

SER 321 B Session

SI Session

Thursday, October 31st 2024

7:00 pm - 8:00 pm MST



Agenda



JSON Practice

Requests

Responses

Sockets!

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 Recognition

How many Objects?

How many Arrays?

How many Members?

```
{  
  "name": "lab3vue_act3_kgrinne3",  
  "version": "0.0.0",  
  "private": true,  
  "scripts": {  
    "dev": "vite",  
    "build": "vite build",  
    "preview": "vite preview"  
  },  
  "dependencies": {  
    "vue": "^3.3.4"  
  },  
  "devDependencies": {  
    "@vitejs/plugin-vue": "^4.3.1",  
    "vite": "^4.4.9"  
  }  
}
```

SER 321

JSON Practice

JSONObject json =

How would we...

Check for the name member?

boolean hasName =

Get the project name?

String name =

```
{
  "name": "lab3vue_act3_kgrinne3",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.3.1",
    "vite": "^4.4.9"
  }
}
```

SER 321

JSON Practice

JSONObject json =

How would we...

Get the dev value?

JSONObject scripts =

String dev=

```
{
  "name": "lab3vue_act3_kgrinne3",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.3.1",
    "vite": "^4.4.9"
  }
}
```

Step 1:
Get the scripts
Object

Step 2:
Get the dev
value

SER 321

JSON Practice

JSONObject json =

How would we...

Get the dev value?

JSONObject scripts =

String dev=

```
{
  "name": "lab3vue_act3_kgrinne3",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.3.1",
    "vite": "^4.4.9"
  }
}
```

Step 1:
Get the scripts
Object

Step 2:
Get the dev
value

SER 321

JSON Practice

JSONObject json =

How would we...

Build the json object?

You *could* do it all inline,
but let's go one object at
a time for clarity

```
{
  "name": "lab3vue_act3_kgrinne3",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.3.1",
    "vite": "^4.4.9"
  }
}
```

SER 321

JSON Practice

JSONObject json =

{

[JSON Guide](#)

```
"name": "lab3vue_act3_kgrinne3",
"version": "0.0.0",
"private": true,
"scripts": {
  "dev": "vite",
  "build": "vite build",
  "preview": "vite preview"
},
"dependencies": {
  "vue": "^3.3.4"
},
"devDependencies": {
  "@vitejs/plugin-vue": "^4.3.1",
  "vite": "^4.4.9"
}
}
```

SER 321

JSON Practice

```
JSONObject json = new JSONObject();  
json.put("name", "lab3vue_act3_kgrinne3");  
json.put("version", "0.0.0");  
json.put("private", "true");
```

```
{  
  "name": "lab3vue_act3_kgrinne3",  
  "version": "0.0.0",  
  "private": true,  
  "scripts": {  
    "dev": "vite",  
    "build": "vite build",  
    "preview": "vite preview"  
  },  
  "dependencies": {  
    "vue": "^3.3.4"  
  },  
  "devDependencies": {  
    "@vitejs/plugin-vue": "^4.3.1",  
    "vite": "^4.4.9"  
  }  
}
```

SER 321

JSON Practice

JSONObject scripts =

```
{
  "name": "lab3vue_act3_kgrinne3",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.3.1",
    "vite": "^4.4.9"
  }
}
```

SER 321

JSON Practice

```
JSONObject scripts = new JSONObject();
scripts.put("dev", "vite");
scripts.put("build", "vite build");
scripts.put("preview", "vite preview");
json.put("scripts", scripts.toMap());
```

```
{
  "name": "lab3vue_act3_kgrinne3",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.3.1",
    "vite": "^4.4.9"
  }
}
```

SER 321

JSON Practice

JSONObject depend =

```
{  
  "name": "lab3vue_act3_kgrinne3",  
  "version": "0.0.0",  
  "private": true,  
  "scripts": {  
    "dev": "vite",  
    "build": "vite build",  
    "preview": "vite preview"  
  },  
  "dependencies": {  
    "vue": "^3.3.4"  
  },  
  "devDependencies": {  
    "@vitejs/plugin-vue": "^4.3.1",  
    "vite": "^4.4.9"  
  }  
}
```

