

SER 321 B Session

SI Session

Sunday, November 5th 2023

7:00 - 8:00 pm MST

Agenda



Threading Pitfalls

Critical Sections

Traffic Analogy

Threading your Server

Serialization

SI Session Expectations

Thanks for coming to the **SER 321** SI session. We have a packed agenda and we are going to try to get through as many of our planned example problems as possible. This session will be recorded and shared with others.

- If after this you want to see additional examples, please visit the drop-in tutoring center.
- We will post the link in the chat now and at the end of the session.
 - tutoring.asu.edu
- Please keep in mind we are recording this session and it will be made available for you to review 24-48 hours after this session concludes.
- Finally, please be respectful to each other during the session.

Interact with us:

Zoom Features




Zoom Chat

- Use the chat feature to interact with the presenter and respond to presenter's questions.
- Annotations are encouraged

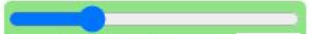
SER 321


Threading Pitfalls!


Dining Philosophers


model speed 

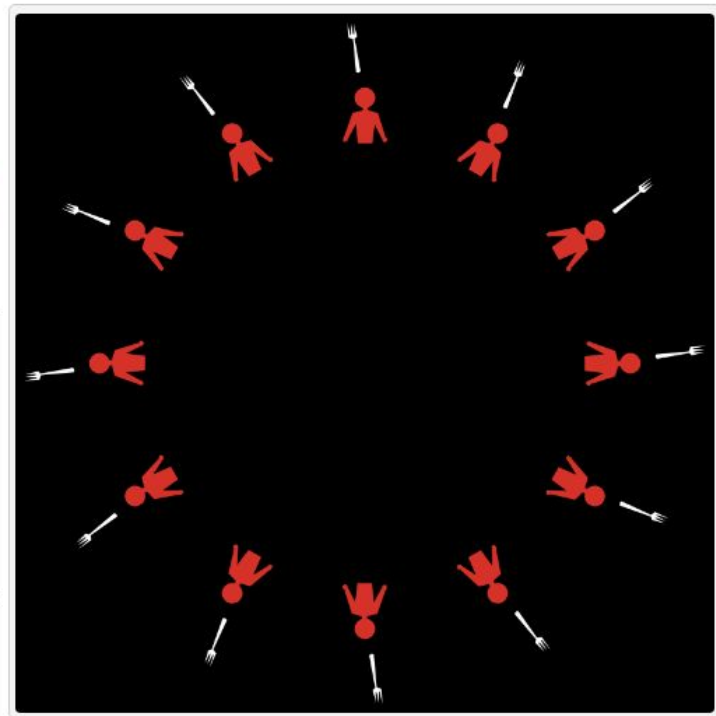
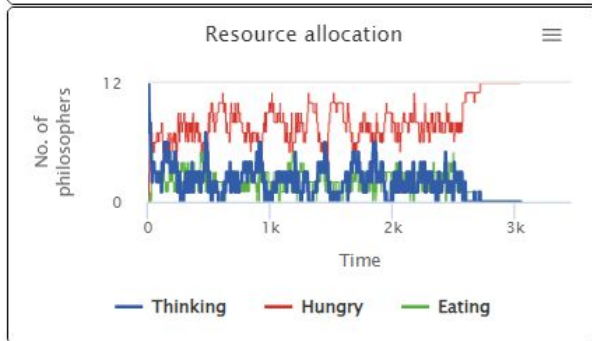
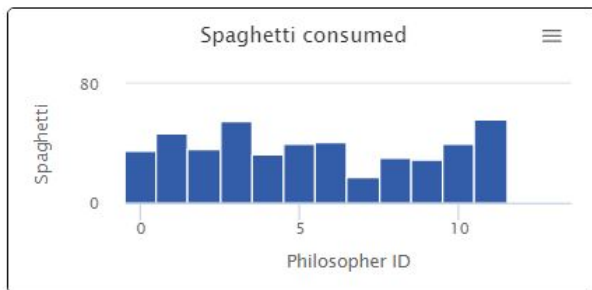
ticks: 3043

 num-philosophers 12

setup go  go once

 hungry-chance 0.5 ☐ cooperation?

