

SER 321 B Session

SI Session

Monday March 25th 2024

7:00 pm - 8:00 pm MST

Agenda



JSON

Socket Communication

Properties

Steps

Diagram

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

JSON

Which of the following would be a valid response?

[Assign 3-1 Starter Code](#)

```
{  
  "type" : "echo", -- echoes the initial response  
  "ok" : <bool>, -- true or false depending on request  
  "echo" : <String>, -- echoed String if ok true  
  "message" : <String>, -- error message if ok false  
}
```

Echo General Response

A. {
 "type" : "echo",
 "echo" : <String>
}

C. {
 "type" : "echo",
 "message" : <String>
}

B. {
 "type" : "echo",
 "ok" : false,
 "echo" : <String>
}

D. {
 "type" : "echo",
 "ok" : true,
 "echo" : <String>
}

SER 321

JSON

Which of the following would be a valid response?

[Assign 3-1 Starter Code](#)

```
{  
  "type" : "echo", -- echoes the initial response  
  "ok" : <bool>, -- true or false depending on request  
  "echo" : <String>, -- echoed String if ok true  
  "message" : <String>, -- error message if ok false  
}
```

Echo General Response

A. {
 "type" : "echo",
 "echo" : <String>
}

C. {
 "type" : "echo",
 "message" : <String>
}

B. {
 "type" : "echo",
 "ok" : false,
 "echo" : <String>
}

D. {
 "type" : "echo",
 "ok" : true,
 "echo" : <String>
}

SER 321

JSON

How would you check if the *type* header exists?

[Assign 3-1 Starter Code](#)

```
{  
  "type" : "echo", -- type of request  
  "data" : <String> -- String to be echoed  
}
```

Echo Request

```
{  
  "type" : "echo", -- echoes the initial response  
  "ok" : <bool>, -- true or false depending on request  
  "echo" : <String>, -- echoed String if ok true  
  "message" : <String>, -- error message if ok false  
}
```

Echo General Response

*Think of
yourself as the
server!*

```
String reqType = ""; //to hold request type  
String content = ""; // to hold data from client  
JSONObject request = new JSONObject(str);
```

You can assume
STR is the data
sent from the
client

```
if (!request.has("type")) {  
    //error - no type header  
    //send error response  
}
```

How would you fetch the type content?

[Assign 3-1 Starter Code](#)

SER 321

JSON

```
{
  "type" : "echo", -- type of request
  "data" : <String> -- String to be echoed
}
```

Echo Request

```
{
  "type" : "echo", -- echoes the initial response
  "ok" : <bool>, -- true or false depending on request
  "echo" : <String>, -- echoed String if ok true
  "message" : <String>, -- error message if ok false
}
```

Echo General Response

*Think of
yourself as the
server!*

You can assume
STR is the data
sent from the
client

```
String reqType = ""; //to hold request type
String content = ""; // to hold data from client
JSONObject request = new JSONObject(str);
if (!request.has("type")) {
    //error - no type header
    //send error response
}
```

```
reqType = request.getString("type");
```


How would you obtain the message sent?

[Assign 3-1 Starter Code](#)

SER 321

JSON

```
{
  "type" : "echo", -- type of request
  "data" : <String> -- String to be echoed
}
```

Echo Request

```
{
  "type" : "echo", -- echoes the initial response
  "ok" : <bool>, -- true or false depending on request
  "echo" : <String>, -- echoed String if ok true
  "message" : <String>, -- error message if ok false
}
```

Echo General Response

*Think of
yourself as the
server!*

You can assume
STR is the data
sent from the
client

```
String reqType = ""; //to hold request type
String content = ""; // to hold data from client
JSONObject request = new JSONObject(str);
if (!request.has("type")) {
    //error - no type header
    //send error response
}
reqType = request.getString("type");
if (reqType.equals("echo")) {
    //fetch message

}
```

```
content = request.getString("data");
```

How would you construct the response?

[Assign 3-1 Starter Code](#)

SER 321

JSON

```
{  
  "type" : "echo", -- type of request  
  "data" : <String> -- String to be echoed  
}
```

Echo Request

```
{  
  "type" : "echo", -- echoes the initial response  
  "ok" : <bool>, -- true or false depending on request  
  "echo" : <String>, -- echoed String if ok true  
  "message" : <String>, -- error message if ok false  
}
```

Echo General Response

*Think of
yourself as the
server!*

```
String content = request.getString("data");  
JSONObject res = new JSONObject();
```

```
{  
  "type" : "echo",  
  "ok" : true,  
  "echo" : <String>  
}
```

```
res.put("type", "echo");  
res.put("ok", true);  
res.put("echo", content);
```

Target Response

Then what do you do?

