Java

面向对象程序设计

软件学院 贾伟峰

复习

第三次课的内容

子类extends父类

改写、新增

super

Object

继承

无法继承的类

无法改写的方法

赋值1次的常量

final

抽象类、接口

abstract

interface

implements

多态

子类“当作”父类使用

运行子类方法

匿名内部类

复习多态：“简单工厂模式”

```java
public interface TV{
    public void play();
}
```

TV. java

```java
public class HaierTV implements TV{
    public void play(){
        System.out.println("This is Haier TV.");
    }
}
```

HaierTV. java

```java
public class HisenseTV implements TV{
    public void play(){
        System.out.println("This is HisenseTV");
    }
}
```

HisenseTV. java

```java
public class TVFactory{
    public static Tv produceTv(String brand){
        if(brand.equals("HaierTv"))
            return new HaierTv();

        if(brand.equals("HisenseTv"))
            return new HisenseTv();

        return null;
    }
}
```

TVFactory. java

```xml
<?xml version="1.0"?>
<config>
    <brandName>HisenseTv</brandName>
</config>
```

config. xml

```java
import javax.xml.parsers.*;
import org.w3c.dom.*;
import org.xml.sax.SAXException;
import java.io.*;
public class XMLUtilTV{
    public static String getBrandName(){
        try{
            DocumentBuilderFactory dFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = dFactory.newDocumentBuilder();
            Document doc = builder.parse(new File("configTV.xml"));

            NodeList nl = doc.getElementsByTagName("brandName");
            Node classNode = nl.item(0).getFirstChild();
            String brandName = classNode.getNodeValue().trim();

            return brandName;
        } catch(Exception e){
            System.out.println(e.getMessage());
            return null;
        }

    }
}
```

XMLUtilTV. java

```java
public class Main{
    public static void main(String args[]){
        Tv tv;
        String brandName = XMLUtilTV.getBrandName();
        tv = TVFactory.produceTv(brandName);
        tv.play();
    }
}
```

Main. java

复习多态：“工厂方法模式”

```java
public interface TV{
    public void play();
}
```

TV. java

```java
public class HaierTV implements TV{
    public void play(){
        System.out.println("This is Haier TV.");
    }
}
```

HaierTV. java

```java
public class HisenseTV implements TV{
    public void play(){
        System.out.println("This is HisenseTV");
    }
}
```

HisenseTV. java

```java
public interface TVFactory{
    public TV produceTV();
}
```

TVFactory. java

```java
public class HaierTVFactory implements TVFactory{
    public TV produceTV(){
        return new HaierTV();
    }
}
```

HaierTVFactory. java

```java
public class HisenseTVFactory implements TVFactory{
    public TV produceTV(){
        return new HisenseTV();
    }
}
```

HisenseTVFactory. java

```xml
<?xml version="1.0"?>
<config>
    <className>HaierTVFactory</className>
</config>
```

XMLUtil. java

```java
import javax.xml.parsers.*;
import org.w3c.dom.*;
import org.xml.sax.SAXException;
import java.io.*;
public class XMLUtil{
    public static Object getBean(){
        try{
            DocumentBuilderFactory dFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = dFactory.newDocumentBuilder();
            Document doc = builder.parse(new File("config.xml"));

            NodeList nl = doc.getElementsByTagName("className");
            Node classNode = nl.item(0).getFirstChild();
            String cName = classNode.getNodeValue();

            Class c = Class.forName(cName);
            Object obj = c.newInstance();
            return obj;
        }
        catch(Exception e){
            e.printStackTrace();
            return null;
        }
    }
}
```

```java
public class Main{
    public static void main(String args[]){
        TV tv;
        TVFactory factory;
        factory = (TVFactory)XMLUtil.getBean();
        tv = factory.produceTV();
        tv.play();
    }
}
```

Main. java

目的——扩展功能时，尽量不改现有代码。

实现开闭原则：对修改关闭，对扩展开放。
Open for extension, but closed for modification.

