SER 321 B Session

SI Session

Monday, November 6th 2023

4:00 - 5:00 pm MST

Agenda

- Serialization
  - JSON Review
  - Protocol Buffers
- Managing Concurrency

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
Java IO Streams

What's the difference?

- Buffered Streams
- Data Streams
- Object Streams

Check out the recording for the solution!

SER 321
Serialization

What is it? 🤔

“Translating data structures or object states for storage or transmission”

Main Forms:

- XML
- JSON
- Java Serialization (Objects)
- Protocol Buffers

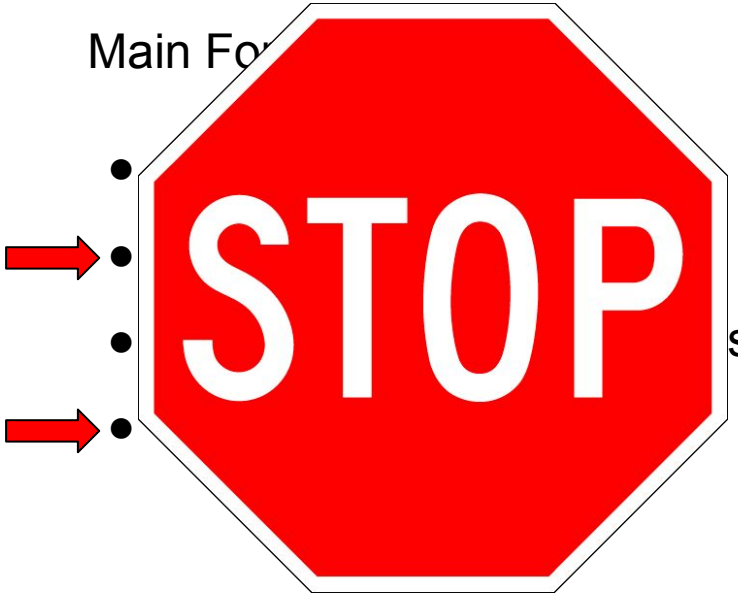
SER 321
Serialization

What is it? 🤔

“Translating data structures or object states for storage or transmission”

Main Fo

Text or Binary ?



Which is more efficient and less expensive?

Check out the recording for the solution!

SER 321
Serialization

What is it? 🤔

“Translating data structures or object states for storage or transmission”

Main Forms: Text or Binary ?

- XML
- JSON
- Java Serialization (Objects)
- Protocol Buffers

Check out the recording for the solution!

org.json API Docs

JSON Helper

SER 321

Serialization - JSON

Spitfire Review!

“Katie”

Data is put into…

org.json API Docs

JSON Helper

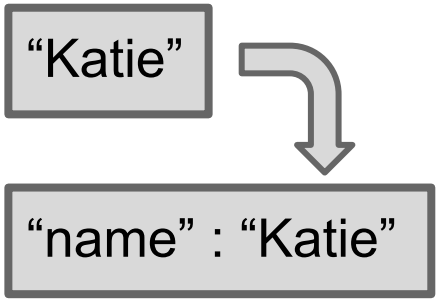
SER 321

Serialization - JSON

Spitfire Review!

Data is put into…

Name : Value pairs



org.json API Docs

JSON Helper

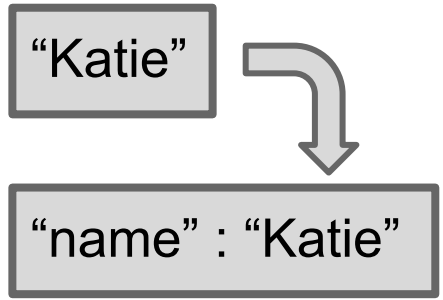
SER 321

Serialization - JSON

Spitfire Review!

