



پر نامه سازی پیش رفته

ورودی و خروجی

مهدی مصطفی زاده

سرفصل مطالب

- جریان‌ها (جویبارها)ی ورودی و خروجی
- برنامه‌نویسی شبکه
- خواندن/نوشتن از/در پرونده به شکل ترتیبی
- خواندن/نوشتن از/در پرونده به شکل تصادفی
- دستیابی به فراداده‌ی پوشه‌ها و پرونده‌ها با استفاده از رده‌ی File



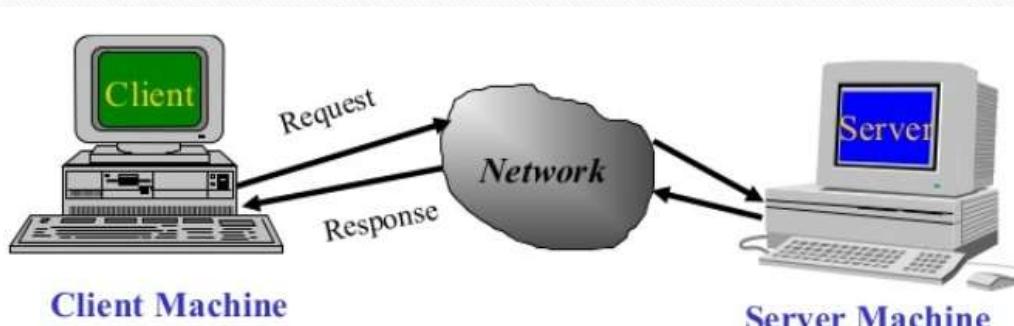
جريان‌های ورودی و خروجی

- در پسیاری از موارد لازم است پرنامه‌ها پا محیط اطراف خود حرف پزند (داده‌ها) پی را دریافت و یا ارسال کنند.
- محیط اطراف شامل: صفحه کلید، نمایشگر، پرونده‌ها (Files)، شبکه و ...
- چاو، مستقل از منبع داده، هر چریان از داده‌ها (پایتهای) به درون پرنامه را یک `InputStream` در نظر می‌گیرد.
- چاو، مستقل از مقصد داده، هر چریان از داده‌ها (پایتهای) به پیرون پرنامه را یک `OutputStream` در نظر می‌گیرد.