SER 321

JSON Practice

```
JSONObject depend = new JSONObject();  
depend.put("vue", "^3.3.4");  
json.put("dependencies", depend.toMap());
```

```
{  
  "name": "lab3vue_act3_kgrinne3",  
  "version": "0.0.0",  
  "private": true,  
  "scripts": {  
    "dev": "vite",  
    "build": "vite build",  
    "preview": "vite preview"  
  },  
  "dependencies": {  
    "vue": "^3.3.4"  
  },  
  "devDependencies": {  
    "@vitejs/plugin-vue": "^4.3.1",  
    "vite": "^4.4.9"  
  }  
}
```

SER 321

JSON Practice

JSONObject devDep =

[JSON Guide](#)

```
{
  "name": "lab3vue_act3_kgrinne3",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.3.1",
    "vite": "^4.4.9"
  }
}
```

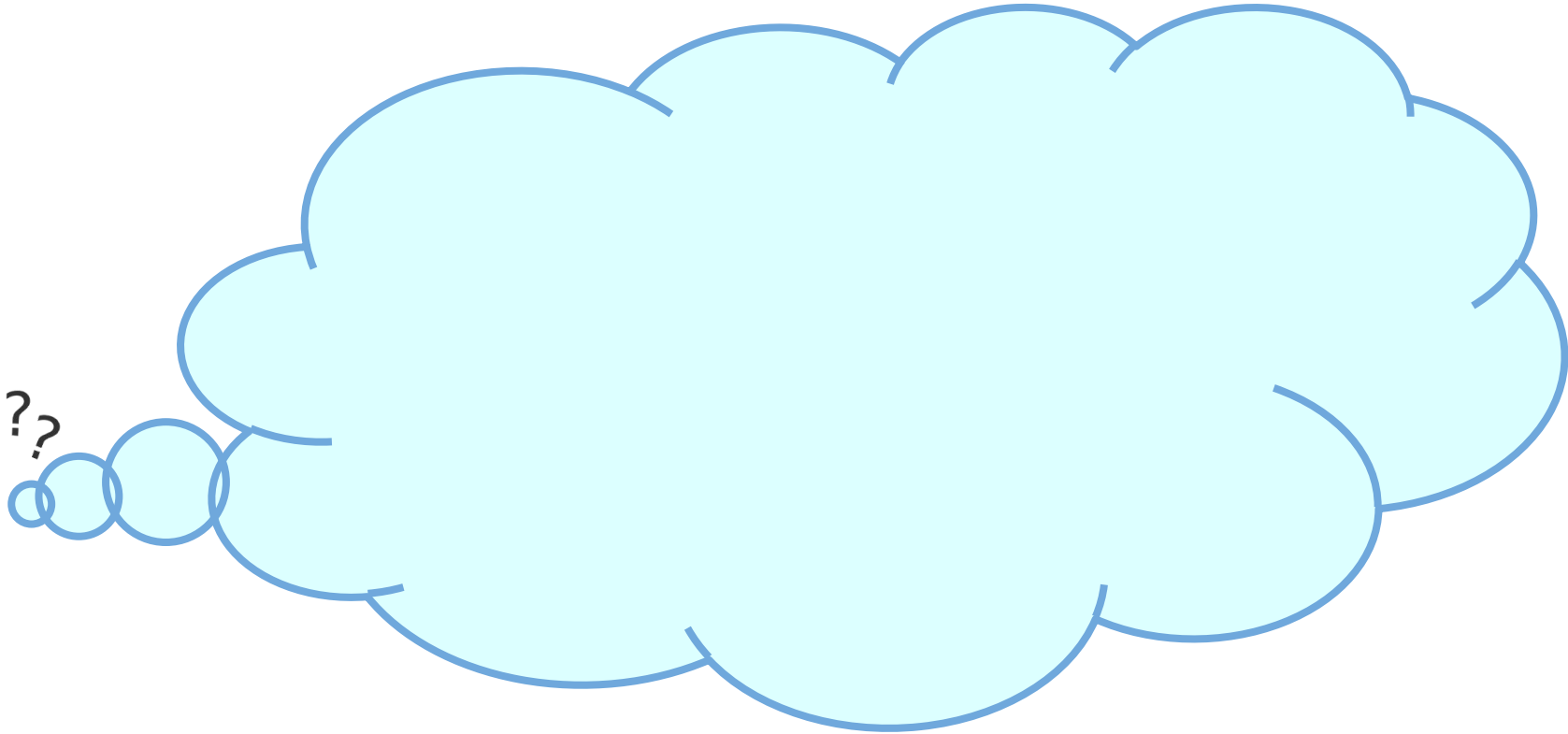

SER 321

JSON Practice

```
JSONObject devDep = new JSONObject();  
devDep.put("@vitejs/plugin-vue", "^4.3.1");  
devDep.put("vite", "^4.4.9");  
json.put("devDependencies", devDep.toMap());
```

```
{  
  "name": "lab3vue_act3_kgrinne3",  
  "version": "0.0.0",  
  "private": true,  
  "scripts": {  
    "dev": "vite",  
    "build": "vite build",  
    "preview": "vite preview"  
  },  
  "dependencies": {  
    "vue": "^3.3.4"  
  },  
  "devDependencies": {  
    "@vitejs/plugin-vue": "^4.3.1",  
    "vite": "^4.4.9"  
  }  
}
```

How do we enable a client/server connection?