Data is put into…

Name : Value pairs AKA



org.json API Docs

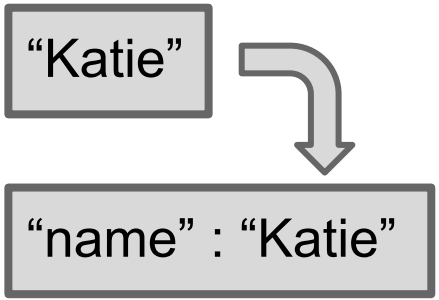
JSON Helper

SER 321

Serialization - JSON

Spitfire Review!

Data is put into…



org.json API Docs

JSON Helper

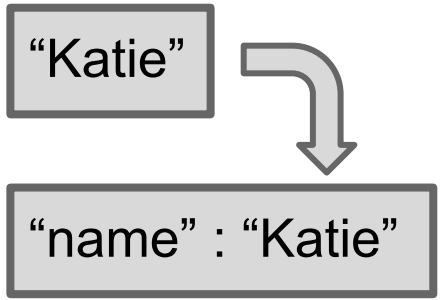
SER 321

Serialization - JSON

Spitfire Review!

Data is put into…

Name : Value pairs AKA Members



What uses curly braces?

{ }

org.json API Docs

JSON Helper

SER 321

Serialization - JSON

Spitfire Review!

Data is put into…

Name : Value pairs AKA Members

"Katie"

"name" : "Katie"

What uses curly braces?

{ }

Objects

org.json API Docs

JSON Helper

SER 321

Serialization - JSON

Spitfire Review!

Data is put into…

Name : Value pairs AKA Members

"Katie"

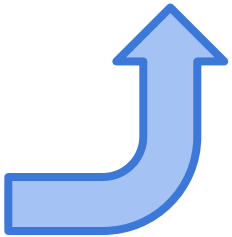
"name" : "Katie"

What uses curly braces?

{ }

Objects

What can objects hold?



org.json API Docs

JSON Helper

SER 321

Serialization - JSON

Spitfire Review!

Data is put into…

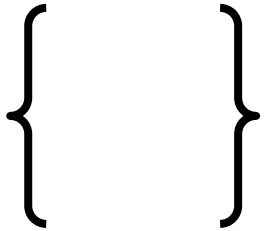
Name : Value pairs AKA Members

"Katie"

"name" : "Katie"

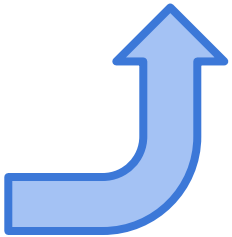
What uses curly braces?

Objects



What can objects hold?

Members or



org.json API Docs

JSON Helper

SER 321

Serialization - JSON

Spitfire Review!

Data is put into…

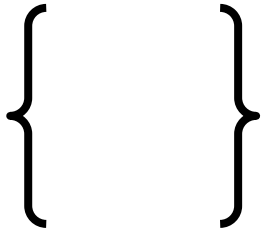
Name : Value pairs AKA Members

"Katie"

"name" : "Katie"

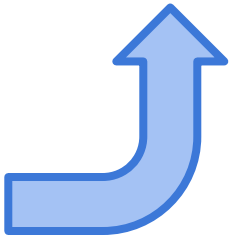
What uses curly braces?

Objects



What can objects hold?

Members or Objects



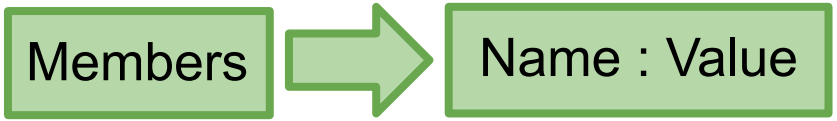
org.json API Docs

JSON Helper

SER 321

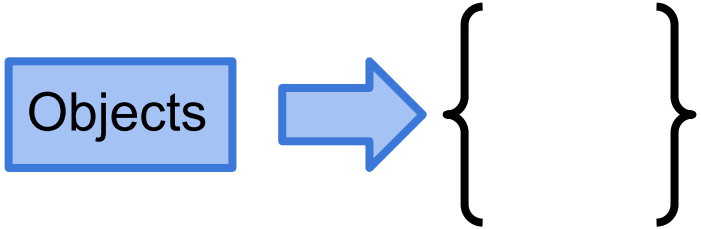
Serialization - JSON

Spitfire Review!



