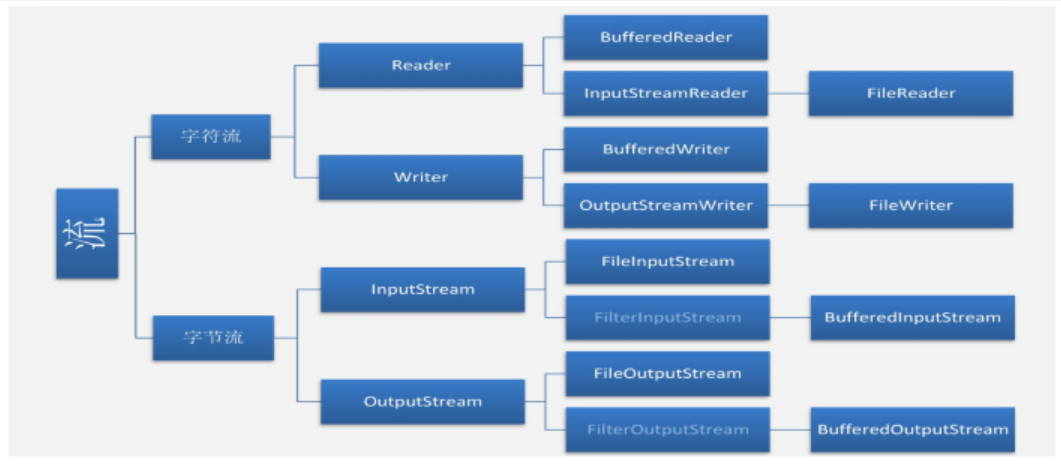
# IO流



流：可以理解为数据的流动，就是一个数据流，IO流最终要以对象来体现，对象都存在于java.io包中。

## 流的分类

（1）输入流（读）和输出流（写）

（2）处理数据的不同，分为字节流和字符流。

## 字节和字符的理解

字节（byte）是计算机中用于计量存储容量的一种计量单位，也表示计算机编程语言中的数据类型和语言字符。byte即字节的意思，是java中的基本数据类型，用来声明字节型的变量，一个字节包含8位，所以，byte类型的取值范围是-128到127。通常在读取非文本文件时（如图片，声音，可执行文件）需要用字节数组来保存文件的内容，在下载文件时，也是用byte数组作临时的缓冲器接收文件内容，所以说byte在文件操作时是必不可少的。不管是对文件写入还是读取都要用到。

计算机只知道字节，而字符是语义上的单位，它是有编码的，一个字符可能编码成1个2个甚至3个4个字节。这跟字符集编码有关系，英文字母和数字是单字节，但汉字这些自然语言中的字符是多字节的。一个字节只能表示

255个字符，不可能用于全球那么多种自然语言的处理，因此肯定需要多字节的存储方式。

## 流的基本知识

（1）Java语言提供了4个非常重要的输入输出类，分别是：InputStream、OutputStream、Reader、Writer，它们都是抽象类。

# 编码

1、gbk编码中中文占用2个字节，英文占用1个字节。

2、utf-8编码中中文占用3个字节，英文占用1个字节。

3、utf-16be中文占用2个字节，英文也是占用2个字节。

4、当你的字节序列是某种编码时，这个时候想把字节序列变成字符串时，也需要用这种编码方式，否则会出现乱码。

5、文本文件就是字节序列，可以是任意编码的字节序列，如果我们在中文机器上直接创建文本文件，那么该文本文件只认识ansi编码，联通、联这是一种巧合，他们正好符合了utf-8编码的规则。

6、我们的java项目的编码方式如何查看：在eclipse中点击项目右键，选择属性，然后查看资源，资源中显示了项目的编码方式。

# File类

## 基本概念

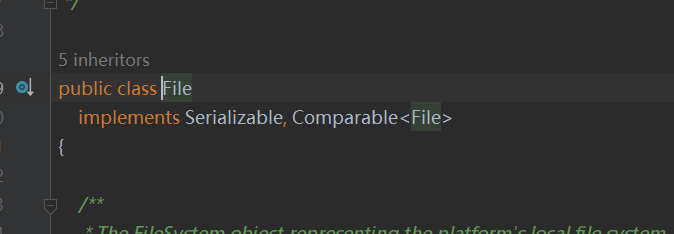
1、java.io.File类用于表示文件（或者目录），可以用于操作硬盘上的文件或者目录。