 full-chance 0.5



SER 321

Threading Pitfalls

- Race Condition

More than one thread accesses a single resource at one time

- Starvation

One thread never gets access to the resource it needs

- Deadlock

A thread is only able to acquire access to part of its resources

SER 321

Threading Pitfalls

- Race Condition



Crash

More than one thread accesses a single resource at one time

- Starvation

One thread never gets access to the resource it needs

- Deadlock

A thread is only able to acquire access to part of its resources

SER 321

Threading Pitfalls

- Race Condition



Crash

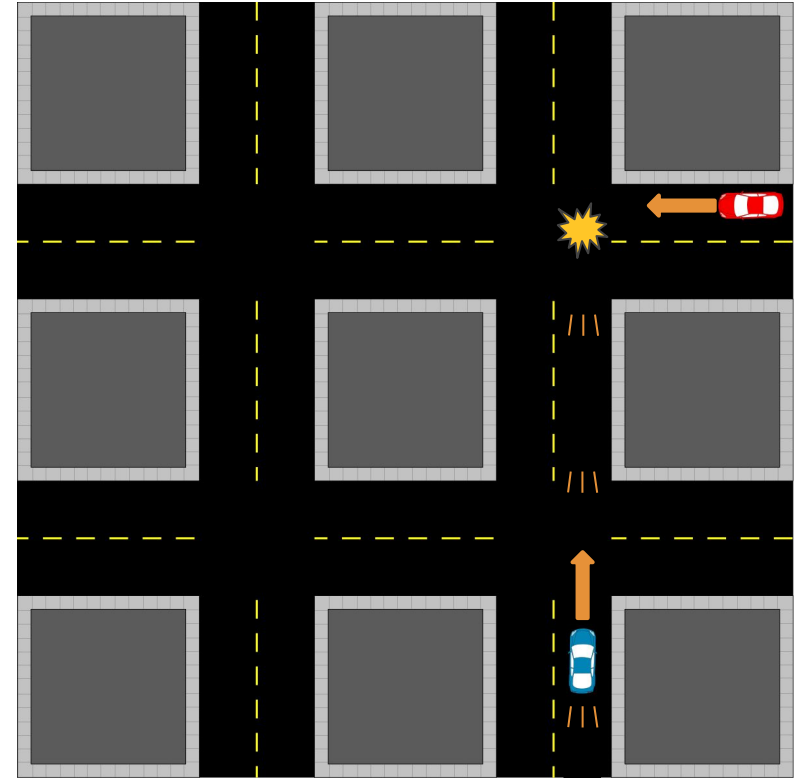
More than one thread accesses a single resource at one time

- Starvation

One thread never gets access to the resource it needs

- Deadlock

A thread is only able to acquire access to part of its resources



SER 321

Threading Pitfalls

- Race Condition



Crash

More than one thread accesses a single resource at one time

- Starvation



Cross Traffic

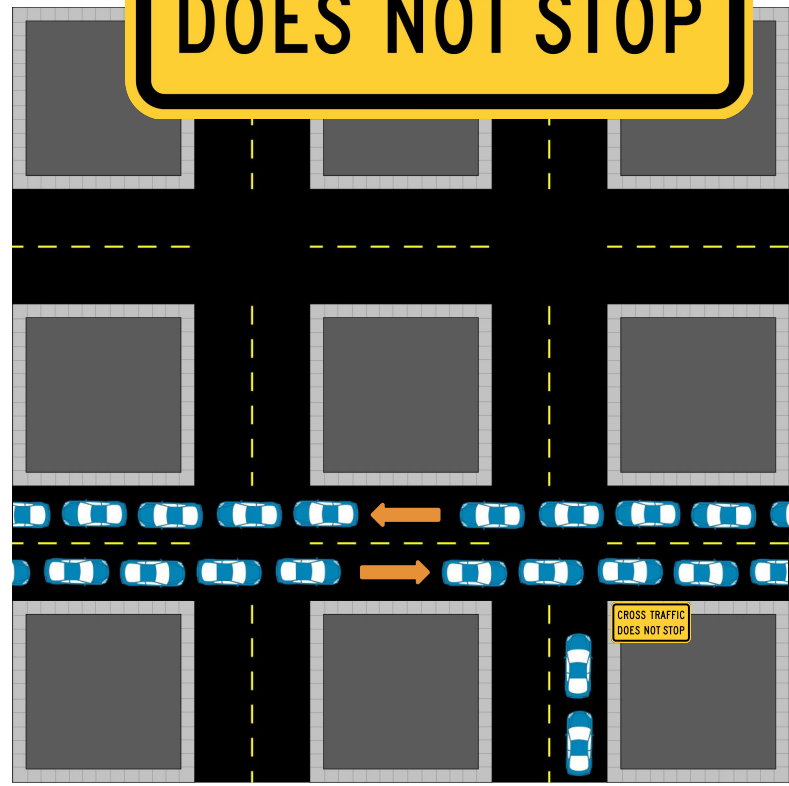
One thread never gets access to the resource it needs