What uses brackets?

[ ]



org.json API Docs

JSON Helper

SER 321

Serialization - JSON

Spitfire Review!

Members → Name : Value

What uses brackets?

Arrays [ ]

Objects { }

org.json API Docs

JSON Helper

SER 321

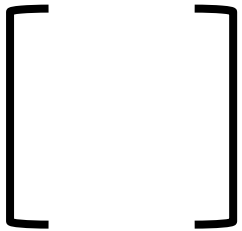
Serialization - JSON

Spitfire Review!

Members → Name : Value

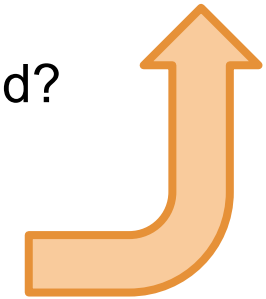
What uses brackets?

Arrays



Objects → { }

What can arrays hold?



org.json API Docs

JSON Helper

SER 321

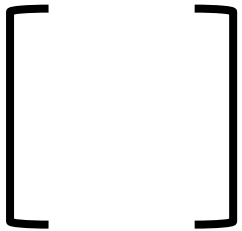
Serialization - JSON

Spitfire Review!

Members → Name : Value

What uses brackets?

Arrays



Objects → { }

What can arrays hold?

Any Valid Value

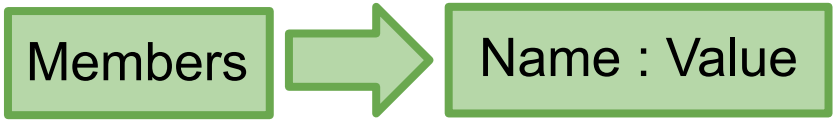
org.json API Docs

JSON Helper

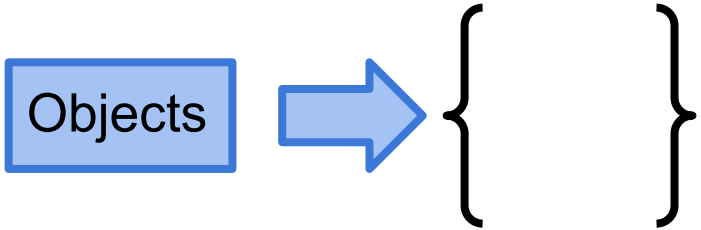
SER 321

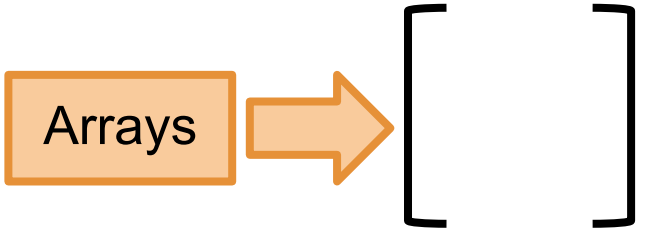
Serialization - JSON

Spitfire Review!



What is a valid value?





Strings	Numbers	Objects
Null	Booleans	Arrays

Questions on JSON?

