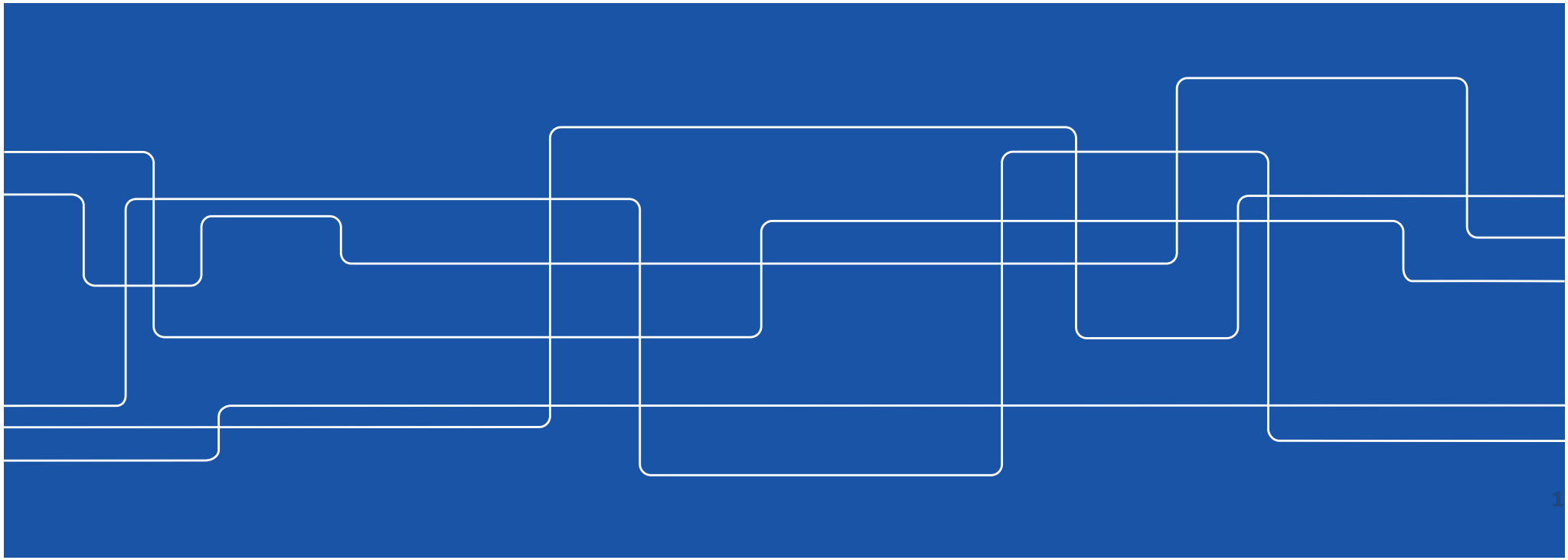
Socket Programming Project

IK1203
Peter Sjödin



Socket Programming Project

- Learn about
  - Socket programming in Java
  - TCP client and server
  - HTTP server
  - Concurrent servers

Project Organization

- A series of tasks
  - A task typically involves implementing a client or server in Java
  - Tasks build upon each other
- Four to five tasks
- About a week to complete each task
- Submit solution in Canvas

Submitting Solutions

- Upload a ZIP archive with your source code
- We use tools to evaluate your solutions
  - Therefore, you solution must be submitted exactly according to the instruction
  - Otherwise it can’t be graded, and you fail
- You get templates with the correct structure from us
  - Don’t change it!

Supervision

- Supervision classes scheduled together with the labs
- Sign up for supervision slots in Canvas
- Discuss, ask questions, get technical help, …

- You must ign up no later than 18:00 the day before the supervision slot to be guaranteed a seat.
- If there are no students sign up, we reserve the right to cancel the slot.
- If you change your mind, please be respectful and cancel your reservation.

Task 1: TCPAsk

- A client that contacts a TCP server (of any kind), and prints out whatever the server returns

```
$ java TCPAsk time.nist.gov 13
time.nist.gov:13 says:

58128 18-01-10 23:18:34 00 0 0  40.2 UTC(NIST) *
```

- You get the source code to TCPAsk
- But most of the work is done by the class TCPClient
- Your job is to implement TCPClient

Task 2: HTTPEcho Server

- A server that returns whatever it receives from the client, as an HTTP response.
- For instance, connect to it from your web browser, and you should see something like this:

```
GET / HTTP/1.1
Host: localhost:8888
Upgrade-Insecure-Requests: 1
Accept:
text/html,application/xhtml+xml,application/xml;q=
0.9,*/*;q=0.8User-Agent: Mozilla/5.0 (Macintosh;
Intel Mac OS X 10_13_2) AppleWebKit/604.4.7
(KHTML, like Gecko) Version/11.0.2 Safari/604.4.7
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

Task 3: HTTPAsk Server

- An HTTP server that uses your TCPClient class from Task 1
  - Specify hostname and port as “query” data to HTTPAsk server
- Server returns response from server, as an HTTP response

```
GET /ask?hostname=time.nist.gov&port=13 HTTP/1.1
```

Task 4: Multi-threaded HTTPAsk Server

- A concurrent server, that can handle many clients at the same time.
  - The HTTPAsk server in Task 3 only needs to deal with one client at a time.

Further Tasks

- More later…

General Instruction

- There are probably many Java libraries that could do the work for you
- See the instructions and examples for what libraries you may use
- If you want to use anything else, you need permission
  - Ask on the forum (and expect “no” for an answer)
  - You are here to learn!
- You may collaborate, but each students submits his/her own solution
- We will check for duplicates/plagiarism

Evaluation

- Your submission needs to meet a number of requirement

- It should:
  1. Have the correct format
  2. Compile without errors
     - We don’t fix your code, not even simple mistakes
  3. Pass a number of test scenarios
  4. Pass manual code review

- If you do not pass, you may get a second chance, if:
  - You have made a serious attempt
  - You have submitted in time

Test Scenarios

- We run your programs and do a number of functional tests
- Some tests described in the instructions (but not necessarily all)
- Output from tests included in feedback
  - Can be terse, and may not always pinpoint the problems exactly
  - May be of some help in debugging
  - It is part of your challenge to test and fix the code
    - You do the debugging!
    - We do not point out exactly what is wrong

Running the Test

- Once a day, for the last three days before the due date, we will check that your submission is complete and compiles correctly.
  - This does not include running the actual test
- Two days before the due date, we will make a pre-run of the automated test, and you will receive the output as feedback
  - Simpler version of the final tests
- After the due date, we will do both automated checks and manual evaluation.
- In this way, hopefully you will not fail because of simple mistakes in the submission
- It is also an incentive for you not to wait with the submission until the last minute

Frequently Asked Question

- **Question:** Can you run the tests more often?
- **Answer:** No, because:
  - The tests are for evaluation and grading, not for debugging
  - It would encourage unsound development habits – brute-force trial-and-error
    - Also known as “whack-a-mole programming”.