SER 321

Response Codes

Think Fast - ID these Codes

1XX

2XX

3XX

4XX

5XX

SER 321

Client/Server

Think Fast - Client or Server?

```
String host = args[0];  
Socket server = new Socket(host, port);  
System.out.println("Connected to server at " + host + ":" + port);
```

SER 321
Client/Server

Think Fast - Client or Server?

```
Socket clientSock;  
ServerSocket sock = new ServerSocket(port);  
System.out.println("Server ready for connections");
```

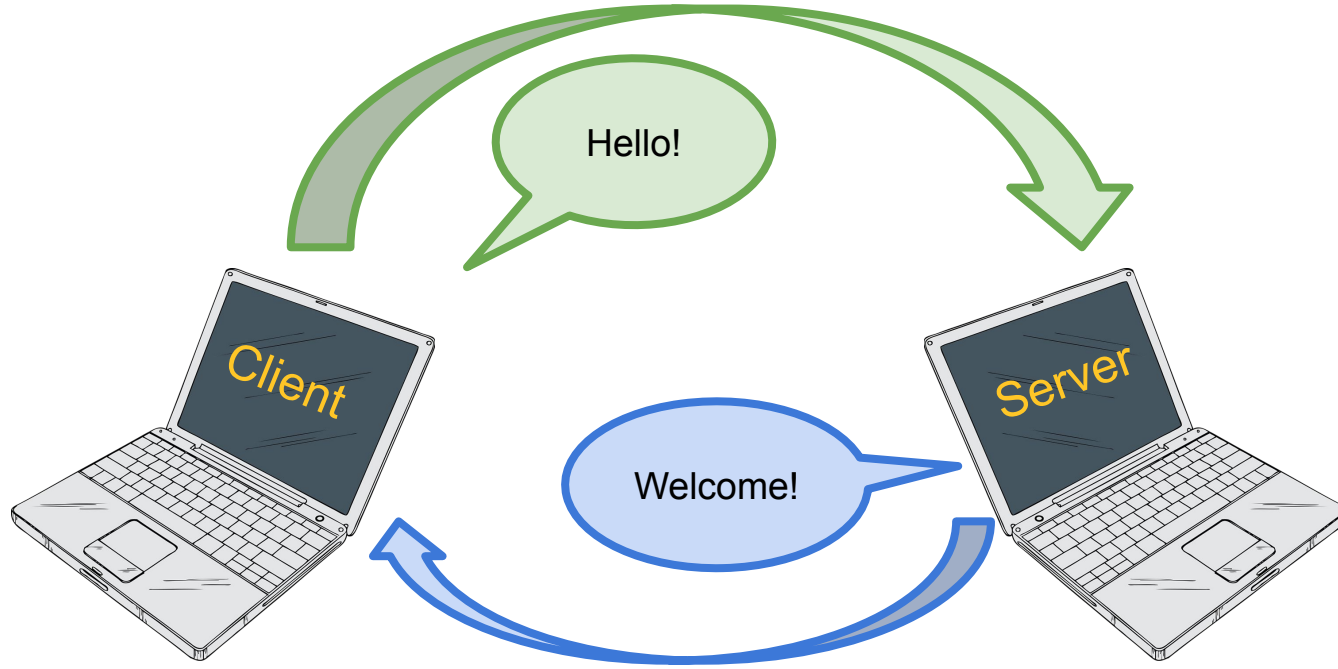
How are we feeling about Socket communications so far?

SER 321

Sockets!

Sockets allow our client and server to communicate!

Enables a client/server **conversation**



SER 321

Sockets!

Sockets allow our client and server to communicate!

Location

Connection
Semantics

Message Format

Need to define **3 properties** before usage

IP or DNS

142.251.46.206

www.google.com

TCP or UDP

Connection
Oriented

Connectionless

Protocol Specs

Synchronous

Asynchronous

Stateless

Stateful

Binary

Text

Headers

No Headers



SER 321

Sockets!

