CS 240: Server Facade Transcript

[00:00:00] **INSTRUCTOR:** So in phase five, this is where you're going to have to write some code that allows your client to communicate with your server. So obviously, there's going to be a lot of interaction between your chess client and your chess server.

[00:00:16] And in the first two parts of the user interface for chess, essentially what is happening is the user is typing in commands, and then based on the command that the user types in, the client is going to make a web API called your server to actually execute the command requested by the user.

[00:00:33] And so you're going to need to write code in your client that calls your web APIs. And so to encapsulate that code that calls your web APIs, we require you to implement a class called a server facade.

[00:00:52] A server facade is really just a class that calls the server. So for example, if a user tries to log in, that means the client is going to need to send a login request to the server's login API, and then the server will return with a response indicating whether that succeeded or failed.

[00:01:13] And so the server facade class will contain the code that creates the HTTP request that goes to the server and then receives the HTTP response back from the server and then processes that. Now in a previous video, we talked about how to write a class like this.

[00:01:33] So it was in a video called client HTTP. And so you want to review that video if you don't remember what that was all about. And so using that information, you'll create a class that looks something like this.

[00:01:47] So you type class server facade. And the methods that are on the server facade would map directly to the web APIs that are supported by the server. So let's see, let's do a register. So it might have a, for example, a register method.

[00:02:13] It takes a register request as a parameter and then it returns a register result as an output. And of course inside the register method that's where you would use the HTTP URL connection class in Java to create an HTTP request that calls the register API on the server, gets back a result, then returns the result.

[00:02:41] So the client, whenever it wants to call the server's web API, it would simply create a server facade object and then call whatever method it wants to call. And then the facade would do all the work of communicating with the server and returning back the result.

[00:02:57] So as far as the rest of the client is concerned, it doesn't really know how we're calling the server. It doesn't know that we're using HTTP. It doesn't know that we're using JSON to format their requests and responses.

[00:03:10] All that information is encapsulated inside the server facade. So if those details ever needed to change, the only class in the client that would be affected would be the server facade. And of course, you'll have several other methods here as well.

[00:03:25] You might have a join result, join a game, join request, et cetera. And so the server facade is probably the first thing I would build if I were working on this project on phase five. I would probably implement my server facade first and get it working.

Start visual description. Instructor types the following code:

Class ServerFacade {

Public RegisterResult register(RegisterRequest request) {…}

Public JoinResult join(JoinRequest request) {…}

End visual description.

[00:03:51] And then based on that working facade, then you can build the rest of the client around that. So consider that as probably a good place to begin your implementation. Now, one thing we can do if we want more help with the server facade is we can look at the pet shop example again.

[00:04:09] So if we go to the pet shop project, if we go to the shared module, go into the server package, there is the pet shop server facade. And this is what we looked at in the client HTTP video. But you can see here that it provides lots of example code for what a server facade would look like.

Start visual description. Instructor shows the server façade founder under the server folder for the pet shop project. End visual description.

[00:04:39] And so you can see all these methods for pet shop or just calling the web APIs on the pet shop server. It's got this nice make request method down here that you can use to actually send the requests and receive the responses and so on.

[00:04:51] So I would go back and study the pet shop server facade before I implemented my chess server facade.