- Deadlock

A thread is only able to acquire access to part of its resources

Austin Walter's Traffic Comparison

CROSS TRAFFIC
DOES NOT STOP



SER 321

Threading Pitfalls

- Race Condition



Crash

More than one thread accesses a single resource at one time

- Starvation



Cross Traffic

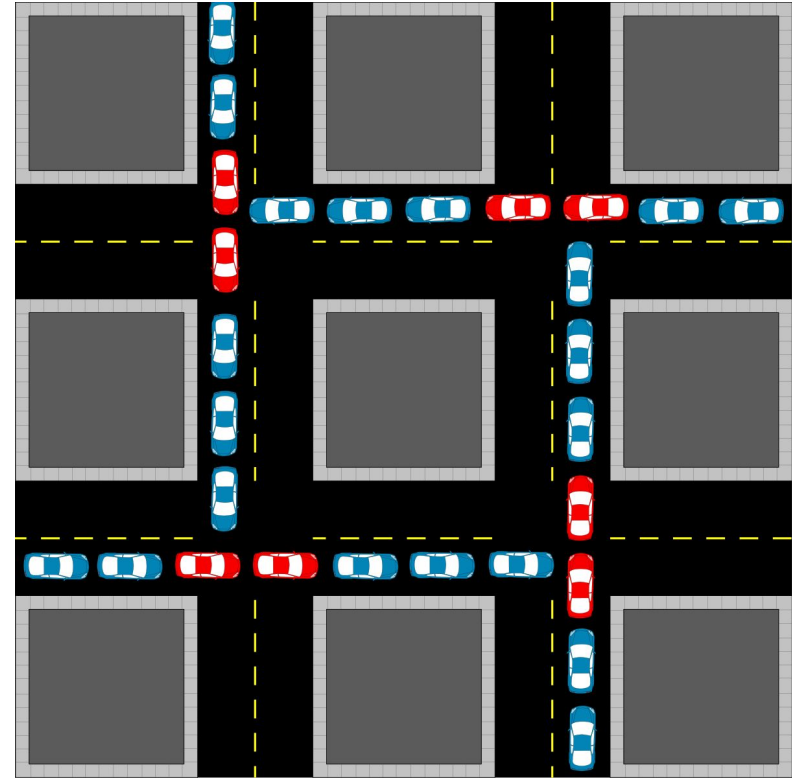
One thread never gets access to the resource it needs

- Deadlock



Gridlock

A thread is only able to acquire access to part of its resources



SER 321

Threading your Server

Standard Server Steps

1. Define Params
2. Make socket
3. Establish connection
4. Handle Connection
5. Close Connection

Okay so how do we thread this?

```
public static void main (String args[]) {  
    Socket sock;  
    try {  
        //open socket  
        ServerSocket serv = new ServerSocket( port 8888); // create server socket on port 8888  
        System.out.println("Server ready for 3 connections");  
        // only does three connections then closes  
        // NOTE: SINGLE-THREADED, only one connection at a time  
        for (int rep = 0; rep < 3; rep++){  
            System.out.println("Server waiting for a connection");  
            sock = serv.accept(); // blocking wait  
            // setup the object reading channel  
            ObjectInputStream in = new ObjectInputStream(sock.getInputStream());  
  
            // read in one object, the message. we know a string was written only by knowing what the client sent.  
            // must cast the object from Object to desired type to be useful  
            String s = (String) in.readObject();  
            System.out.println("Received the String "+s);  
            // read in the number, we know it's an integer because that's the second thing sent by the client.  
            Integer i = (Integer) in.readObject();  
            System.out.println("Received the Integer "+ i);  
  
            // generate an output  
            // get output channel  
            OutputStream out = sock.getOutputStream();  
            // create an object output writer (Java only)  
            ObjectOutputStream os = new ObjectOutputStream(out);  
            // write the whole message  
            os.writeObject("Got it!");  
            // make sure it wrote and doesn't get cached in a buffer  
            os.flush();  
        }  
    } catch (Exception e) {e.printStackTrace();}  
}
```

[SockServer](#) from [JavaSimpleSock2](#) in [examples Repo](#)

```
public class ThreadedSockServer extends Thread {
```

