上級プログラミング2（第1回）

工学部　情報工学科
木村昌臣

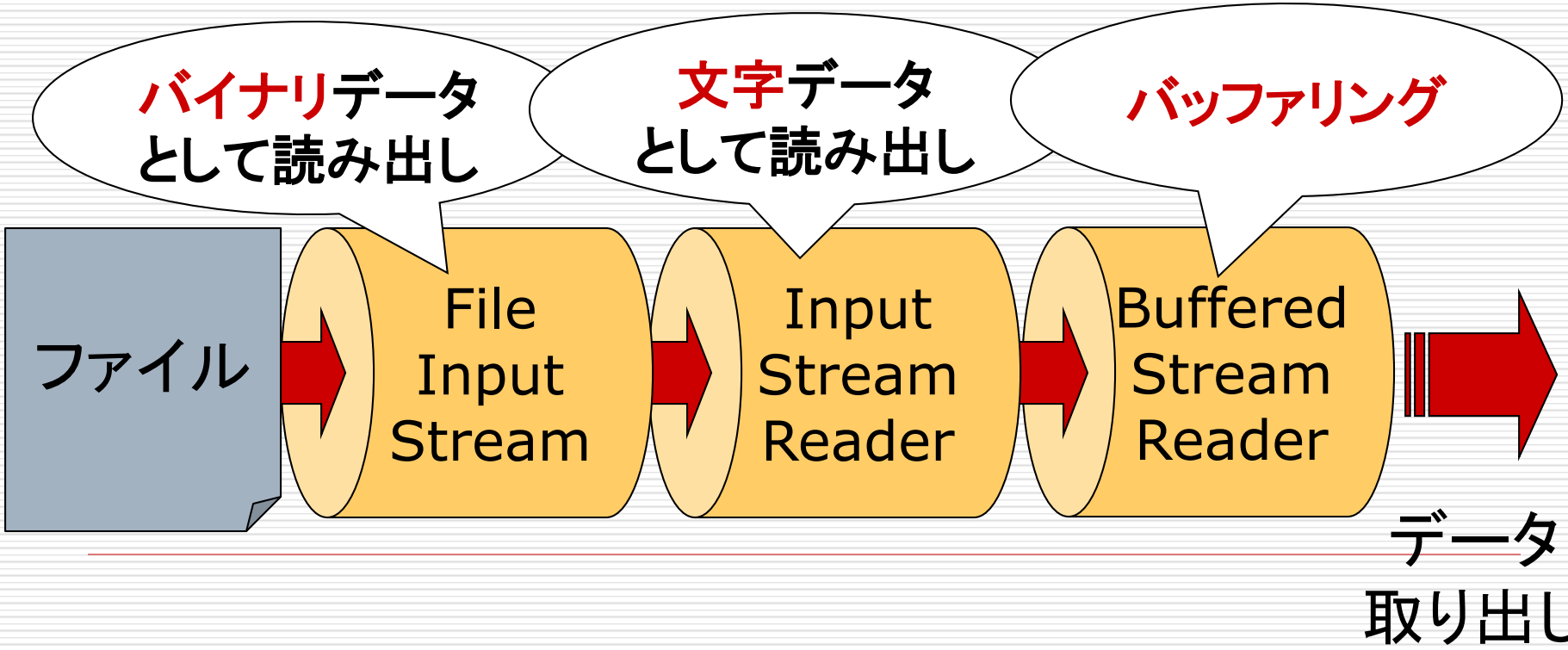
今日のテーマ

- 入出力に関わるプログラムの作り方
- ネットワークプログラミングの続き
  - TCPの場合のプログラム
    - 先週のプログラムの詳細な説明
    - URLクラス
  - サーバープログラムの例

データ入出力プログラミングの復習

テキストの読み込み関係のクラス

テキストからデータを読み込むときには、
通常、三段構えで行う



テキストファイルの読み出し

```
String line="";

FileInputStream fi
            = new FileInputStream("S
InputStreamReader is
            =new InputStreamReader(fi,"SJIS");
BufferedReader br
            =new BufferedReader(is);
while((line=br.readLine())!=null){
            System.out.println(line);
}
```

