1、File类的使用

1. File类的一个对象，代表一个文件或一个文件目录(俗称：文件夹)

2. File类声明在java.io包下

3. File类中涉及到关于文件或文件目录的创建、删除、重命名、修改时间、文件大小等方法，并未涉及到写入或读取文件内容的操作。如果需要读取或写入文件内容，必须使用IO流来完成。

4. 后续File类的对象常会作为参数传递到流的构造器中，指明读取或写入的"终点".

2、File类的构造

1、File的实例化

1. public File(String pathname)

以pathname为路径创建File对象，可以是绝对路径或者相对路径，如果pathname是相对路径，当默认的当前路径在系统属性user.dir中存储

》 相对路径：相较于某个路径下，指明的路径

》 绝对路径：包含盘符在内的文件或文件目录的路径

2. public File（String parent, String child）

以parent 为父路径， child 为子路径创建 File 对象

3. public File(File parent,String child)

根据一个父File对象和子文件路径创建File对象

2、路径分隔符

1. 路径中的每级目录之间用一个路径分隔符隔开
2. 路径分隔符和系统有关

》 windows和DOS系统默认使用“\”来表示

》 UNIX和URL使用“/”来表示

3. Java程序支持跨平台运行，因此路径分隔符要慎用
4. File提供了一个常量：

public static final String separator.根据系统，动态的提供分隔符

5. 举例

```
File file1 = new File(File("d:\\atguig\\info.txt")
File file2 = new File(File("d:"+ File.separator + "atguigu"+ File.separator + "info.txt");
File file3 = new File(File("d:/atguig");
```

3、File的实例化举例

```
@Test
public void test1(){
//构造器1
File file1 = new File("hello.txt");//相对于当前module
File file2 = new File("D:\\workspace_idea1\\JavaSenior\\day08\\he.txt");

System.out.println(file1);
System.out.println(file2);
```

```
9
10  //构造器2:
11  File file3 = new File("D:\\workspace_idea1","JavaSenior");
12  System.out.println(file3);
13
14  //构造器3:
15  File file4 = new File(file3,"hi.txt");
16  System.out.println(file4);
17  }
```

3、File类的常用方法

1、File的获取方法

1. public String getAbsolutePath()：获取绝对路径
2. public String getPath() ： 获取路径
3. public String getName() ： 获取名称
4. public String getParent()：获取上层文件目录路径。若无，返回null
5. public long length() ： 获取文件长度（即：字节数）。不能获取目录的长度。
6. public long lastModified() ： 获取最后一次的修改时间，毫秒值

```
1   @Test
2   public void test2(){
3   File file1 = new File("hello.txt");
4   File file2 = new File("d:\\io\\hi.txt");
5
6   System.out.println(file1.getAbsolutePath());
7   System.out.println(file1.getPath());
8   System.out.println(file1.getName());
9   System.out.println(file1.getParent());
10   System.out.println(file1.length());
11   System.out.println(new Date(file1.lastModified()));
12
13   System.out.println();
```

```
14
15  System.out.println(file2.getAbsolutePath());
16  System.out.println(file2.getPath());
17  System.out.println(file2.getName());
18  System.out.println(file2.getParent());
19  System.out.println(file2.length());
20  System.out.println(file2.lastModified());
21  }
```

7. 如下的两个方法适用于文件目录：

》 public String[] list()：获取指定目录下的所有文件或者文件目录的名称数组

》 public File[] listFiles()：获取指定目录下的所有文件或者文件目录的File数组

```
1  @Test
2  public void test3(){
3  File file = new File("D:\\workspace_idea1\\JavaSenior");
4
5  String[] list = file.list();
6  for(String s : list){
7  System.out.println(s);
8  }
9
10  System.out.println();
11
12  File[] files = file.listFiles();
13  for(File f : files){
14  System.out.println(f);
15  }
16
17  }
```

2、File类的重命名功能

public boolean renameTo(File dest):把文件重命名为指定的文件路径

比如：file1.renameTo(file2)为例：要想保证返回true,需要file1在硬盘中是存在的，且file2不能在硬盘中存在。

```java
@Test
public void test4(){
File file1 = new File("hello.txt");
File file2 = new File("D:\\io\\hi.txt");

boolean renameTo = file2.renameTo(file1);
System.out.println(renameTo);

}
```

3、File的判断功能

1. public boolean isDirectory()：判断是否是文件目录
2. public boolean isFile()：判断是否是文件
3. public boolean exists()：判断是否存在
4. public boolean canRead()：判断是否可读
5. public boolean canWrite()：判断是否可写
6. public boolean isHidden()：判断是否隐藏

```java
@Test
public void test5(){
File file1 = new File("hello.txt");
file1 = new File("hello1.txt");

System.out.println(file1.isDirectory());
System.out.println(file1.isFile());
System.out.println(file1.exists());
System.out.println(file1.canRead());
 System.out.println(file1.canWrite());
 System.out.println(file1.isHidden());

 System.out.println();

 File file2 = new File("d:\\io");
 file2 = new File("d:\\io1");
 System.out.println(file2.isDirectory());
 System.out.println(file2.isFile());
```

```
System.out.println(file2.exists());
System.out.println(file2.canRead());
System.out.println(file2.canWrite());
System.out.println(file2.isHidden());

}
```

4、File的创建和删除功能

4.1. 文件或文件目录的创建(磁盘中)

1. public boolean createNewFile()：创建文件。若文件存在，则不创建，返回false

2. public boolean mkdir()：创建文件目录。如果此文件目录存在，就不创建了。如果此文件目录的上层目录不存在，也不创建。

3. public boolean mkdirs()：创建文件目录。如果此文件目录存在，就不创建了。如果上层文件目录不存在，一并创建

4. 注意事项：如果你创建文件或者文件目录没有写盘符路径，那么默认在项目路径下

4.2. 文件或文件目录(磁盘中)

```
@Test
public void test6() throws IOException {
File file1 = new File("hi.txt");
if(!file1.exists()){
//文件的创建
file1.createNewFile();
System.out.println("创建成功");
}else{//文件存在
file1.delete();
 System.out.println("删除成功");
 }
 }

 @Test
 public void test7(){
 //文件目录的创建
 File file1 = new File("d:\\io\\io1\\io3");
```

```
18
19  boolean mkdir = file1.mkdir();
20  if(mkdir){
21  System.out.println("创建成功1");
22  }
23
24  File file2 = new File("d:\\io\\io1\\io4");
25
26  boolean mkdir1 = file2.mkdirs();
27  if(mkdir1){
28  System.out.println("创建成功2");
29  }
30
31  //要想删除成功，io4文件目录下不能有子目录或文件
32  File file3 = new File("D:\\io\\io1\\io4");
33  file3 = new File("D:\\io\\io1");
34  System.out.println(file3.delete());
35  }
```