Two Main Conversation Models

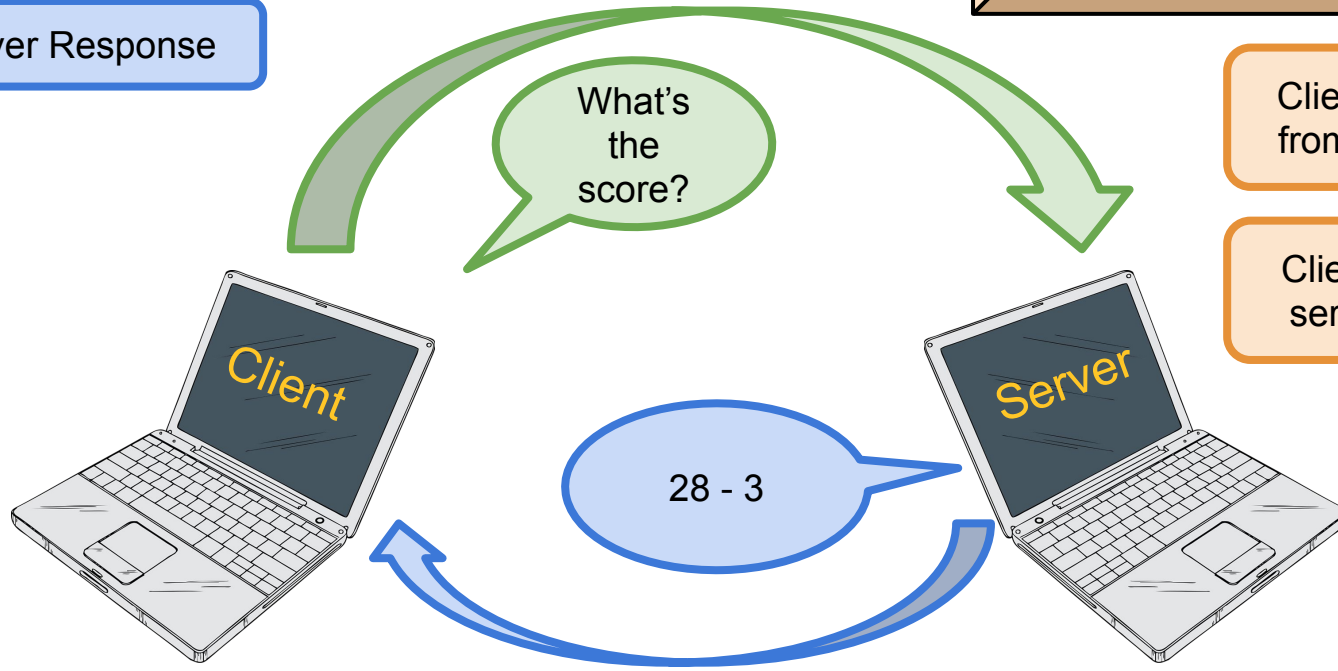
1. Client Request

2. Server Response

Pull/Polling Model

Client *pulls* info from the server

Client *polls* the server for info



SER 321

Sockets!

Two Main Conversation Models

1. Server sends update

2. Client acknowledges

Push Model

Server *pushes* info to client

Push notifications



SER 321

Sockets!