バイナリデータ
として読み出し

文字データ
として読み出し

バッファリング

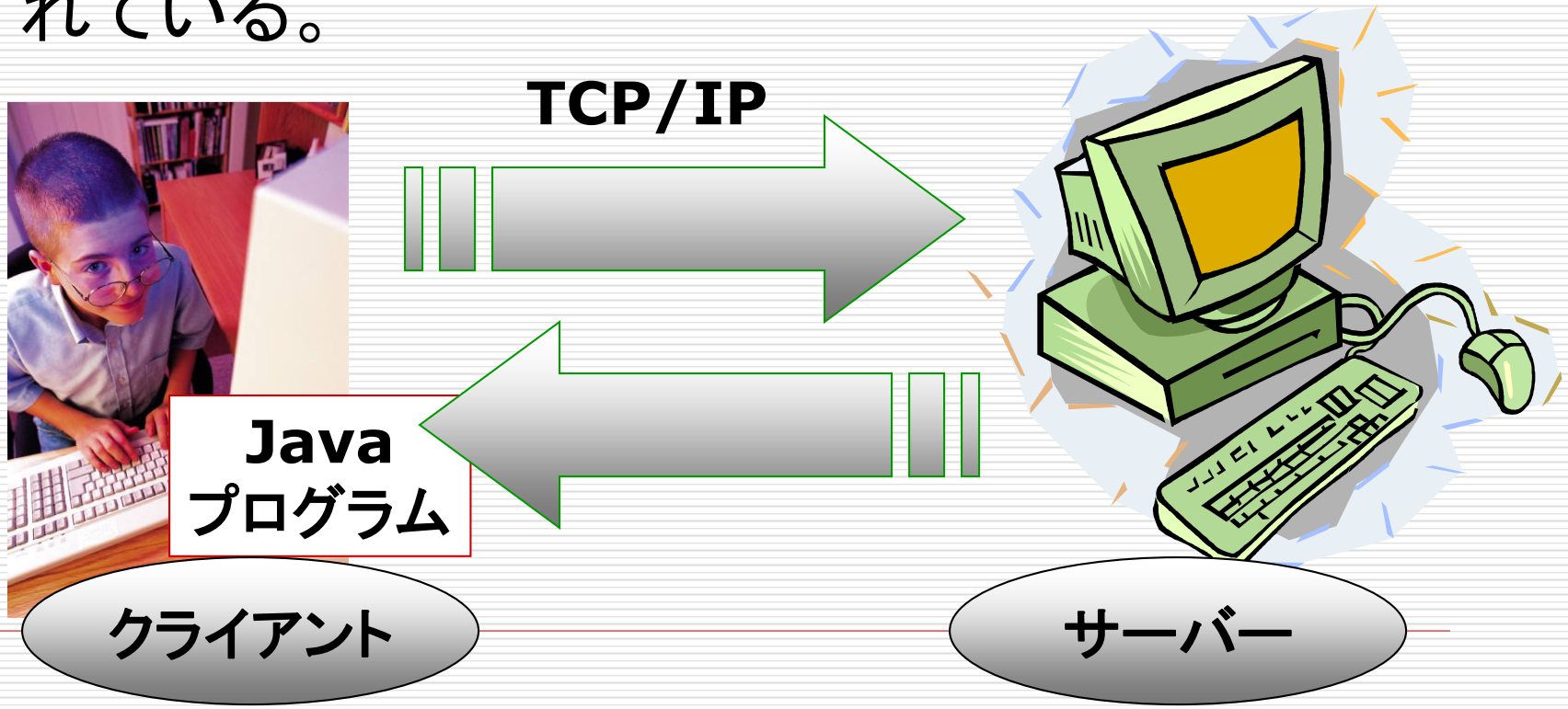
出力については

- 入力と考え方は同じ（順番は逆）
  - Input ⇔ Output
  - Reader ⇔ Writer
- PrintWriterなどを使うとき、出力先に書き込み内容を反映するにはflushメソッドの使用が必要になる場合がある。

ネットワークプログラミング

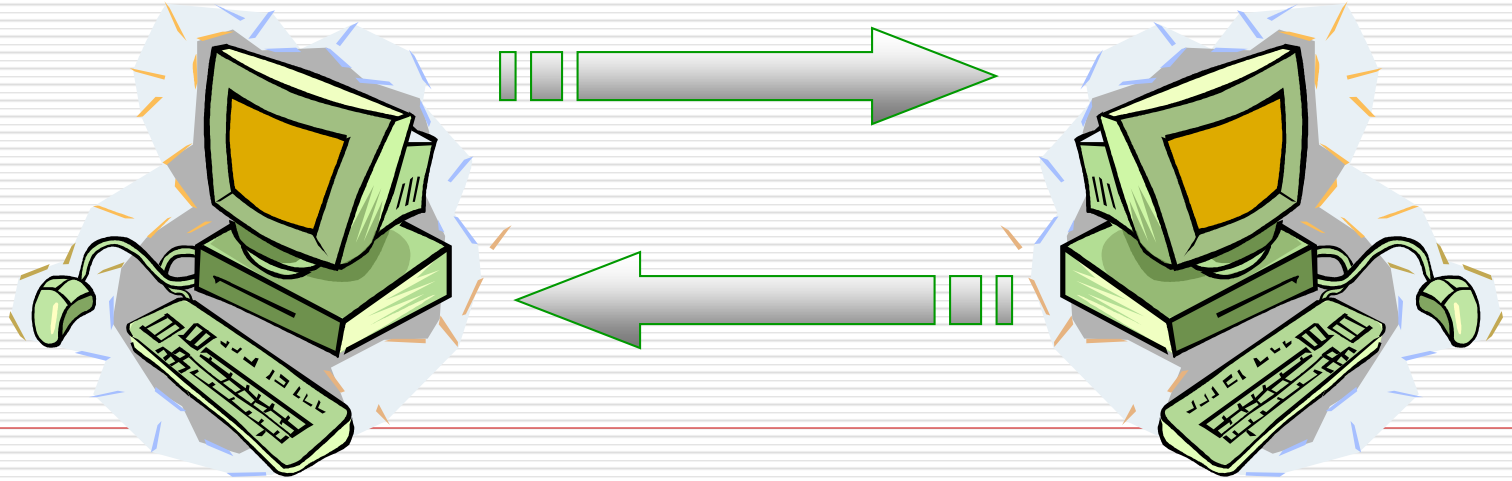
Javaでネットワークプログラミング

- Javaには、ネットワークを使ったプログラムを簡単に作成できるよう、様々なクラスが用意されている。



ネットワークの復習(1)

