传智播客 www.itcast.cn —高级软件人才实作培训专家！

Java 预热班讲义

讲师：毕向东

IO(Input Output)流

- IO流用来处理设备之间的数据传输
- Java对数据的操作是通过流的方式
- Java用于操作流的对象都在IO包中
- 流按操作数据分为两种：字节流与字符流。
- 流按流向分为：输入流，输出流。

IO流常用基类

- 字节流的抽象基类：
  - InputStream ，OutputStream。
- 字符流的抽象基类：
  - Reader ， Writer。
- 注：由这四个类派生出来的子类名称都是以其父类名作为子类名的后缀。
  - 如：InputStream的子类FileInputStream。
  - 如：Reader的子类FileReader。

IO程序的书写

- 导入IO包中的类
- 进行IO异常处理
- 在finally中对流进行关闭

思考：

- 有了垃圾回收机制为什么还要调用close方法进行关闭。
- 为什么IO异常一定要处理。

例程

字符流——创建文件

- 创建流对象，建立数据存放文件
  - FileWriter fw = new FileWriter(“Test.txt”);
- 调用流对象的写入方法，将数据写入流
  - fw.write(“text”);
- 关闭流资源，并将流中的数据清空到文件中。
  - fw.close();

不写close方法会有什么结果呢？

如果想在原有文件上继续加入新的数据呢？

完整代码

```
FileWriter fw = null;
try{
        fw = new FileWriter("Test.txt");
        fw.write("text");
}
catch (IOException e){
        System.out.println(e.toString());
}
finally{
        If(fw!=null)

        try{

                fw.close();
        }
        catch (IOException e){
                System.out.println(e.toString());

        }
}
```

字符流——读取文件

- 建立一个流对象，将已存在的一个文件加载进流。
  - FileReader fr = new FileReader(“Test.txt”);
- 创建一个临时存放数据的数组。
  - char[] ch = new char[1024];
- 调用流对象的读取方法将流中的数据读入到数组中。
  - fr.read(ch);

思考：

- 在加载文件时候是否是将文件全部加载进流
- 为什么定义数组，要定义多大呢？

完整代码

```
FileReader fr = null;
    try{
            fr = new FileReader("c:\\test.txt");
            char[] buf = new char[1024];
            int len= 0;
            while((len=fr.read(buf))!=-1){
                    System.out.println(new String(buf,0,len));
            }
    }
    catch (IOException e){
            System.out.println("read-Exception :"+e.toString());
    }
    finally{
            if(fr!=null){
                    try{
                            fr.close();
                    }
                    catch (IOException e){
                            System.out.println("close-Exception :"+e.toString());
                    }
```

注意：

- 定义文件路径时，可以用“/”或者“\\”。
- 在创建一个文件时，如果目录下有同名文件将被覆盖。
- 在读取文件时，必须保证该文件已存在，否则出异常。

练习：Copy一个文本文件。

字符流的缓冲区

- 缓冲区的出现提高了对数据的读写效率。
- 对应类
  - BufferedWriter
  - BufferedReader
- 缓冲区要结合流才可以使用。
- 在流的基础上对流的功能进行了增强。

装饰设计模式

- 对原有类进行了功能的改变，增强。
- 装饰模式的基本格式。
- 它与继承有什么不同？
- 了解BufferedReader的原理。

练习：

- 模拟一个BufferedReader类。
- 模拟一个LineNumberReader类。

字节流

- 基本操作与字符流类相同
- 但它不仅可以操作字符，还可以操作其他媒体文件
- 例程
  - Copy一个Jpg文件。

字节流的缓冲区

- 同样是提高了字节流的读写效率。

练习：

- 通过几种方式对MP3的进行拷贝，比较它们的效率。
- 模拟一个BufferedInputStream

转换流

- InputStreamReader,OutputStreamWriter
- 转换流的由来
  - 字符流与字节流之间的桥梁
  - 方便了字符流与字节流之间的操作
- 转换流的应用
  - 字节流中的数据都是字符时，转成字符流操作更高效。
- 例程：标准输入输出。

标准输入输出流

- System类中的字段：in，out。
- 它们各代表了系统标准的输入和输出设备。
- 默认输入设备是键盘，输出设备是显示器。
- System.in的类型是InputStream.
- System.out的类型是PrintStream是OutputStream的子类FilterOutputStream 的子类.

