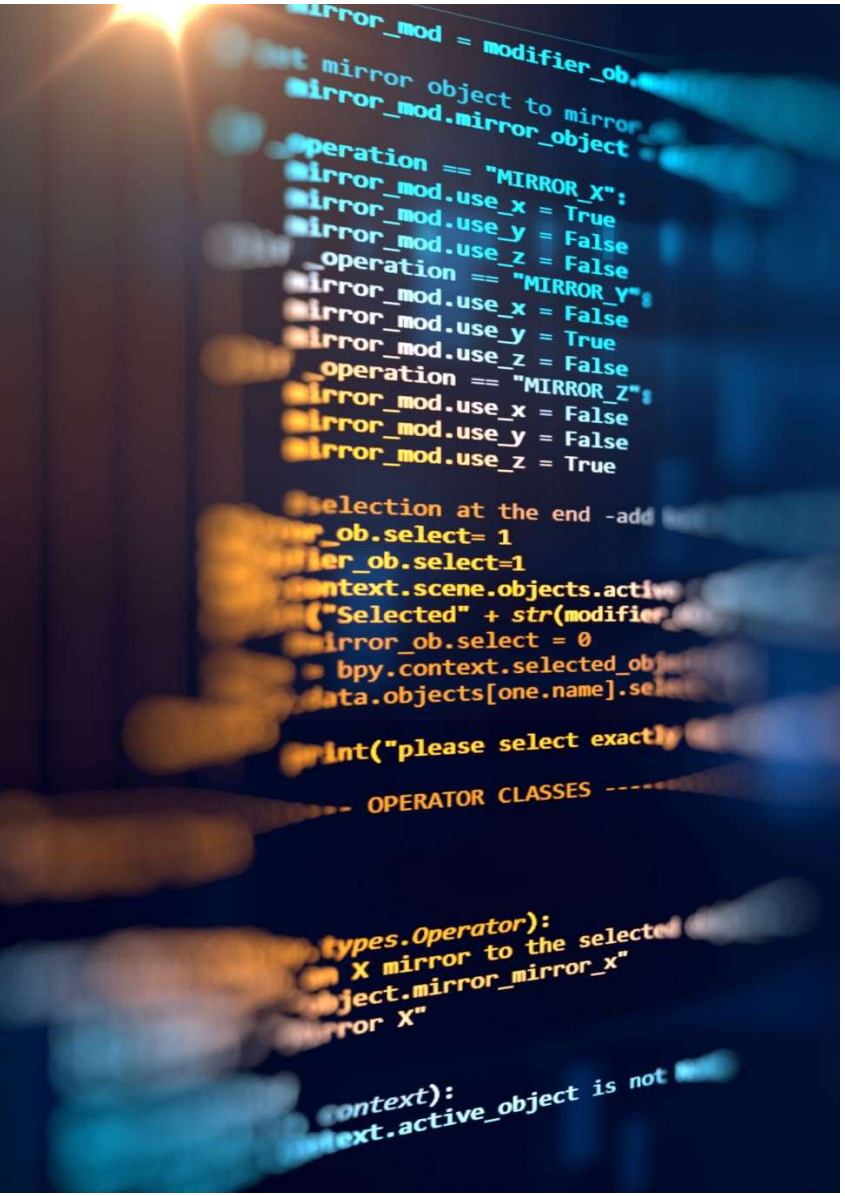


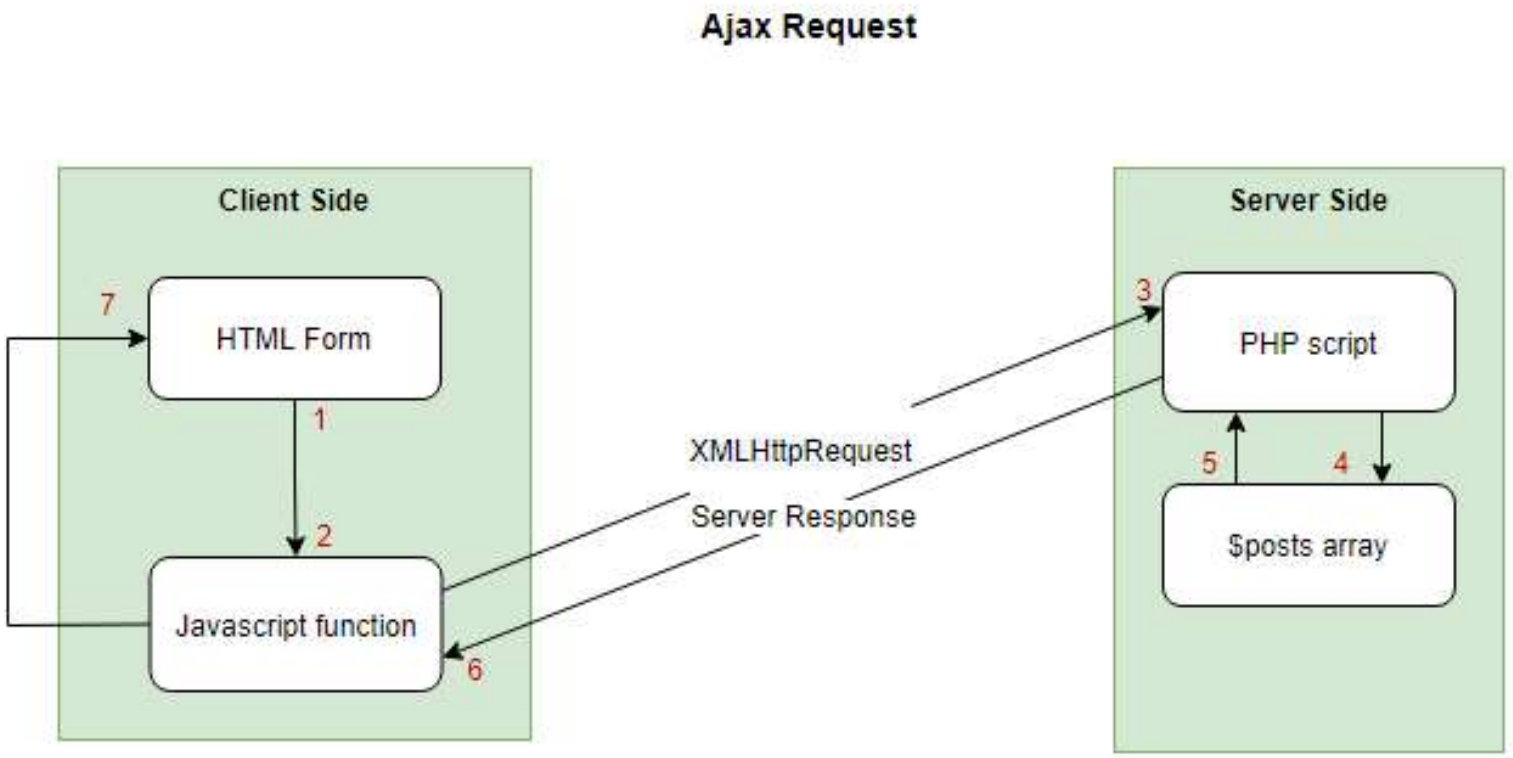
PHP AJAX requests

- How to handle AJAX calls at the server

DEMO 02b

Refresh client part of making Ajax calls from CMPE2000.

Handling AJAX Calls



Handling AJAX Calls on Server

- There is nothing special that must be completed on the server.
- Because the request was made via an AJAX call instead of a form post,
- but you will not have any HTML generation following PHP processing as you are not responding with a whole page, just the relevant data that the client can poke into the page.

Handling AJAX Calls on Server

- The apache web engine automagically populates the superglobal arrays when a request is made for resources on the server.
- if you send a GET request, the GET superglobal will be populated with the sent name-value pairs, and if a POST request is sent, the POST superglobal will be populated.
- It does not make a difference if your request is a form submission or an AJAX call.
- Thus, the general flow of your processing of the data that you are sent can be set up identically.
- Received data is cleaned (trim and strip_tags for now) into a clean array.
- Then you apply whatever rules and/or make any calculations or comparisons needed.
- Data may be pulled from the SESSION and updated information may be stored back into the SESSION.

Handling AJAX Calls on Server

- The major visual difference will be that when responding to an AJAX call, you are not attempting to regenerate a page and send it to the client.
- It is just a data exchange, essentially a web service that provides nothing but the resulting data from an action or request from the client.
- This means that the target page for your AJAX request need only contain PHP.
- Once you have completed all comparisons, calculations, and updated any SESSION values, you echo your information back to the client.
- Remember that echo puts the information into a "page", but if there is no page structure, then the echo spits back text information. Echo is used only once.

Handling AJAX Calls on Server

- If you have complex information (multiple pieces) to send back, then you will want to have it organized in an associative array (name-value pairs).
- To make this as smooth as possible, you will want a format for data transfer.
- The simplest one is generally a JSON encoded string.
- There just so happens to be a function in PHP, json_encode(), that will do this for you.
- As an aside, you may also json_encode() data for saving to the SESSION, and json_decode() the data to a local variable when you want to manipulate it.
- On the client, if as part of your AJAX request you specified "json" data, then the browser should decode the JSON string automatically for you, and your success function will have that data available to you as the first variable in the response. If not, you will have to use the Javascript function for parsing JSON.

