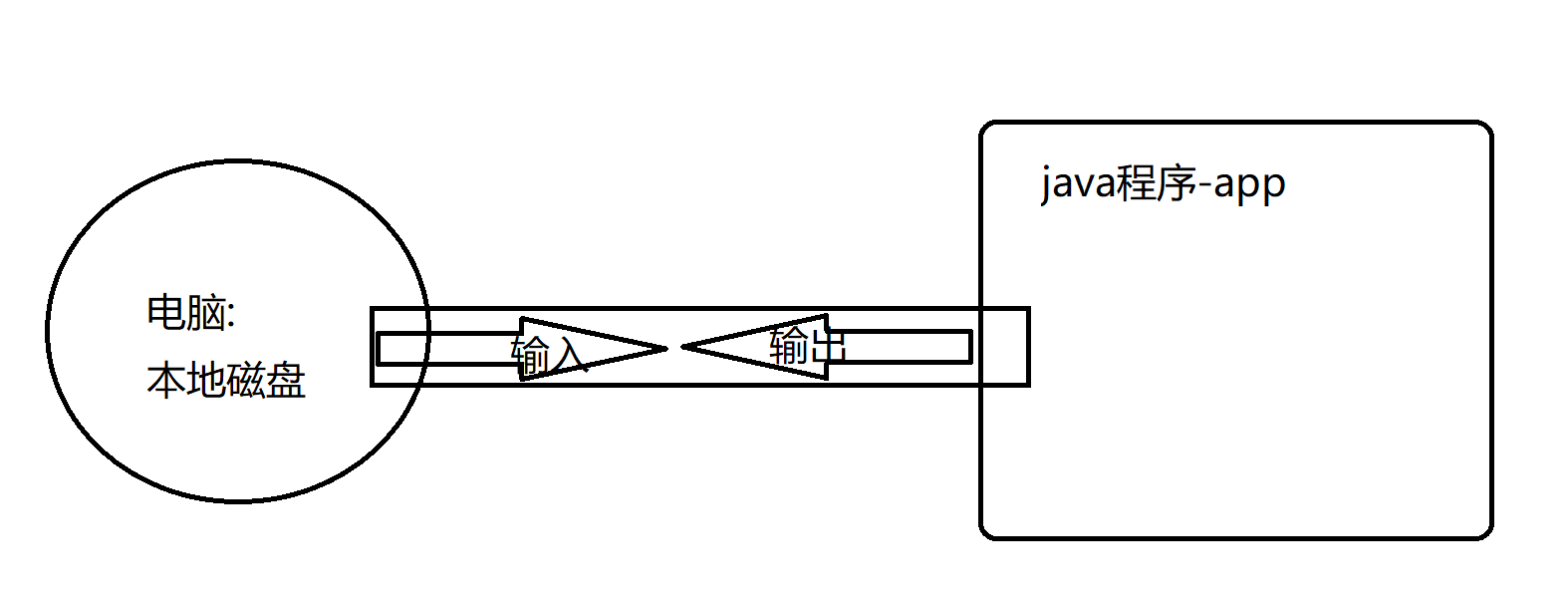
# IO

Input输入 Output输出



按照流向: 输入流和输出流

按照类型: 字节流和字符流

字节输入流的 最大父类 InputStream(抽象类)

字节输出流的 最大父类 OutputStream(抽象类)

# 字节输入流

InputStream是抽象类,一般使用其子类FileInputStream.

使用FileInputStream字节流来读取图像等二进制文件.

输入是指: 从本地磁盘将数据读取到java程序

ps: 为了演示出效果,暂时是使用字节流读取字符了,工作时不应如此!