SER 321
Protocol Buffers

Require a few steps before use - listed in the README

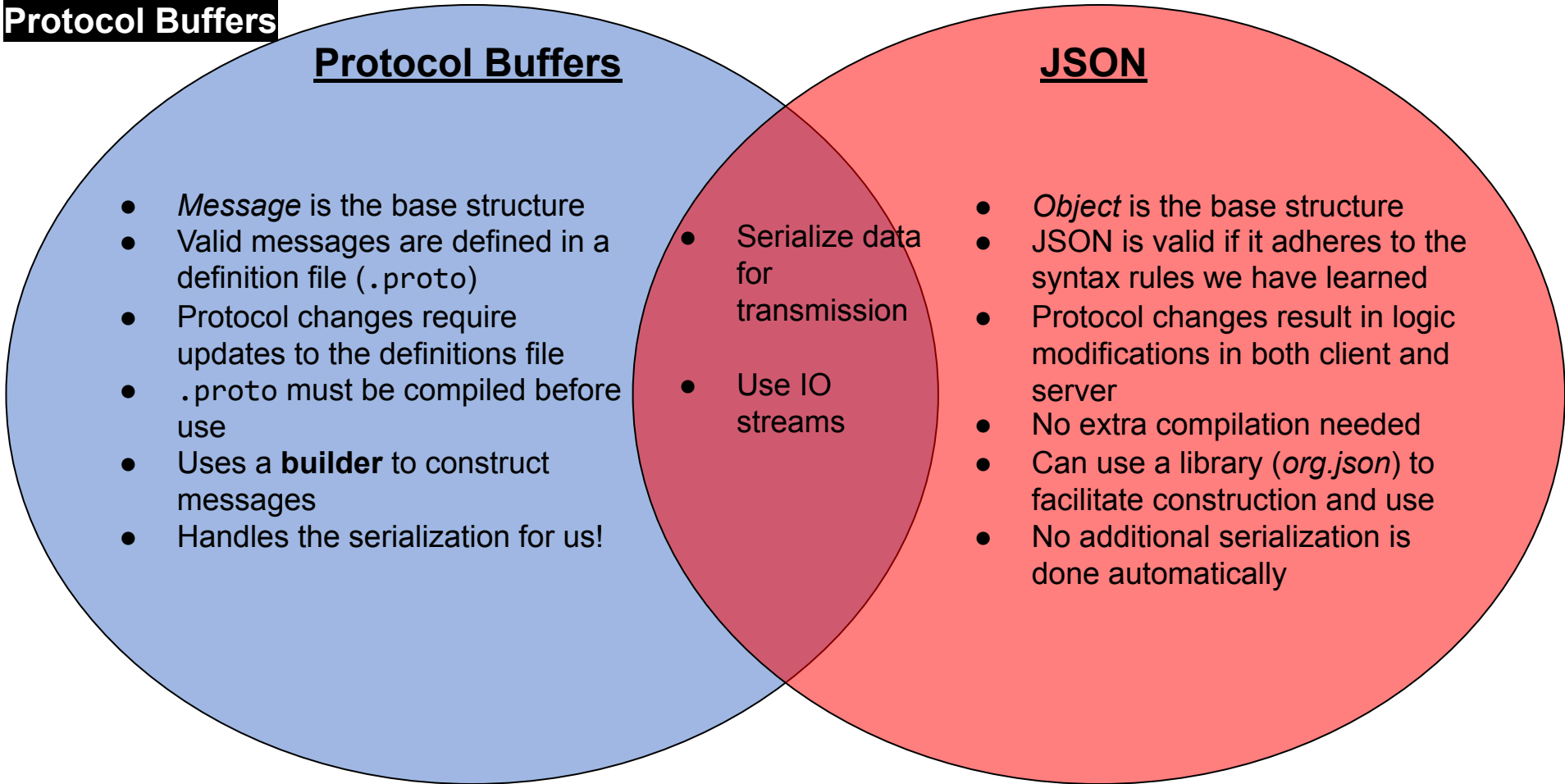
Step 1: Generate the Protobuf Code

```
gradle generateProto
```

Step 2: If you use IntelliJ, add the following to your build.gradle file:

```
sourceSets {
      main {
            java {
                  srcDirs 'build/generated/source/proto/main/java'
            }
      }
}
```

SER 321
Protocol Buffers



SER 321

Serialization - Protocol Buffers

How many message type definitions do we have here?

So what does a `Response` message need to contain?

Need to check the protocol!

```
## Protocol Elaboration
```

Check out the recording for the full walkthrough!