Two Main Conversation Models

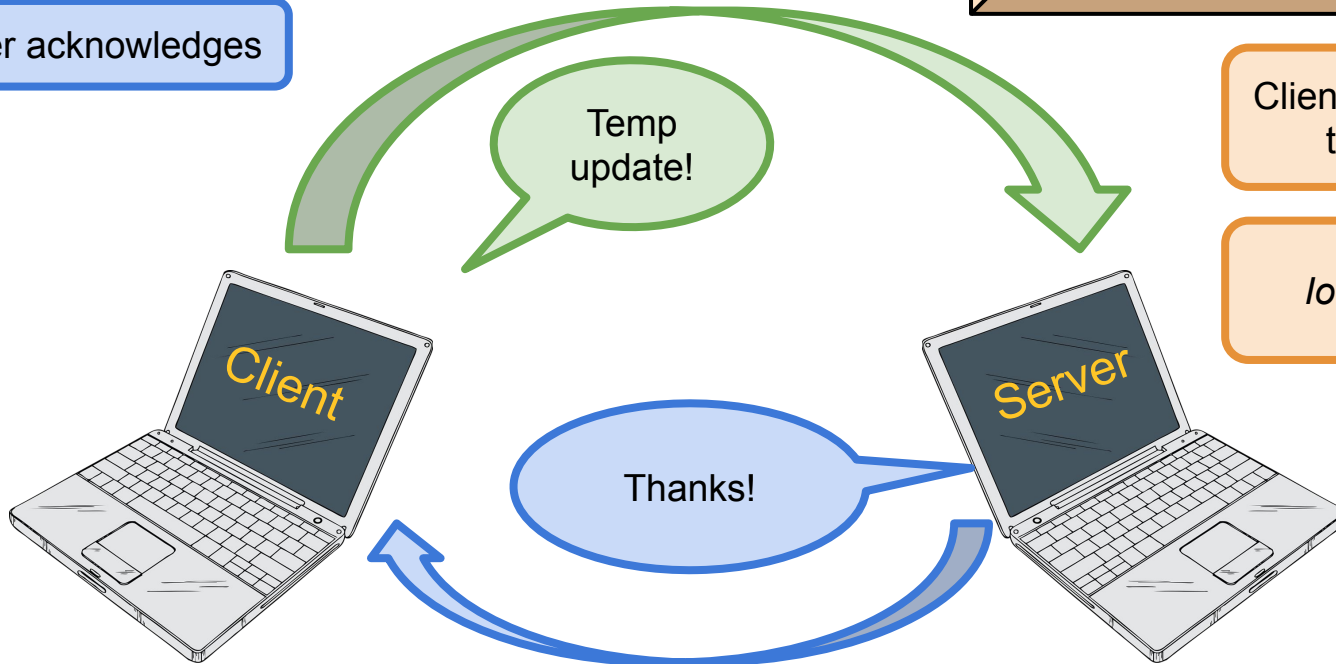
Client Push Model

1. Client sends update

2. Server acknowledges

Client *pushes* info
to Server

IoT sensors



SER 321

Client Socket

1.

2.

3.

4.

5.

6.

7.

8.

SER 321

Server Socket

1.

2.

3.

4.

5.

6.

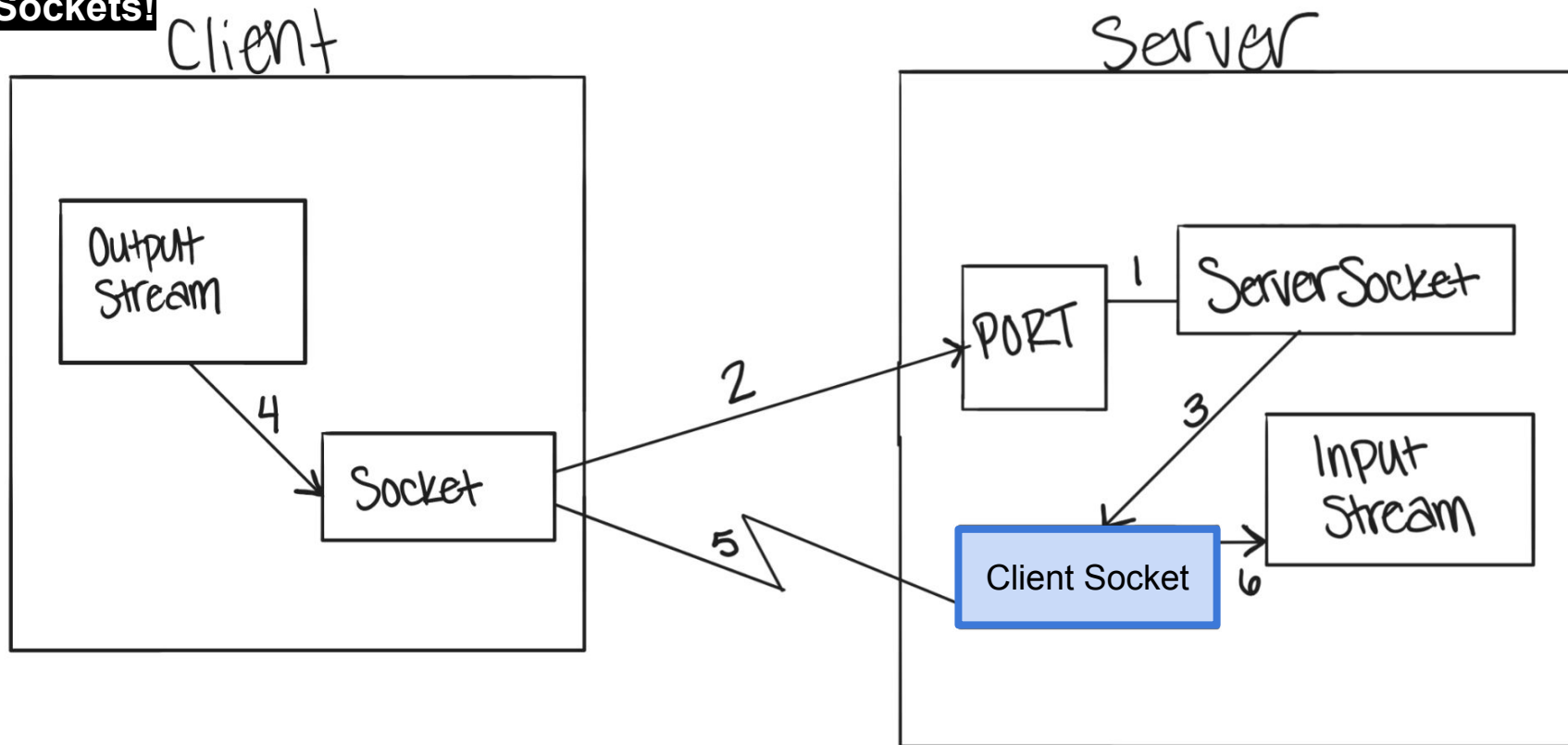
7.

8.

9.