- プロトコル
  - ネットワークを使ってデータをやり取りする手順
  - TCP/IPが主流
  - HTTP,FTP,TELNET,SMTP,POP3,NTPなど



ネットワークの復習(2)

- ホスト名
  - ネットワーク上のマシンの名前
- IPアドレス
  - マシンに振られるネットワーク上の番号(住所)
  - 192.168.1.5など32bitで表す
- ポート
  - データをやり取りするための補助番号
  - IPアドレスとポート番号がペアでデータをやりとり
  - プロトコルと対応づけされている(HTTPでは80)

ネットワークの復習(3)

- DNS
  - ホスト名とIPアドレスの対応情報を保存
  - ホスト名からIPアドレスを引いたり、IPアドレスからホスト名を引くことが可能
  - UnixやWindowsではhostsファイルで代用可能
- TCPとUDP
  - TCP:サーバーとの接続を確立する(データの到達が保証される)
  - UDP:データは送りっぱなし(データの到達は保証されない)

HTTP

GET /index.html HTTP/1.0

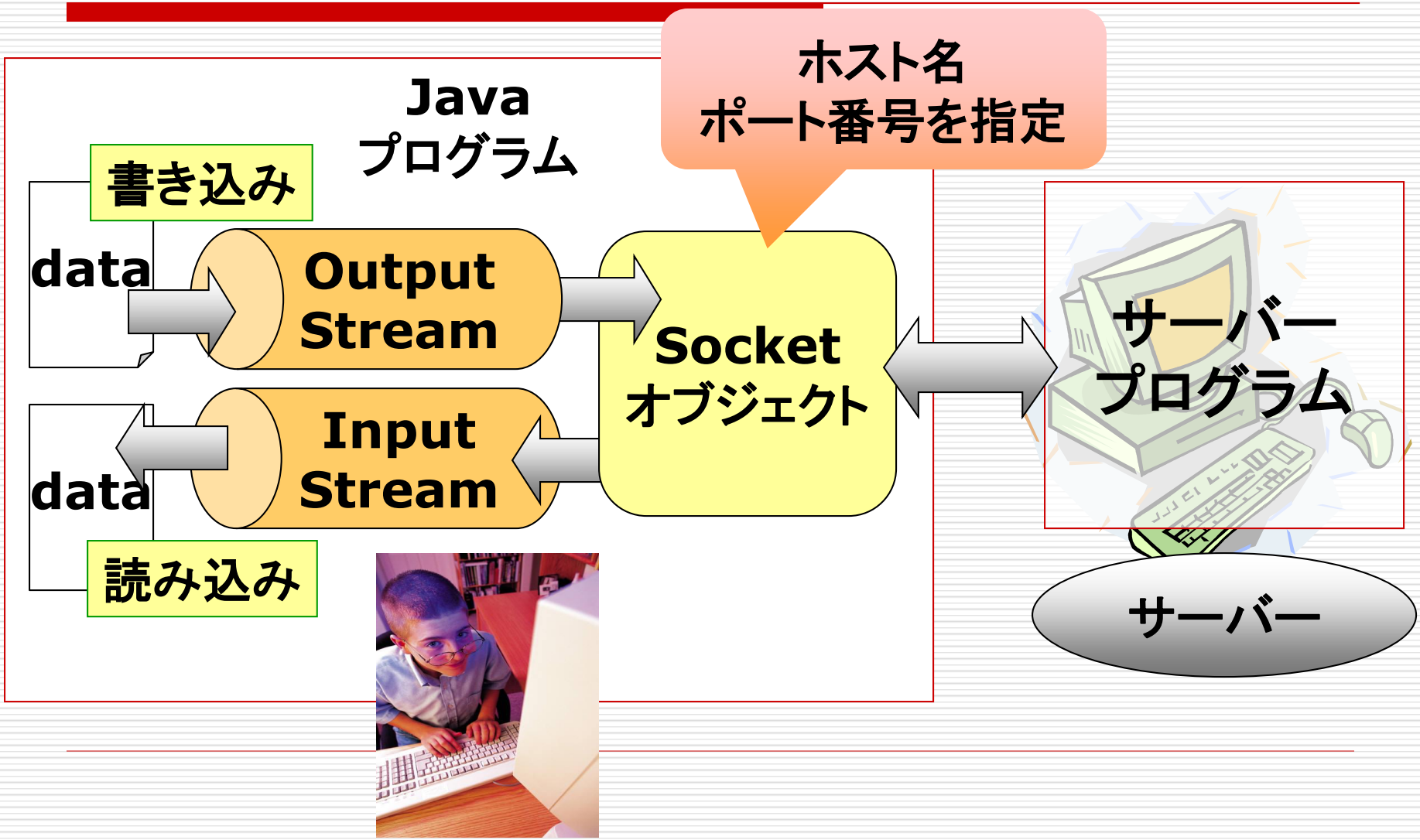
要求

HTML

<HTML>
<HEAD>芝浦工業大学</HEAD>
<BODY>芝浦工大は、…</BODY>
</HTML>

Web サーバー

Javaでのネットワークプログラミングの考え方



HTTPでデータをWebサーバから取得するプログラム

```
import java.io.*;
import java.net.*;
```

必要なパッケージをインポート

```
class TCP_Http {

    public static void main(String[] args) {

        String serverName;
        //サーバー名
        serverName="orion.data.ise.shibaura-it.ac.jp";

        // ポート番号
        int portNo=80;

        InetAddress sAdr;
        StringBuffer sBuff=new StringBuffer();
```

HTTPのポート番号=80

【続く】

【続き】

```
try {
    // ホスト名からIPアドレスに変換
    sAdr=InetAddress.getByName(serverName);
    // Socketの生成
    Socket skt = new Socket(sAdr,portNo);
    // サーバーへのデータ送信用ライター