Starter Code for Assign 4-2

```
message Response {
  enum ResponseType {
    HELLO = 0;
    LEADERBOARD = 1;
    TASK = 2;
    WON = 3;
    ERROR = 4;
    BYE = 5;
  }

  optional ResponseType responseType = 1 [default = HELLO];
  optional string hello = 2;

  repeated Leader leaderboard = 3;

  optional string task = 4;
  optional string image = 5;

  optional bool eval = 6;
  optional string message = 7;
}

message Leader {
  optional string name = 1;
  optional int32 wins = 2;
  optional int32 logins = 3;
}
```

SER 321

Serialization - Protocol Buffers

Let's do an error example…

So what does an `Error Response` message need to contain?

Need to check the protocol!

Check out the recording for the full walkthrough!

Starter Code for Assign 4-2

```
message Response {
  enum ResponseType {
    HELLO = 0;
    LEADERBOARD = 1;
    TASK = 2;
    WON = 3;
    ERROR = 4;
    BYE = 5;
  }

  optional ResponseType responseType = 1 [default = HELLO];
  optional string hello = 2;

  repeated Leader leaderboard = 3;

  optional string task = 4;
  optional string image = 5;

  optional bool eval = 6;
  optional string message = 7;
}

message Leader {
  optional string name = 1;
  optional int32 wins = 2;
  optional int32 logins = 3;
}
```

SER 321

Serialization - Protocol Buffers

Let's do an error example…

So what does an `Error Response` message need to contain?

```
SV Response
```
ResponseType: ERROR
RequiredFields: message (description of error), type
```
Some error types to use:
1 - required field missing -- in message name the field
2 - request not supported -- in message name the request that is not supported
0 - any other errors, in this case the message will just be displayed
```

```
message Response {
  enum ResponseType {
    HELLO = 0;
    LEADERBOARD = 1;
    TASK = 2;
    WON = 3;
    ERROR = 4;
    BYE = 5;
  }

  al ResponseType responseType = 1 [default = HELLO];
  al string hello = 2;

  ed Leader leaderboard = 3;

  al string task = 4;
  al string image = 5;

  al bool eval = 6;
  al string message = 7;
}

message Leader {
  optional string name = 1;
  optional int32 wins = 2;
  optional int32 logins = 3;
}
```

Check out the recording for the full walkthrough!

Starter Code for Assign 4-2

SER 321

Serialization - Protocol Buffers

ResponseType

Message

```
message Response {
  enum ResponseType {
    HELLO = 0;
    LEADERBOARD = 1;
    TASK = 2;
    WON = 3;
    ERROR = 4;
    BYE = 5;
  }
```

Let's do an error example…

So what does an `Error Response` message need to contain?

```
SV Response
```
ResponseType: ERROR
RequiredFields: message (description of error), type
```
Some error types to use:
1 - required field missing -- in message name the field
2 - request not supported -- in message name the request that is not supported
0 - any other errors, in this case the message will just be displayed
```

```
    al ResponseType responseType = 1 [default = HELLO];
    al string hello = 2;

    ed Leader leaderboard = 3;

    al string task = 4;
    al string image = 5;

    al bool eval = 6;
    al string message = 7;
}

message Leader {
  optional string name = 1;
  optional int32 wins = 2;
  optional int32 logins = 3;
}
```

Check out the recording for the full walkthrough!

Starter Code for Assign 4-2

SER 321

Serialization - Protocol Buffers

ResponseType

Message

Let's do an error example…

So what does an `Error Response` message need to contain?

```
SV Response
```
ResponseType: ERROR
RequiredFields: message (description of error), type
```
Some error types to use:
1 - required field missing -- in message name the field
2 - request not supported -- in message name the request that is not supported
0 - any other errors, in this case the message will just be displayed
```

```
message Response {
  enum ResponseType {
    HELLO = 0;
    LEADERBOARD = 1;
    TASK = 2;
    WON = 3;
    ERROR = 4;
    BYE = 5;
  }

  al ResponseType responseType = 1 [default = HELLO];
  al string hello = 2;

  ed Leader leaderboard = 3;

  al string task = 4;
  al string image = 5;

  al bool eval = 6;
  al string message = 7;
}

message Leader {
  optional string name = 1;
  optional int32 wins = 2;
  optional int32 logins = 3;
}
```

Check out the recording for the full walkthrough!

