

HELSINGIN YLIOPISTO  
HELSINGFORS UNIVERSITET  
UNIVERSITY OF HELSINKI

# Introduction to Java Network Programming

Jussi Kangasharju





## Java Network Programming

- Network programming in Java in general much easier than in C...
- ...except some advanced things which are harder ☹
  - Setting socket options, no `select()`-call
  - But threads help with missing `select()`
- Java supports both TCP and UDP sockets
- Many different ways to read/write sockets
  - Differentiates between text and binary ☹
  - Often several correct ways to handle socket
  - TIMTOWTDI: There Is More Than One Way To Do It



## Using TCP Sockets

- Client side:

```
Socket sock = new Socket(host, port);
```

- String host = host to contact, int port = port
- Host can also be InetAddress instead of String

- Server side

```
ServerSocket sock = new ServerSocket(port);
```

- Listen for incoming connections

```
Socket client = sock.accept();
```



## Using UDP Sockets

- Same for client and server

```
DatagramSocket sock = new DatagramSocket ();
```

- For server, give port number as argument
- Send packets with **send ()**
- Receive packets with **receive ()**
- UDP packets implemented in **DatagramPacket**-class



## Reading and Writing TCP Sockets

- Socket has `InputStream` and `OutputStream`
- Need to wrap other streams around them
- Some wrappers implement buffers
  
- Java has many different I/O Streams
  - See Java API for others (e.g., reading files)
- Relevant for sockets:
  - `InputStreamReader`, `OutputStreamWriter`
  - `BufferedReader`, `BufferedWriter`
  - `DataInputStream`, `DataOutputStream`



## Reading from a Socket

- Typical code:

```
InputStream is = socket.getInputStream();
```

```
InputStreamReader isr = new InputStreamReader(is);
```

```
BufferedReader br = new BufferedReader(isr);
```

- Read text by calling `br.readLine()`

- Can be used only for reading text!



## Writing to a Socket

- Typical code

```
OutputStream os = socket.getOutputStream();
```

```
OutputStreamWriter osw = new OutputStreamWriter(os);
```

```
BufferedWriter bw = new BufferedWriter(osw);
```

- Write by calling one of many `write()` -functions

- See the different classes for different possibilities

- Strings need to be converted to bytes with `getBytes()`

- Can also write directly to `OutputStream`

- **BufferedWriter only for text output!**



## DataInputStream

- **DataInputStream** can read binary data from socket
- Also can send primitive data types
- Typical code

```
InputStream is = socket.getInputStream();
```

```
DataInputStream dis = new DataInputStream(is);
```

- Read binary data with **read()** (see API for details)
- Bonus functionality: Read text with **readLine()**
  - But **DataInputStream.readLine()** is deprecated ☹️





## DataOutputStream

- **DataOutputStream** can be used to write

- Typical code:

```
OutputStream os = socket.getOutputStream();
```

```
DataOutputStream dos = new DataOutputStream(os);
```

- **DataOutputStream** can also write text and binary

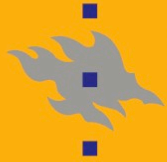
- Has `writeBytes()` -function

- no need for `String.getBytes()`



## Differences Between Output Streams?!?

- What is the difference between `DataOutputStream` and normal `OutputStream` wrapped with `BufferedWriter`?
- **Answer:** There is no difference in practice
- Some subtleties:
  - Possible problems with conversion between 8-bit and 16-bit characters (e.g., `DataInputStream.readLine()`)
  - Possible text/binary data issues
  - Possible problems with buffering (use `flush()`)
  - `dos.writeBytes(str)` VS. `bw.write(str.getBytes())`
- No “correct” way, use either as long as it works
  - Be careful not to get confused!



HELSINGIN YLIOPISTO  
HELSINGFORS UNIVERSITET  
UNIVERSITY OF HELSINKI

# Assignment

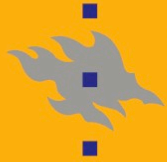
Java Network Programming





## Assignment Details

1. TCP client and server
  2. Simple Web server
  3. Web server improvements (+ optionals)
- <http://www.cs.helsinki.fi/u/jakangas/Teaching/CBU/lab1.html>



HELSINGIN YLIOPISTO  
HELSINGFORS UNIVERSITET  
UNIVERSITY OF HELSINKI

**Questions?**