    BufferedWriter outS
        =new BufferedWriter(
            new OutputStreamWriter(skt.getOutputStream()));
    // サーバーからのデータ受信用リーダー
    BufferedReader inS
        = new BufferedReader(
            new InputStreamReader(skt.getInputStream()));
    //
    String[] msg={"GET /index.html HTTP/1.0¥r¥n","¥r¥n"};

    for(int i=0; i<msg.length; i++){
            outS.write(msg[i]);
    }
    outS.flush();
```

IPアドレスを取得

ソケットとリーダーやライターの作成

サーバーへ送るメッセージ

サーバーへ送信

【続く】

【続き】

サーバーからのデータ受信

ソケット等のクローズ

例外処理

```
            String tmpString;
            while ((tmpString=inS.readLine())!=null){
                    sBuff.append(tmpString+"¥n");
            }
            System.out.println(sBuff.toString());
            outS.close();
            inS.close();
            skt.close();
        } catch (UnknownHostException e) {

            e.printStackTrace();
        } catch (IOException e) {

            e.printStackTrace();
        }
    }
}
```

実行結果抜粋(www.shibaura-it.ac.jp)

```
HTTP/1.1 200 OK
Date: Tue, 18 Oct 2005 13:34:08 GMT
Server: Apache
Last-Modified: Thu, 13 Oct 2005 01:14:32 GMT
ETag: "126763-66f0-434db4f8"
Accept-Ranges: bytes
Content-Length: 26352
Connection: close
Content-Type: text/html
```

