**File类**

File类是一个主要专门对文件存储路径和文件进行增删改查的类，但是不能读写文件。

* **构造函数**

**public File(File dir，String name)**

新建的File可以在指定目录下创建一个文件。dir是一个路径，name是一个文件名字（没有尾缀也可以）。需要注意的是name字符串前面不需要添加斜线‘/’，这个构造函数会自动在name前面添加斜线。

**Public File(String path)**

构造一个path路径构造一个File对象。

**public File(String dirPath,String name)**

新建的File可以在指定目录下创建一个文件。效果同File(File dir，String name)。

**public File(URI uri)**

使用URI指定路径构造新的文件。

* **创建文件夹**

**boolean mkdir()**

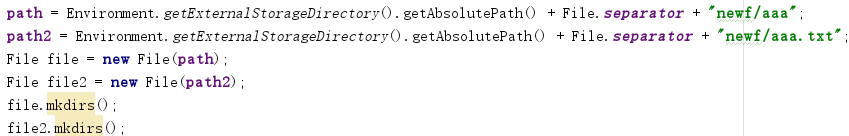
创建一个文件夹，当父目录存在才能成功创建。

**boolean mkdirs()**

创建一个文件夹，可以附带创建父目录。就是说可以创建多层目录。

注意：

不管是你创建的Flie对象时，构造的路径是什么样子，调用这两个方法后只能生成路径，不会生成文件，如下：





即使path2是一个文件的绝对路径，最后也会生成一个文件夹，如果file2再次调用createNewFile方法也不会创建这个aaa.txt文件了。

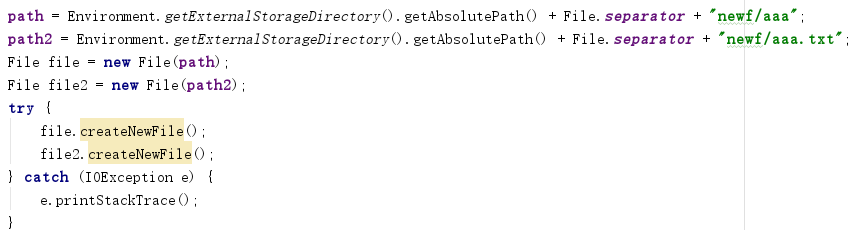
* **创建文件**

**boolean createNewFile（）**

创建一个文件，当父目录存在才能成功创建。

注意：

不管是你创建的Flie对象时，构造的路径是什么样子，在调用这个方法之前只要没有调用mkdir()和mkdirs()这两个方法，就能成功创建文件。并且不管路径最后的一个斜线后的字符串带不带尾缀，都会创建一个文件，如下：





还有一点，如上路径，如果系统中没有newf这个路径（这个文件夹），那么aaa或者aaa.txt文件会创建失败。因为createNewFile()不会创建路径。

* **删除文件**

**boolean delete（）**

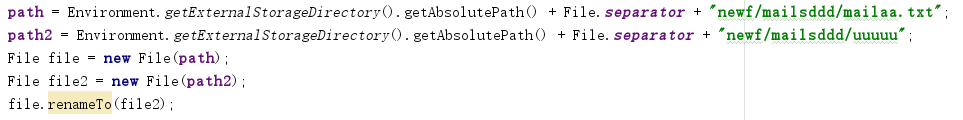
只能删除路径末端的文件或者文件夹。

例如：使用storage/emulated/0/a/b/c.txt这个路径创建一个file对象，然后调用delete()方法，那么只能删除c.txt文件（如果c是一个文件夹，那么照样会被删除）。如果我们用storage/emulated/0/a这个路径创建一个file对象，然后调用delete()方法，则会删除失败。所以要想删除一个目录，只能是遍历遍历这个目录下所有目录，然后从路径末端逐层向上删除目录。

* **修改文件或文件夹**

**boolean renameTo(File newPath)**

调用file.rename(newFile)会将file路径末端文件或者文件夹的名称修改为newPath对象路径末端的文件或者文件夹的名称。





* **查询**

**File getAbsoluteFile()**

返回一个新的文件，该文件的路径为调用这个方法的file对象的路径。

**String getName()**

返回路径尾端文件或者文件夹的名称。

**String getPath()**

返回相对于当前目录的相对路径

**String getAbsolutePath()**

返回该文件的绝对路径。

**String getParent()**

返回当前文件或者文件夹的完全父目录。

**String getCanonicalPath()**

返回一个唯一的绝对路径。即使构造File时传入的是相对路径，调用此方法也会返回

一个绝对路径，而getAbsolutePath和getPath方法则不是。

**long getFreeSpace()**

返回在分区上剩余的字节数量，包括当前File的路径

**long getTotalSpace()**

返回分区的总字节大小。

**long getUsableSpace()**

返回分区可用字节的大小，会与getFreeSpace方法返回结果有偏差。

**long length()**

返回路径下文件的大小，单位为字节。

* **其他**

**boolean equals (Object obj)**

判断obj和调用的对象是否相同，只要两个file对象的构造路径一样则返回true。

**boolean IsDirectory()**

判断是否为文件夹。

**boolean IsFile()**

判断是否为文件

**boolean isHidden**

判断是否为操作系统定义的隐藏文件

**long lastModified()**

返回最后一次修改该文件的时间，以毫秒计算，从1970年1月1日开始算

**URI toURI()**

返回一个文件的URI

**boolean setWritable(boolean writable,boolean ownerOnly)**

writable 如果为真，对该文件有写权限，否则没有

ownerOnly 如果为真，只对文件的拥有者有写权限，否则，对其他人也有写权限。

**boolean setWritable(boolean writable,true)**

同setWritable(writable,true)一个意思

**boolean setReadOnly()**

对文件只有读权限

**boolean setReadable(boolean readable,boolean ownerOnly)**

readable 如果为真，对该文件有读权限

ownerOnly 如果为真，只对文件的拥有者有读权限，否则，对其他人也有读权限

**boolean setReadable(boolean readable)**

同setReadable(boolean readable,true)一个意思

**File[] listFiles()**

列出文件夹下的所有文件和文件夹名

**boolean exists**()

判断文件或者文件夹是否存在。

* **判断SD卡是否插入（即是否被挂载）**

Environment.getExternalStorageState().equals(android.os.Environment.MEDIA\_MOUNTED);

* **文件读写操作模式**

Context.MODE\_PRIVATE：新内容覆盖原内容

Context.MODE\_APPEND：新内容追加到原内容后

Context.MODE\_WORLD\_READABLE：允许其他应用程序读取

Context.MODE\_WORLD\_WRITEABLE：允许其他应用程序写入，会覆盖原数据。