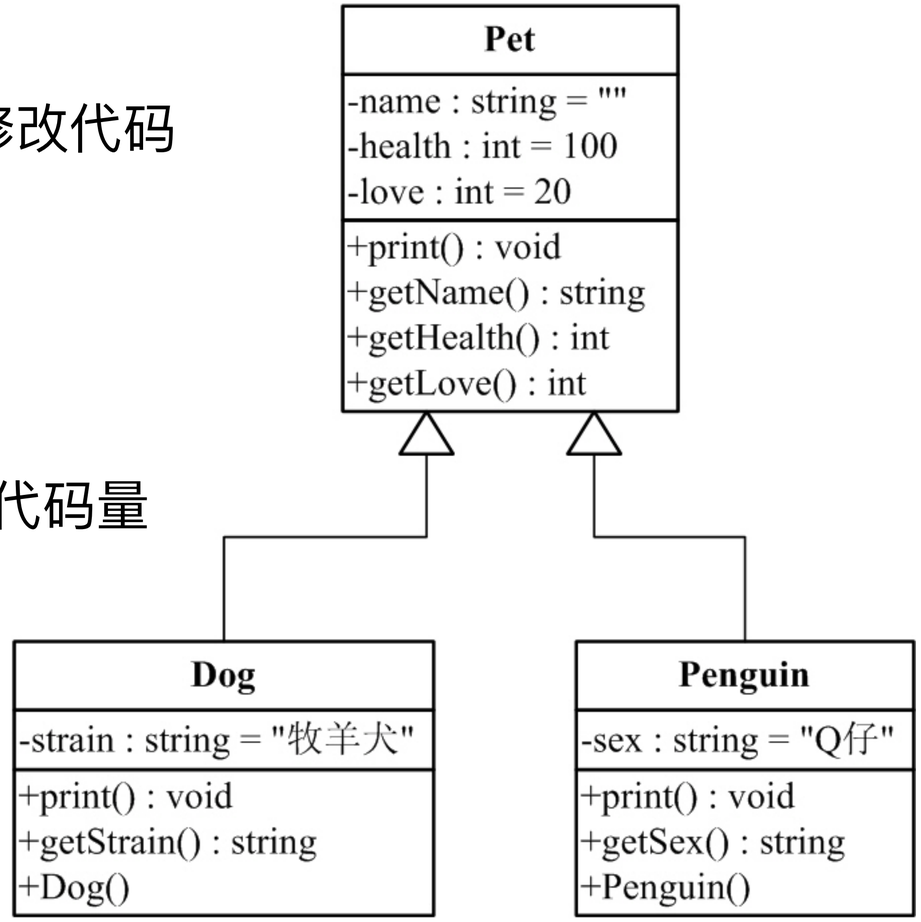
使用继承优化设计

为什么使用继承

▷ 使用继承优化后：

方便修改代码

减少代码量



如何使用继承

- 使用继承
  - 编写父类

```
class Pet {
    //公共的属性和方法
}
```

  - 编写子类，继承父类

```
class Dog extends Pet {
    //子类特有的属性和方法
}

class Penguin extends Pet {
}
```

只能继承一个父类

继承关键字

理解继承

- 有些父类成员不能继承
  - private成员
  - 子类与父类不在同包，使用默认访问权限的成员
  - 构造方法

理解继承---super

- 子类访问父类成员

使用super关键字

super代表父类对象

- 访问父类属性 super.name;

- 访问父类方法 super.print();

继承中的构造方法

- 如何在子类中调用父类的构造方法?

可以被默认添加

super();

super(参数表);

只能是构造方法的第一条语句

理解继承

- 继承后的初始化顺序

父类属性

父类构造方法

子类属性

子类构造方法

理解继承

- 在子类中可以通过super关键字来访问父类的成员。
- super必须是出现在子类中（子类的方法和构造方法中），而不是其他位置。
- super可以访问父类的成员，例如父类的属性、方法、构造方法。
- 注意访问权限的限制，例如无法通过super访问private成员。

理解继承

- 访问修饰符protected
  - 可以修饰属性和方法
  - 本类、同包、子类可以访问
- 访问修饰符总结

访问修饰符	本类	同包	子类	其他
private	√			
默认(default)	√	√		
protected	√	√	√	
public	√	√	√	√

方法重写

子类重写父类方法

- 方法重写必须满足如下要求：
- 重写方法和被重写方法必须具有相同的方法名；
- 重写方法和被重写方法必须具有相同的参数列表；
- 重写方法的返回值类型必须和被重写方法的返回值类型相同或者是其子类；
- 重写方法的不能缩小被重写方法的访问权限。

抽象类

- Java中也使用抽象类，限制实例化

```
public abstract class Pet {
}
```

抽象方法

- abstract也可用于方法——抽象方法
  - 抽象方法没有方法体　　public abstract void print();
  - 抽象方法必须在抽象类里
  - 抽象方法必须在子类中被实现，除非子类是抽象类

final用法

- 类不希望再被其他类继承?
  - 使用final类

最终版的类

- 方法不希望被重写?
  - 使用final方法

最终版的方法

- 属性值不希望被修改?
  - 使用final属性

最终版的属性值

THE END