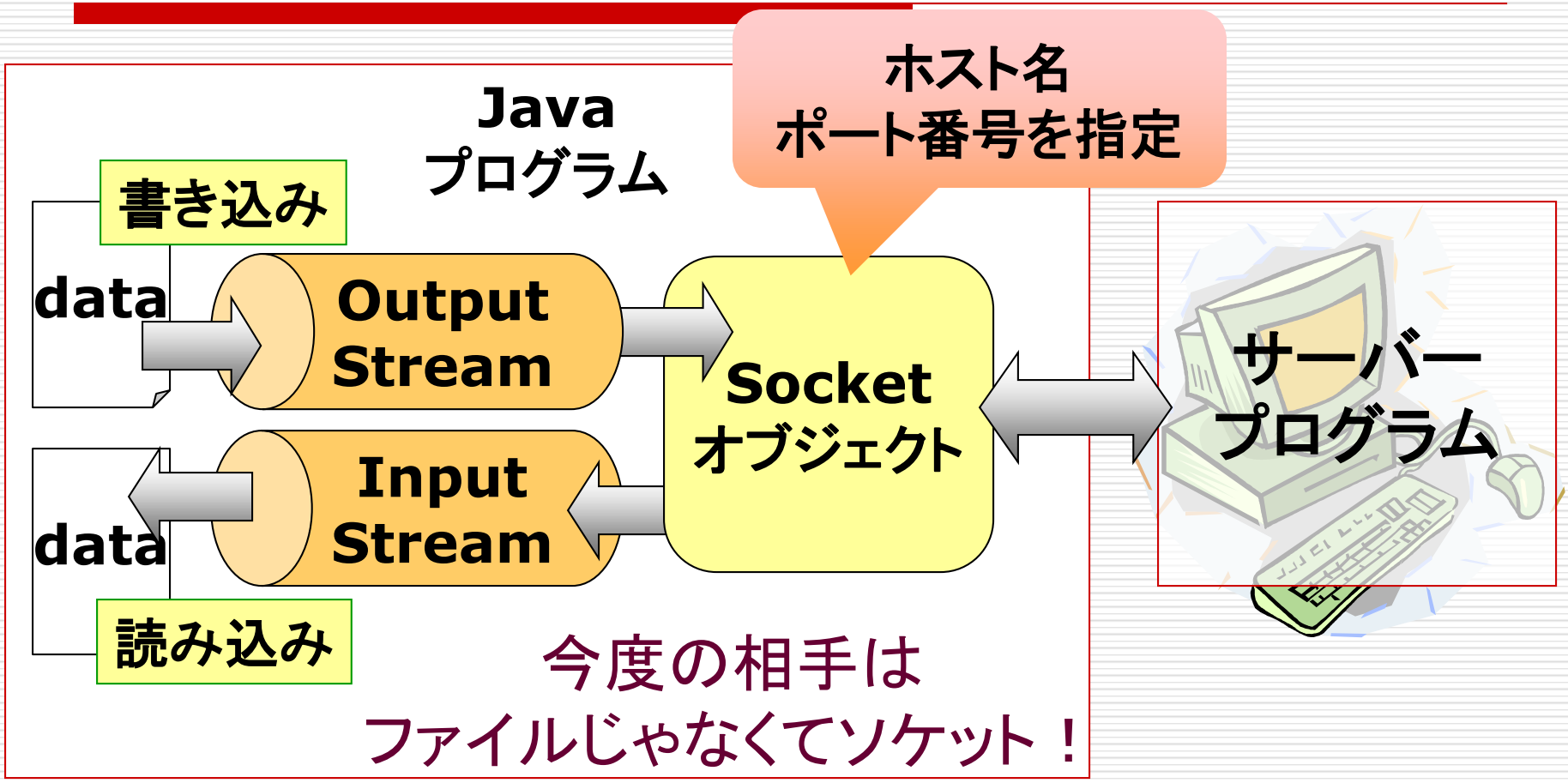
HTTPの応答ヘッダ

HTML

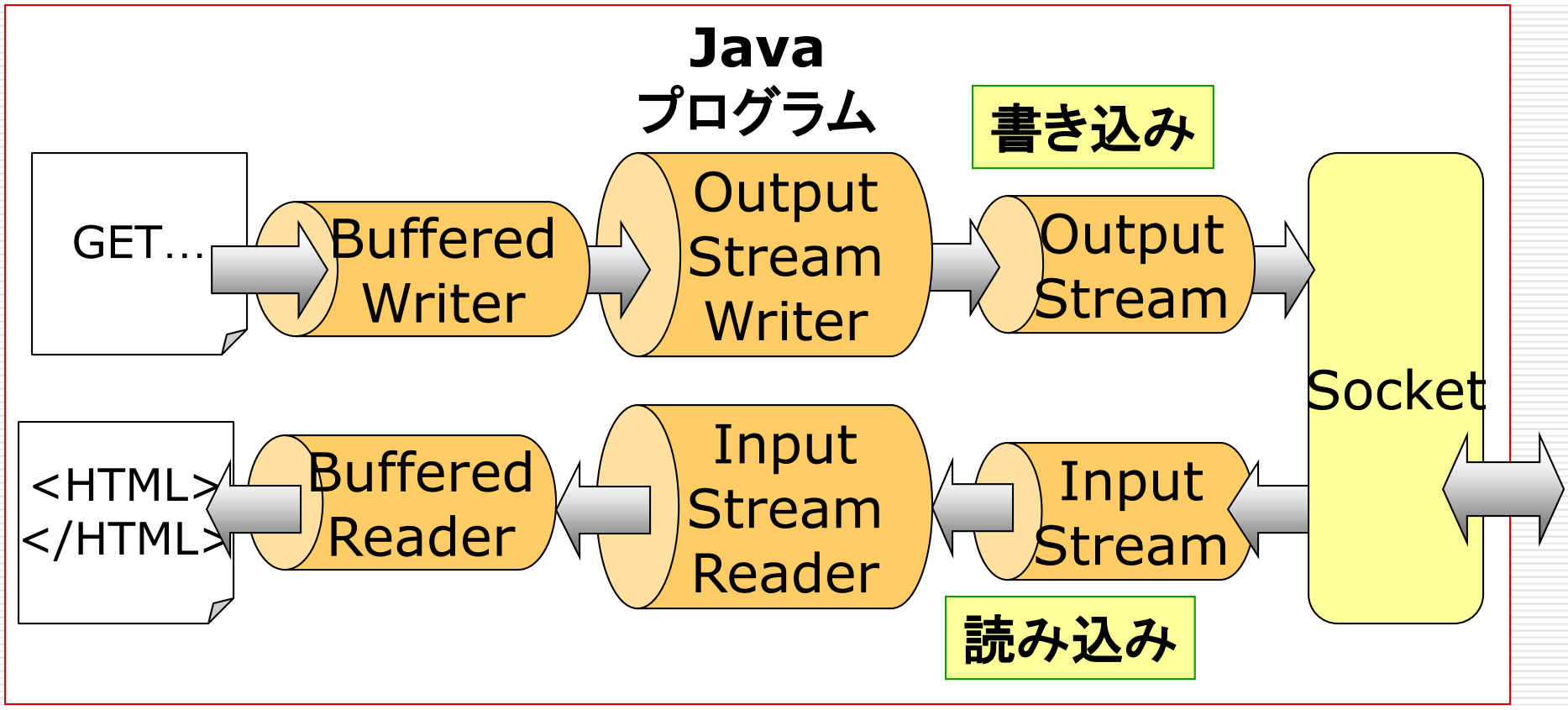
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML><HEAD><TITLE>芝浦工業大学</TITLE>
<META http-equiv=Content-Type content="text/html; charset=Shift_JIS">
<STYLE>
A {
    FONT-SIZE: 10pt; TEXT-DECORATION: none
```

