# Assignment, part1: feedback and common mistakes

INFO-0010

Samuel HIARD

## Grades?

- Not yet
- Work done at ~50%
- ETA : Couple of  weeks after Spring holidays (*should* be)

## Content of the presentation

- All 71 submissions were downloaded.
  - Compiled all of them.
  - Run some automatic test procedures.
- 26 submissions completely graded
- Selected examples illustrating mis-conceptions (incl. source code when relevant) are shown today.
- These examples are sorted according to the frequency of the problem in the submissions, when available.

## Localhost (100%)

**(NOT sanctioned this project)**

(This is the client)

```
int Port = 2[redacted];
InetAddress Address = InetAddress.getByName("127.0.0.1");
Socket serviceSocket = new Socket(Address,Port);
```

Server's address is thus hardcoded in the client.

3 ways to improve :

1. Have a config file that can be changed without recompiling (but then, program manipulates file, which was forbidden for this project)
2. Allow user to specify IP address in program's arguments
3. Ask user for the IP address in the program itself

# Robustness (~80%)

(This is the server, parsing the requests)

```
if(type == 0) {
        //Start Method
        server.start();
} else if (type == 1) {
        //This is a guess
        String follow = readNext(s);
        Analyse(follow);
} else {
        //List
        server.returnList();
}
```

My request is 1 3, then what?

Checked on this project :

1. 1 5 (~20%)
2. 2 0 (~65%)
3. 1 1 200 (~70%)

**Don't forget to respond in any case**

**Never expect, always check**

# Naming (62%)

**(This is the server)**

```
ServerSocket ss = new ServerSocket(2[redacted]);
Scanner sc = new Scanner(System.in);
char rep = 'n';
while(rep != 'y'){
        System.out.println("Waiting for a new connection to handle...");
        Socket ts = ss.accept();
        Worker w = new Worker(ts);
        Thread t = new Thread(w);
        t.start();
        System.out.println("New connection handled.");
}
```

**Use self-explanatory names for variables, methods and classes.**

## Report (~60%)

Report is evaluated mainly on two criterias:

- Presentation
  - Nice frontpage
  - Justified text
  - Numbered sections
  - No misprints/errors
  - Images non blurry if any
- Content
  - Most important : software architecture
  - If several classes, at least explain what each class does
  - In this context, good idea to describe how each request is processed and by which classes/methods
    - ✓ But no penalty if not done

# TCP (~55%)

(This is the server)

```
byte msg[ ] = new byte [2] ;

//receiving header
int len = in.read(msg);
if(len < 2)
{
        System.out.println("Error, received too few bytes");
        s.close();
}

//Initializing the subroutine
SubRoutine sub = new SubRoutine(out,in,s);
```

**Respect the stream-oriented property of TCP**

# Comments (~52% )

(Don't focus on the code, but on what the code looks like)

```
public class BattleshipGame{
   private boolean inProgress;
   private String secretPositions;
   private boolean[] showFilter;
   private int lives;
   private final ArrayList<Character> proposedLetters;

   static{
      PredefinedWords.addWords();
   }

   public BattleshipGame(){
      proposedLetters = new ArrayList<Character>();
   }

   private void pickAWord(){
      final Random randomNumberGenerator = new Random(new Date().getTime());
      secretWord = PredefinedWords.alWords.get
         (randomNumberGenerator.nextInt(PredefinedWords.iCapacity));
      showFilter = new boolean[secretWord.length()];
   }

   public void newGame(){
      inProgress = true;
      pickAWord();
      lives = 6;
      proposedLetters.clear();
   }

   public boolean inProgress(){
      return inProgress;
   }
```

```
public int lives(){
      return lives;
   }

   public int proposedLettersNumber(){
      return proposedLetters.size();
   }

   public String secretWordWithQuestionMarks(){
      StringBuilder secretWordWithQuestionMarksBuilder =
         new StringBuilder(secretWord);
      for(
         int letter = 0;
         letter < secretWord.length();
         ++letter
      )
         if(!showFilter[letter])
            secretWordWithQuestionMarksBuilder.setCharAt(letter, '?');

      return secretWordWithQuestionMarksBuilder.toString();
   }

   public String proposedLetters(){
      StringBuilder proposedLettersStringBuilder = new StringBuilder();

      for(Character letter: proposedLetters)
         proposedLettersStringBuilder.append(letter);

      return proposedLettersStringBuilder.toString();
   }
```