2、File类只用于表示文件（或者目录）的信息，如名称、大小，不能用于文件内容的访问。

3、File.separator 用于设置分隔符，不管是window系统还是linux系统都是适用的。

4、直接打印file对象输出的是文件的目录。

## 源码



## 构造函数

（1）new File(String pathName) pathName:文件的路径名

（2）new File(String parent,String child) parent:父路径字符串 child:子路径字符串

## 方法

### （1）String getName()

//获取文件（或者文件夹）的名称

### （2）boolean canRead()

//判断文件是否是可读的。

### （3）boolean canWrite()

//判断文件是否可被写入

### （4）boolean exists()

//判断文件是否存在

### （5）long length()

//获取文件的长度（以字节为单位）

### （6）String getAbsolutePath()

//获取文件的绝对路径

### （7）String getParent()

//获取文件的父路径

### （8）boolean isFile()

//判断是否是一个文件

### （9）boolean isDirectory()

//判断文件是否是一个目录

### （10）boolean isHidden()

//判断文件是否是隐藏文件

### （11）Long lastModified()

//获取文件最后的修改时间

### （12）boolean renameTo()

//重新命名此抽象路径名表示的文件

### （13）File[] listFiles()

//获取同一目录下文件的列表（包含文件和子目录中的文件抽象）

### （14）String[] list()

//用于列出当前目录下的子目录和文件（不会列出子目录下面的文件）

### （15）createNewFile()

//当且仅当不存在指定路径的文件时，创建一个新的空文件。

### （16）delete()

//删除抽象路径名表示的文件或目录

### （17）mkdir()

//创建文件夹

### （18）mkdirs()

//创建多级文件夹

### public String getCanonicalPath()

返回一个规范化的绝对路径名字符串

createTempFile

# 遍历目录

//列出指定目录下（包括其子目录）的所有文件

public static void listDirectory(File dir) throws IOException{

if(!dir.exists()){

throw new IllegalArgumentException("目录"+dir+"不存在");

}

if(!dir.isDirectory()){

throw new IllegalArgumentException("目录"+dir+"不是目录");

}

//返回的是直接子目录（文件）的抽象

File[] files = dir.listFiles();

if(files!=null && files.length>0){

for(File file:files){

if(file.isDirectory()){

//递归

listDirectory(file);

}else{

System.out.println(file);

}

}

}

}

# RandomAccessFile

（1）RandomAccessFile支持随机访问文件，可以访问文件的任意位置。

（2）Java文件模型，在硬盘上的文件是byte存储的，是数据的集合。

（3）打开文件有两种模式，分别是：rw（读写），r（只读），文件指针，打开文件时指针在开头pointer=0;

（4）写方法：write(int)—只写一个字节（后8位），同时指针指向下一个位置，准备再次写入。

（5）读方法：read()—读一个字节。

（6）文件读写完成以后一定要关闭。

示例代码：

# Resource接口

## 基本概念

## 方法

## 实现类

UrlResource URL类的包装

ClassPathResource 类路径资源类

FileSystemResource 文件系统资源类 可转成File或Path

ServletContextResource 在web容器中使用

InputStreamResource 没有选择的情况下使用

ByteArrayResource 处理字节数组时使用

# ResourceLoader接口

## 基本概念

ResourceLoader 接口是 Spring 框架中的一个核心接口，用于加载资源文件。它提供了一种统一的方式来获取不同类型的资源，例如文件、类路径下的文件、URL、ServletContext 等。

主要作用是将资源的加载过程进行抽象和封装，使得应用程序可以以一致的方式获取所需的资源，并且不依赖于底层的资源加载细节。它使得开发者可以更加方便地在 Spring 应用中使用和管理资源。

## 常量

String CLASSPATH\_URL\_PREFIX = ResourceUtils.CLASSPATH\_URL\_PREFIX;

这是一个常量，表示类路径（classpath）的前缀，可以与资源路径拼接来获取类路径下的资源。

## 方法

### Resource getResource(String location)

根据给定的资源位置（location）返回一个 Resource 对象，Resource 对象可以用于进一步操作，如读取资源内容、获取资源的 URL、判断资源是否存在等。

### ClassLoader getClassLoader()

返回一个 ClassLoader 对象，用于加载类路径下的资源文件。

## 代码示例

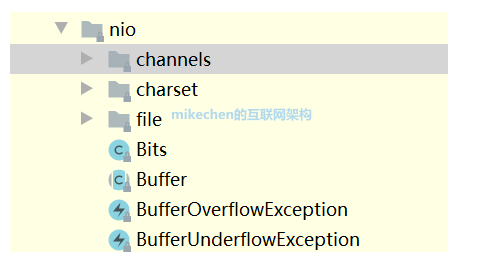
# ClassPathResource类

## 基本概念

# NIO

## 基本概念

NIO中的N可以理解为Non-blocking，不单纯是 New，是解决高并发、I/O高性能的有效方式。Java NIO是Java1.4之后推出来的一套IO接口，NIO提供了一种完全不同的操作方式，NIO支持面向缓冲区的、基于通道的IO操作。新增了许多用于处理输入输出的类，这些类都被放在java.nio包及子包下，并且对原java.io包中的很多类进行改写，新增了满足NIO的功能。



下面是Java IO和Java NIO的一些比较：



## 核心组件

### Buffer

### Channel

### Selector

# Files类