・・・・・（以下略）

Javaでのネットワークプログラミングの考え方

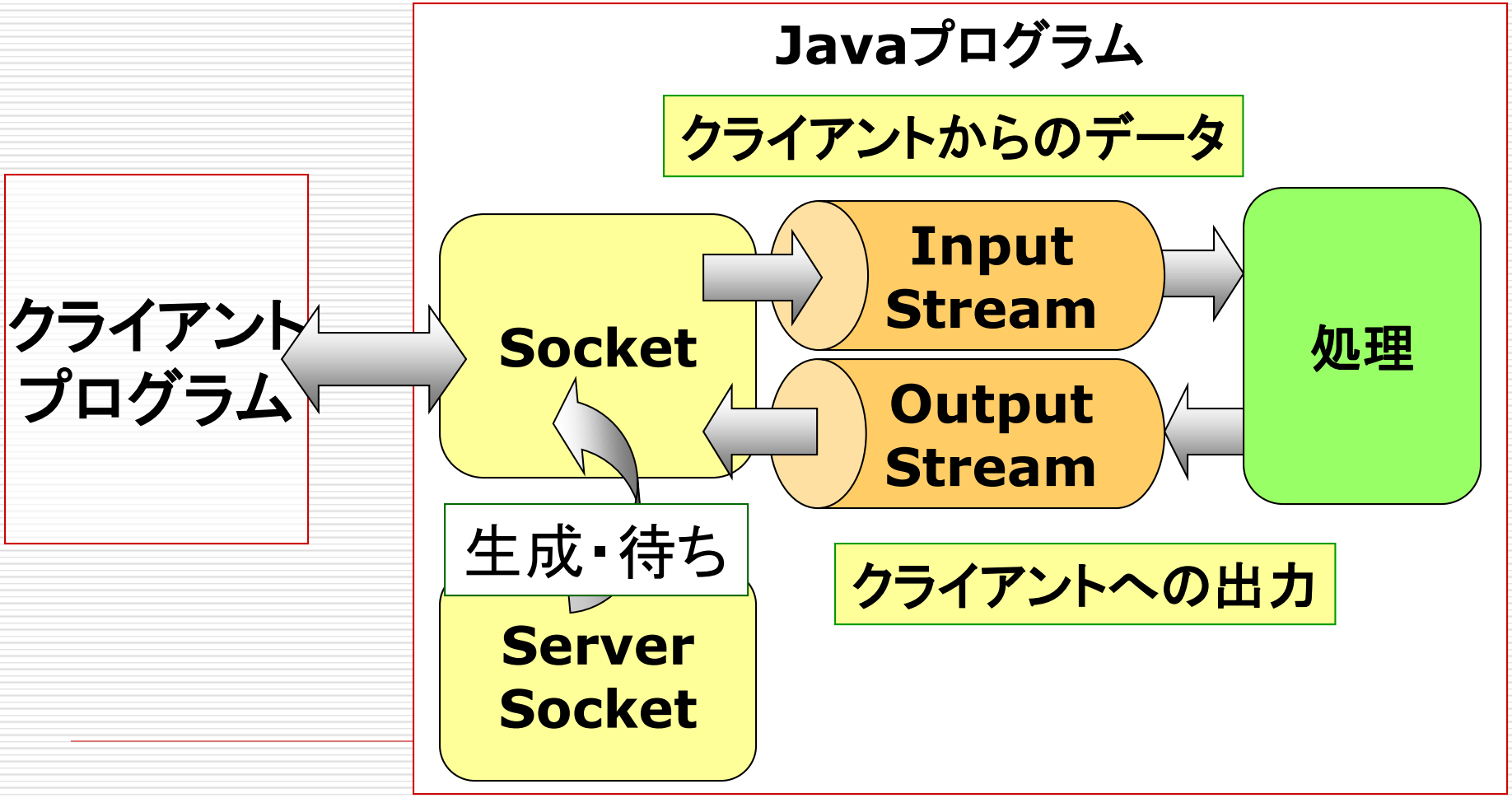


このプログラムでは


Java
プログラム
書き込み
GET…
Buffered
Writer
Output
Stream
Writer
Output
Stream
Socket
<HTML>
</HTML>
Buffered
Reader
Input
Stream
Reader
Input
Stream
読み込み

サーバープログラムを作る

サーバープログラムの構造



サーバー側で動くプログラムを作る

1. ServerSocketオブジェクトを作る

ポート番号のみ！

