

反射

Java Platform Standard Edition

郑春光

课程目标

CONTENTS

ITEMS **1** 什么是类对象

ITEMS **2** 获取类对象的方法

ITEMS **3** 常见操作

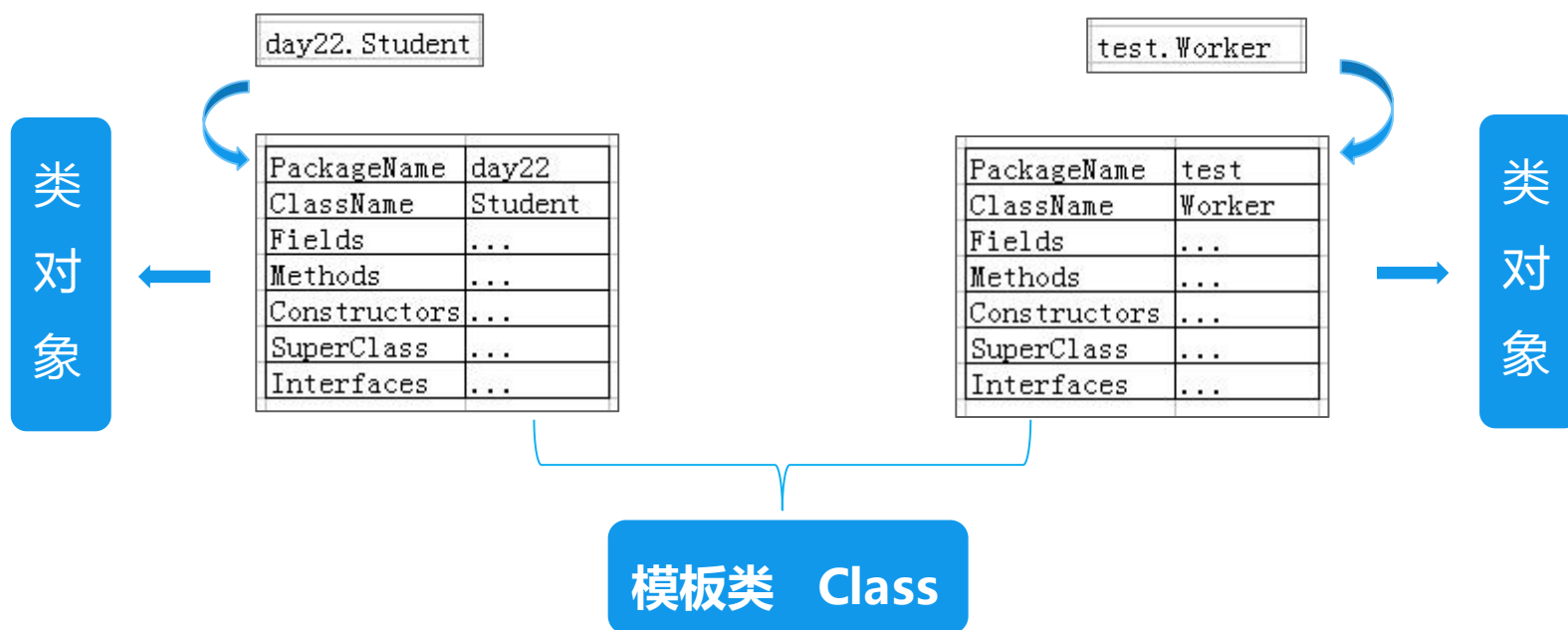
ITEMS **4** 设计模式介绍

ITEMS **5** 单例设计模式

ITEMS **6** 工厂设计模式

什么是类对象

- 类的对象：基于某个类 new 出来的对象，也称为实例对象。
- 类对象：类加载的产物，封装了一个类的所有信息（类名、父类、接口、属性、方法、构造方法）



- 通过类的对象，获取类对象
 - `Student s = new Student();`
 - `Class c = s.getClass();`
- 通过类名获取类对象
 - `Class c = 类名.class;`
- 通过静态方法获取类对象
 - `Class c=Class.forName(“包名.类名”);`

- `public String getName()`
- `public Package getPackage()`
- `public Class<? super T> getSuperclass()`
- `public Class<?>[] getInterfaces()`
- `public Field[] getFields()`
- `public Method[] getMethods()`
- `public Constructor<?>[] getConstructors()`
- `public T newInstance()`

- 开发中有一个非常重要的原则“开闭原则”，对拓展开放、对修改关闭。
- 工厂模式主要负责对象创建的问题。
- 可通过反射进行工厂模式的设计，完成动态的对象创建。

- 单例 (Singleton) : 只允许创建一个该类的对象。
- 方式1: 饿汉式 (类加载时创建, 天生线程安全)

```
class Singleton {  
    private static final Singleton instance = new Singleton();  
    private Singleton(){}  
    public static Singleton getInstance(){  
        return instance;  
    }  
}
```

- 方式2：懒汉式（使用时创建，线程不安全，加同步）

```
class Singleton{  
    private static Singleton instance = null;  
  
    private Singleton(){}  
  
    public static synchronized Singleton getInstance(){  
        if(instance == null){  
            instance = new Singleton();  
        }  
        return instance;  
    }  
}
```


- 方式3：懒汉式（使用时创建，线程安全）

```
class Singleton {  
    private Singleton() {}  
  
    private static class Holder {  
        static Singleton s = new Singleton();  
    }  
  
    public static Singleton instance() {  
        return Holder.s;  
    }  
}
```

- 类对象：
 - Class对象，封装了一个类的所有信息；程序运行中，可通过Class对象获取类的信息。
- 获取类对象的三种方式：
 - `Class c = 对象.getClass();`
 - `Class c = 类名.class;`
 - `Class c=Class.forName(“包名.类名”);`
- 工厂模式：
 - 主要用于创建对象，通过反射进行工厂模式的设计，完成动态的对象创建。
- 单例模式：
 - Singleton，只允许创建一个该类的对象。