ps: 以后会使用工具框架POI进行读取输出excel,word,pdf等

|  |
| --- |
| **package** com.zhiyou100.io;  **import** java.io.FileInputStream; **import** java.io.IOException;  */\*\*  \** ***@author*** *QiuShiju  \** ***@date*** *2021/1/5  \** ***@desc*** *\*/* **public class** Demo1FileInputStream {   **public static void** main(String[] args) **throws** IOException {  */\*\*  \* FileInputStream(File file)  通过打开一个到实际文件的连接来创建一个 FileInputStream，该文件通过文件系统中的 File 对象 file 指定。  FileInputStream(String name)  通过打开一个到实际文件的连接来创建一个 FileInputStream，该文件通过文件系统中的路径名 name 指定。   \*/  // 通过本地a文件获得一个字节输入流* FileInputStream fis = **new** FileInputStream(**"E:\\A\\a.txt"**);   *// 读取下一个字节 // int i1 = fis.read( ); // System.out.println((char)i1);   // 读取下一个字节 // int i2 = fis.read( ); // System.out.println((char)i2);   // 读取下一个字节 // int i3 = fis.read( ); // System.out.println((char)i3);   // 读取下一个字节 : 返回-1 即说明读完 // int i4 = fis.read( ); // System.out.println(i4);   // OK* **while**(**true**) {  **int** read = fis.read( );  **if**(read != -1) {  System.***out***.println((**char**)read );  }**else**{  **break**;  }  }   **int** val = -1;  **while**((val = fis.read( )) != -1) {  System.***out***.println((**char**)val );  }   *// 关流* fis.close();   } } |

# 字节输出流

OutputStream是抽象父类,一般使用其子类FileOutputStream.

输出是指: 从java程序中将数据写出到本地磁盘.

|  |
| --- |
| **package** com.zhiyou100.io;  **import** java.io.FileNotFoundException; **import** java.io.FileOutputStream; **import** java.io.IOException;  */\*\*  \** ***@author*** *QiuShiju  \** ***@date*** *2021/1/5  \** ***@desc*** *\*/* **public class** Demo2FileOutputStream {   **public static void** main(String[] args) **throws** IOException {  */\*\*  \* 输入流的文件必须存在,否则抛异常  \* 输出流的文件,可以不存在,会自动创建出!  \*/  // 通过文件获得一个输出流  /\*\*  \* 默认的构造方法,是不允许追加数据,即每次都会将之前的数据覆盖  \* 如果想要将数据追加在之前文件的后面,就需要设置append参数为true  \*/ // FileOutputStream fos = new FileOutputStream("E:\\A\\b.txt");* FileOutputStream fos = **new** FileOutputStream(**"E:\\A\\b.txt"**,**true**);   *// 输出一个字节* fos.write(67);   *// 关流* fos.close();   } } |

# 练习

复制文件

将E:\A\a.txt 文件 复制并粘贴到 E:\A\C\acopy.txt

思路: 边读边写

|  |
| --- |
| **package** com.zhiyou100.io;  **import** java.io.FileInputStream; **import** java.io.FileNotFoundException; **import** java.io.FileOutputStream; **import** java.io.IOException;  */\*\*  \** ***@author*** *QiuShiju  \** ***@date*** *2021/1/5  \** ***@desc*** *练习复制文件  \*/* **public class** Demo3 {   **public static void** main(String[] args) **throws** IOException {  *// 获得时间* **long** begin = System.*currentTimeMillis*( );   *// 输入流* FileInputStream fis = **new** FileInputStream(**"E:\\A\\baby.mp3"**);   *// 输出流* FileOutputStream fos = **new** FileOutputStream(**"E:\\A\\C\\gan.mp3"**);   *// 边读边写* **int** val = -1;  **while**((val = fis.read()) != -1) {  fos.write(val);  }   fis.close();  fos.close();   *// 获得时间* **long** end = System.*currentTimeMillis*();  System.***out***.println(end-begin );   }   **private static void** copy1() **throws** IOException {  *// 获得时间* **long** begin = System.*currentTimeMillis*( );   *// 输入流* FileInputStream fis = **new** FileInputStream(**"E:\\A\\a.txt"**);   *// 输出流* FileOutputStream fos = **new** FileOutputStream(**"E:\\A\\C\\acopy.txt"**);   *// 边读边写* **int** val = -1;  **while**((val = fis.read()) != -1) {  fos.write(val);  }   fis.close();  fos.close();   *// 获得时间* **long** end = System.*currentTimeMillis*( );  System.***out***.println(end-begin );  }  } |

问题:

当读取大文件时,非常慢!

