Programming Assignment I

Computer Networks

Zheng Cao
Sriramkumar Balasubramanian
Li Yan

Introduction

- We prefer Java (Download Eclipse)
- Easy for socket programming and multi-threading
- Can use C as well

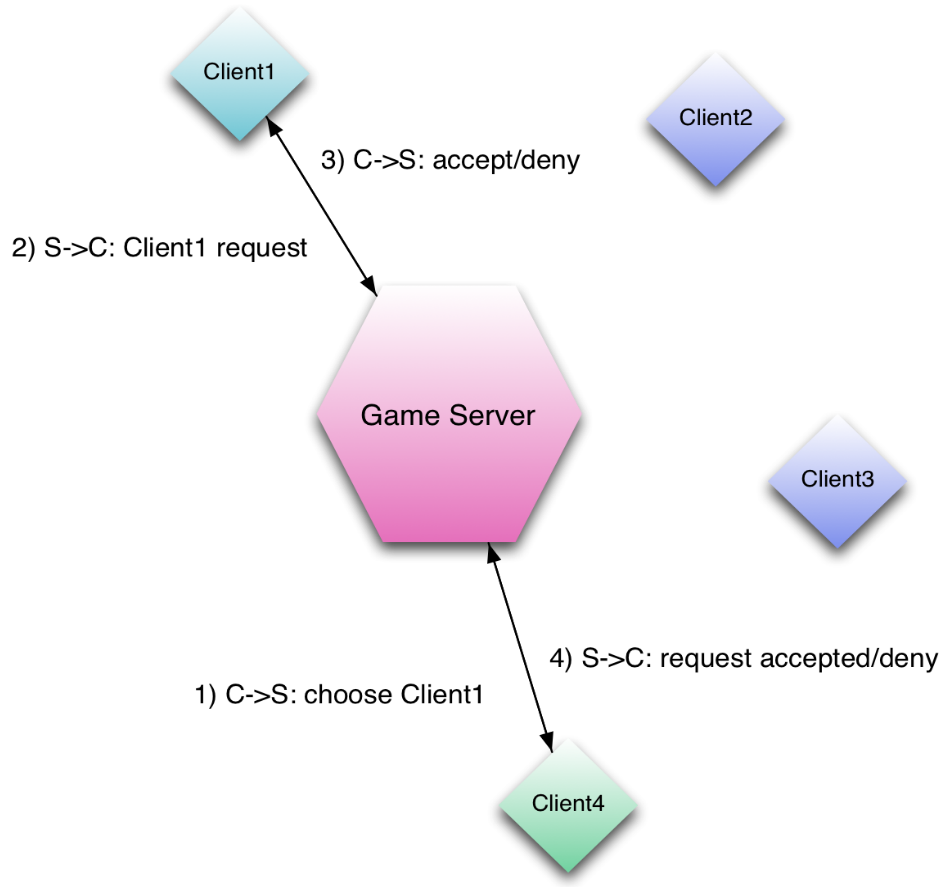
- Download **from wiki website**
  - The zip file containing code
  - Assignment manual

Overview

- Design a system with a game server and clients
- Each entity listen's on it's own port (unique)
- UDP protocol from client to server and back

- Practical aspect: Client-server programs are going to be run on the same machine, so "127.0.0.1" – standard IP (for part 1)

Structure



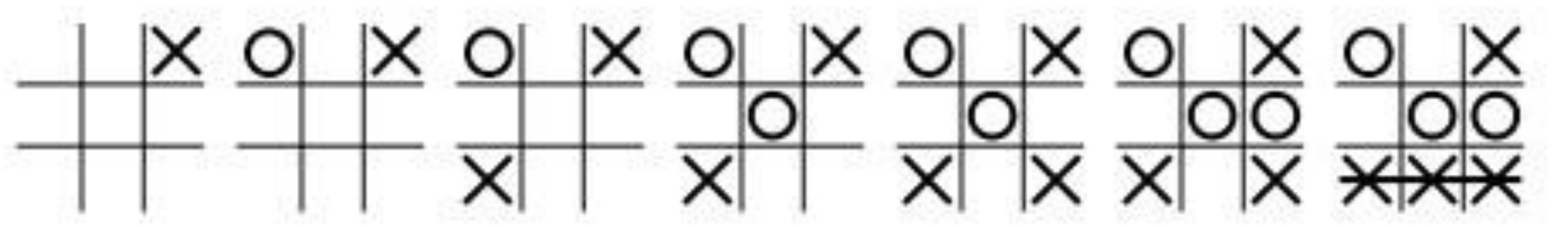
- 1 game server
- ≤ 5 players

In one local host machine

The Game

- Tic-tac-toe
- Fairly standard game

http://en.wikipedia.org/wiki/Tic-tac-toe



The server

- Clients register with the server
- The server maintains a list of clients with three states(busy, free, decision)
- The server also handles the game logic for each game, choosing a player/playing a valid cell /finishing game

The client(s)

- The client basically logs in to the server
- Chooses an opponent from player list
- Establish connection and play the game
- Continue when finishing game or logout

Part 1

- Clients- server communication
- Just combine the previous definitions into a system
- The format for packets will be provided

Part 2

- Communication over an unreliable channel
- We provide a channel "jar" file, arguments <port-number>
- It ensures that packets are lost at a high frequency
- Note: introduce "acknowledgement" packets
- The format of the packet will be provided with the assignment document

Part 3

- The server can accept only 5 clients from the same IP
- Ensure that the server blocks the 6th client from the same IP trying to access it
- So we provide a proxy server
- Use the proxy to bypass the restriction

