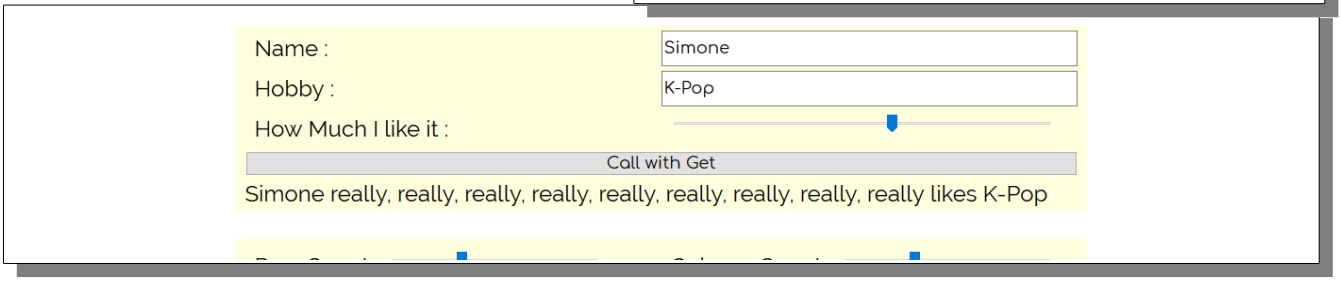
School of Applied Sciences And Technology
CMPE2000 – ICA10 – Ajax Intro

In this exercise you will utilize the Ajax capabilities of the jQuery library to exercise a simple in place web service retrieval.

Start a new page/style/js flee structured as usual with separate style and javascript files. Include jQuery as well.

There are use a nested grid placing each part in its own container. Your inner panel grid can consist of 2 columns with a style type that will allow certain elements to span 2 columns.

Part I – Hobbies



HTML - Reproduce the block as shown. No form objects are used. The "How Much.." input element is a range element with a default value of 8 with minimum value of 1 and maximum value of 13. ID as appropriate to access your elements via jQuery.

JS – For the [Call with Get] button create an explicit callback function. You will code an in-place ajax() call. Declare a data object and populate it with your data use :

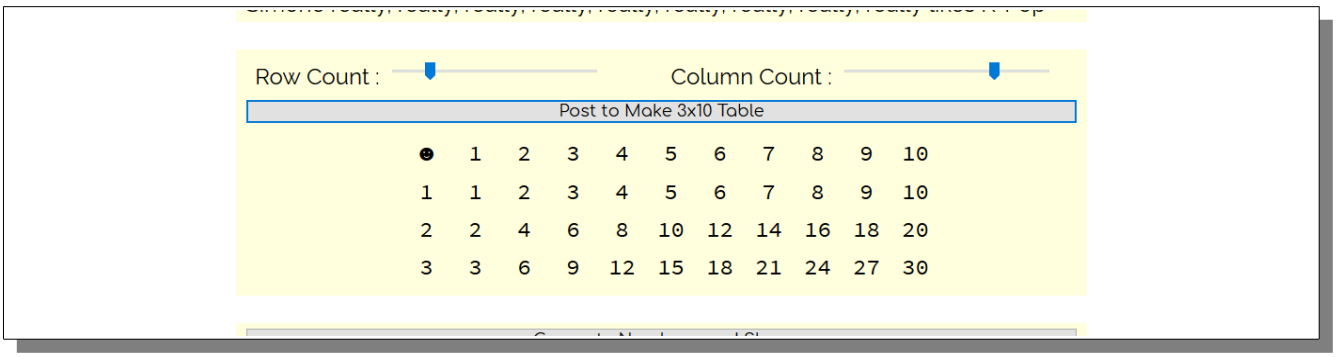
- Name as property name for Name input element
- Hobby as property name for Hobby input element
- HowMuch as property name for Range input element

For your .ajax() options object you will use a type = "GET" and a dataType = "html" Use target URL : **/~demo/cmpe2000/ica_Hobby.php**

Invoke the .ajax() saving the return object then use it to define the done()e fail() and always() callbacks.

