java io读取写入文件效率比较

作者：JavaHope
原文链接：https://ld246.com/article/1472004508201
来源网站：链滴

```
public class ReadTxtJson {

    public static String readTxtFile(String FileName) throws Exception {
        BufferedInputStream bufferedInputStream = null;
        ByteArrayOutputStream memStream = null;
        byte[] data = null;
        try {
            bufferedInputStream = new BufferedInputStream(new FileInputStream(FileName));
            memStream = new ByteArrayOutputStream();
            byte[] buffer = new byte[1024];
            int len = 0;
            while ((len = bufferedInputStream.read(buffer)) != -1){
                memStream.write(buffer, 0, len);
            }
            data = memStream.toByteArray();
        } catch (Exception e) {
            e.printStackTrace();
        }finally{
            try {
                    if(memStream != null){
                      memStream.close();
                    }
                    if(bufferedInputStream != null){
                        bufferedInputStream.close();
                    }
          } catch (IOException e) {
              e.getStackTrace();
          }
        }

        String s = new String(data);
        if(s != null){
            bufferedWriter(s,"D:\\FtpFile\\test1.txt");
        }
        return new String(data);
    }


    /**
     * 以行为单位读写文件内容
     *
     * @param filePath
     */
    public static String readTxtFileJson(String filePath) throws Exception{
        File file = new File(filePath);
        InputStreamReader read = null;
        StringBuffer sb = null;
        try {
            //判断文件是否存在
            if(file.isFile() && file.exists()){
                read = new InputStreamReader(new FileInputStream(file),"utf-8");
                BufferedReader bufferedReader = new BufferedReader(read);
                sb = new StringBuffer();
                String lineTxt = null;
```

```
            while((lineTxt = bufferedReader.readLine()) != null){
                sb.append(lineTxt);
            }
        }else{
            System.out.println("找不到指定的文件");
        }
    } catch (Exception e) {
        e.getStackTrace();
    } finally {
        if (read != null) {
            try {
                read.close();
            } catch (IOException e) {
                e.getStackTrace();
            }
        }
    }

    if(sb != null){
        bufferedWriter(sb.toString(),"D:\\FtpFile\\test2.txt");
    }
    return sb != null ? sb.toString() : null; // GsonUtil.transJsonStrToObject(sb.toString(), Kub
Data.class)
}

/**
 * 缓冲字符写入文件，写字符串，数组或字符数据
 * @param content
 * @throws Exception
 */
public static void bufferedWriter(String content,String filePath) throws Exception{
    FileWriter fw = null;
    BufferedWriter bw = null;
    try {
        fw = new FileWriter(new File(filePath).getAbsoluteFile());
        bw = new BufferedWriter(fw);
        bw.write(content);
    } catch (Exception e) {
        e.printStackTrace();
    } finally{
        try {
            if(bw != null){
                bw.close();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

/**
 * 文件输出流,必须将数据转换为字节，并保存到文件
 * @param content
 * @throws Exception
```

```
     */
    public static void fileOutputStream(String content,String filePath) throws Exception{
        FileOutputStream fop = null;
        try {
            fop = new FileOutputStream(new File(filePath));
            byte[] contentInBytes = content.getBytes();
            fop.write(contentInBytes);
            fop.flush();
        } catch (Exception e) {
             e.printStackTrace();
        } finally{
            try {
                if (fop != null) {
                    fop.close();
                }
            } catch (IOException e) {
             e.printStackTrace();
            }
        }
    }

    /**
     * 测试1：
     * 文件大小：2m
     * 读取:readTxtFileJson 行读，写入：bufferedWriter 缓冲字符写入
     * 用时：51秒
     *
     * 读取:readTxtFile 缓冲读取，写入：bufferedWriter 缓冲字符写入
     * 用时：31秒
     *
     * 测试2：
     * 文件大小：10m
     * 读取:readTxtFileJson 行读，写入：fileOutputStream 文件输出流写入
     * 用时：501秒
     *
     * 读取:readTxtFile 缓冲读取，写入：fileOutputStream 文件输出流写入
     * 用时：172秒
     *
     * 文件大小：10m
     * 读取:readTxtFileJson 行读，写入：bufferedWriter 缓冲字符写入
     * 用时：293秒
     *
     * 读取:readTxtFile 缓冲读取，写入：bufferedWriter 缓冲字符写入
     * 用时：132秒
     *
     * 总结：
     * 不按格式读取效率高写入文件后大小比源文件小：readTxtFile 缓冲读取，bufferedWriter 缓冲
符写入
     * 按格式读取效率偏低(是第一种方式的一倍左右)写入文件后大小比源文件大小相当：readTxtFileJ
on 行读 ，bufferedWriter 缓冲字符写入
     * @param args
     */
    public static void main(String[] args) {
        try {
```

```
            long date1 = System.currentTimeMillis();
            String s = readTxtFileJson("D://FtpFile//get_services.txt");
            //System.out.println(s);
            System.out.println(System.currentTimeMillis()-date1);

//            if(Util.isNotNull(kubeData)){
//                System.out.println(kubeData.getKind()+"=="+kubeData.getApiVersion());
//            }

            long date2 = System.currentTimeMillis();
            String s1 = readTxtFile("D://FtpFile//get_services.txt");
            //System.out.println(s1);
            System.out.println(System.currentTimeMillis()-date2);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```