Introduction to Threads

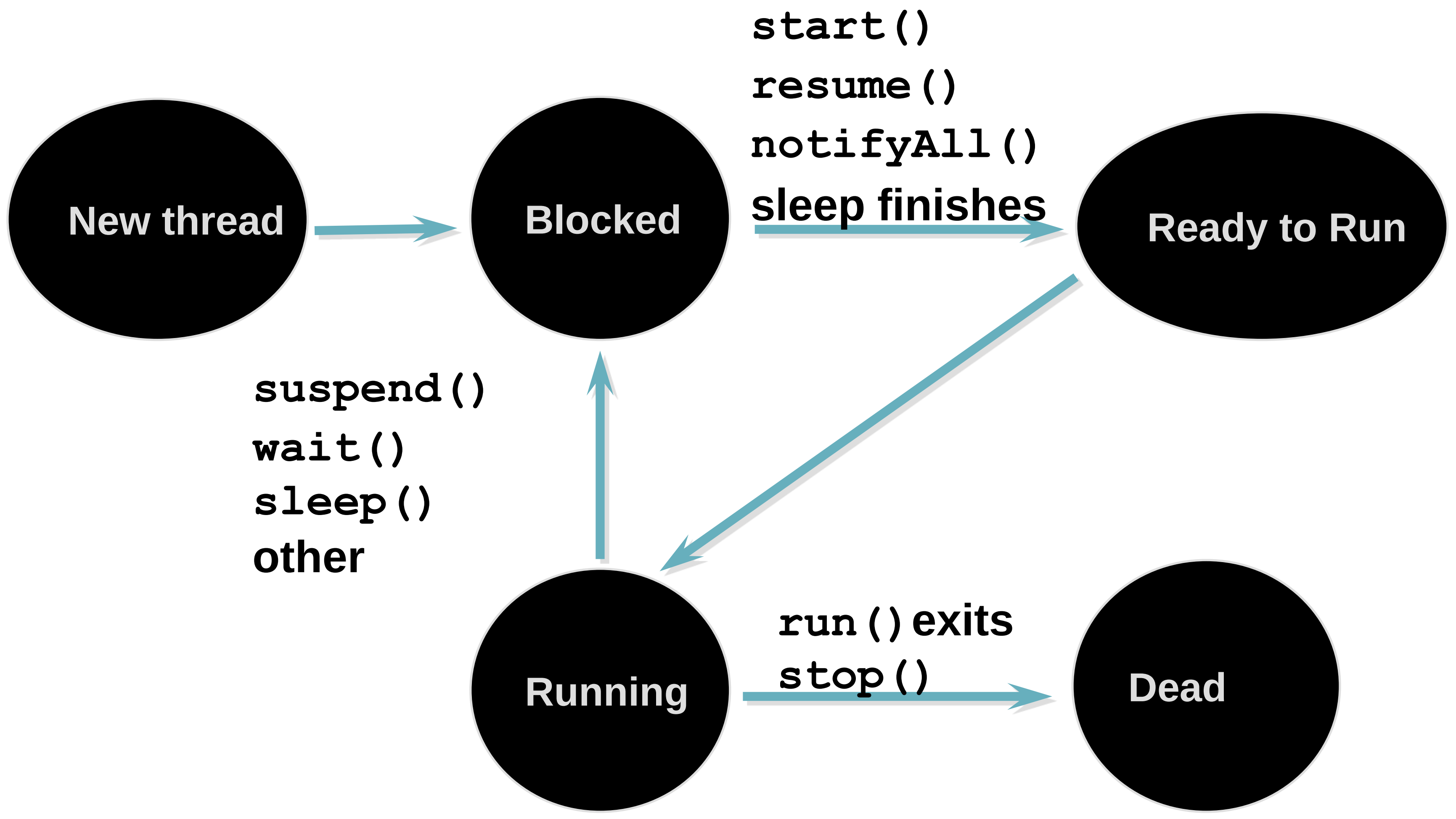
- Threads and processes
- Running in parallel

- **Practical aspect**:
- Client and server are processes
- Each process contains one or more threads for the tasks
- Context of threads - Java

What is a thread?

- A single sequential execution path in a program
- Concentrates on a particular subtask
- Efficient usage of CPU time
- Different from a process

Lifecycle of a thread - Java



Creating a Thread

```
public class myThread implements Runnable {
…….
public void run(){
     ….
 }
}
```

```
myThread t1 = new myThread();

Thread t = new Thread(t1);
t.start();
```

```
public class myThread extends Thread {
…….
public void run(){
     ….
 }
}
```

```
myThread t = new myThread();
t.start();
```

Thread API definition

- http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html
- More about the thread class

- Note: ensure there are no issues when threads access a common resource
- http://en.wikipedia.org/wiki/Semaphore_%28programming%29
- Examples will be provided with the assignment

Socket Programming

Java

Two Types

- TCP
  - Set up connection
  - Send data through the connection
- UDP
  - No connection needed
  - Send UDP packets

TCP Programming Steps

TCP Receiver

- Create ServerSocket
- Bind listening port
- Create Socket
- Accept connection
- Receive
- Close connection

TCP Sender

- Create Socket
- Connect to the receiver
- Send
- Close

Related Class & Method

- ServerSocket
  - accept()

- Socket
  - getInputStream()
  - getInetAddress()
  - getPort()

UDP Programming Steps

UDP Receiver

- Create DatagramSocket
- Bind receiving port
- Receive DatagramPacket

UDP Sender

- Create DatagramSocket
- Create DatagramPacket
- Send DatagramPacket

Related Class & Method

- DatagramSocket
  - send()
  - receive()

- DatagramPacket
  - getAddress()
  - getPort()

Tools

- Eclipse
    - Helpful in development
    - http://www.eclipse.org/
- Terminal
    - Compile source code
    - Run program
    - Submit
- Link:
    - http://docs.oracle.com/javase/tutorial/networking/sockets/index.html

Q&A