SER 321

Sockets!



```
ServerSocket sock = new ServerSocket(port);
```

SER 321

Sockets!

Client

```
> Task :runServer
```

```
Server ready for connections
```

```
Server is listening on port: 9099
```

```
-----
```

```
Values of the ServerSocket Object:
```

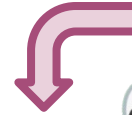
```
Inet Address: 0.0.0.0/0.0.0.0
```

```
Local Port: 9099
```

```
Server waiting for a connection
```

Server

```
ServerSocket.getLocalPort()
```



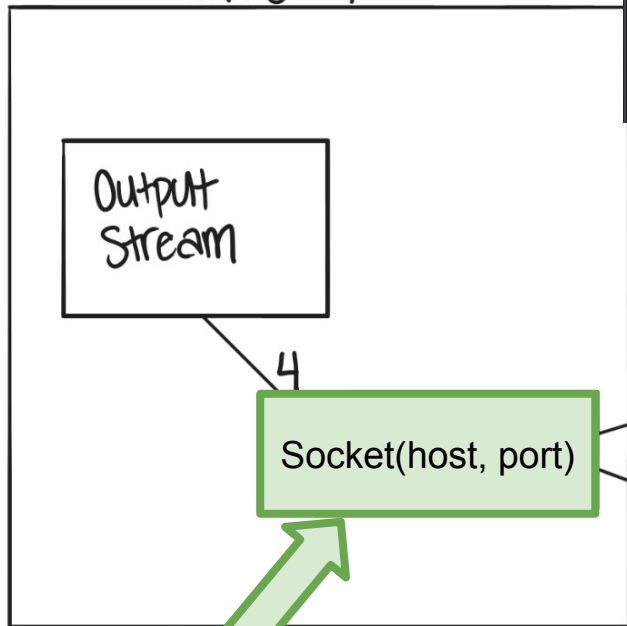
9099

sock

SER 321

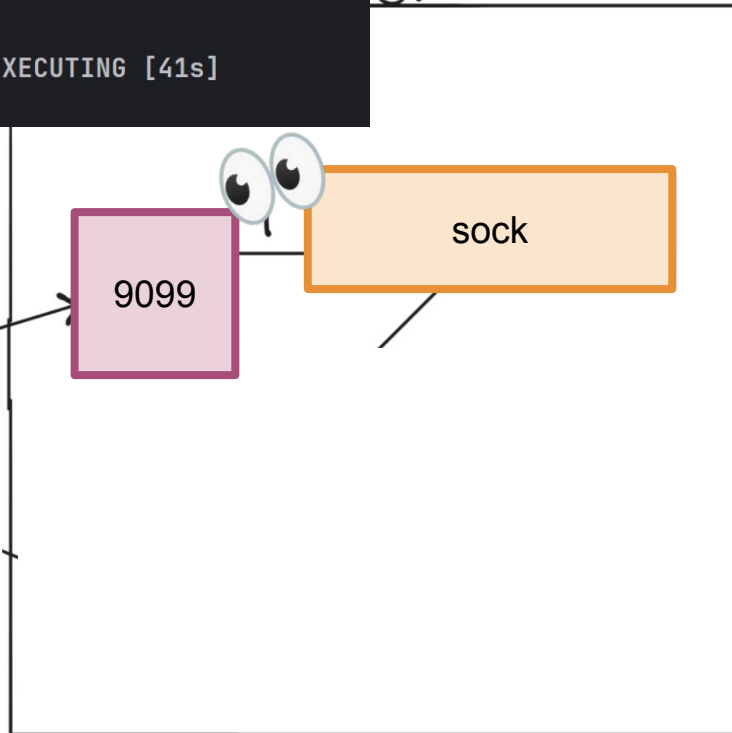
Sockets!

Client



```
> Task :runClient
Connected to server at localhost:9099
Values of the Socket Object for the Server:
    Host: /127.0.0.1
    Port: 9099
    Local Port: 14507
String to send>
<=====--> 75% EXECUTING [41s]
> :runClient
```