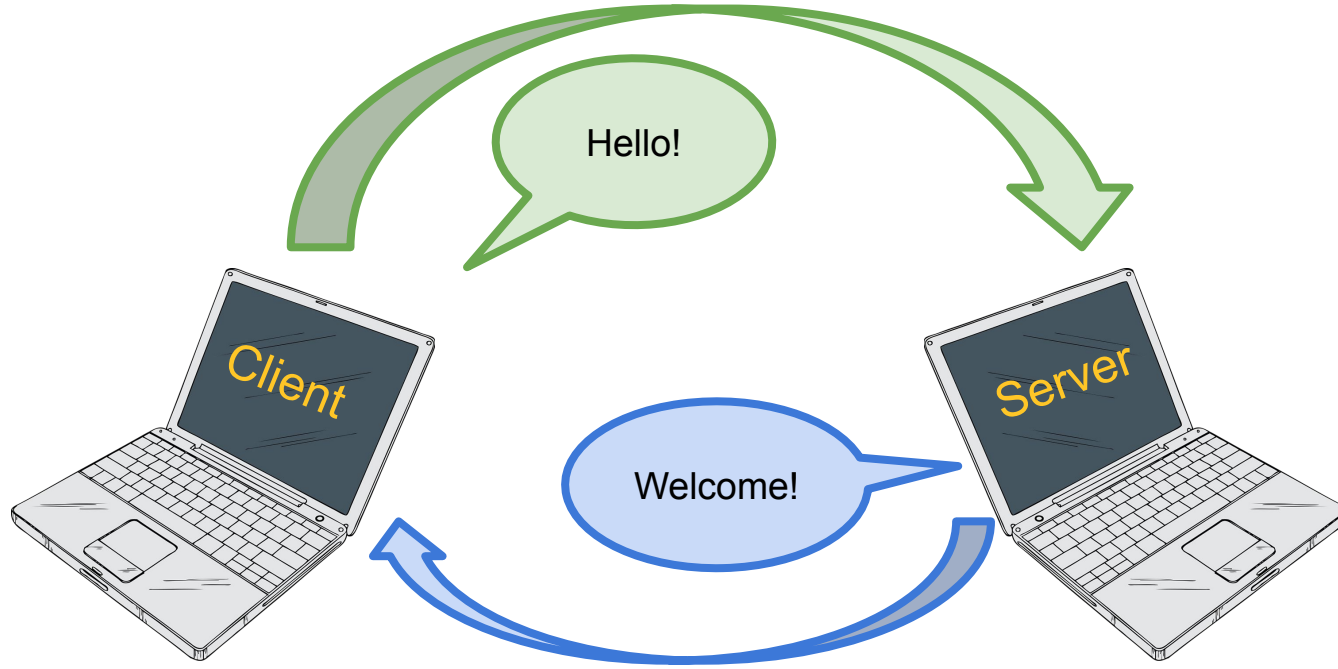


SER 321

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

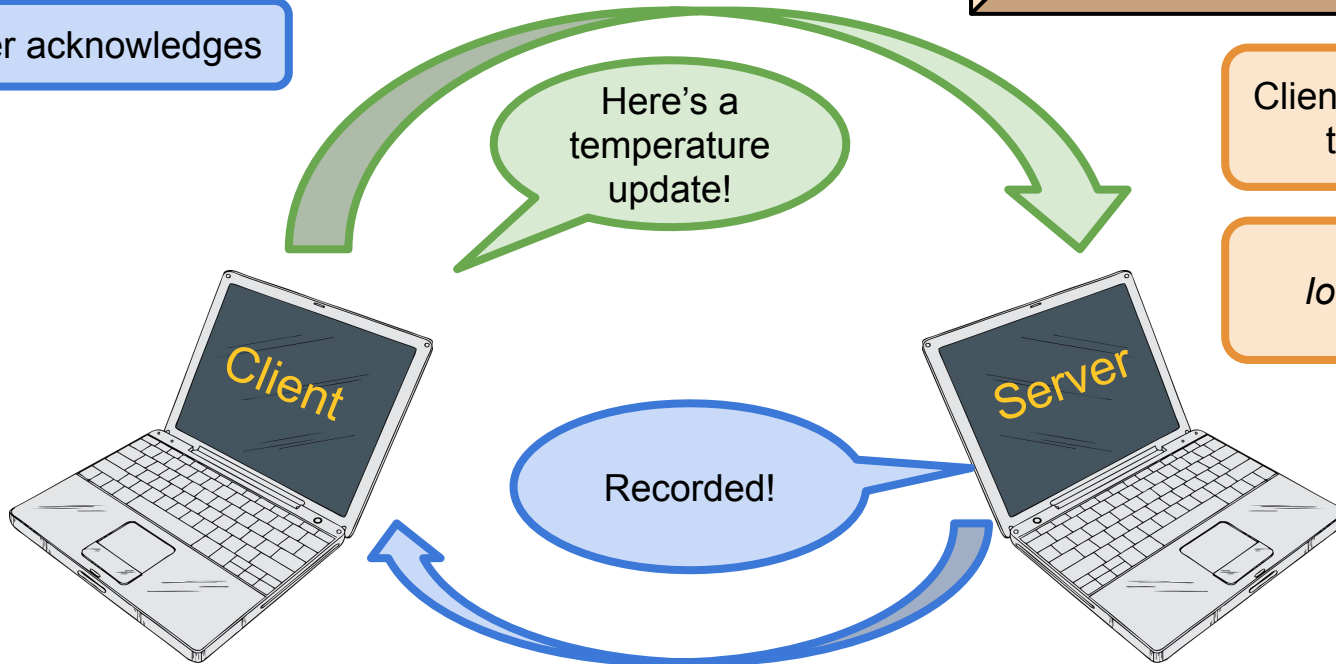
Client Push Model

1. Client sends update

2. Server acknowledges

Client *pushes* info
to Server

IoT sensors



SER 321

Client Socket

Steps for the **Client Socket**

1.

2.

3.

4.

5.

6.

7.

8.