标准输入输出流示例

- 例:获取键盘录入数据，然后将数据流向显示器，那么显示器就是目的地。
- 通过System类的setIn，setOut方法对默认设备进行改变。
  - System.setIn(new FileInputStream(“1.txt”));//将源改成文件1.txt。
  - System.setOut(new FileOutputStream(“2.txt”));//将目的改成文件2.txt
- 因为是字节流处理的是文本数据，可以转换成字符流，操作更方便。
- BfferedReader bufr =

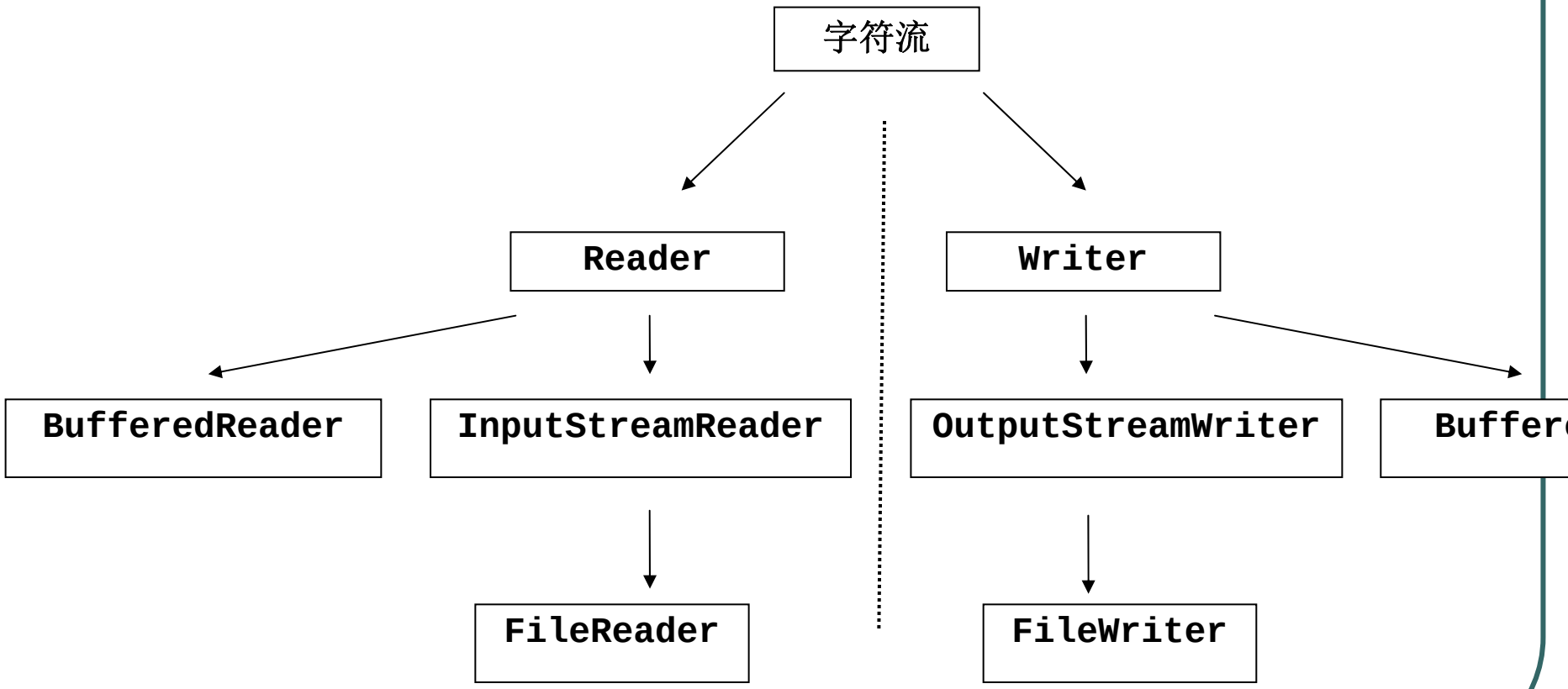
  new BufferedReader(new InputStreamReader(System.in));
- BufferedWriter bufw =

