

# SER 321 C Session

**SI Session**

**Thursday, June 6th 2024**

*6:00 pm - 7:00 pm MST*

# Agenda



Working with JSON

Sockets!

Review Steps

Diagram Connection

# 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](https://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

## Error Handling

### *Excellent* question concerning Error Handling for Assign 3



**Dr. Mehlhase** 1 day ago

You do not have to add error handling to the client side **BUT your server should be able to handle all kinds of issues.** After all others can call your client and can write a client for your server and could send invalid requests and invalid values. E.g. they could send a string instead of an int, could send something out of range and your server should handle that well and not crash. The advantage of writing your own client for your server is that you can make sure that your client only sends correct things, often others write a client or make calls on your server though with their own things and thus you need to make sure your server can handle that. I hope that makes sense [@slmcgill](#).



1



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>  
}

*Check out the recording for the solution!*

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

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

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

[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 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

*Check out the recording for the solution and discussion!*

## 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!*

```
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

*Check out the recording for the solution and discussion!*



## 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!*

```
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

*Check out the recording for the solution and discussion!*

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();
```

*Check out the recording for the solution and discussion!*

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

Target Response

Then what do you do?

**SER 321**

**Client Socket**

## Steps for the **Client Socket**

1. *Check out the recording for the discussion!*

2.

3.

4.

5.

6.

7.

8.

**SER 321**

**Server Socket**

# Steps for the **Server Socket**

1. *Check out the recording for the discussion!*

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);
```

```
    }
```

*Check out the recording for the discussion!*

```
    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");
```

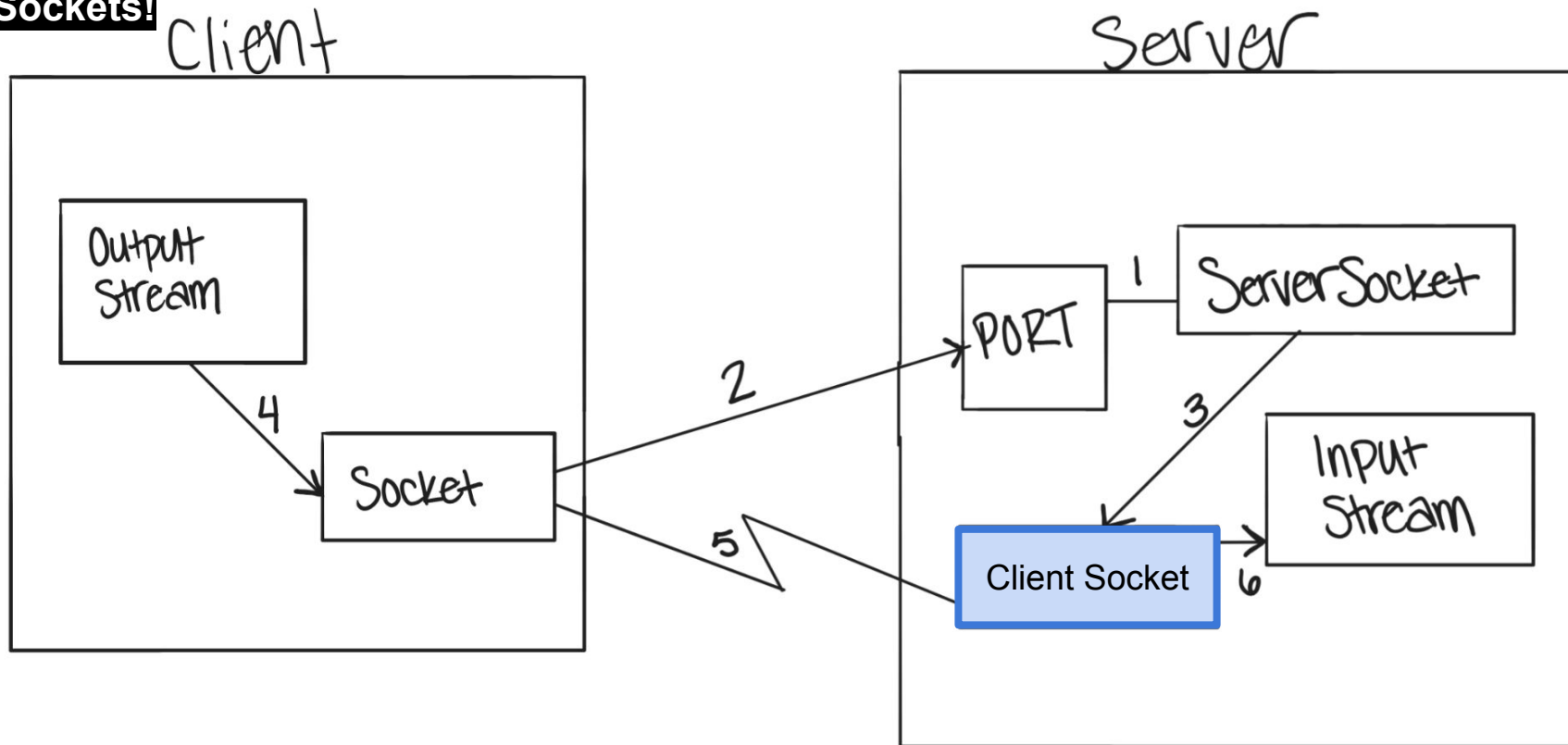
```
            sock = serv.accept(); // blocking wait
```

