001.File类

基本了解File类

File类主要是JAVA为文件这块的操作(如删除、新建等)而设计的相关类

File类的包名是java.io，其实现了Serializable, Comparable两大接口以便于其对象可序列化和比较

创建一个文件/文件夹

删除文件/文件夹

获取文件/文件夹

判断文件/文件夹是否存在

对文件夹进行遍历

获取文件的大小

File类是一个与系统无关的类，任何的操作系统都可以使用这个类中的方法

重点：记住这第三个单词

file——文件

directory——文件夹/目录

path——路径

彻底搞明白绝对路径和相对路径

绝对路径————是一个完整的路径

以盘符（ 比如C：）开始的路径

c:\a.txt

c:\demo\b.txt

相对路径————是一个简化的路径

相对指的是相对于当前项目的根目录

如果使用当前项目的根目录，路径可以简化书写

D:\java\Java语言高级\File类\File类\File类的概述.avi ->简化为——File类的概述.avi(可以省略项目的根目录)

注意

1. 路径是不区分大小写
2. 路径中的文件名称分隔符widows使用反斜杠，反斜杠是转义字符，两个反斜杠代表一个普通反斜杠。

补充在Java程序中获取项目根路径的方法：

```java
public class FileTest1 {
    public static void main(String[] args) {
        FileTest1 fileTest1 = new FileTest1();
        try {
            fileTest1.getUrl();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void getUrl() throws IOException {
        String path = "";

        //第一种：获取类加载的根路径
        path = this.getClass().getResource("/").getPath();
        System.out.println(path);

     //获取当前类的所在工程路径,不加“/”  获取当前类的加载目录
        path = this.getClass().getResource("").getPath();
        System.out.println(path);

        // 第二种：获取项目路径
        File file = new File("");
        path = file.getCanonicalPath();
        System.out.println(path);

        // 第三种：
        URL path1 = this.getClass().getResource("");
        System.out.println(path1);

        // 第四种：
        path = System.getProperty("user.dir");
        System.out.println(path);


        // 第五种：  获取项目路径
        path = Thread.currentThread().getContextClassLoader().getResource("").getPath();
        System.out.println(path);

        //第六种  表示到项目的根目录下，要是想到目录下的子文件夹,修改"/"即可
        //request.getSession().getServletContext().getRealPath("/"));
    }
}
```

File类的静态属性

static String pathSeparator——与系统有关的路径分隔符，为了方便，它被表示为一个字符串。

static char pathSeparatorChar——与系统有关的路径分隔符。

windows下是分号,Linux下是冒号

static String Separator——与系统有关的默认名称分隔符，它被表示为一个字符串。

static char SeparatorChar——与系统有关的默认名称分隔符。

windows下是 \,Linux下是 /

File类的构造方法

File(String pathname)——通过将给定路径名字符串转换为抽象路径名来创建一个新的File实例。

参数

String pathname————字符串的路径名称

路径可以是以文件结尾，也可以是以文件夹结尾。

路径可以是相对路径，也可以是绝对路径。

路径可以是存在的，也可以是不存在的。

创建File对象，只是把字符串路径封装为File对象，不考虑路径的真假情况。

File(String parent, String child)——根据parent路径名字符串和child路径名字符串来创建一个新的File实例。

参数————————把路径分成了两部分

String parent————父路径

String child————子路径

好处

父路径和子路径可以单独书写，使用起来非常灵活；父路径和子路径都可以变化。

File(File parent, String child)——根据parent路径名字符串和child路径名字符串来创建一个新的File实例。

好处

父路径和子路径可以单独书写，使用起来非常灵活；父路径和子路径都可以变化。

父路径是File类型，可以使用File的方法对路径进行一些操作，在使用路径创建对象。

File类中访问文件名相关的常用方法

public String getName()————返回此File标识的文件或目录的名称。

public String getPath()————将此File转换为路径名字符串。

public String getAbsolutePath()————返回此File的绝对路径名字符串，返回的是绝对路径。

public String getParent()————返回File对象所对应的目录（最后一级子目录）的父目录名称。

public boolean renameTo(File newName)————重命名此File对象所对应的文件或目录，成返回true

File类检查文件的相关常用方法

public boolean exists()————此File表示的文件或目录是否实际存在 。

public boolean isDirectory()————此File表示的是否为目录 。

public boolean isFile()————此File表示的是否为文件 。

public canWrite()————此File是否可写

public canRead()————此File是否可读

public isAbsolute()————此判断创建File对象时是不是用的是绝对路径。

获取文件常规信息的常用方法

public long lastModified()————返回文件的最后修改时间。

public long length()————返回文件内容的长度。

File类创建删除文件的常用方法

public boolean creatNewFile()——当且仅当具有该名称的文件尚不存在时，创建一个新的空文件。

creatNewFile声明抛出了IOException，我们调用这个方法，就必须处理这个异常，要么throws，要么try catch

public boolean delete()——删除由此File表示的文件或目录。

此方法，可以删除构造方法路径中给出的文件/文件夹。注意delete方法直接在硬盘删除文件/文件夹，不走回收站，删除要谨慎。

public boolean mkdir()——创建由此File表示的目录。

创建单级空文件夹

public boolean mkdirs()——创建由此File表示的目录，包括任何必需但不存在的父目录。

既可以创建单级空文件夹，也可以创建多级文件夹

File类遍历(文件夹)目录功能

public String[ ] list( )——返回一个String数组，表示该File目录中的所有子文件或子目录。

public File[ ] listFiles( )——返回一个File数组，表示该File目录中的所有子文件或子目录。

注意

list方法和listFiles方法遍历的是构造方法中给出的目录。

如果构造方法中给出的目录的路径不存在，会抛出空指针异常。

如果构造方法中给出的路径不是一个目录，也会抛出空指针异常。

文件过滤器的原理和使用

在File类中有两个listFiles重载的方法，方法的参数传递的就是过滤器。

File[ ] listFiles(**FilesFilter filter**)

java.io.FilesFilter 接口————用于抽象路径名（File对象）的过滤器。

FilesFilter 接口里包含了一个抽象方法：**boolean accept(File pathname)**

参数

File pathname————使用listFiles方法遍历目录，得到的每一个文件对象。

File[ ] listFiles(**FilenameFilter filter**)

java.io.FilenameFilter 接口————实现此接口的类实例可用于过滤文件名。

FilenameFilter接口里包含了一个抽象方法：**boolean accept(File dir，String name)**

参数

File dir ——————构造方法中传递的被遍历的目录

String name————使用lisFiles方法遍历目录，获取的每一个文件/文件夹的名称。

注意

两个过滤器接口是没有实现类的，需要我们自己写实现类，重写过滤的方法accept，在方法中自己定义过滤的规则。

例子：

遍历D:\abc文件夹，以及子文件夹下，只要.java后缀的文件

```
public class FileTest1 {
    public static void main(String[] args) {
        File path = new File("D:\\abc");
        getAllFiles(path);
    }

    public static void getAllFiles(File path) {
        File[] fileList = path.listFiles(new FileFilter() {
            @Override
            public boolean accept(File pathname) {
                if(pathname.isDirectory())
                    return true;
                return pathname.getName().toLowerCase().endsWith(".java");
            }
        });

        for (File f : fileList) {
            if (f.isDirectory()) {
                getAllFiles(f);
            } else {
                System.out.println(f);
            }
        }
    }
```

```
}
```