SER 321

Threading your Server

Threaded Server Steps

0. Server extends Thread
1. Define Params
2. Make socket
3. Establish connection
- 4.
5. Handle Connection
6. Close Connection

SockServer in [JavaSimpleSock2](#)

```
Socket sock;  
try {  
    //open socket  
    ServerSocket serv = new ServerSocket(port: 8888); // create server socket on port 8888  
    System.out.println("Server ready for 3 connections");  
    // only does three connections then closes  
    // NOTE: SINGLE-THREADED, only one connection at a time  
    for (int rep = 0; rep < 3; rep++){  
        System.out.println("Server waiting for a connection");  
        sock = serv.accept(); // blocking wait  
        // setup the object reading channel  
        ObjectInputStream in = new ObjectInputStream(sock.getInputStream());
```

```
ServerSocket serv = new ServerSocket(portNo);  
while (true) {  
    System.out.println("Threaded server waiting for connects on port " + portNo);  
    sock = serv.accept();  
    System.out.println("Threaded server connected to client-" + id);
```

[JavaThreadedSock](#) in Sockets

```
public class ThreadedSockServer extends Thread {
```

SER 321

Threading your Server

Threaded Server Steps

0. Server extends Thread
1. Define Params
2. Make socket
3. Establish connection
4. Start Thread!
5. Handle Connection
6. Close Connection

```
Socket sock; public ThreadedSockServer(Socket sock, int id) {
try {
    //open so
    ServerSocket serv = new ServerSocket(portNo);
    System.out.println("Threaded server waiting for connects on port " + portNo);
    sock = serv.accept(); // blocking wait
    // setup the object reading channel
    ObjectInputStream in = new ObjectInputStream(sock.getInputStream());

    // only does three connections then closes
    // NOTE: SINGLE-THREADED, only one connection at a time
    for (int rep = 0; rep < 3; rep++){
        System.out.println("Server waiting for a connection");
        sock = serv.accept(); // blocking wait
        // setup the object reading channel
        ObjectInputStream in = new ObjectInputStream(sock.getInputStream());
    }
}
```

t 8888

```
ServerSocket serv = new ServerSocket(portNo);
while (true) {
    System.out.println("Threaded server waiting for connects on port " + portNo);
    sock = serv.accept();
    System.out.println("Threaded server connected to client-" + id);
    // create thread
    ThreadedSockServer myServerThread = new ThreadedSockServer(sock, id++);
    // run thread and don't care about managing it
    myServerThread.start();
}

public void run() {
```