```
ServerSocket srvSock
        =new ServerSocket(10007);
```

2. acceptメソッドで待ち。
   クライアントからのアクセスが来たら
   クライアントと通信するための
   Socketオブジェクトが作られる。

```
Socket skt=srvSock.accept();
```

何故、ServerSocketクラスには ポート番号しか指定しない？

クライアントプログラム
からのアクセス待ち

誰と通信するかわからない！
わかるのは、ポート番号だけ

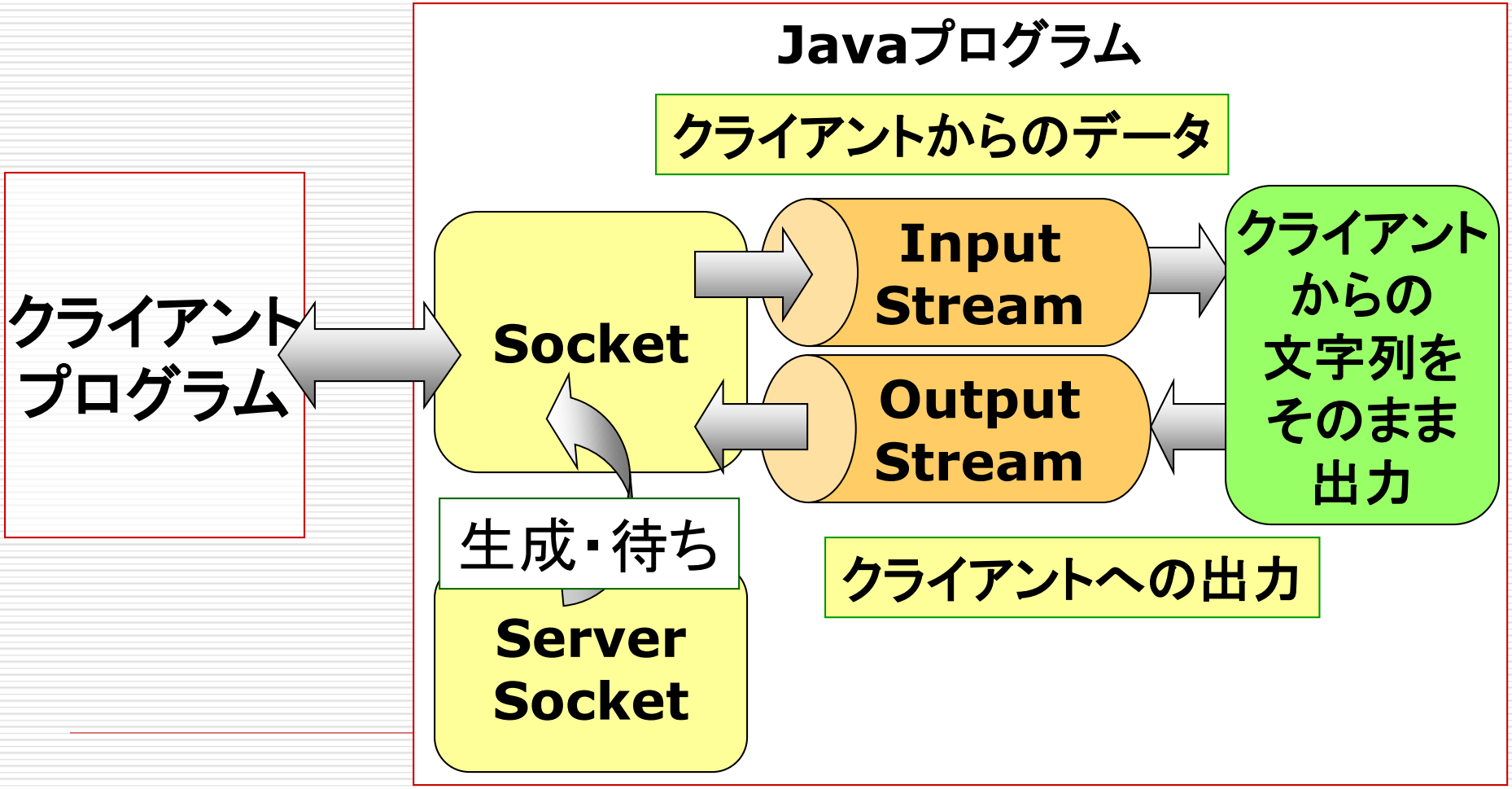
何故、ServerSocketクラスにはポート番号しか指定しない？

クライアントプログラムからのアクセスきた！
Socketオブジェクトでやりとりしよう！

クライアント
プログラム

サーバー
プログラム

例)echoサーバー


Javaプログラム
クライアントからのデータ
クライアント
プログラム
Socket
Input
Stream
Output
Stream
クライアント
からの
文字列を
そのまま
出力
生成・待ち
Server
Socket
クライアントへの出力

TCPサーバープログラム（抜粋）

```
ServerSocket srvSock=new ServerSocket(10007);
boolean flg=true;
while(flg==true){
        Socket skt=srvSock.accept();

