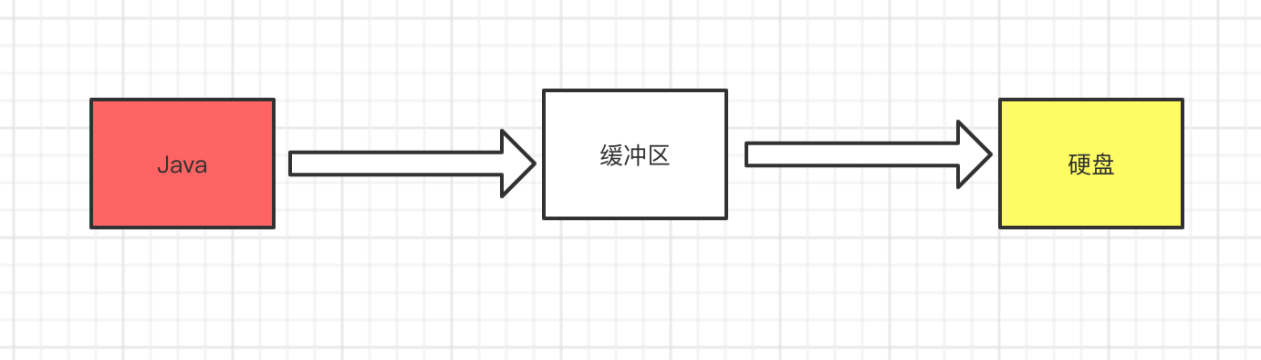
缓冲流

无论是字节流还是字符流，使用的时候都会频繁访问硬盘，对硬盘是一种损伤，同时效率不高，如何解决?

可以使用缓冲流，缓冲流自带缓冲区，可以一次性从硬盘中读取部分数据存入缓冲区，再写入内存，这样就可以有效减少对硬盘的直接访问。



缓冲流属于处理流，如何来区分节点流和处理流?