Handling AJAX Calls on Server

- If you have complex information (multiple pieces) to send back, then you will want to have it organized in an associative array (name-value pairs).
- To make this as smooth as possible, you will want a format for data transfer.
- The simplest one is generally a JSON encoded string.
- There just so happens to be a function in PHP, json_encode(), that will do this for you.
- As an aside, you may also json_encode() data for saving to the SESSION, and json_decode() the data to a local variable when you want to manipulate it.
- On the client, if as part of your AJAX request you specified "json" data, then the browser should decode the JSON string automatically for you, and your success function will have that data available to you as the first variable in the response. If not, you will have to use the Javascript function for parsing JSON.

PHP Basic Stuff Demo 02b

PHP AJAX call Validation

Validate and sanitize your data on server before processing it further.

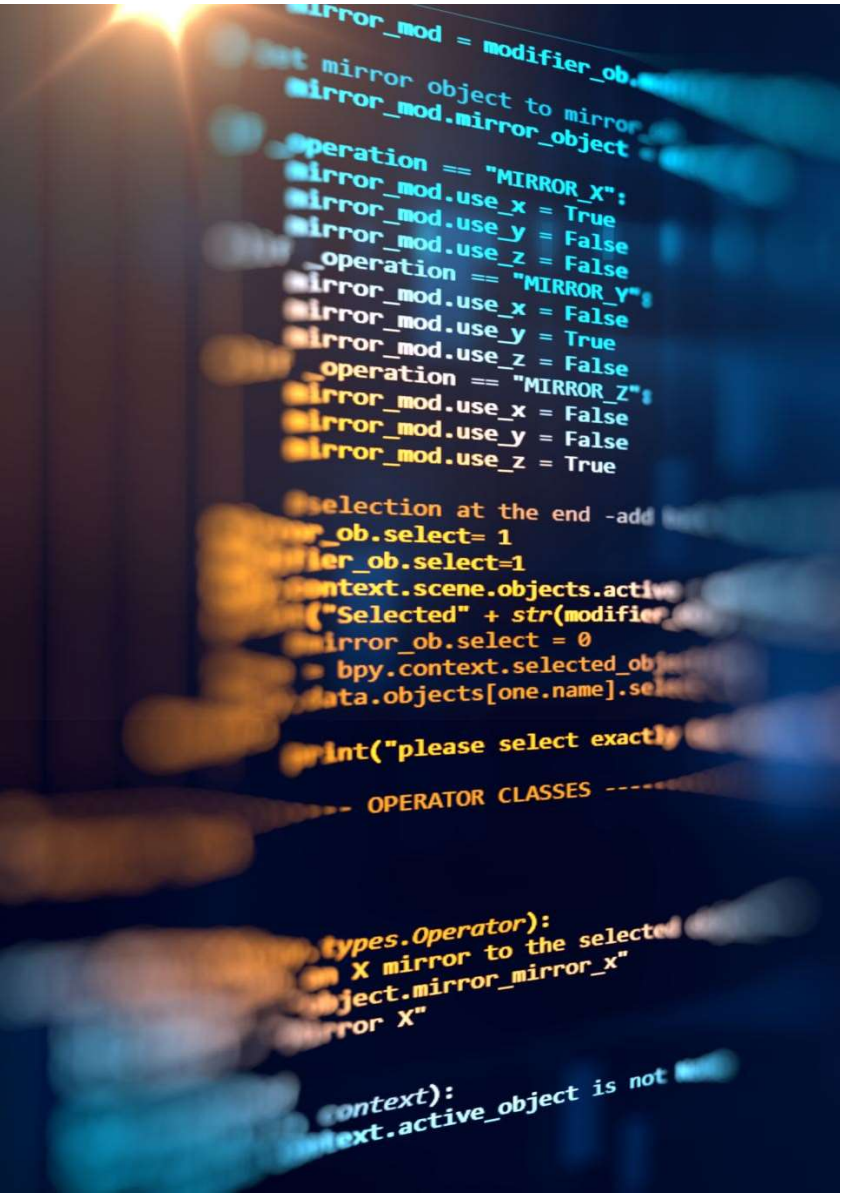
isset(): checks whether a variable is set, which means it is declared and is not null.

```
if(isset($_POST["name"]) && strlen($_POST["name"]) > 0 && isset($_POST['age']) &&
   strlen($_POST['age'])>0)
{ // validating that both variables are set and are not empty.
   // Sanitize submitted data

   // Trim()- trim function removes white spaces from both sides of the string.
    //strip_tags()- strips a string from HTML, XML and PHP tags.

    $name= strip_tags( trim($_POST["name"]) );
    $age= strip_tags( trim($_POST["age"]) );

    // We are good to use submitted data
}
```



PHP AJAX requests

- Review class DEMO 02b files