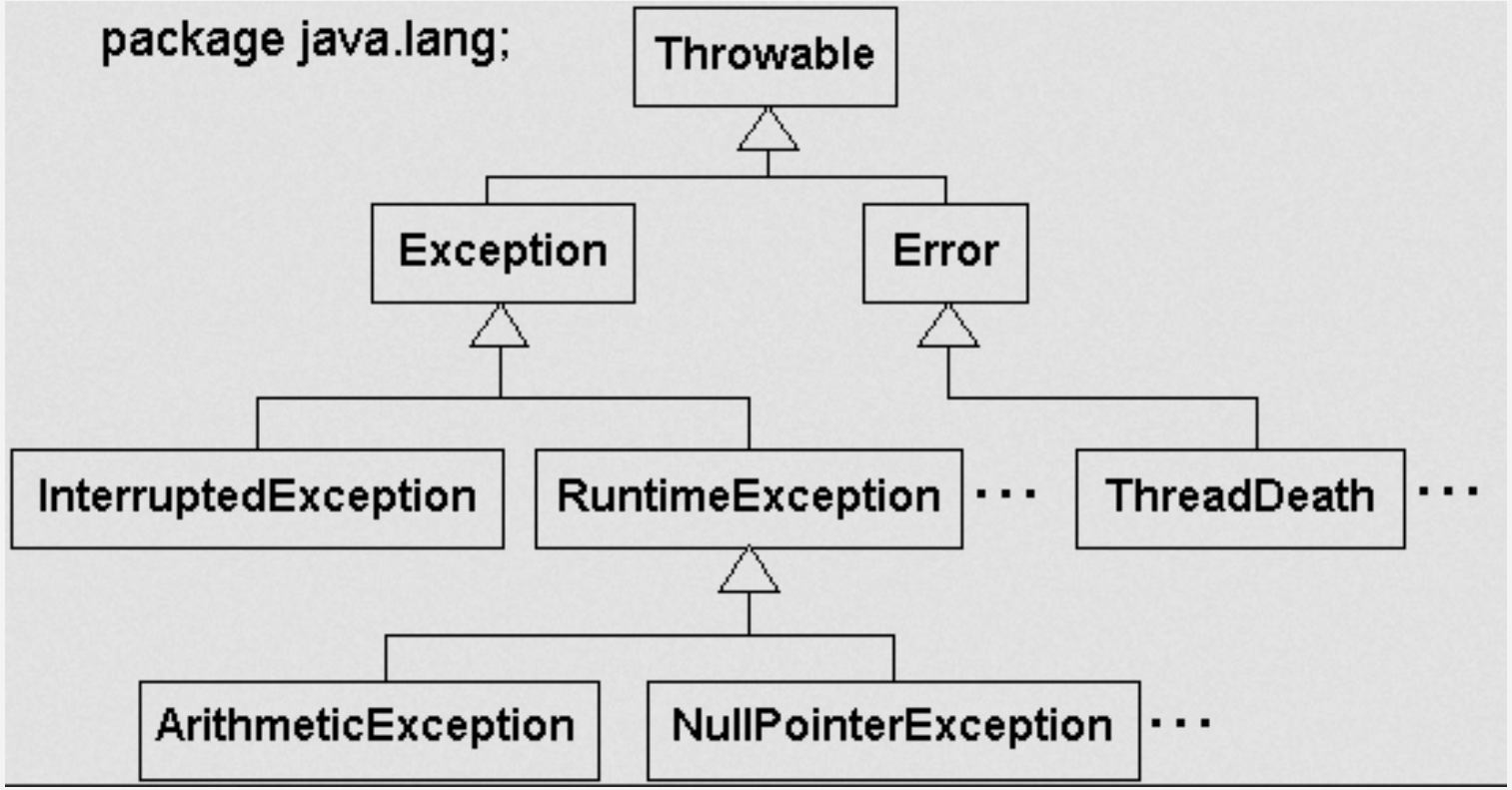
编程有时候会“出错”，比如下面的程序。

```java
1  public class Example20{
2      public static void main(String[] args){
3          int result = divide(4, 0);
4          System.out.println(result);
5      }
6      public static int divide(int x, int y){
7          int result = x/y;
8          return result;
9      }
10 }
```

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at Example20.divide(Example20.java:7)
        at Example20.main(Example20.java:3)
```

有“异常”：Exception

Java中的异常(Exception)、错误(Error)



能否捕捉程序中的异常？如图所示。

```java
public class Example20{
    public static void main(String[] args){
        try{
            int result = divide(4, 0);
            System.out.println(result);
        }catch (Exception e){
            System.out.println("捕捉的异常信息为: "+e.getMessage());
        }
    }
    public static int divide(int x, int y){
        int result = x/y;
        return result;
    }
}
```