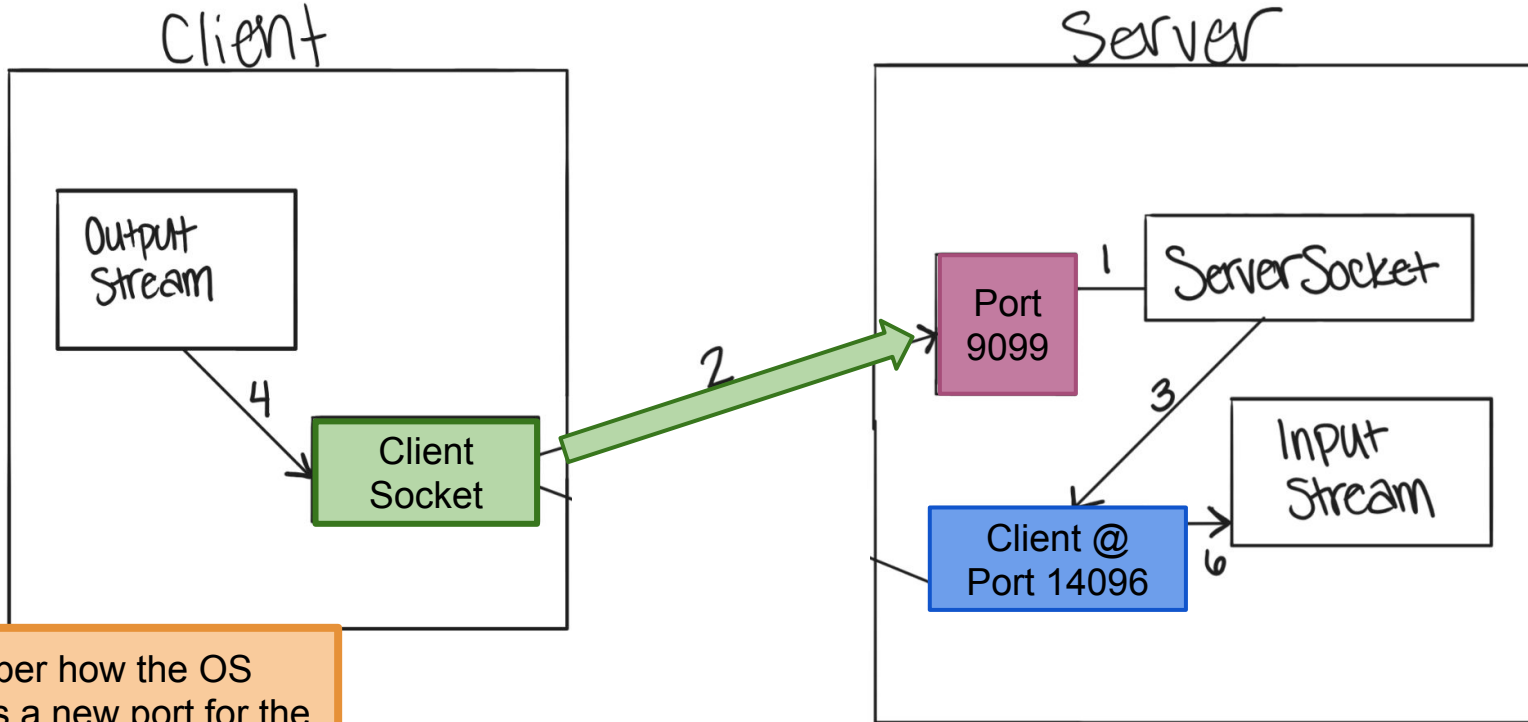


JavaThreadedSock in Sockets

SER 321

Client Socket

Values of the Client Socket Object after Connection:
Inet Address: /127.0.0.1
Local Address: /127.0.0.1
Local Port: 9099
Allocated Client Socket (Remote Port): 14096