Assign 3-1 Starter Code

SER 321

Client Socket

1. Define Params
2. Create Socket
3. **C ONLY** Create a struct for the address
4. Establish Connection
5. Send Message
6. Receive Message
7. Repeat #5 and #6 as needed
8. Close Socket

```
class SocketClient {
    static Socket sock = null; 4 usages
    static String host = "localhost"; 2 usages
    static int port = 8888; 2 usages
    static OutputStream out; 2 usages
    // Using and Object Stream here and a Data Stream as return. Could both
    // to show the difference. Do not change these types.
    static ObjectOutputStream os; 4 usages
    static DataInputStream in; 3 usages
    public static void main (String args[]) {

        if (args.length != 2) {...}

        try {
            host = args[0];
            port = Integer.parseInt(args[1]);
        } catch (NumberFormatException nfe) {
            System.out.println("[Port|sleepDelay] must be an integer");
            System.exit( status: 2);
        }

        try {
            connect(host, port); // connecting to server
            System.out.println("Client connected to server.");
            boolean requesting = true;
            while (requesting) {
                System.out.println("What would you like to do: 1 - echo, 2 - add, 3 - quit");
                Scanner scanner = new Scanner(System.in);
                int choice = Integer.parseInt(scanner.nextLine());
                // You can assume the user put in a correct input, you do not need
                // You can assume the user inputs a String when asked and an int when asked
                JSONObject json = new JSONObject(); // request object
                switch(choice) {
                    case 0:
                        System.out.println("Choose quit. Thank you for using our service");
                        requesting = false;
                        break;
                    case 1:
                        System.out.println("Choose echo, which String do you want to send?");
                        String message = scanner.nextLine();
                        json.put("type", "echo");
                        json.put("data", message);
                        break;
                    case 2:
                        break;
                }
                if(!requesting) {
                    continue;
                }

                // write the whole message
                os.writeObject(json.toString());
                // make sure it wrote and doesn't get cached in a buffer
                os.flush();

                // TODO: handle the response
                // - not doing anything other than printing payload
                // !! you will most likely need to parse the response for the payload
                String i = (String) in.readUTF();
                JSONObject res = new JSONObject(i);
                System.out.println("Got response: " + res);
                if (res.getBoolean( key: "ok")){
                    if (res.getString( key: "type").equals("echo")) {
                        System.out.println(res.getString( key: "echo"));
                    } else {
                        System.out.println(res.getInt( key: "result"));
                    }
                } else {
                    System.out.println(res.getString( key: "message"));
                }
            }
            // want to keep requesting services so don't close connection
            //overandout();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

SER 321

Server Socket

Steps for the **Server Socket**

1.

2.

3.

4.

5.

6.

7.

8.

9.

SER 321

Server Socket

Java
handles
a few
steps for
us...

1. Define Params

2. Create Socket

3. **C ONLY** Create a struct for the address

3-5. Mark Socket to Listen

5. Mark Socket to Listen for Connections

6. Wait for Connection

7. Handle Client Connection

8. Close Client Connection

9. Continue Listening for Connections

Assign 3-1 Starter Code

SER 321

Server Socket

1. Define Params

2. Create Socket

3-5. Mark Socket to Listen

6. Wait for Connection

7. Handle Client Connection

8. Close Client Connection

9. Continue Listening

1

2 & 3-5

9

6

```
public static void main (String args[]) {
```

```
    if (args.length != 1) {
```

```
        System.out.println("Expected arguments: <port(int)>");
```

```
        System.exit( status: 1);
```

```
    }
```

```
    try {
```

```
        port = Integer.parseInt(args[0]);
```

```
    } catch (NumberFormatException nfe) {
```

```
        System.out.println("[Port|sleepDelay] must be an integer");
```

```
        System.exit( status: 2);
```

```
    }
```

```
    try {
```

```
        //open socket
```

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

```
        System.out.println("Server ready for connections");
```

```
        /** Simple loop accepting one client and calling handling one request. */
```