Server

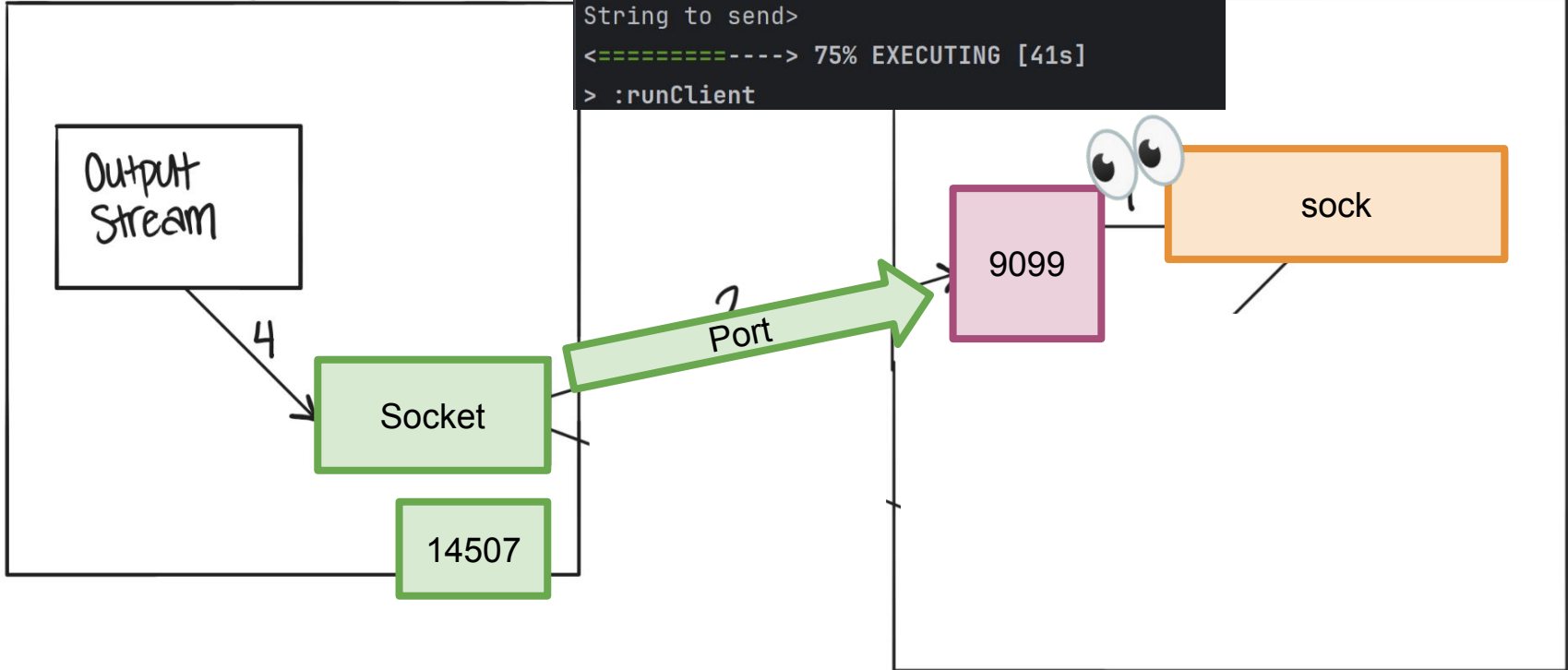


```
Socket server = new Socket(host, port);
```

SER 321

Sockets!

Client



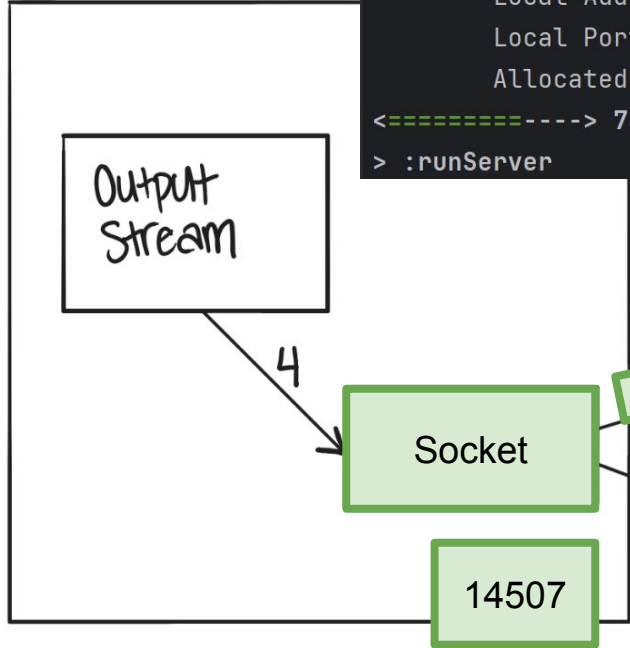
```
> Task :runClient
Connected to server at localhost:9099
Values of the Socket Object for the Server:
    Host: /127.0.0.1
    Port: 9099
    Local Port: 14507
String to send>
<=====--> 75% EXECUTING [41s]
> :runClient
```

er

SER 321

Sockets!

