11 IO流_字符流

JAVAEE高级

一：字符流描述

> 1、字符流是可以直接读写字符的IO流
>
> 2、符流读取字符、就要先读取到字节数据、 然后转为字符.
>
> 3、如果要写出字符、 需要把字符转为字节再写出.
>
> 4、读:字符流FileReader

```
1.      @Test
2.      public void getVoid() throws IOException {
3.          FileReader fr = new FileReader("D:\\IO\\修改后.txt");
4.          int c;
5.          while ((c = fr.read()) != -1) { // 通过项目默认的码表一次读取一个字符
6.              System.out.print((char) c);
7.          }
8.          fr.close();
9.      }
```

> 5、写:字符流FileWriter

```
1.      @Test
2.      public void getVoid() throws IOException {
3.          FileWriter fw = new FileWriter("D:\\IO\\修改后.txt",true);
4.          fw.write("大家好大家好大家好大家好大家好大家好大家好");
5.          fw.write(97);
6.          fw.close();
7.      }
```

二：字符流的拷贝

1、拷贝文件

```
@Test
    public void getVoid() throws IOException {
        FileReader fr = new FileReader("D:\\IO\\修改后.txt");
        FileWriter fw = new FileWriter("D:\\IO\\修改后2.txt");
        int c;
        while ((c = fr.read()) != -1) {
            fw.write(c);
        }
        fr.close();
        fw.close();
    }
```

2、拷贝纯文本<是不拷贝纯文本文件的>

```
@Test
    public void getVoid() throws IOException {
        FileReader fr = new FileReader("D:\\IO\\微信图片.png");
        FileWriter fw = new FileWriter("D:\\IO\\微信图片4.png");
        int c;
        while ((c = fr.read()) != -1) {
            fw.write(c);
        }
        fr.close();
        fw.close();
    }
```

3、使用数组的方式进行拷贝

```
    @Test
    public void getVoid() throws IOException {
        FileReader fr = new FileReader("D:\\IO\\修改后.txt");
        FileWriter fw = new FileWriter("D:\\IO\\修改后111.txt");
        char[] arr = new char[1024];
        int len;
```

```
        while ((len = fr.read(arr)) != -1) { // 将文件上的数据读取到字符数组中
            fw.write(arr, 0, len); // 将字符数组中的数据写到文件上
        }
        fr.close();
        fw.close();
    }
```

三：带缓冲的字符流

> 1、BufferedReader的read()方法读取字符时会一次读取若干字符到缓冲区, 然后逐个返回给程序, 降低读取文件的次数, 提高效率
>
> 2、BufferedWriter的write()方法写出字符时会先写到缓冲区, 缓冲区写满时才会写到文件, 降低写文件的次数, 提高效率

```
@Test
    public void getVoid() throws IOException {
        BufferedReader br = new BufferedReader(new FileReader("D:\\IO\\修改前.txt"));
        BufferedWriter bw = new BufferedWriter(new FileWriter("D:\\IO\\修改后111.txt"));
        int c;
        while ((c = br.read()) != -1) {
            bw.write(c);
        }
        br.close();
        bw.close();
    }
```

①、readLine()方法

> 1、可以读取一行字符(不包含换行符号)

```
    @Test
    public void getVoid() throws IOException {
        BufferedReader br = new BufferedReader(new FileReader("D:\\IO\\修改后.txt"));
```

```
        BufferedWriter bw = new BufferedWriter(new FileWriter("D:\\IO\\修改后1212.txt"));
        String line;
        while((line = br.readLine()) != null) {
            bw.write(line);
            bw.newLine();
            System.out.println(line);
            //写出回车换行符
            bw.write("\r\n");
        }
        br.close();
        bw.close();
    }
```

四：LineNumberReader

> 1、LineNumberReader是BufferedReader的子类, 具有相同的功能, 并且可以统计行号
>
> 2、getLineNumber()方法可以获取当前行号
>
> 3、setLineNumber()方法可以设置当前行号

```
@Test
    public void getVoid() throws IOException {
        // 创建输入输出流
        LineNumberReader lnr = new LineNumberReader(new FileReader("D:\\IO\\修改后.txt"));
        String line;
        lnr.setLineNumber(10);
        while ((line = lnr.readLine()) != null) {
            System.out.println(lnr.getLineNumber() + ":" + line);
        }
        lnr.close();
    }
```

五：案列

1、<文件反转>

1、将一个文本文档上的文本反转,第一行和倒数第一行交换,第二行和倒数第二行交换

2、分析

创建输入输出流
创建集合
将读到的数据存储在集合
倒着遍历集合将数据写到文件上
关流

```
@Test
    public void getVoid() throws IOException {
        // 创建输入输出流
        BufferedReader br = new BufferedReader(new FileReader("D:\\IO\\修改后.txt"));
        // 创建集合
        ArrayList<String> list = new ArrayList<>();
        //将读到的数据存储在集合中
        String line;
        while ((line = br.readLine()) != null) {
            list.add(line);
        }
        br.close(); // 关流

        // 倒着遍历集合将数据写到文件上
        BufferedWriter bw = new BufferedWriter(new FileWriter("D:\\IO\\修改后2.txt"));
        for (int i = list.size() - 1; i >= 0; i--) {
            bw.write(list.get(i));
            bw.newLine();
        }
        // 关流
        bw.close();
    }
```

2、<获取文件出现的字符的次数>

1、获取一个文本上每个字符出现的次数,并将结果写在.txt文件上

```
    @Test
```

```
    public void getVoid() throws IOException {
        //1创建带缓冲区的输入流对象
        BufferedReader br = new BufferedReader(new FileReader("D:\\IO\\修改后.txt"));
        //2创建双列集合对象,目的是把字符当作键,把字符出现的次数当作值
        HashMap<Character, Integer> hm = new HashMap<>();
        //3通过读取不断向集合中存储,存储的时候要判断,如果不包含这个键就将键和值为1存储,如果包含就将键和值加1存储
        int c;
        while((c = br.read()) != -1) {
            char ch = (char)c;
            hm.put(ch, !hm.containsKey(ch)? 1 : hm.get(ch) + 1);
        }
        //4关闭输入流
        br.close();
        //5创建输出流对象
        BufferedWriter bw = new BufferedWriter(new FileWriter("D:\\IO\\修改后1212.txt"));
        //6将结果写出

        for (Character key : hm.keySet()) {
            bw.write(key + "=" + hm.get(key));
        }
        bw.close();
    }
```