```
        while (true){
```

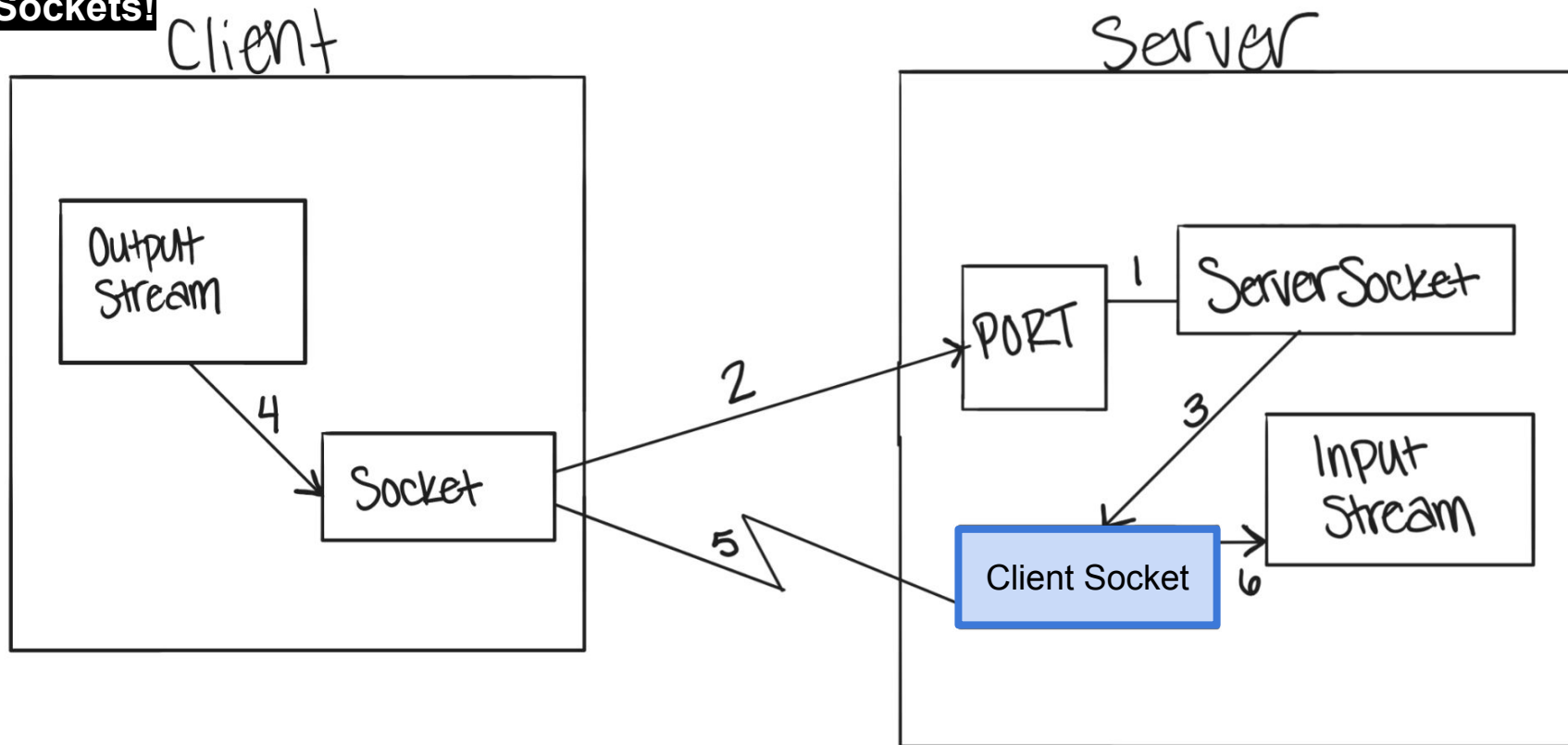
```
            System.out.println("Server waiting for a connection");
```