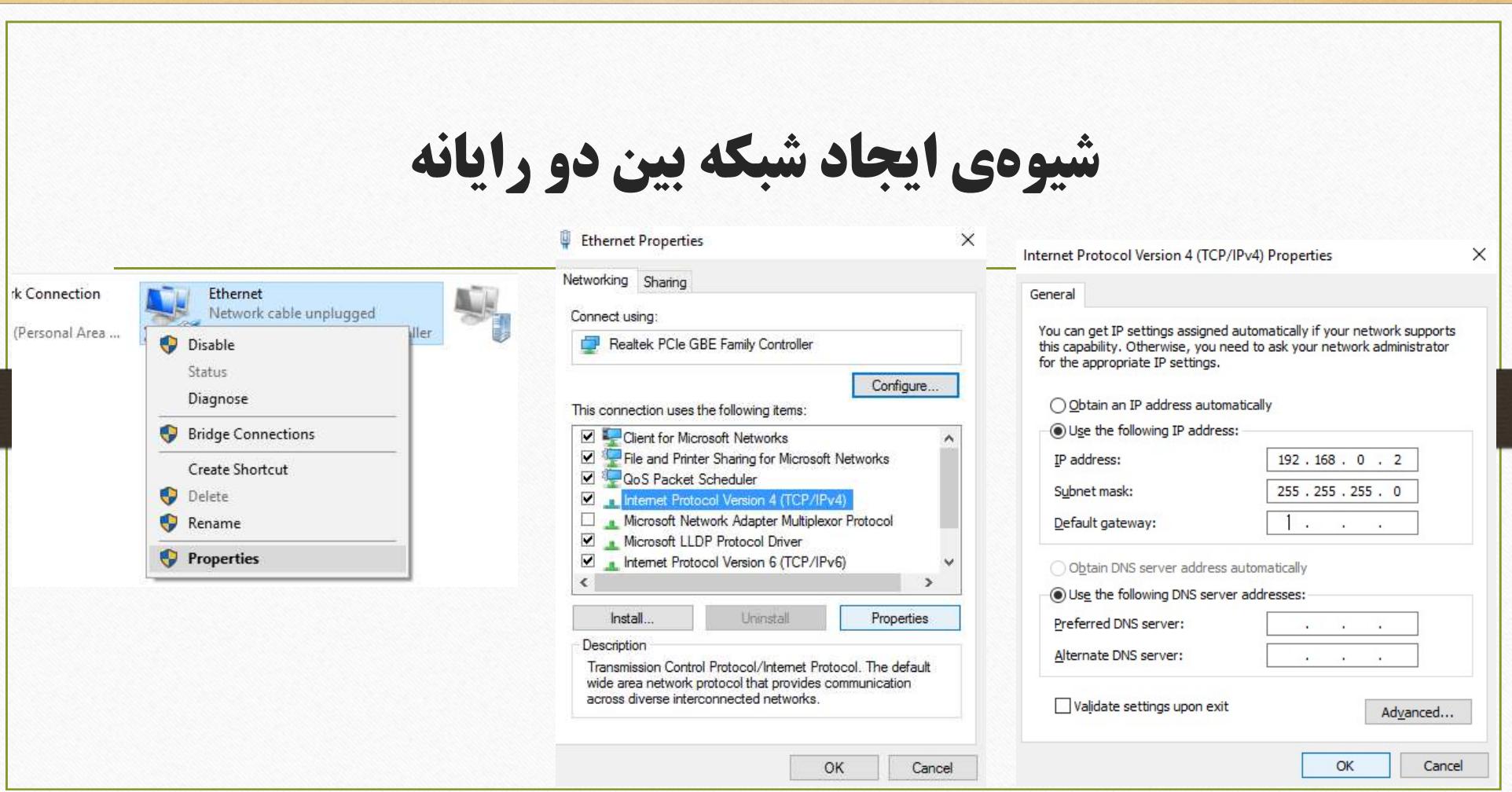


مقدمه‌ای بر شبکه‌های رایانه‌ای

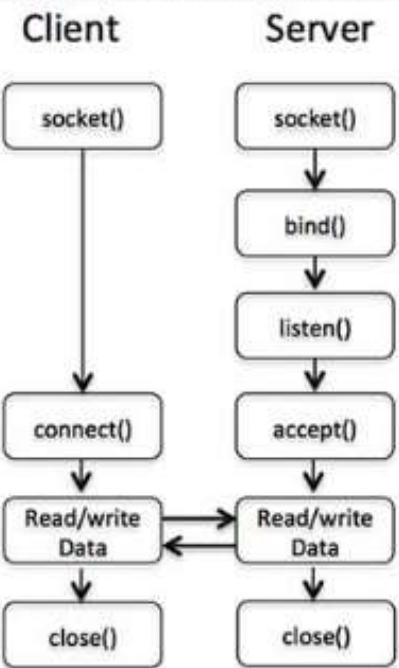
- هر دستگاه در شبکه‌ی رایانه‌ای، یک آدرس یکتا (IP Address) دارد (مثلاً ۱۹۲.۱۶۸.۰.۱۰).
- معروف‌ترین و پرکاربردترین مدل شبکه، مدل مشتری-خدمت‌گزار (Client-Server) است.



شیوه‌ی ایجاد شبکه بین دو رایانه



مدل برنامه‌نویسی مشتری-خدمتگزار در جاوا



- ردیف انتزاعی پر روی درگاه (port) مورد استفاده در ارتباط است.

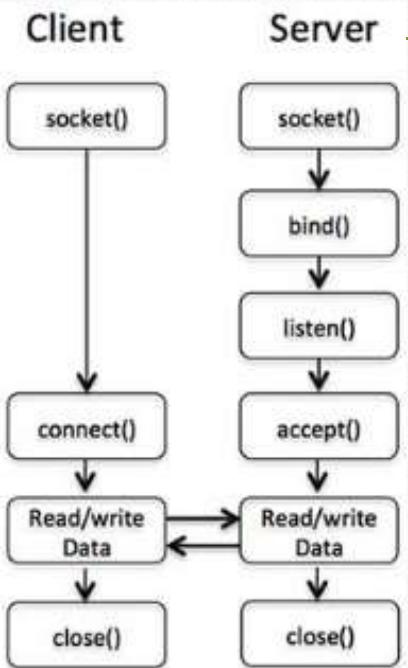
- خدمتگزار (Server) با مشخص کردن یک شماره‌ی درگاه، شعایی از چنین ServerSocket (ییجاد می‌کند).

- خدمتگزار پر فراخوانی متد accept، روی درگاه مشخص شده پر گوش می‌شیند (Listen).

- مشتری (Client)، پر ایجاد یک شعی Socket با IP و Port معین در خدمتگزار، درخواست پرقداری ارتباط را پر خدمتگزار می‌فرستد.



مدل برنامه‌نویسی مشتری-خدمتگزار در جاوا (ادامه)



- پیرو درخواست مشتری، متدهای accept و read در Socket را پردازی کرداند.

- در این نقطه، هم مشتری و هم خدمتگزار، یک شیء Socket دارند که به هم متصل‌اند.

- هریک از طریقین، می‌توانند از شیء `InputStream` مخود یک `OutputStream` و یک `Socket` پیگیرند و اطلاعات را به واسطه‌ی آن مبادله کنند.