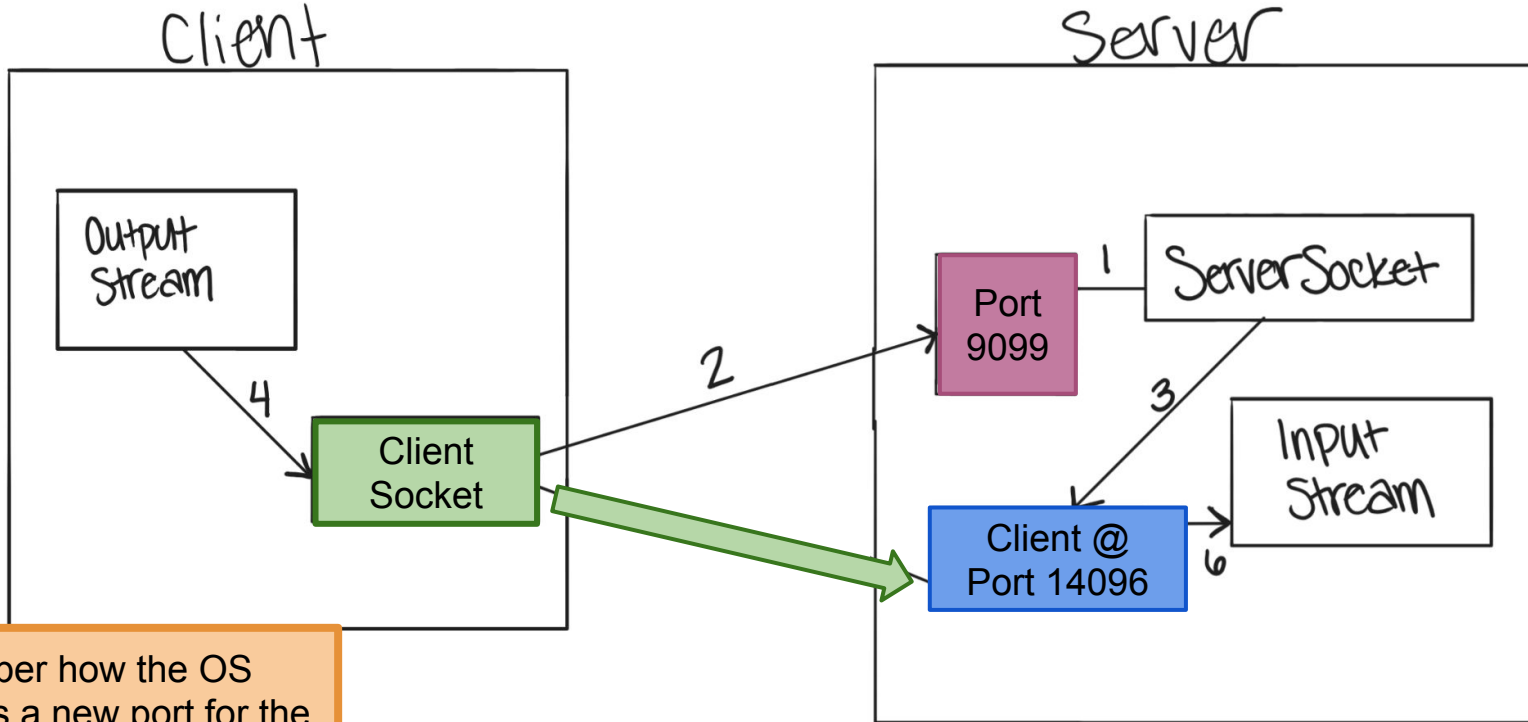


Remember how the OS
allocates a new port for the
client?

SER 321

Client Socket

```
Values of the Client Socket Object after Connection:  
Inet Address: /127.0.0.1  
Local Address: /127.0.0.1  
Local Port: 9099  
Allocated Client Socket (Remote Port): 14096
```