```java
public class Example20{
    public static void main(String[] args){
        try{
            int result = divide(4, 0);
            System.out.println(result);
        }catch (Exception e){
            System.out.println("捕捉的异常信息为: "+e.getMessage());
            return;
        }finally{
            System.out.println("进入finally代码块");
        }
        //此处代码不会执行
        System.out.println("程序继续向下执行……");
    }
    public static int divide(int x, int y){
        int result = x/y;
        return result;
    }
}
```

程序员就是不写异常捕捉（处理）代码怎么办？

throws

throws关键词

```
1  public class Example20{
2      public static void main(String[] args){
3          try{
4              int result = divide(4, 0);
5              System.out.println(result);
6          }catch (Exception e){
7              System.out.println("捕捉的异常信息为: "+e.getMessage());
8              return;
9          }finally{
10             System.out.println("进入finally代码块");
11         }
12         //此处代码不会执行
13         System.out.println("程序继续向下执行……");
14     }
15     public static int divide(int x, int y) throws Exception{
16         int result = x/y;
17         return result;
18     }
19 }
```

或者，继续throws，就像p146那样

Java中的异常

编译时异常

01

02

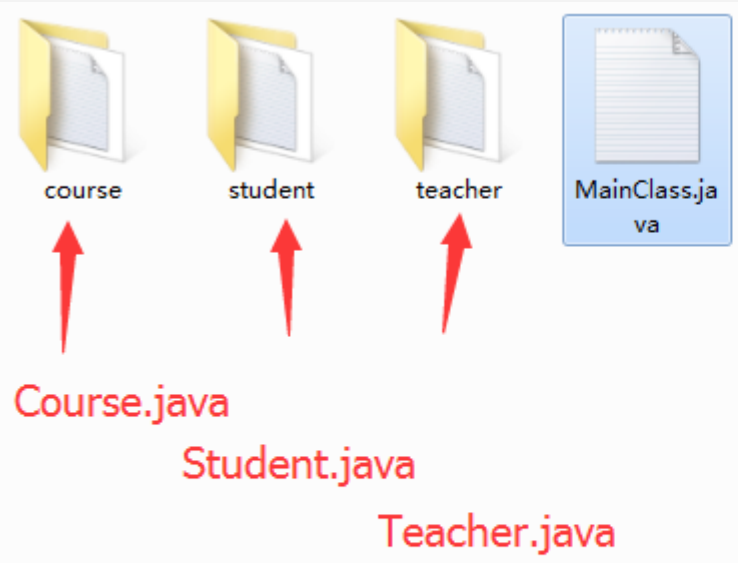
运行时异常

还可以自定义异常！P147-149

包

太多文件，如何更合适地管理？用package

java文件所在目录src



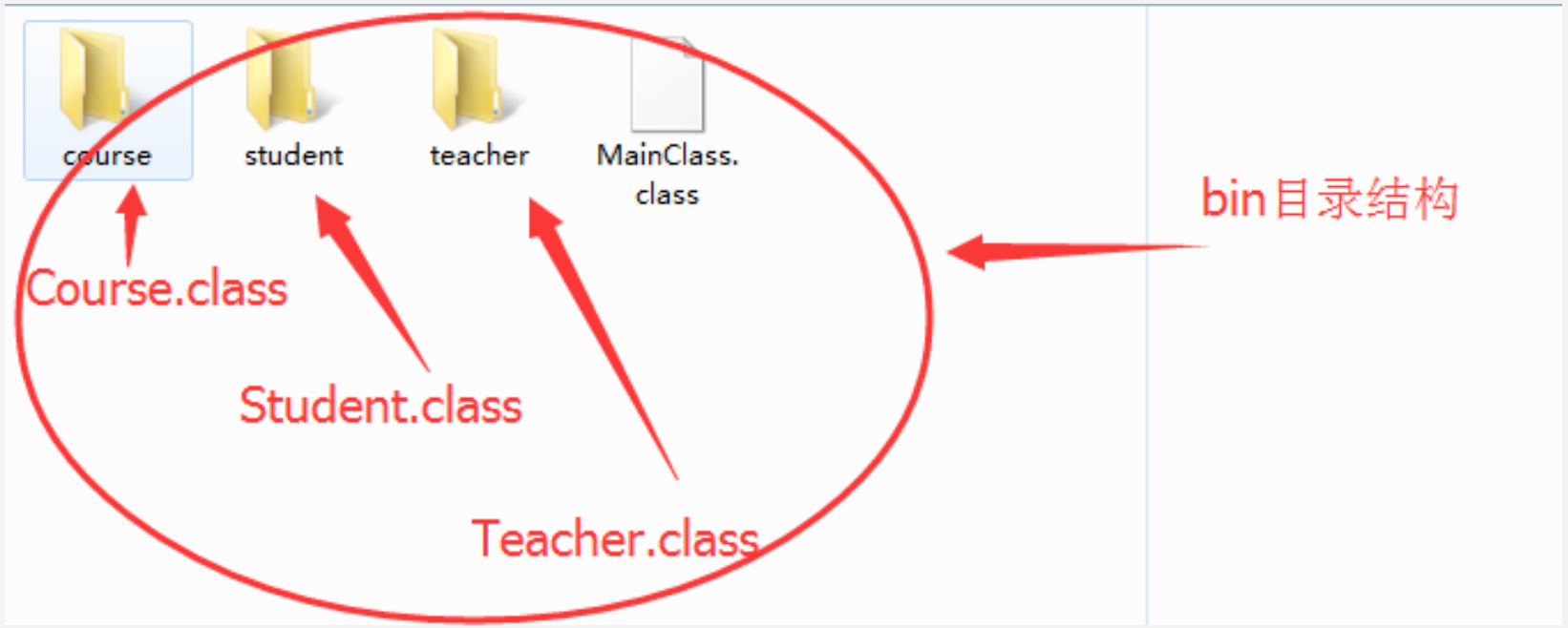