Client



Server waiting for a connection
Server connected to client



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

Values of the Client Socket Object after Connection:

Inet Address: /127.0.0.1

Local Address: /127.0.0.1

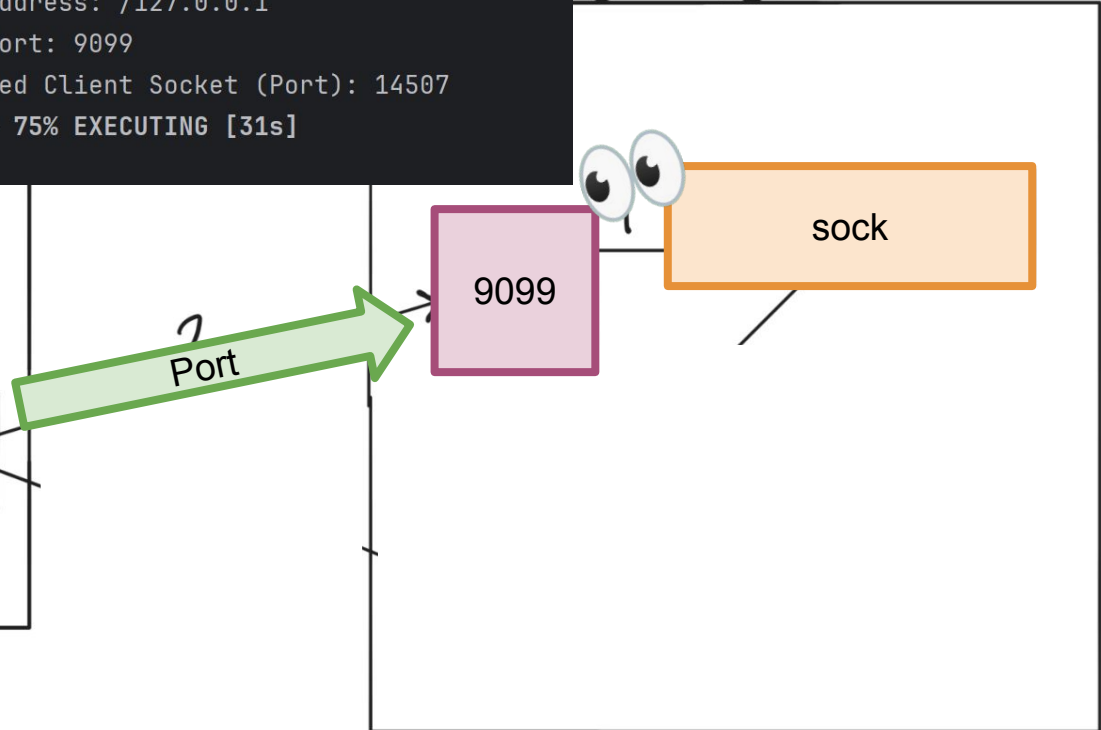
Local Port: 9099

Allocated Client Socket (Port): 14507

<=====--> 75% EXECUTING [31s]