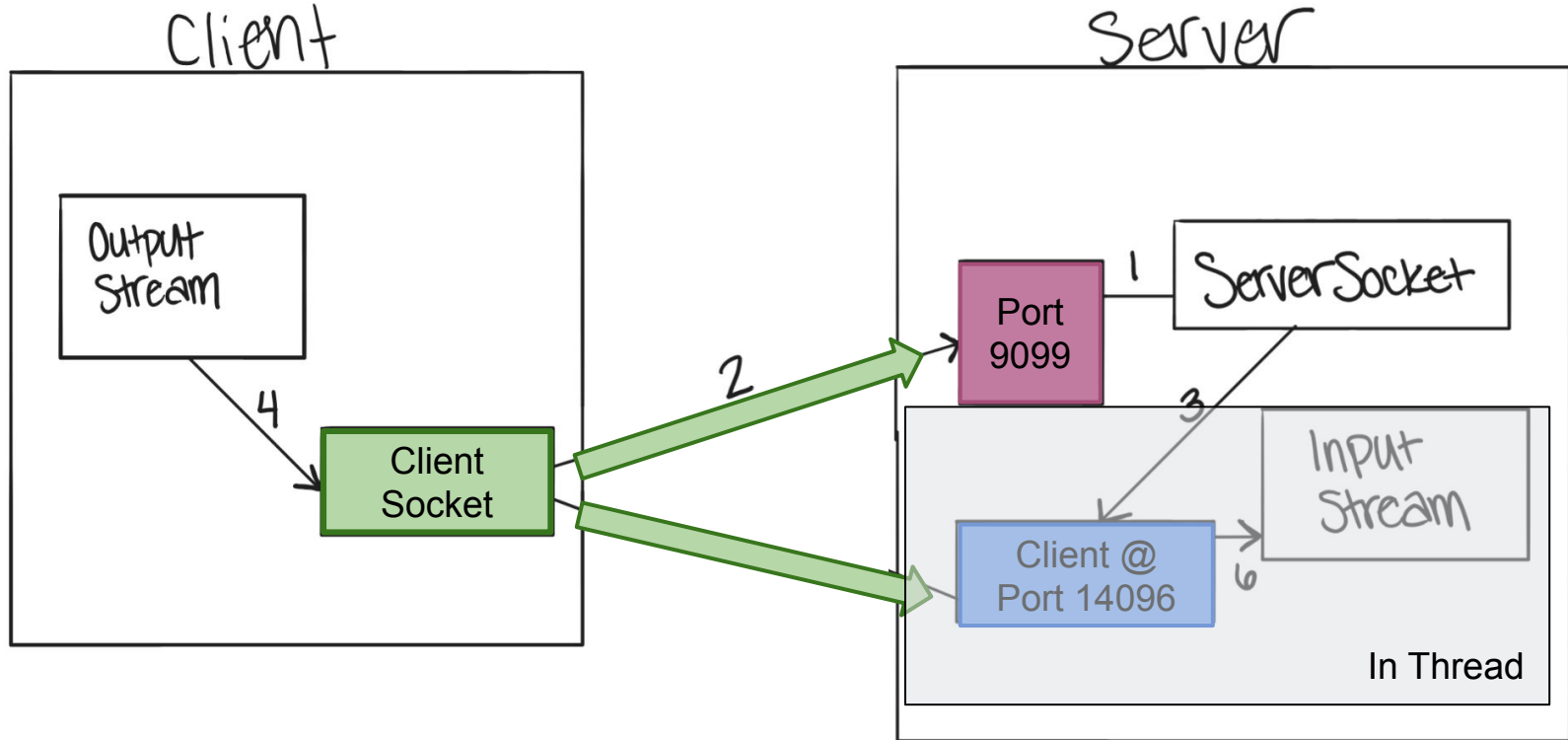


Remember how the OS
allocates a new port for the
client?

SER 321

Threaded Server Communication

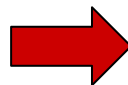
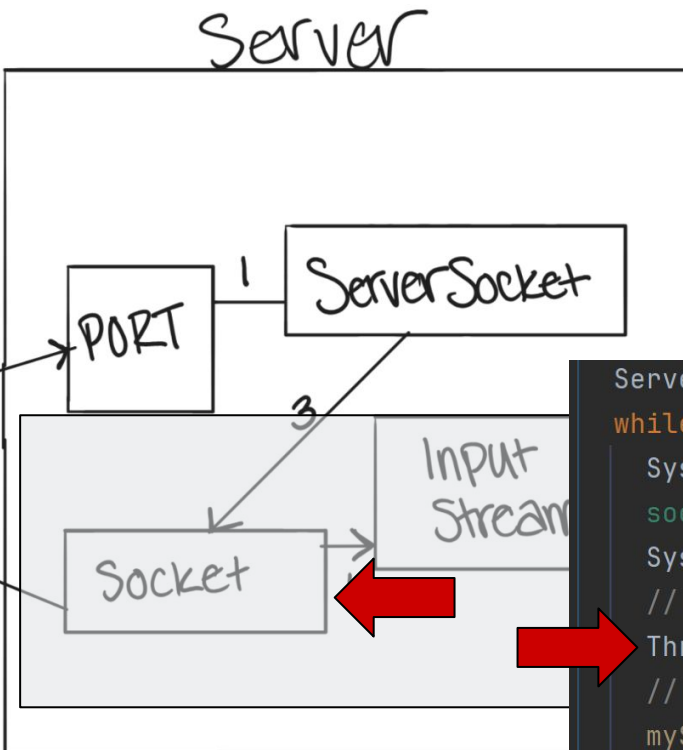
How do we do that in a threaded server?



SER 321

Threaded Server Communication

So how does the communication get passed from the main thread into the Client Thread?



```
sock = serv.accept();
```



```
public ThreadedSockServer(Socket sock, int id) {
    this.conn = sock;
    this.id = id;
}
```



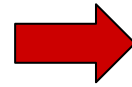
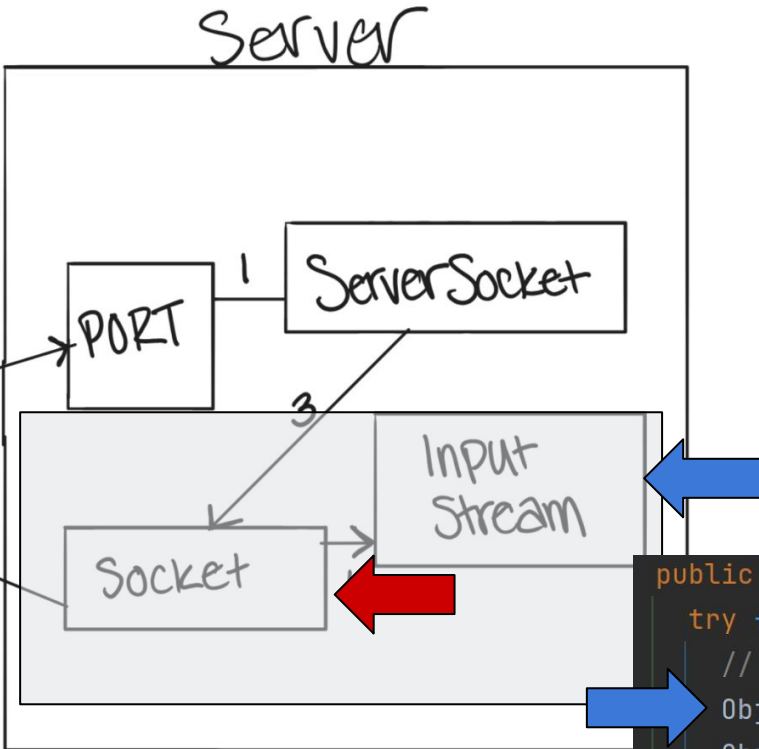
```
ServerSocket serv = new ServerSocket(portNo);
while (true) {
    System.out.println("Threaded server waiting for connects on port " + portNo);
    sock = serv.accept();
    System.out.println("Threaded server connected to client-" + id);
    // create thread
    ThreadedSockServer myServerThread = new ThreadedSockServer(sock, id++);
    // run thread and don't care about managing it
    myServerThread.start();
}
```