```java
import course.*;
import student.*;
import teacher.*;

public class MainClass
{
    public static void main(String[] args){
        Course cs = new Course();
        Student st = new Student();
        Teacher te = new Teacher();
    }
}
```

```java
package teacher;
public class Teacher
{
    public Teacher(){
        System.out.println("This is a teacher.");
    }
}
```

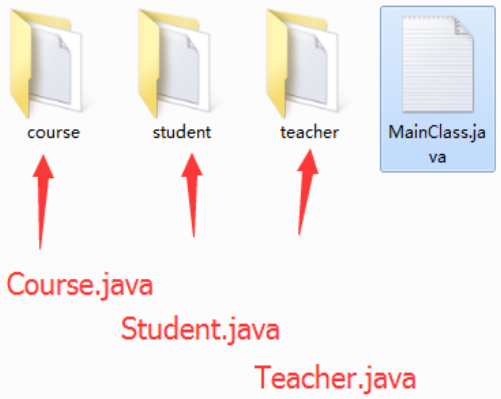
```java
package course;
public class Course
{
    public Course(){
        System.out.println("This is a course.");
    }
}
```

```java
package student;
public class Student
{
    public Student()
    {
        System.out.println("This is a student.");
    }
}
```

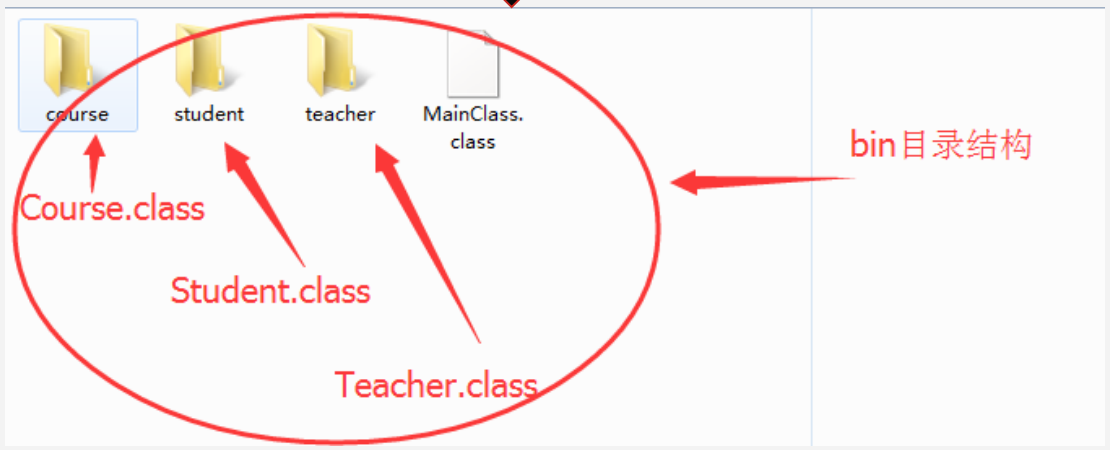
class文件所在目录bin



如何从src的java文件，变成bin中的class文件?