For :
.done : put the return data into the label use console.log to output "GET done : " and the status
.fail : use console.log to output "GET failed : " and the status
.always : use console.log to output : "Always : " and the status

Part II – Table Maker



HTML – Reproduce the block as shown note the 2 labeled range elements for capturing table size requests. These start with a min/max of 1/13 and a default value of 5. Add handlers to the range elements (use .change() shortcut on your selector)e and update the Button text to indicate the proposed table size that will be requested. The table of numbers is actually just an empty full width div (image is shown after data has been returned).

JS – For this part and the next part we need to start by coding a generic ajax() method to use. Call it AjaxRequest with parameters for urle type data dataType successFunction and errorFunction. Using full form of the .ajax call construct an options object to populate the required fields. Set the appropriate properties to their corresponding arguments.

Set the success property to your success function argument.

Set the error property to your error function argument.

*We will use this function for Part III as well but we will code a general Error handler for use as well. Code your explicit error handler called ErrorHandler which will take the requestor object textStatus and errorThrown arguments – format the textStatus and errorThrown into a string and and log it to the console - also invoke alert() with the same string.

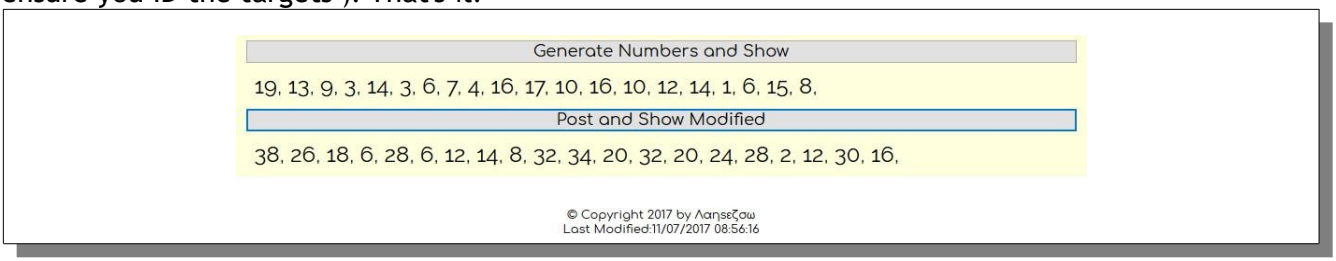
In your callback for the button declare and populate your post data object use : RowCount
as property name for the row range element value
ColumnCount as property name for the column range element value
url : **/~demo/cmpe2000/ica_Table.php**

Create a function for your success handler to accept the data and textStatus incoming arguments then take the data and inject it into the html of your target object - the empty div element put the status to the console.

You should now invoke your AjaxRequest() method with your initialized arguments. You will pass the literal value “html” for the dataType indicating that the response data will be expected in html format. Your successful response should populate the empty table ensure it is styled as shown above.

Part III – Passing Arrays

HTML – Reproduce the block as shown : 4 rows 2 with buttons and 2 empty full width target divs – ensure you ID the targets). That's it.



JS – Functionally this is a 2 step process. [Generate …] button callback will populate a global js array with 20 random ints between 0 and 20. (Use Math.foor() or Math.ceil()). Once populated clear your frst target DI and iterate through the array and display the elements as a comma separated string.

In your [Post..] button callback declare and populate your post data object use : Numbers as the property name for your array assign your array directly.
Url : **/~demo/cmpe2000/ica_Numbers.php**

Code a new success function for use here it will populate the response data into your 2nd target div. Use the previous error handler for the errorFunction it will handle any error conditions.

Now invoke your AjaxRequest() method with your arguments using “html” as the type again and your numbers will have magically doubled.

Now to see if anyone reads the entire ica - include another button at the bottom with the text “Fail!”. The click callback for this should invoke AjaxRequest. In this case we want to force an ajax error to test your coding and handler use a url that we previously passed - but inject a spelling error (should be easy for most of you) thereby having a failure condition- you should see your alert and logging show up.

** Try everything TWICE to ensure you are cleaning up old information as needed.