



## School of Applied Sciences And Technology

### CMPE2000 - ICA07 - JS Forms

In this exercise you will implement on Motorcycle purchase application using javascript. Standard rules apply.

**Ica06 - Purchase a Ride**

Make :	<input type="button" value="Please Pick One"/>	<input type="button" value="Japanese"/>	<input type="button" value="European"/>	<input type="button" value="American"/>	
Model :	<input type="radio"/> Scooter <input checked="" type="radio"/> Naked <input type="radio"/> Track				
Options :	<input checked="" type="checkbox"/> Automatic Braking System <input checked="" type="checkbox"/> Traction Control System <input type="checkbox"/> Flight Mode Mapping				
Pre-Payment :	4567				
Selection : Honda : \$15000.00 : 2 options selected \$15000.00 + \$700.00 - \$4567.00 = \$11133.00					
<input type="button" value="Purchase"/>					
<small>© /home/ica06 Last Modified 05/25/2021 14:24:14</small>					

Functionally, this is a standard web form, looking for information to forward regarding the purchase of a motorcycle.

The page is 3 column ( 1fr, 4fr, 1fr ) grid, with a simple 2 column ( 1fr, 3fr ) nested grid for the form. The header and footer span all 3 columns, The status output and Purchase/Submit button in our nested form will span the form container. Justification should be as shown, using a class for all elements that require right justification.

Presentation should be essentially as shown, background and foreground colors must be specified, but your choice unless indicated. Use padding, margins, grid-gaps etc, to make a pleasing UI.

The mobile layout should collapse to a single column, with left justified input captions.

Also, specify a monospaced font for all textbox values and status output.

**\*\* For this ica, all access to form elements will done with `querySelector` and `querySelectorAll`. The only page element with an ID is the status output container. This means you will be using selectors like `name`, `type` and pseudo-selectors like `:checked`.**

## Structure

Form is a GET to the [https://thor.cnt.sast.ca/~demo/cmpe2000/ica04\\_formtest.php](https://thor.cnt.sast.ca/~demo/cmpe2000/ica04_formtest.php) page. Remember that only “named” form elements will be submitted.

Make Selector is a select object named brand, sized to 5 to appear as a listbox. The selections will have name/value pairs of { Please Pick One/”” }, { Japanese/Honda }, { European/Ducati }, { American/Harley }.

Model set are radio buttons named model, which have label/value pairs of {Scooter/5000}, {Naked/10000}, {Track/15000}.

Options are a group of checkboxes with name/value pairs, respectively { ABS/300 }, { TCS/400 }, { FMM/500 } labeled as shown.

Pre-Payment is an input box of type text named down\_payment

and an addition hidden field named total\_cost, populated prior to submission.

[Purchase] is the submit button.

## Functionality :

To help construct this ica, each functional element should be tackled incrementally, The following is your suggested approach at completion :

### Step 1 : Make Selection

Create an event function called UpdateMake. In your page onload - Bind UpdateMake to the onchange event of your Make selector. UpdateMake will uncheck all Model radio buttons - this will have to be done with querySelectorAll(). It will then invoke another new function called UpdateStatus() - create this function and put some test text into the status output div to ensure it is working.

### Test.

### Step 2 : Model selection.

Create an event function called UpdateModel. In your page onload - Bind it to the onclick event of all the radio buttons. Do this with querySelectorAll(). The function UpdateModel will uncheck all the Option checkboxes if the current radio button value ( this ) is less than 10000 ( the cheap bikes don't have any options ). Upon completion, invoke your UpdateStatus method.

**Test - Changing a Make should clear any/all Model selections. If the Model selected or changed - and is a scooter (< 10000), then all the options should be cleared..**

### Step 3 : Checkbox selections

In your page onload, bind all the checkbox onclick events to invoke the UpdateStatus method.

### Test

### Step 4: Down payment processing

In your page onload, use an anonymous method ( or the arrow operator ) to bind the onchange event of the down payment text input box to invoke the UpdateStatus method.

## Test

### Step 4 : Final Wiring

Finish your onload event processing. Set the onsubmit event of the form to a new function called Validate() - create the empty stub. The last line of onload should explicitly invoke UpdateStatus.

UpdateStatus() - Depending on form state, the status will output as follows and in this order of processing :

- If down payment is not a number, output “Invalid Downpayment”, put focus and select the bad input.
- If no Make is selected ( use the selectedIndex property ), output “No Make Selected”.
- If no Model is selected, output “No Model Selected”.
- Otherwise, we must build an output string in this form :

“Selection : MAKE : \$MODELVALUE : N options selected  
\$MODELVALUE + \$OPTIONSUM - \$DOWNPAYMENT = \$BALANCE

Where Brand is the Brand value, N is the number of options currently selected. Followed by the full fixed money format equation evaluating the outstanding balance. See sample figure.

Leveraging UpdateStatus() will be used for form validation. For this, in every error condition, return false, otherwise if processing is successful to the end, return something else ( ? hmm )..

## Test

### Step 5 : Form Submission

To complete our ica, we need to ensure correctness prior to submission.

For function Validate, invoke your UpdateStatus, leveraging the return value whereby if UpdateStatus returned false, an error occurred and submission can not continue. If UpdateStatus was successful, meaning most data has been validated, perform an addition check. Only let the form submit if the down payment is at least half the total cost, If it is less, append to the status message the Minimum Downpayment value in this form :

( Min \$MinValue )

Also, set focus and select() the current down payment text, ready for user modification.

If everything checks out, place the current calculated total cost into the hidden form field and return to allow submission of the form.

## Test

	Options :	<input checked="" type="checkbox"/> Traction Control System <input type="checkbox"/> Flight Mode Mapping
	Pre-Payment :	4567
	Selection : Honda : \$15000.00 : 2 options selected \$15000.00 + \$700.00 - \$4567.00 = \$11133.00 ( Min \$7850.00)	
	Purchase	

## Step 6 : Prettification

Finally, since a picture is worth a thousand words, we will include images to finish our form. Download the supplied image folder 7z, and save them into an images folder within your ica folder. Add an image tag beside the select element. Include processing to populate the src property of the image. The file name can be constructed by examining ( switch ? ) the Make and Model. Set the image source only if a valid Brand/Model has been selected, empty otherwise ( or pick an appropriate “empty” bike image if you wish ).

We want the image to be the same height as the select element to keep the form from unnecessary resizing, so use window.getComputedStyle() on your select element, then apply the retrieved height property to your image.

Signoff : No sub-part marks. Section B, C, D must be functional for any marks.	
A - 20%	CSS Grid Mobile/Desktop layout correct - all elements shown and display correctly
B - 30%	Form complete and functional - button functionality, submits properly, no validation
C - 25%	Status messaging complete - Brand/Model/Options+Count/Equation
D - 15%	Validation complete with leveraging UpdateStatus
E - 10%	Proper image processing on valid Brand/Model selections