```
import course.*;
import student.*;
import teacher.*;

public class MainClass
{
    public static void main(String[] args){
        Course cs = new Course();
        Student st = new Student();
        Teacher te = new Teacher();
    }
}
```




```
pkgdaemon\src>javac -d ../bin course/Course.java
```

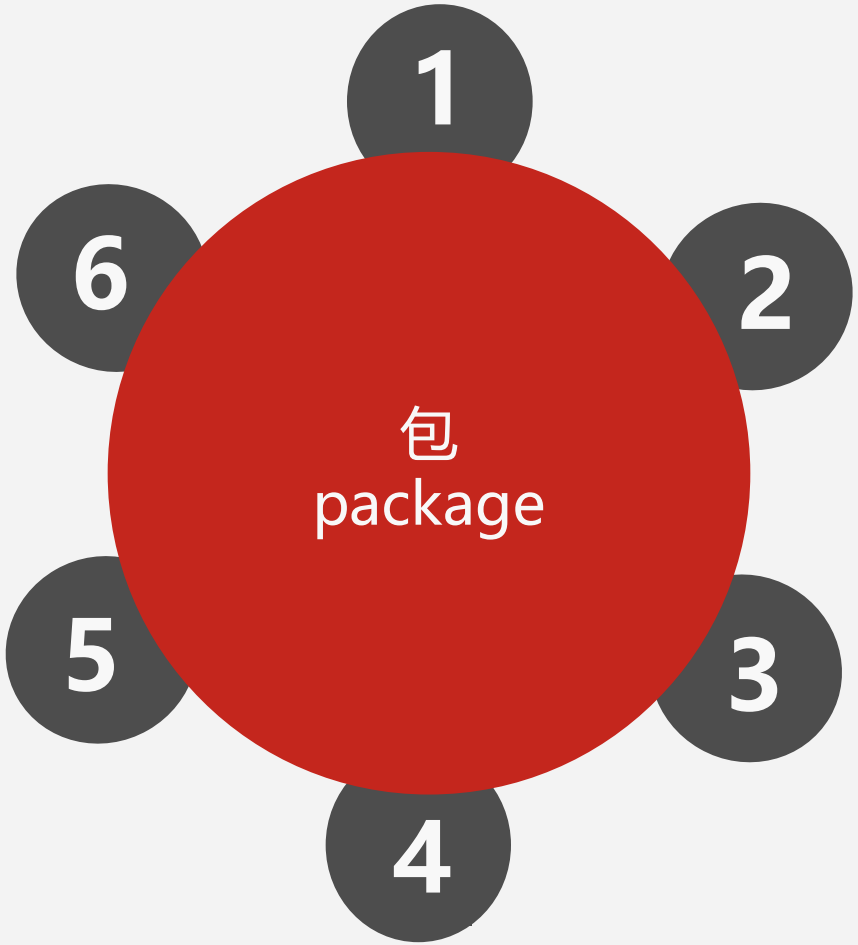
```
pkgdaemon\src>javac -d ../bin student/Student.java
```

```
pkgdaemon\src>javac -d ../bin teacher/Teacher.java
```

```
pkgdaemon\src>javac -d ../bin MainClass.java
```

```
pkgdaemon\bin>java MainClass
```

java中自带的包



java.lang

java.util

java.net

java.io

java.awt

java.sql

这么多的class，如何打包成1个供使用?

jar

```
\pkgdaemon\bin>jar -cvf MainClass.jar .
```

在有Main方法的那个类所在的文件夹中执行。

改MANIFEST.MF，加入口类（P156）。

```
C:\>java -jar MainClass.jar
```

运行打包后的文件

- private：类内访问
- default：包内访问
- protect：包内访问+继承（子类）访问
- public：不受限

总结


第四次课的内容
try{}catch{}
finally
throws
捕捉异常
编译时
运行时
自定义异常
异常分类
包
package
import
用途
jar
访问控制
private
default
protected
public