**Comments!**

**At least one introductory block at the beginning of the file**
**At least one block for each function**
**At least one line for each code block bigger than 8 lines**

# Exceptions (~50%)

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

public class BattleshipServer {
      public static void main (String argv[]) throws Exception{
                  ServerSocket ss = new ServerSocket(1[redacted]);//init of the socket

                  try{
                        while(true){
                                    Socket accepter = ss.accept();//accept the connection of a client
                                    ClientWorker client = new ClientWorker(accepter);//create a thread
corresponding to the client
                                    client.start();//start the thread
                                    }
                  }catch(InterruptedIOException iioe){//exception handler for the timeout
                        System.err.println ("Remote host timed out during read operation");
                  }catch(IOException e){//any other exception for the thread
                        System.err.println("ioexception:"  + e);
                  }
                  finally{//closing the socket
                                    ss.close();
                  }
      }
}
```

**ALWAYS catch Exceptions!**

# Compilation (~46%)

```
ms808:~/cpp/test/Etudiants/TP1/s******$ javac -Xlint *.java
Game.java:12: warning: [rawtypes] found raw type: ArrayList
        this.listGuess = new ArrayList();
                             ^
  missing type arguments for generic class ArrayList<E>
  where E is a type-variable:
    E extends Object declared in class ArrayList
Game.java:12: warning: [unchecked] unchecked conversion
        this.listGuess = new ArrayList();
                         ^
  required: ArrayList<Characters>
  found:    ArrayList
2 warnings
ms808:~/cpp/test/Etudiants/TP1/s******$
```

**Every submission should compile without warnings (nor errors)**

**Use –Xlint to display compilation warnings**

# Reading (~45%)

(This is the server reading the header)

```
protected Message RetrieveResponse() throws IOException {
	// Compare version numbers
	boolean correctVersion = version == (byte) is.read();

	// Retrieve the rest of the header
	byte messageType = (byte) is.read();
```

**Never read one character at a time!**

## Time-outs (~36%)