SER 321

Threaded Server Communication

So how does the communication get passed from the main thread into the Client Thread?



```
sock = serv.accept();
```



```
public ThreadedSockServer(Socket sock, int id) {
    this.conn = sock;
    this.id = id;
}
```

Now we can just handle the connection as we have been!

```
public void run() {
    try {
```

```
        // setup read/write channels for connection
```

```
        ObjectInputStream in = new ObjectInputStream(conn.getInputStream());
```

```
        ObjectOutputStream out = new ObjectOutputStream(conn.getOutputStream());
```

Questions?

Survey:

https://bit.ly/asn_survey



Upcoming Events

SI Sessions:

- Monday, November 6th 2023 at 4:00 pm MST
- Thursday, November 9th 2023 at 7:00 pm MST
- Sunday, November 12th 2023 at 7:00 pm MST

Review Sessions:

- Planning to post a poll this week!

More Questions?

Check out our other resources!

tutoring.asu.edu



Academic Support

Academic Support Network (ASN) provides a variety of free services in-person and online to help currently enrolled ASU students succeed academically.

Services



Subject Area Tutoring

Need in-person or online help with math, science, business, or engineering courses? Just hop into our Zoom room or drop into a center for small group tutoring. We'll take it from there.

[Need help using Zoom?](#)

[View the tutoring schedule](#)

[View digital resources](#)

Go to Zoom



Writing Tutoring

Need help with undergraduate or graduate writing assignments? Schedule an in-person or online appointment, access your appointment link, or wait in our drop-in queue.

[Access your appointment link](#)

[Access the drop-in queue](#)

Schedule Appointment



Online Study Hub

Join our online peer communities to connect with your fellow Sun Devils. Engage with our tools to search our bank of resources, videos, and previously asked questions. Or, ask our Tutorbot questions.

Now supporting courses in Math, Science, Business, Engineering, and Writing.

Online Study Hub

1-

Go to Zoom

2-

[Need help using Zoom?](#)

[View the tutoring schedule](#)

[View digital resources](#)



1. Click on 'Go to Zoom' to log onto our Online Tutoring Center.
2. Click on 'View the tutoring schedule' to see when tutors are available for specific courses.

More Questions?

Check out our other resources!

tutoring.asu.edu/online-study-hub

 **Academic Support Network**

 [Services](#)  [Faculty and Staff Resources](#) [About Us](#) 

[University College](#)

Online Study Hub

Online peer communities for students and tutors, YouTube channels, and Tutorbots.



What are online peer communities?

Individual courses have an online peer community that allows you to connect with your peers to post and answer questions and to develop study groups.



How can tutoring center videos help?

Videos can help supplement the learning you're doing in and outside of class and include step-by-step methods for how to understand concepts.



How does the Tutorbot work?

You can ask the Tutorbot questions about course concepts and the Tutorbot will recommend additional resources and examples to help address your questions.

Select a subject

- Any -

Apply



Academic Support Network



[Services](#) 

[Faculty and Staff Resources](#)

[About Us](#) 

[University College](#)

Select a subject

- Any -

Apply

Business

ACC 231

Uses of Accounting Info I

 [Peer Community](#)

ACC 241

Uses of Accounting Info II

 [Peer Community](#)

CIS 105

Computer Applications and Information Technology

 [Peer Community](#)

Don't forget to check out the Online Study Hub for additional resources!

Additional Resources

[CoureRepo](#)

[org.json API Docs](#)

[JSON Helper](#)

[Dining Philosophers Interactive](#)

[Austin Walter's Traffic Comparison](#)