怎么解决?使用下方的高效缓冲流

# 高效流/缓冲流

缓冲字节输出流 BufferedOutputStream

缓冲字节输入流 BufferedInputStream

|  |
| --- |
| **package** com.zhiyou100.io;  **import** java.io.\*;  */\*\* 101387  \** ***@author*** *QiuShiju  \** ***@date*** *2021/1/5  \** ***@desc*** *缓冲区输入输出字节流  \*/* **public class** Demo4Buffered {   **public static void** main(String[] args) **throws** IOException {  *// 获得时间* **long** begin = System.*currentTimeMillis*( );   FileInputStream fis = **new** FileInputStream(**"E:\\A\\baby.mp3"**);  *// 创建缓冲输入流* BufferedInputStream bis = **new** BufferedInputStream(fis);   *// 创建缓冲输出流* BufferedOutputStream bos = **new** BufferedOutputStream(**new** FileOutputStream(**"E:\\A\\C\\copy.mp3"**));   *// 边读边写* **int** val = -1;  **while**((val = bis.read()) != -1) {  bos.write(val);  }  *// 关流* bis.close();  bos.close();   *// 获得时间* **long** end = System.*currentTimeMillis*( );  System.***out***.println(end-begin );  }  } |

复制同样一个大文件,时间大大缩减!

原理:

缓冲输入输出,在其内部都默认创建了一个长度8192的byte数组,用来读写.

# 字符输入流

Reader是字符输入流的抽象父类,一般使用其子类FileReader

|  |
| --- |
| **package** com.zhiyou100.io;  **import** java.io.FileNotFoundException; **import** java.io.FileReader; **import** java.io.IOException;  */\*\*  \** ***@author*** *QiuShiju  \** ***@date*** *2021/1/5  \** ***@desc*** *字符输入  \*/* **public class** Demo5FileReader {   **public static void** main(String[] args) **throws** IOException {   *// 创建字符输入流* FileReader fr = **new** FileReader(**"E:\\A\\a.txt"**);   **int** val = -1;  **while**((val = fr.read()) != -1) {  System.***out***.println((**char**)val );  }  fr.close();  }  } |

# 字符输出流

|  |
| --- |
| **package** com.zhiyou100.io;  **import** java.io.FileWriter; **import** java.io.IOException;  */\*\*  \** ***@author*** *QiuShiju  \** ***@date*** *2021/1/5  \** ***@desc*** *\*/* **public class** Demo6FileWriter {   **public static void** main(String[] args) **throws** IOException {   */\*\*  \* 向磁盘输出文件,文件不存在时会自动创建出  \*/ // FileWriter writer = new FileWriter("E:\\A\\c.txt");* FileWriter writer = **new** FileWriter(**"E:\\A\\c.txt"**,**true**);   writer.write(98);  writer.write(**"bigdata"**);  writer.close();  } } |

Writer是字符输出流的抽象父类,一般使用子类FileWriter

# 缓冲字符输入流

BufferedReader

# 缓冲字符输出流

BufferedWriter

|  |
| --- |
| **package** com.zhiyou100.io;  **import** java.io.\*;  */\*\*  \** ***@author*** *QiuShiju  \** ***@date*** *2021/1/5  \** ***@desc*** *缓冲输入输出字符流  \*/* **public class** Demo7Buffered {   **public static void** main(String[] args) **throws** IOException {   *// 创建缓冲字符输入流* BufferedReader br = **new** BufferedReader(**new** FileReader(**"E:\\A\\a.txt"**));   *// 创建缓冲字符输出流* BufferedWriter bw = **new** BufferedWriter(**new** FileWriter(**"E:\\A\\C\\acopy.txt"**));    *// 读取一个字符 // int val = br.read();   // 读取一行,以换行符为读取一行的标志.但是,并没有读取都换行符* String line = **null**;  **while** ((line = br.readLine()) != **null**) {  */\*\*  \* 所有,在向外输出时,没有换行符.导致结果都在一行.  \*/* bw.write(line);  *// 手动向外写换行符 // bw.write("\r\n");* bw.newLine();  }    br.close();  bw.close();   }    **private static void** 缓冲输入() **throws** IOException {  *// 创建缓冲字符输入流* BufferedReader br = **new** BufferedReader(**new** FileReader(**"E:\\A\\a.txt"**));   *// 读取一个字符 // int val = br.read();   // 读取一行* String line = **null**;  **while** ((line = br.readLine()) != **null**) {  System.***out***.println(line );  }   br.close();  } } |

# 作业

使用字符流把a文件中的数据转换后写到b文件中 大写转换为小写 小写转换为大写 删除数字

public static void changeFile(String filea,String fileb)