CS 342 Software Design

- Start Networking
- Review for midterm
- Midterm is Thursday night!

TCP/IP

Transmission Control Protocol/ Internet Protocol:

how data is exchanged over the internet by providing end-to-end communications

Hosts have ports. In Java, Socket objects connect to ports and communicate to other sockets objects via streams

Java tcp/ip:

ServerSocket: gets a port to listen to; listens for connections.

Socket: endpoint of communication between two machines; reading and writing of data happen here

InputStream: each socket has one for input

OutputStream: each socket has one for output

That's all you need for basic communication!

ServerSocket:

```
ServerSocket mySocket = new ServerSocket(5555);
```

create a new ServerSocket listening for connections on port 5555;

```
Socket connection = mySocket.accept();
```

The accept() method returns a socket that is the connection between one client and the server. Each connection gets their own socket.

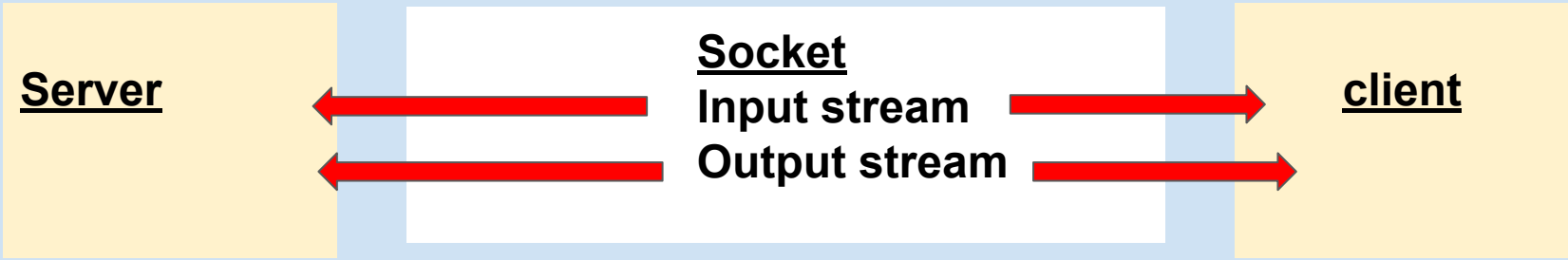
Client socket:

```
Socket socketClient = new Socket(“127.0.0.1”, 5555);
```

Create a new socket and try to connect to a certain address and port at that address

Streams:

Each socket has an input stream that reads information in and an output stream that sends information out.



What can we pass?

Serialization: Convert an object into a byte stream

- The object must implement the Serializable interface
- Static and transient values are not saved in the serialization process

serialVersionUID: used to verify that the sender and receiver of a serialized object have compatible classes for deserialization

Lots of errors! How do we handle them?

JVM Errors:

OutOfMemoryError

StackOverflowError

LinkageError

System errors:

FileNotFoundException

IOException

SocketTimeoutException

Programming errors:

NullPointerException

ArrayIndexOutOfBoundsException

ArithmeticException

Exceptions:

- Exceptions are events that occur during the execution of programs that disrupt the normal flow of instructions (e.g. divide by zero, array access out of bound, etc.).
- An exception is an object containing
  - Info about error
  - State of program when error happened
  - Can be thrown and caught

Checked and unchecked exceptions

Checked: programmer must handle them

Unchecked: optional; generally things your program can't recover from.

1) Methods that generate checked exceptions must declare that they throw them.
2) Methods that invoke methods that throw exceptions must handle them or let them propagate by declaring that they throw them.

Some Methods that throw exceptions:

Socket();

ServerSocket();

getInputStream();

getOutputStream();

PrintWriter();

Scanner();

Clicker Question:

If a method generates an exception, that exception:

A. Is Unchecked
B. Is Checked
C. Must be handled
D. A and C
E. B and C

Try, Catch, Finally:

Try: put the method with the checked exception here

Catch: catches the exception is error occurs. Put code to handle it here.

Finally: usually clean up code (close connections…). Gets called regardless of error occuring or not.

Try with resources:

Exception handling and tear down in one place; automatically closes resources when done.

```
try(  FileInputStream input = new FileInputStream("file.txt");
      BufferedInputStream bufferedInput = new
BufferedInputStream(input)){

}
```

JavaFX application lifecycle:

1. Constructs instance of the specified Application class
2. Calls the init() method
3. Calls the start() method
4. Waits for the application to finish
    a. Application calls Platform.exit()
    b. Last window is closed
5. Calls the stop() method

Application runs on a single thread...Application Thread

init() runs on Launcher Thread

Blocking Methods:

Blocking methods put Current thread on blocking position until method returns like ServerSocket accept() method which blocks until a client Socket connects to Server.

Most GUI applications have a single UI thread: most GUI applications are multithreaded

Clicker Question:

.java files can have multiple public classes?

A) True
B) False