1、课程名称：泛型操作范例


JAVA培训先锋

JAVA 应用开发详解

第10章：泛型
——泛型操作范例

MLDN 软件教学研发部

2、实例要求


本章目标

- 加深泛型的理解
- 掌握标识接口的定义

E-MAIL：mldnqa@163.com www.MLDNJAVA.cn


题目要求

- 用户在设计类的时候往往会使用类的关联关系，例如，一个人中可以定义一个信息的属性，但是一个人可能有各种各样的信息（**例如：**联系方式、基本信息等），所以此信息属性的类型就可以通过泛型进行声明，之后只要设计相应的信息类即可。

E-MAIL：mldnqa@163.com www.MLDNJAVA.cn

3、本实例主要采用的知识

关联、泛型、接口

4、具体内容

如果现在假设要表示一个人的信息：

- 基本信息
- 联系方式

在此时，肯定要设计出一个接口，因为只有实现了此接口的类才应该可以表示出人的信息。

```
interface Info{		// 只有此接口的子类才是表示人的信息
```

```
}
```

此接口定义完成，但是在此接口中没有任何的方法，所以此时，这样的接口称为标识接口。之后定义人的类，人的类中只要是此接口的子类都可以表示人的信息。

```
interface Info{		// 只有此接口的子类才是表示人的信息
}
class Contact implements Info{	// 表示联系方式
	private String address ;// 联系地址
	private String telphone ;	// 联系方式
	private String zipcode ;	// 邮政编码
	public Contact(String address,String telphone,String zipcode){
		this.setAddress(address) ;
		this.setTelphone(telphone) ;
		this.setZipcode(zipcode) ;
	}
	public void setAddress(String address){
		this.address = address ;
	}
	public void setTelphone(String telphone){
		this.telphone = telphone ;
	}
	public void setZipcode(String zipcode){
		this.zipcode = zipcode ;
	}
	public String getAddress(){
		return this.address ;
	}
	public String getTelphone(){
		return this.telphone ;
	}
	public String getZipcode(){
		return this.zipcode ;
	}
	public String toString(){
		return "联系方式：" + "\n" +
				"\t|- 联系电话：" + this.telphone + "\n" + 
				"\t|- 联系地址：" + this.address + "\n" + 
				"\t|- 邮政编码：" + this.zipcode ;
	}
```

```
};
class Introduction implements Info{
	private String name ;		// 姓名
	private String sex ;		// 性别
	private int age ;		// 年龄
	public Introduction(String name,String sex,int age){
		this.setName(name) ;
		this.setSex(sex) ;
		this.setAge(age) ;
	}
	public void setName(String name){
		this.name = name ;
	}
	public void setSex(String sex){
		this.sex = sex ;
	}
	public void setAge(int age){
		this.age = age ;
	}
	public String getName(){
		return this.name ;
	}
	public String getSex(){
		return this.sex ;
	}
	public int getAge(){
		return this.age ;
	}
	public String toString(){
		return "基本信息：" + "\n" +
				"\t|- 姓名：" + this.name + "\n" + 
				"\t|- 性别：" + this.sex + "\n" + 
				"\t|- 年龄：" + this.age ;
	}
};
class Person<T extends Info>{
	private T info ;
	public Person(T info){		// 通过构造方法设置信息属性内容
```

```
		this.setInfo(info);
	}
	public void setInfo(T info){
		this.info = info ;
	}
	public T getInfo(){
		return this.info ;
	}
	public String toString(){	// 覆写 Object 类中的 toString()方法
		return this.info.toString() ;
	}
};
```

先将联系方式作为基本信息。

```
public class GenericsDemo32{
	public static void main(String args[]){
		Person<Contact> per = null ;		// 声明 Person 对象
		per = new Person<Contact>(new Contact("北京市","01051283346","100088")) ;
		System.out.println(per) ;
	}
};
```

将基本信息作为信息

```
public class GenericsDemo33{
	public static void main(String args[]){
		Person<Introduction> per = null ;		// 声明 Person 对象
		per = new Person<Introduction>(new Introduction("李兴华","男",30)) ;
		System.out.println(per) ;
	}
};
```

5、总结

使用泛型操作之后，从整个操作的代码上更加合理一些。在日后讲解类集的部分要大量的使用泛型，本章中要把泛型的基本概念全部掌握清楚。


下一章内容

Java 常用类库

E-MAIL：mldnqa@163.com www.MLDNJAVA.cn