  new BufferedWriter(new OutputStreamWriter(System.out));

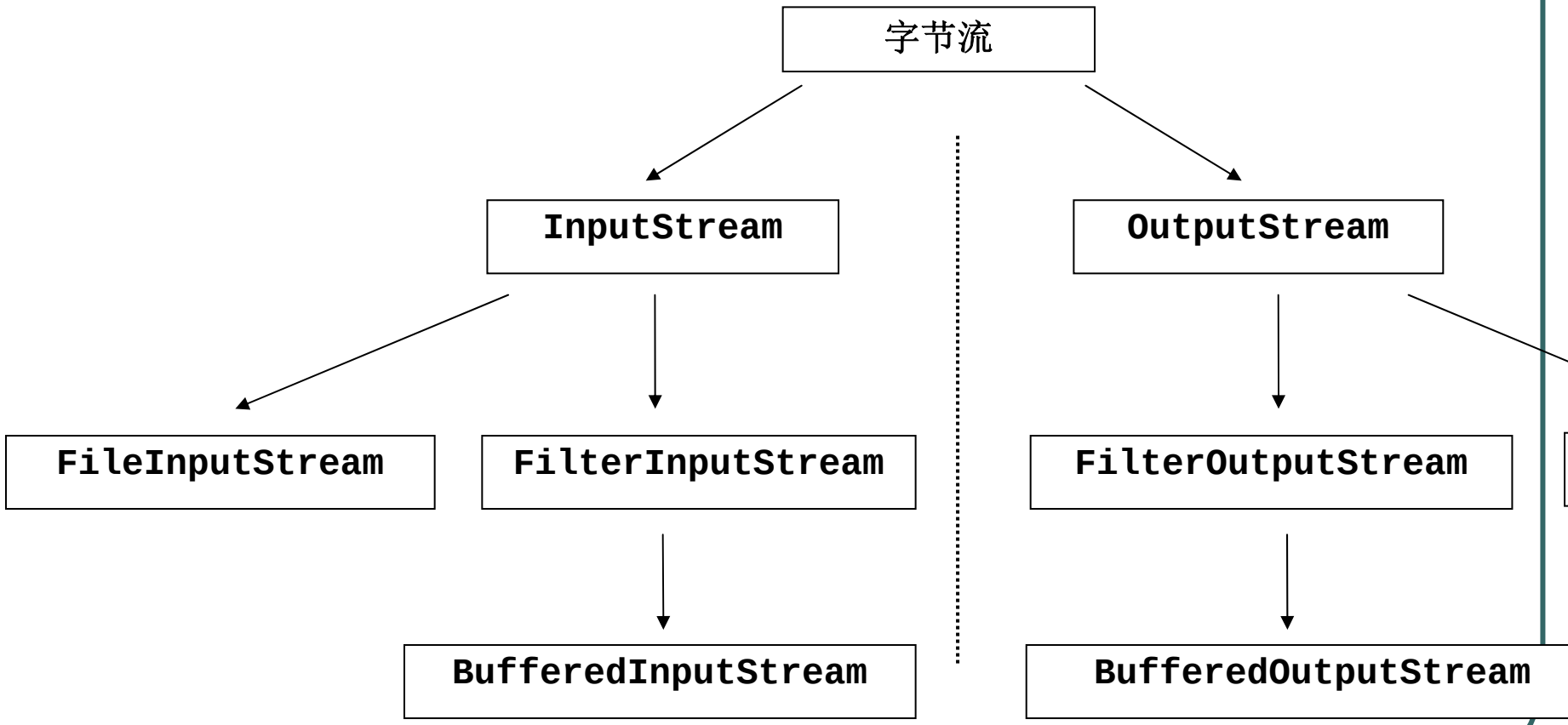
流的基本应用小节

- 流是用来处理数据的。
- 处理数据时，一定要先明确数据源，与数据目的地（数据汇）。
- 数据源可以是文件，可以是键盘。
- 数据目的地可以是文件、显示器或者其他设备。
- 而流只是在帮助数据进行传输,并对传输的数据进行处理，比如过滤处理.转换处理等。

字符流继承体系简图


字符流
Reader
Writer
BufferedReader
InputStreamReader
OutputStreamWriter
Buffer
FileReader
FileWriter

字节流继承体系简图


字节流
InputStream
OutputStream
FileInputStream
FilterInputStream
FilterOutputStream
BufferedInputStream
BufferedOutputStream

File类

- 用来将文件或者文件夹封装成对象
- 方便对文件与文件夹的属性信息进行操作。
- File对象可以作为参数传递给流的构造函数。
- 了解File类中的常用方法。

递归

- 函数自己调用自己。
- 注意：递归时一定要明确结束条件。
- 应用场景：
  - 当某一功能要重复使用时。
- 练习：
  - 列出一个文件夹下所有的子文件夹以及子文件
- 思考：
  - 1，删除一个目录的过程是如何进行的？

IO包中的其他类

- RandomAccessFile
  - 随机访问文件，自身具备读写的方法。
  - 通过skipBytes(int x),seek(int x)来达到随机访问。
- 管道流
  - PipedInputStream和PipedOutputStream
    - 输入输出可以直接进行连接，通过结合线程使用。

IO包中的其他类

- 打印流
  - PrintWriter与PrintStream