Starter Code for Assign 4-2

SER 321

Serialization - Protocol Buffers

How would we send the first task to the client?

```
SV Response
Server responds with a message specifying if the game is joined or started.
```
```
ResponseType: TASK
Required Fields: image, task
```

Check out the recording for the full walkthrough!

Starter Code for Assign 4-2

```
message Response {
  enum ResponseType {
    HELLO = 0;
    LEADERBOARD = 1;
    TASK = 2;
    WON = 3;
    ERROR = 4;
    BYE = 5;
  }

  optional ResponseType responseType = 1 [default = HELLO];
  optional string hello = 2;

  repeated Leader leaderboard = 3;

  optional string task = 4;
  optional string image = 5;

  optional bool eval = 6;
  optional string message = 7;
}

message Leader {
  optional string name = 1;
  optional int32 wins = 2;
  optional int32 logins = 3;
}
```

SER 321

Serialization - Protocol Buffers

```
Response response2 = Response.newBuilder()
    .setResponseType(Response.ResponseType.TASK)
    .setImage(game.getImage())
    .setTask("Great task goes here")
    .build();
```

What if I need to construct the message slowly?

```
Leader leader2 = Leader.newBuilder()
    .setName("name2")
    .setWins(1)
    .setLogins(1)
    .build();
```

Use `Response.Builder` much like a StringBuilder!

```
Response.Builder res = Response.newBuilder()
    .setResponseType(Response.ResponseType.LEADERBOARD);
```

Is that it?

```
res.addLeaderboard(leader);
res.addLeaderboard(leader2);
```

Check out the recording for the full walkthrough!

```
response3.writeDelimitedTo(out);
```

Starter Code for Assign 4-2

Questions?

Survey:

https://bit.ly/asn_survey

32

Upcoming Events

SI Sessions:

- Thursday, November 9th 2023 at 7:00 pm MST
- Sunday, November 12th 2023 at 7:00 pm MST
- Monday, November 13th 2023 at 4:00 pm MST

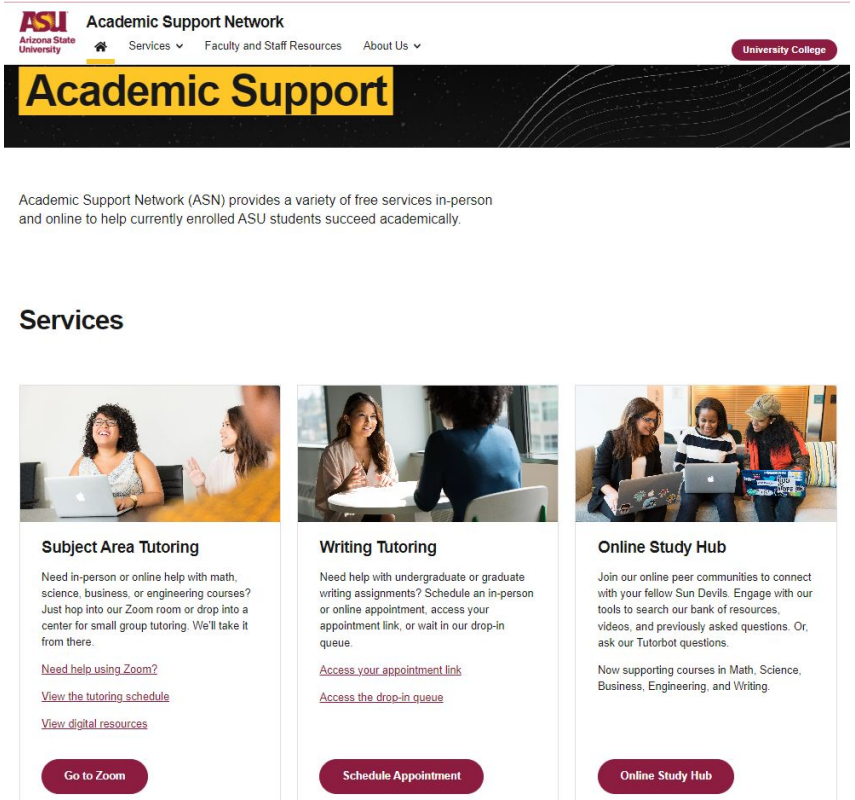
Review Sessions:

- Finishing up the Scheduling Poll today, will post tonight or tomorrow!

More Questions?

Check out our other resources!

tutoring.asu.edu

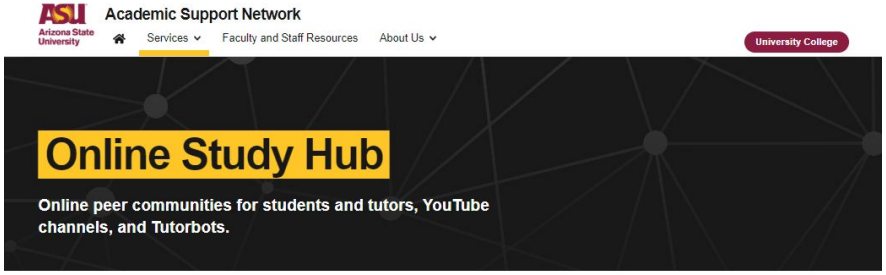


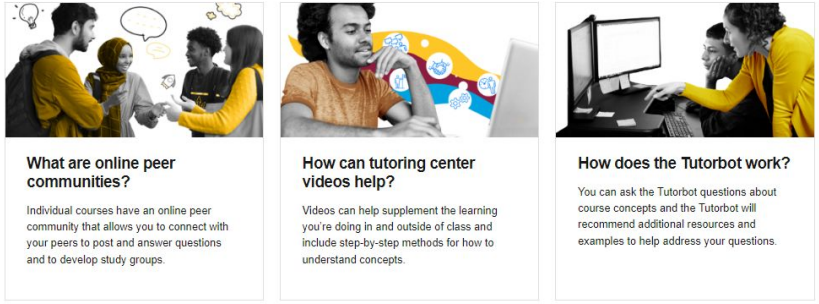
1. **Click on ‘Go to Zoom’ to log onto our Online Tutoring Center.**
2. **Click on ‘View the tutoring schedule’ to see when tutors are available for specific courses.**

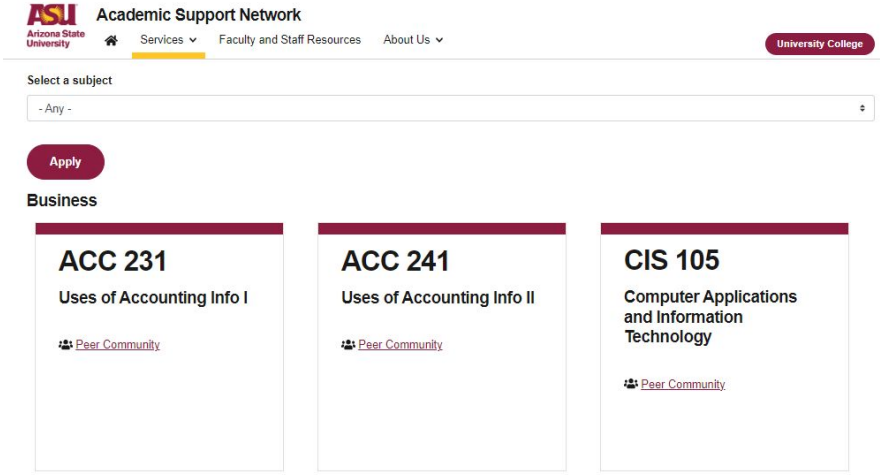
More Questions?

Check out our other resources!

tutoring.asu.edu/online-study-hub







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