1、节点流使用的时候可以直接对接到文件对象 File

2、处理流使用的时候不可以直接对接到文件对象 File，必须要建立在字节流的基础上才能创建。

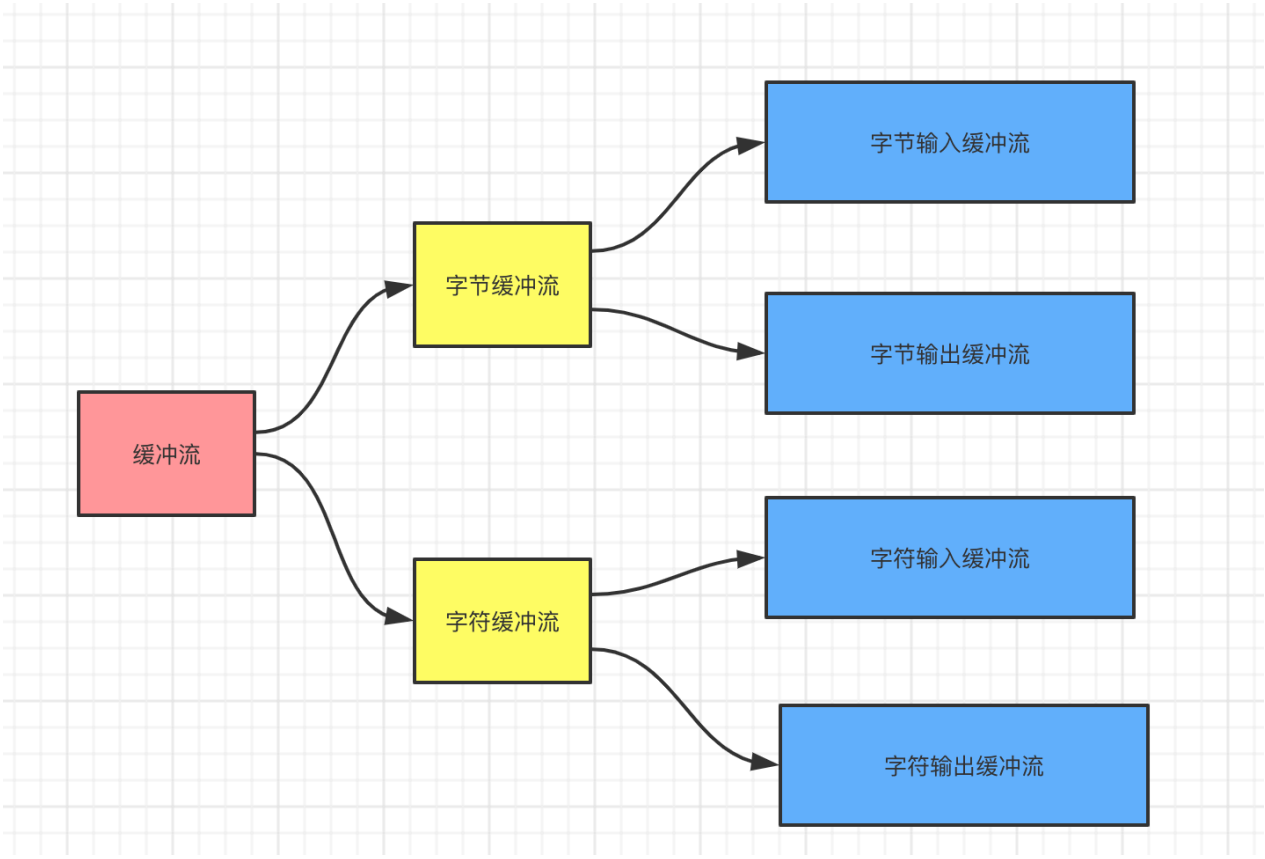
```
public class Test {
    public static void main(String[] args) throws Exception {
        File file = null;
        //节点流
        InputStream inputStream = new FileInputStream(file);
        OutputStream outputStream = new FileOutputStream(file);

        Reader reader = new FileReader(file);
        Writer writer = new FileWriter(file);

        //处理流
        InputStreamReader inputStreamReader = new InputStreamReader(file);
        OutputStreamWriter outputStreamWriter = new OutputStreamWriter(file);

    }
}
```

缓冲流又可以分为字节缓冲流和字符缓冲流，按照方向再细分，又可以分为字节输入缓冲流和字节输出缓冲流，以及字符输入缓冲流和字符输出缓冲流。



> 字节输入缓冲流

```
package com.southwind.demo;

import java.io.BufferedInputStream;
import java.io.FileInputStream;
import java.io.InputStream;

public class Test {
    public static void main(String[] args) throws
Exception {
        //1、创建节点流
```

```
        InputStream inputStream = new
FileInputStream("/Users/southwind/Desktop/test.txt")
;
        //2、创建缓冲流
        BufferedInputStream bufferedInputStream =
new BufferedInputStream(inputStream);
//         int temp = 0;
//         while ((temp =
bufferedInputStream.read())!=-1){
//             System.out.println(temp);
//         }

        byte[] bytes = new byte[1024];
        int length =
bufferedInputStream.read(bytes,10,10);
        System.out.println(length);
        for (byte aByte : bytes) {
            System.out.println(aByte);
        }

        bufferedInputStream.close();
        inputStream.close();
    }
}
```

> 字符输入缓冲流

readLine 方法

```
package com.southwind.demo;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.Reader;
```

```
public class Test2 {
    public static void main(String[] args) throws
Exception {
        //1、创建字符流（节点流）
        Reader reader = new
FileReader("/Users/southwind/Desktop/test.txt");
        //2、创建缓冲流（处理流）
        BufferedReader bufferedReader = new
BufferedReader(reader);
        String str = null;
        int num = 0;
        System.out.println("***start***");
        while ((str =
bufferedReader.readLine())!=null){
            System.out.println(str);
            num++;
        }
        System.out.println("***end***,共读取
了"+num+"次");
        bufferedReader.close();
        reader.close();
    }
}
```