```
            sock = serv.accept(); // blocking wait
```

```
            System.out.println("Client connected");
```

SER 321

Sockets!

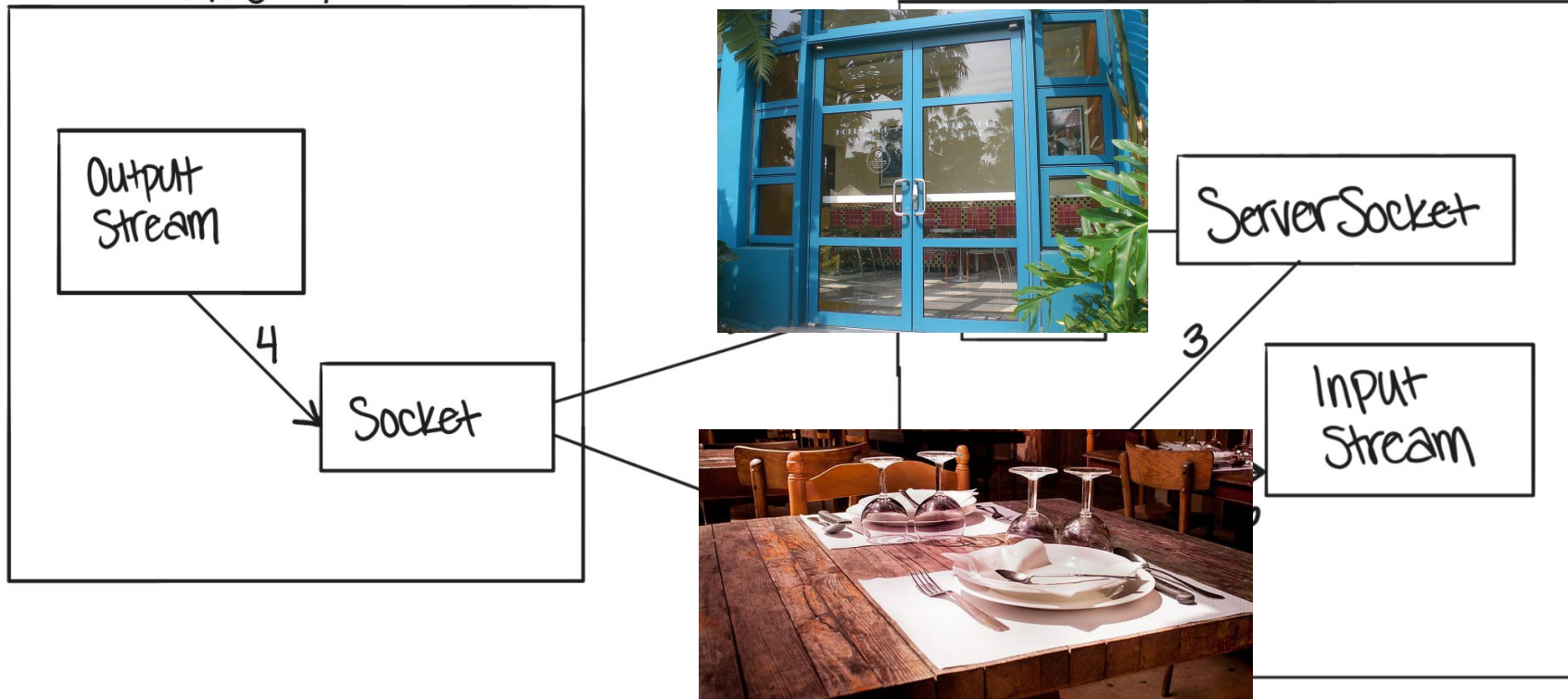


SER 321

Sockets!

Client

Server



SER 321

Scratch Space

Upcoming Events

SI Sessions:

- Sunday, November 3rd 2024 at 7:00 pm MST
- Tuesday, November 5th 2024 at 10:00 am MST
- Thursday, November 7th at 7:00 pm MST

Review Sessions:

- TBD

Don't forget about
Daylight Savings!

Questions?

Survey:

<https://asuasn.info/ASNSurvey>



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

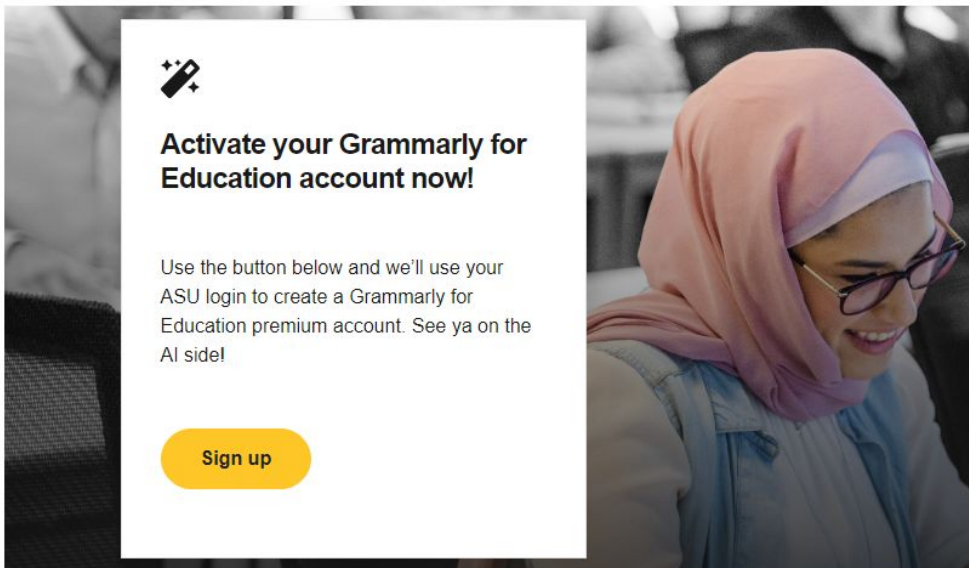
Computer Applications and Information Technology


 [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!





Activate your Grammarly for Education account now!

Use the button below and we'll use your ASU login to create a Grammarly for Education premium account. See ya on the AI side!

[Sign up](#)



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](#)