# 引用数据类型操作

说明

一个引用数据类型：在一个执行代码过程中引入另一个方法把引用数据类型作为参数传值，引用数据类型在此方法中的成员变量发成改变，不用进行返回值也会改变的

public static void main(String[] args){

User u= new User(); //创建对象user

u.setAge(5); //为age赋值5

update(u); //进入方法update为Age重新赋值：注意这里不用返回值了(以前老笨还会弄出返回值)

System.out.println(u.getAge());//打印10

}

private static void update(User u){

u.setAge(10); //为引用类型User中的age重新赋值10

}

例子：IO转File

例子：已知有一个文件流(FileInputStream),但是方法里需要的是一个File类型的转换操作：

第一步 创建一个File f = new File(”这里path随便但要指定一个文件(不存在也可以)不是目录”)

File f= new File("d:e/eee"); //主目录路径

f.mkdirs(); //系统自带的递归创建目录方法

File f2=new File(f,"e.txt"); //指定的file文件

f2.createNewFile();

System.out.println(f2.length()); //此时长度为0

FileInputStream fis = new FileInputStream("d:/a.txt");

第二步 创建一个方法用于把此流输出到此文件中：参数：已知的FileInputStream和 file

createFile(fis, f2); //不用返回值，此时指定的文件不是空文件了

System.out.println(f2.length()); //有数据的长度了

public void createFile(FileInputStream input,File file){

//下面是把流写入到文件中操作

try {

FileOutputStream fos= new FileOutputStream(file); //在给出的路径文件没有直接创建文件(但不会创建目录)

byte[] b= new byte[1024];

int i=0;

while((i=fis.read(b)) !=-1) {

fos.write(b, 0, i);

}

fis.close();

fos.close();

} catch (Exception e) {

e.printStackTrace();

}

}