مثالی از برنامه‌ی خدمتگزار

```
public static void main(String[] args) throws IOException {
    System.out.println("Server");
    ServerSocket serverSocket = new ServerSocket(9090);
    Socket socket = serverSocket.accept();
    OutputStream outputStream = socket.getOutputStream();
    InputStream inputStream = socket.getInputStream();
    Formatter formatter = new Formatter(outputStream);
    Scanner scanner = new Scanner(inputStream);
    Scanner keyboardScanner = new Scanner(System.in);
    while(true) {
        System.out.println(scanner.nextLine());
        formatter.format(keyboardScanner.nextLine() + "\n");
        formatter.flush();
    }
}
```



مثالی از برنامه‌ی مشتری

```
public static void main(String[] args) throws IOException {
    System.out.println("client");
    Socket socket = new Socket("localhost", 9090);
    OutputStream outputStream = socket.getOutputStream();
    InputStream inputStream = socket.getInputStream();
    Formatter formatter = new Formatter(outputStream);
    Scanner scanner = new Scanner(inputStream);
    Scanner keyboardScanner = new Scanner(System.in);
    while(true) {
        formatter.format(keyboardScanner.nextLine() + "\n");
        formatter.flush();
        System.out.println(scanner.nextLine() + "\n");
    }
}
```



چندخی‌سازی برنامه‌ها

```
public static void main(String[] args) throws IOException {
    System.out.println("Server");
    ServerSocket serverSocket = new ServerSocket(9090);
    Socket socket = serverSocket.accept();
    new MyThread(socket).start();
    InputStream inputStream = socket.getInputStream();
    Scanner scanner = new Scanner(inputStream);
    while(true) {
        System.out.println(scanner.nextLine());
    }
}
```



چندنخی‌سازی برنامه‌ها (ادامه)

```
public class MyThread extends Thread {  
    OutputStream outputStream;  
  
    public MyThread(Socket socket) throws IOException {  
        this.outputStream = socket.getOutputStream();  
    }  
  
    @Override  
    public void run() {  
        Formatter formatter = new Formatter(outputStream);  
        Scanner keyboardScanner = new Scanner(System.in);  
        while(true) {  
            formatter.format(keyboardScanner.nextLine() + "\n");  
            formatter.flush();  
        }  
    }  
}
```



کار با پرونده‌ها (Files)

- **پرونده (File):** دنباله‌ای از پایتی‌ها که در حافظه‌ی ذخیره‌سازی چانپی (دیسک سخت و ...) مان، ذخیره شده‌اند.
- **انواع پرونده:**
- **پایندی (Binary):**
- **متنی (Text):**



خواندن/نوشتن از/در پرونده

- روال کلی خواندن/نوشتن از/در پرونده عبارت است از:
- پار کردن پرونده (Open)
- گرفتن یک `InputStream` (پای خواندن) و یا یک `OutputStream` (پای نوشتن)
- خواندن/نوشتن از/در پرونده با استفاده از چویپار (و احتمالاً به کمک رده‌های کمکی مثل `FileReader`, `Formatter`, `Scanner`)
- پستن پرونده (Close)



مثال: کپی یک تصویر

```
public static void main(String[] args) {
    try(InputStream inputStream = new FileInputStream("1.jpg");
        OutputStream outputStream = new FileOutputStream("2.jpg")) {
        int nextByte;
        while((nextByte = inputStream.read()) != -1) {
            outputStream.write(nextByte);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```



مثال: پردازش پروندهای متنی با Formatter و Scanner

• مثال: شمارش نویسه‌های هر خط

```
public static void main(String[] args) {
    try(InputStream inputStream = new FileInputStream("sample.txt");
        OutputStream outputStream = new FileOutputStream("len.txt");
        Scanner scanner = new Scanner(inputStream);
        Formatter formatter = new Formatter(outputStream)) {
        int lineNumber = 1;
        while(scanner.hasNext()) {
            formatter.format("Line # %d: %d\n",
                            lineNumber++, scanner.nextLine().length());
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```



مثال: پردازش پروندهای متنی با FileReader و FileWriter

• مثال: حذف نقاط

```
public static void main(String[] args) {
    try(FileReader fileReader = new FileReader("sample.txt");
        FileWriter fileWriter = new FileWriter("sampleWithoutDot.txt")) {
        int nextCharacter;
        while((nextCharacter = fileReader.read()) != -1) {
            if(nextCharacter == '.')
                nextCharacter = ' ';
            fileWriter.write(nextCharacter);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```



مثال: پردازش پروندها به شکل تصادفی

```
public static void main(String[] args) {
    try(RandomAccessFile randomAccessFile =
        new RandomAccessFile("sample.txt", "rw")) {
        randomAccessFile.writeChars("ABC");
        randomAccessFile.seek(2);
        randomAccessFile.writeChar('T');
        randomAccessFile.seek(2);
        System.out.println(randomAccessFile.getFilePointer()); // 2
        System.out.println(randomAccessFile.readChar()); // T
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```



دستیابی به فراداده‌ی (metadata) پوشه‌ها و پرونده‌ها

```
public static void main(String[] args) {
    File file = new File("sampleWithoutDot.txt");
    System.out.println(file.exists()); // true

    /* F:\Mahdi\workspace\Test\sample.txt */
    System.out.println(file.getAbsolutePath());

    System.out.println(file.isDirectory()); // false
    System.out.println(file.length()); // 62
    System.out.println(file.delete()); // true
    System.out.println(file.exists()); // false
}
```