> :runServer

Server

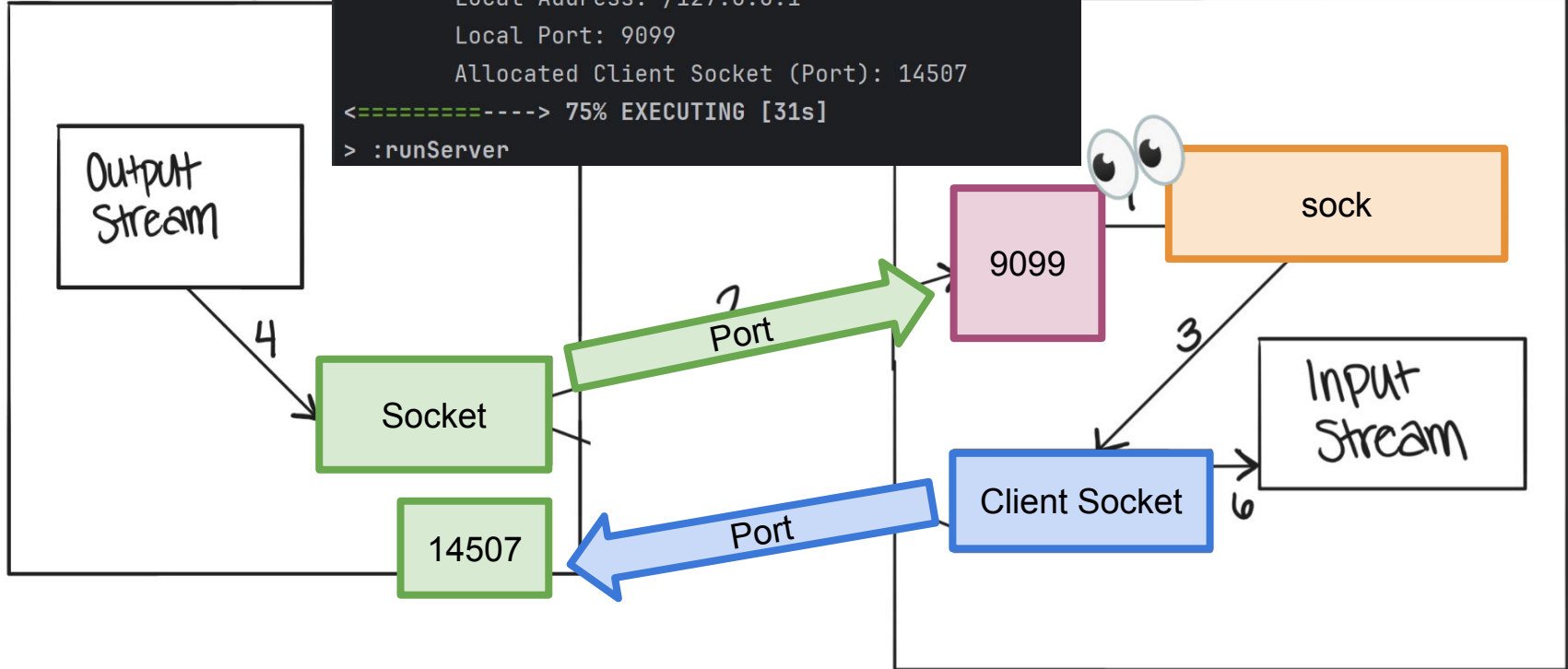


SER 321

Sockets!

Client

```
System.out.println("Values of the Client Socket Object after Connection:");
System.out.println("\tInet Address: " + clientSock.getInetAddress());
System.out.println("\tLocal Address: " + clientSock.getLocalAddress());
System.out.println("\tLocal Port: " + clientSock.getLocalPort());
System.out.println("\tAllocated Client Socket (Port): " + clientSock.getPort());
Local Address: /127.0.0.1
Local Port: 9099
Allocated Client Socket (Port): 14507
<=====--> 75% EXECUTING [31s]
> :runServer
```



SER 321

Scratch Space

Questions?

Survey:

<http://bit.ly/ASN2324>



Upcoming Events

SI Sessions:

- Thursday, March 28th at 7:00 pm MST
- Sunday, March 31st at 7:00 pm MST
- Monday, April 1st at 7:00 pm MST

Review Sessions:

- TBD

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

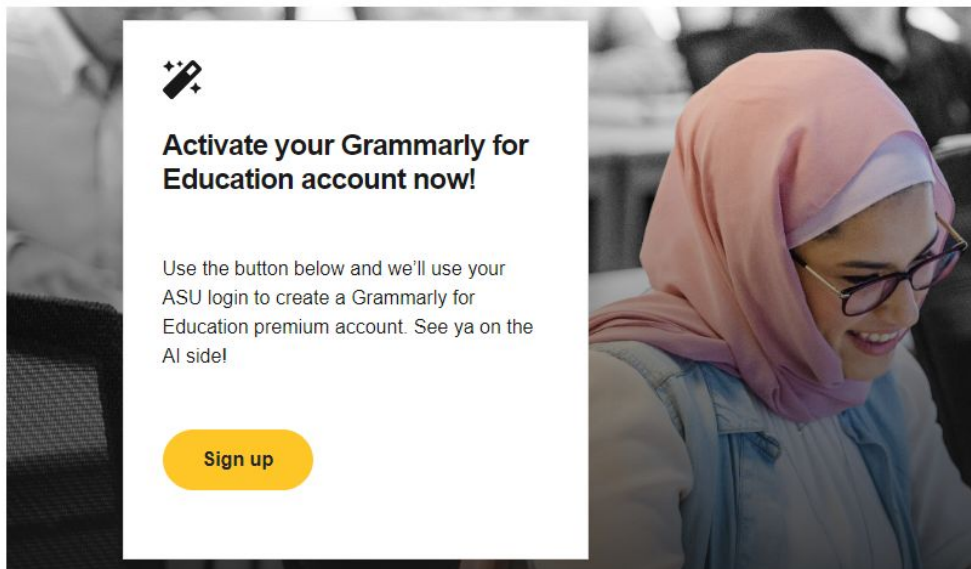
Computer Applications and Information Technology

 [Peer Community](#)

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

Expanded Writing Support Available

Including Grammarly for Education, at no cost!



tutoring.asu.edu/expanded-writing-support

*Available slots for this pilot are limited

Additional Resources

- [Course Repo](#)
- [Gradle Documentation](#)
- [GitHub SSH Help](#)
- [Linux Man Pages](#)
- [OSI Interactive](#)
- [MDN HTTP Docs](#)
 - [Requests](#)
 - [Responses](#)
- [JSON Guide](#)
- [org.json Docs](#)
- [javax.swing package API](#)
- [Swing Tutorials](#)