> 字节输出缓冲流

```
package com.southwind.demo2;

import java.io.BufferedOutputStream;
import java.io.FileOutputStream;
import java.io.OutputStream;
```

```
public class Test {
    public static void main(String[] args) throws
Exception {
        OutputStream outputStream = new
FileOutputStream("/Users/southwind/Desktop/test2.txt
");
        BufferedOutputStream bufferedOutputStream =
new BufferedOutputStream(outputStream);
        String str = "由于在开发Oak语言时，尚且不存在运行
字节码的硬件平台，所以为了在开发时可以对这种语言进行实验研究，
他们就在已有的硬件和软件平台基础上，按照自己所指定的规范，用软
件建设了一个运行平台，整个系统除了比C++更加简单之外，没有什么
大的区别。";
        byte[] bytes = str.getBytes();
//        for (byte aByte : bytes) {
//            bufferedOutputStream.write(aByte);
//        }
        bufferedOutputStream.write(bytes,9,9);
        bufferedOutputStream.flush();
        bufferedOutputStream.close();
        outputStream.close();
    }
}
```

> 字符输出缓冲流

```
package com.southwind.demo2;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.Writer;

public class Test2 {
```

```
    public static void main(String[] args) throws
Exception {
        Writer writer = new
FileWriter("/Users/southwind/Desktop/test2.txt");
        BufferedWriter bufferedWriter = new
BufferedWriter(writer);
//        String str = "由于在开发语言时尚且不存在运行字节
码的硬件平台，所以为了在开发时可以对这种语言进行实验研究，他们
就在已有的硬件和软件平台基础上，按照自己所指定的规范，用软件建
设了一个运行平台，整个系统除了比C++更加简单之外，没有什么大的
区别。";
//        bufferedWriter.write(str,5,10);
        char[] chars = {'J','a','v','a'};
//        bufferedWriter.write(chars,2,1);
        bufferedWriter.write(22902);
        bufferedWriter.flush();
        bufferedWriter.close();
        writer.close();
    }
}
```

输入流没有 flush 方法，但不代表它没有缓冲流，输出流是有 flush 方法的，实际开发中在关闭输出缓冲流之前，需要调用 flush 方法。

序列化和反序列化

序列化就是将内存中的对象输出到硬盘文件中保存。

反序列化就是相反的操作，从文件中读取数据并还原成内存中的对象。

> 序列化

1、实体类需要实现序列化接口，Serializable

```
package com.southwind.entity;

import java.io.Serializable;

public class User implements Serializable {
    private Integer id;
    private String name;
    private Integer age;

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getAge() {
        return age;
    }

    public void setAge(Integer age) {
        this.age = age;
```

```
    }

    @Override
    public String toString() {
        return "User{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", age=" + age +
                '}';
    }

    public User(Integer id, String name, Integer
age) {
        this.id = id;
        this.name = name;
        this.age = age;
    }
}
```

2、实体类对象进行序列化处理，通过数据流写入到文件中，ObjectOutputStream。

```
package com.southwind.demo3;

import com.southwind.entity.User;

import java.io.File;
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.io.OutputStream;

public class Test {
```

```
    public static void main(String[] args) throws
Exception {
        User user = new User(1,"张三",22);
        OutputStream outputStream = new
FileOutputStream("/Users/southwind/Desktop/obj.txt")
;
        ObjectOutputStream objectOutputStream = new
ObjectOutputStream(outputStream);
        objectOutputStream.writeObject(user);
        objectOutputStream.flush();
        objectOutputStream.close();
        outputStream.close();
    }
}
```

> 反序列化

```
package com.southwind.demo3;

import com.southwind.entity.User;

import java.io.FileInputStream;
import java.io.InputStream;
import java.io.ObjectInputStream;

public class Test2 {
    public static void main(String[] args) throws
Exception {
        InputStream inputStream = new
FileInputStream("/Users/southwind/Desktop/obj.txt");
        ObjectInputStream objectInputStream = new
ObjectInputStream(inputStream);
```

```
        User user = (User)
objectInputStream.readObject();
        System.out.println(user);
        objectInputStream.close();
        inputStream.close();
    }
}
```