```
serverSocket = new ServerSocket(Port);
clientSocket = serverSocket.accept();
dos = new DataOutputStream(clientSocket.getOutputStream());
dis = new DataInputStream(clientSocket.getInputStream());
byte[] header = new byte[2];
dis.read(header, 0, 2);
```

**Use Socket Time-outs (default : none)**

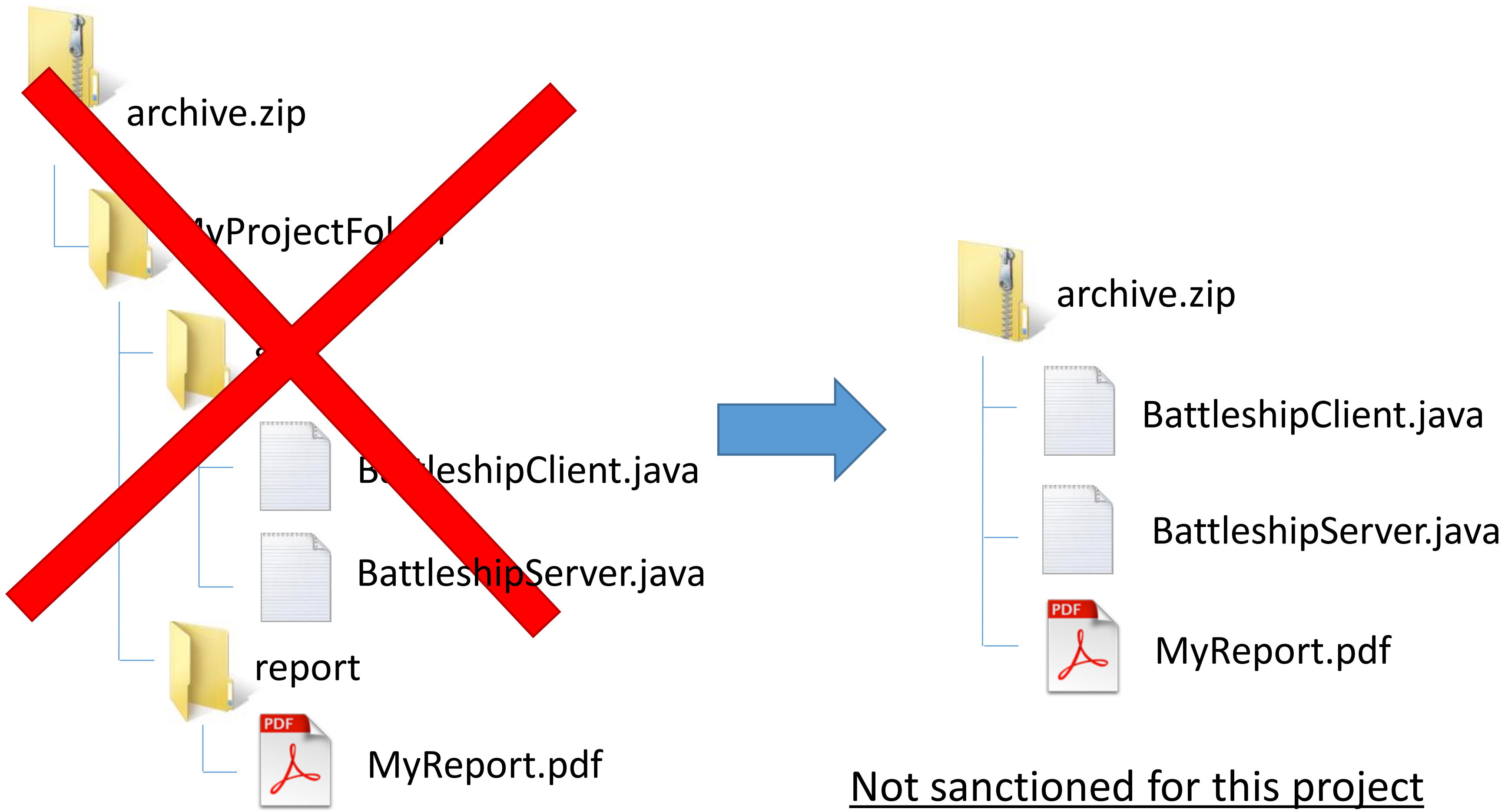
## Respect the protocol (~35%)

- Messages always start with a 2 byte header and sometimes have one byte more
  - Not the same as sending 3 bytes each time (nor 64 or even 201)
- 1 3 2 4 5 2 0 is not "1 3 2 4 5 2 0"
- Menus should not travel through the network
- If a bad request arrives, respond with the bad request reply, don't just close the connexion

- Especially important when the development is split into several teams, and that all teams agree on a protocol.

# Archive ( ~26% )

- *"You will ensure your main class is named (…) located (…) at the root of the archive, (…).“*
- *“You will not cluster your program in packages or directories. All java files should be found in the same directory.”*



<ins>Not sanctioned for this project</ins>

Icons from Windows 8.1™

## Guidelines (~10% )

- You will implement the programs using Java 1.8, with packages java.lang, java.io, java.net and java.util.
  - **import org.omg.CORBA.portable.UnknownException;**
- You will ensure that your program can be terminated at any time simply using CTRL+C, and avoid the use of ShutdownHooks
  - **Runtime.getRuntime().addShutdownHook(new Thread() { …**
- The server listens on port 2*xxx* – where *xxx* are the last three digits of your Ulg ID
  - **socket = new Socket(44444, port);**

**Respect the guidelines**

## And the rest

Not useful to show example

- Forget to synchronize the access to shared data (if any)
- Inconsistent indentation (~15%)
- Multi-threading (~5%)

## That's all for the first part

Keep on the good work!