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

# SER 321

## Sockets!

*Check out the recording for the discussion!*



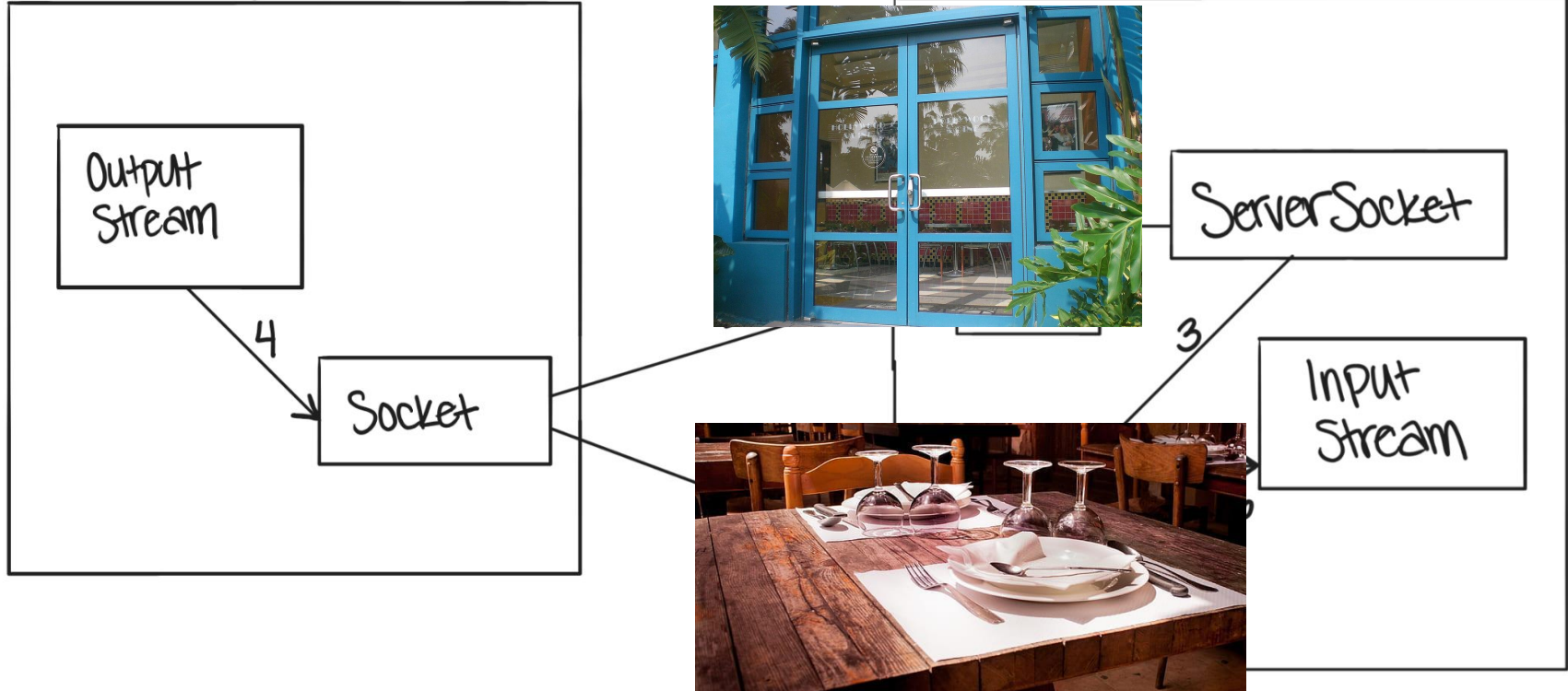
# SER 321

## Sockets!

Check out the recording for the discussion!

Client

Server





**SER 321**

**Scratch Space**

# Questions?



## Survey:

<http://bit.ly/ASN2324>



## Upcoming Events

### SI Sessions:

- Sunday, June 9th at 6:00 pm MST
- Monday, June 10th at 6:00 pm MST
- Thursday, June 13th at 6:00 pm MST

### Review Sessions:

- Review Session - **Wednesday**, July 3rd at 6:00 pm MST (2 hr Session)
- Q&A Session - Sunday, July 7th at 6:00 pm MST (Final Session)

# 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](https://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

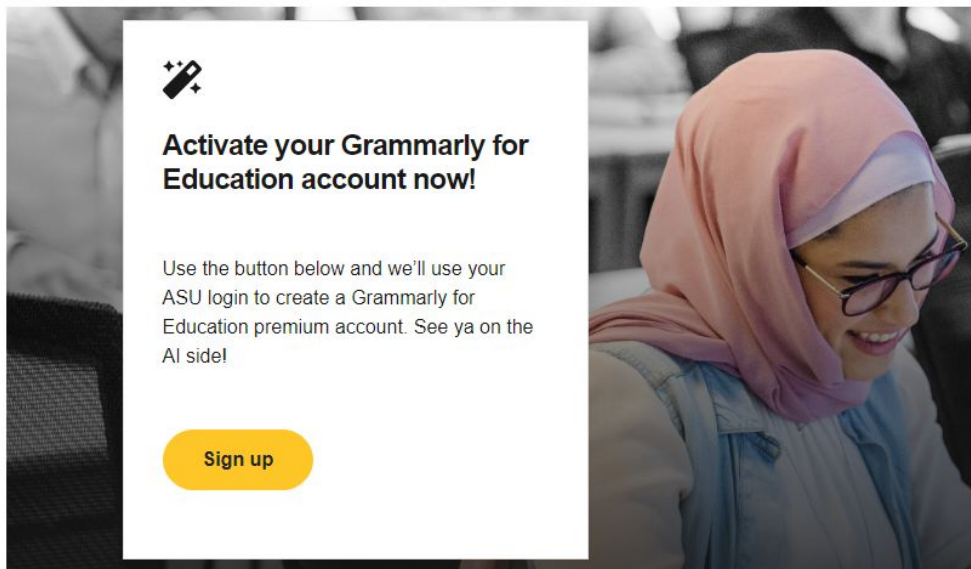
Computer Applications and Information Technology

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](https://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](#)