        ・・・・(省略。inSは入力ストリームのバッファリーダ、
                outSは出力ストリームのプリントライタ)・・・・

        String msg=inS.readLine();

        ・・・・(省略。Msgから出力文字列tmpStringを生成、
                        Msgが空文字列ならflgをfalseに)・・・・

        outS.println(tmpString);
        inS.close();  outS.close();
        skt.close();
}
```

クライアントプログラム（抜粋）

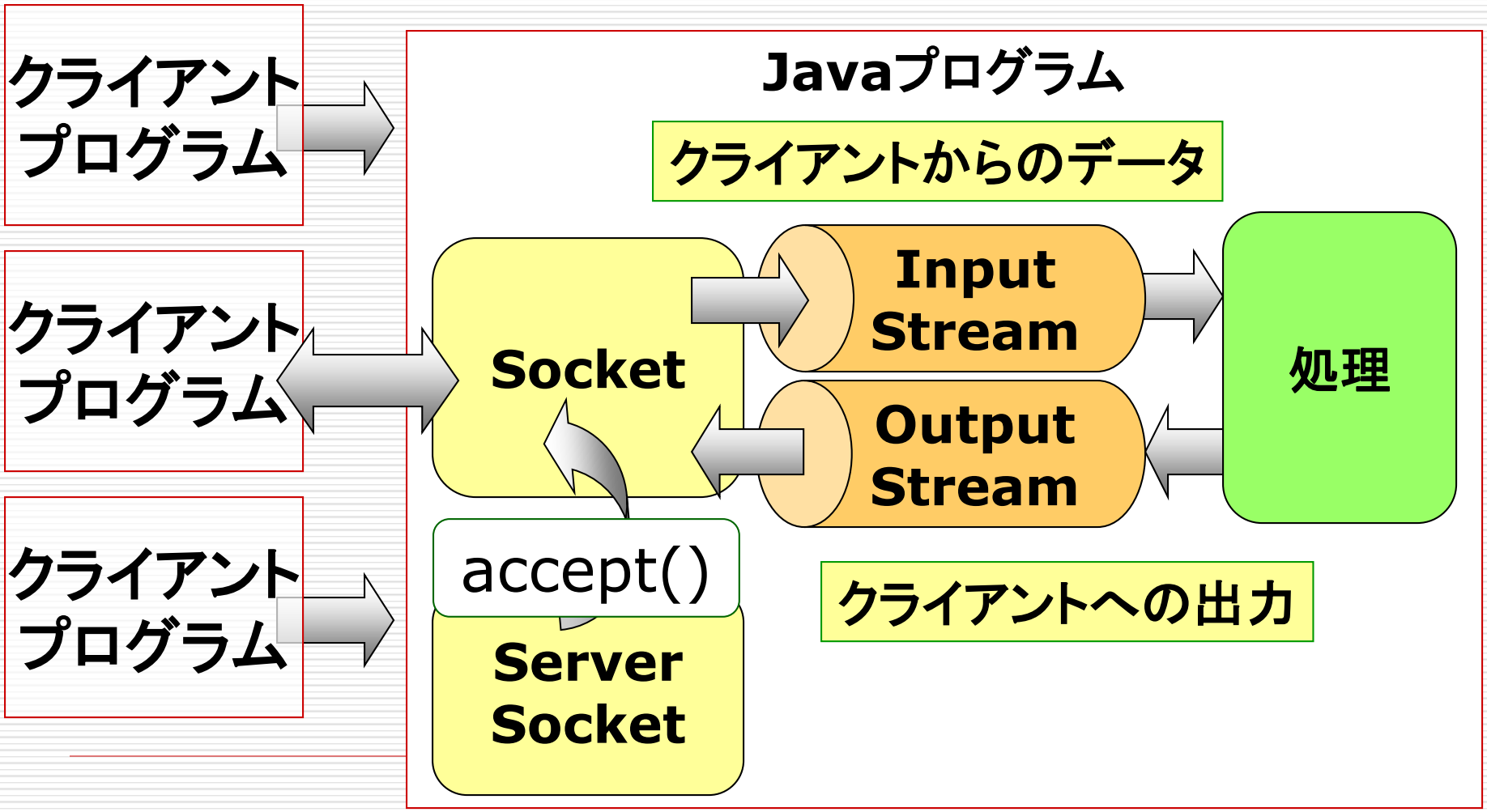
```
Socket skt=new Socket("127.0.0.1",10007);

        ・・・・(省略。inSは入力ストリームのバッファリーダ、
                    outSは出力ストリームのプリントライタ)・・・・

outS.println(args[0]);
while((tmpString=inS.readLine())!=null){
                    System.out.println(tmpString);
}

inS.close();
outS.close();
skt.close();
```

複数のクライアントプログラムがアクセスしてきたら？



実は

- さっきのプログラムは、1対1の通信しかサポートできない
  - サーバー内の処理が短ければ、見かけ上複数のクライアントに対応できる
  - しかし、時間がかかる処理の場合は、1つのクライアントの処理がおわるまで他のクライアントは待たされる
- これを解決するには、複数の処理を同時に実行できる仕組みが必要