Java 核心技术

第五章 继承、接口和抽象类

第一节 继承

华东师范大学 陈良育

继承(1)

- 面向对象编程语言和面向过程的编程语言最突出的特点就是变量类型的继承
- 更符合大自然规律：父亲有的，儿子就有
- 首先看个例子

继承(2)

```
class Father
{
    public void f1()
    {
        System.out.println("hi");
    }
}

public class Son extends Father
{
    public static void main(String[] a)
    {
        Son s = new Son();
        s.f1();   //Son没有定义f1,而是通过父类继承的
    }
}
```

继承(3)

- 面向过程编程语言没有继承，导致出现很多类型重复定义
- 物以类聚，世间万物皆对象，对象也可以分成若干类别。
- 类别内的对象属性和方法都具有一定的共同点。
- 将共同点提取出来，即形成了父类/基类/超类
  - Parent class/Base class/Super class
- 而其他类则自动成为子类/派生类
  - Child class/Derived class

继承(4)


```java
public class Man {
    int height;
    int weight;
    public void eat(){};
    public void plough(){}; //耕田

}
```

```java
public class Woman {
    int height;
    int weight;
    public void eat(){};
    public void weave(){}; //织布
}
```

```java
public class Human {
    int height;
    int weight;
    public void eat(){};
}
```

```java
public class Man extends Human{
    public void plough(){}; //耕田

}
```

```java
public class Woman extends Human {
    public void weave(){}; //织布
}
```

继承(5)

- Man extends Human 表示Man继承于Human
- Human是父类，Man是子类
- 子类继承父类所有的属性和方法（但不能直接访问private成员）
- 根据信息隐藏原则：子类会继承父类所有的方法。可以直接使用。
- 子类也会继承父类的父类的所有的属性和方法（但不能直接访问private成员）

继承(7)

- 单根继承原则：每个类都只能继承一个类
- 如果不写extends，Java类都默认继承java.lang.Object类
- class Human extends java.lang.Object
- Java所有类从java.lang.Object开始，构建出一个类型继承树
- Object类里面默认就有clone, equals, finalize, getClass, hashCode, toString等方法

继承(8)

- 每个Java类都必须有构造函数。
- 如果没有显式定义构造函数，Java编译器自动为该类产生一个空的无形参构造函数。如果已经有了显式的有参构造函数，编译器就不会越俎代庖了。
- 每个子类的构造函数的第一句话，都默认调用父类的无参数构造函数super()，除非子类的构造函数第一句话是super，而且super语句必须放在第一条，不会出现连续两条super语句。
- 查看A.java和B.java 程序

继承(8)

- 总结
  - 子类继承父类们所有的东西(但不能直接访问private成员)
  - Java所有类都继承自java.lang.Object类
  - Java所有的类都是单根继承的
  - 子类构造函数默认第一句话都会去调用父类的构造函数

代码(1) Father.java/Son.java


```java
public class Father
{
    public void f1()
    {
        System.out.println("hi");
    }
}
```

```java
public class Son extends Father
{
    public static void main(String[] a)
    {
        Son s = new Son();
        s.f1();  //Son没有定义f1,而是通过父类继承的
    }
}
```

代码(2)Human/Woman/Man.java


```java
public class Human {
    int height;
    int weight;
    public void eat()
    {
        System.out.println("I can eat!");
    }
}
```

```java
public class Woman extends Human {
    public void weave()
    {
        System.out.println("I can weave");
    }//织布
}
```

```java
public class Man extends Human
{
    //overwrite重写父类方法
    public void eat(){
        System.out.println("I can eat more");
    }
    //扩展，比父类拥有更多的方法
    public void plough(){
        System.out.println("I can plough");
    } //耕田

    public static void main(String[] a) {
        Man obj1 = new Man();
        obj1.eat();
    }
}
```

代码(3)Base.java/Derived.java


```java
public class Base {
    private int num = 10;

    public int getNum()
    {
        return this.num;
    }
}
```

```java
public class Derived extends Base{
    private int num = 20;

    public int getNum() {
        return this.num;
    }

    public static void main(String[] args) {
        Derived foo = new Derived();
        System.out.println(foo.getNum());
    }
}
```

代码(4)A.java/B.java

```java
public class A {
    public A()
    {
        System.out.println("111111");
    }
    public A(int a)
    {
        System.out.println("222222");
    }
}
```

```java
public class B extends A{
    public B()
    {
        //super(); 编译器自动增加super()
        System.out.println("333333");
    }
    public B(int a)
    {
        super(a);  //编译器不会自动增加super();
        System.out.println("444444");
    }
    public static void main(String[] a)
    {
        B obj1 = new B();
        System.out.println("===============");
        B obj2 = new B(10);
    }

}
```

谢 谢！