    - 可以直接操作输入流和文件。
- 序列流
  - SequenceInputStream
    - 对多个流进行合并。
- 操作对象
  - ObjectInputStream与ObjectOutputStream
    - 被操作的对象需要实现Serializable （标记接口);
- 练习：文件分割程序。

IO包中的其他类

- 操作基本数据类型
  - DataInputStream与DataOutputStream
- 操作字节数组
  - ByteArrayInputStream与ByteArrayOutputStream
- 操作字符数组
  - CharArrayReader与CharArrayWrite
- 操作字符串
  - StringReader 与 StringWriter

字符编码

- 字符流的出现为了方便操作字符。
- 更重要是的加入了编码转换。
- 通过子类转换流来完成。
  - InputStreamReader
  - OutputStreamWriter
- 在两个对象进行构造的时候可以加入字符集。

编码表的由来

- 计算机只能识别二进制数据，早期由来是电信号。
- 为了方便应用计算机，让它可以识别各个国家的文字。
- 就将各个国家的文字用数字来表示，并一一对应，形成一张表。
- 这就是编码表。

常见的编码表

- ASCII：美国标准信息交换码。
  - 用一个字节的7位可以表示。
- ISO8859-1：拉丁码表。欧洲码表
  - 用一个字节的8位表示。
- GB2312：中国的中文编码表。
- GBK：中国的中文编码表升级，融合了更多的中文文字符号。
- Unicode：国际标准码，融合了多种文字。
  - 所有文字都用两个字节来表示,Java语言使用的就是unicode
- UTF-8：最多用三个字节来表示一个字符。
- .....

转换流的编码应用

- 可以将字符以指定编码格式存储。
- 可以对文本数据指定编码格式来解读。
- 指定编码表的动作由构造函数完成。

字符编码

- 编码：字符串→字节数组
- 解码：字节数组→字符串