

## EECS 1012 – *Practice* Lab Test #2

### INSTRUCTIONS

You will have 90 minutes to complete this lab test. When 90 minutes have finished, you will be asked to stop and upload your files based on the submission instructions. **Do not turn off your computer, leave your files all in the www directory, we will come by and mark**

Edit the files in the www directory, **you do not need to create any files.**

### Additional Files Provided

(1) HTML+FORMS, (2) CSS and (3) JavaScript - “Cheat sheet”

There is also a video showing you the behaviour of the tasks.

### *Tentative* Grading Scheme

Task	Marks
Task 1: Javascript + Forms	30%
Task 2: JavaScript Basic	15%
Task 3: JavaScript DOM	25%
Task 4: JavaScript Events	30%
<b>Total</b>	100%

## Task 1 (FORMS and PHP): files - task1.html, task1.css, and task1.js.

You only need to modify the task1.js, however, you are allowed to modify all files if you want. Your task is to validate the input as it is typed in. Where appropriate, use a suitable regular expression to check the input. See the boxes below to explain what is allowed as input. When the submit button is pressed, check to see if all fields have been validated as correct. (**Note** – you do not need to submit the form, only change the color of the button).

### Task 1 - [Write your name here]

Write your name here.

### Sunlife Dental Insurance Claim

Enter Card ID: Format A (adult) or C (child), 4 numbers, Letter

Sunlife Health Card ID:

Enter Country Code: CAN, MEX, USA, EUR, OTHER

Country Code:

Claim amounts:

This is the HTML file with no JavaScript. Please look at the HTML file carefully to see the layout and the IDs of the elements.

### Sunlife Dental Insurance Claim

Enter Card ID: Format A (adult) or C (child), 4 numbers, Letter

Sunlife Health Card ID:

Enter Country Code: CAN, MEX, USA, EUR, OTHER

Country Code:

Claim amounts:

When the page loads, set up all event observations. Change the background of all input backgrounds to the color "mistyrose".

### Sunlife Dental Insurance Claim

Enter Card ID: Format A (adult) or C (child), 4 numbers, Letter

Sunlife Health Card ID:

Enter Country Code: CAN, MEX, USA, EUR, OTHER

Country Code:

Claim amounts:

This field should match the following pattern: First letter A or C, followed by four numbers, followed by any letter. The input can only be 6 characters.

This field has to be one of the strings shown above (CAN, MEX, ..). Input can be upper or lower case.

Input should match the following pattern: start with a dollar sign, followed by one or more numbers. Then followed by an optional dot, followed by 0 or more numbers.

### Sunlife Dental Insurance Claim

Enter Card ID: Format A (adult) or C (child), 4 numbers, Letter

Sunlife Health Card ID:

Enter Country Code: CAN, MEX, USA, EUR, OTHER

Country Code:

Claim amounts:

When an input is valid, change the background color to "lightblue", otherwise, change it to "mistyrose".

**HINT:** When validating that all inputs are correct, you can check to see if the color of the background are lightblue. For example: `(replace element name with the appropriate id) $("[element name]").style.backgroundColor == "lightblue"`

When the user clicks the submit button, if all fields are valid, then change the button's background color to "lightblue". If any of the fields are not validate, then change the button to "mistyrose" (see image above this one). [**NOTE: You don't need to submit the form!**]

Task 2: (Javascript) – files: task2.html and task2.js.

Your task is to implement a simple number pad emulator using Javascript as shown below.

## NumberPad Emulator



When a key is pressed, update the div's (id= `entry`) innerHTML to add in the value of the button just pressed.

Hint: Use the string concatenate operator to add the innerHTML of the button pressed.

For example, button id=3's innerHTML is "3".

So, div's innerHTML = div's innerHTML + button id's innerHTML. (where + is the string concatenate operator).

The delete button is the hardest example. It requires you remove the last character of a string. See info below on how to do this.

```
<div id="pad">
<div id="entry"></div> <br>
<button id="7">7</button><button id="8">8</button><button id="9">9</button> <br>
<button id="4">4</button><button id="5">5</button><button id="6">6</button> <br>
<button id="1">1</button><button id="2">2</button><button id="3">3</button> <br>
<button id="0">0</button><button id="back">Delete</button>
</div>
```

The "delete" button is the trickiest part since we have not seen how to do this in Java Script. However, it is pretty easy to do. Consider you have a string `s`. There is a method "slice" that can remove the last character. See code:

```
var s = "string";
var new_s = null;
new_s = s.slice(0, -1);    (new_s now equals "strin")
```

When delete is pressed, get the innerHTML of the entry div, set this to a variable.

For example (using prototype)

```
var s = $("entry").innerHTML;
```

If `s` isn't the empty string, then set the `$("entry").innerHTML = s.slice(0, -1);`

### Task 3: Javascript (DOM + Events) - task3.html and task3.js, this task is modeled after Lab 7's task #2.

Write the Javascript code that allows the user to type in text that is added to an ordered list.

Create two buttons (add item and delete last) that allows the user to add the item when pressed or delete the last item added.

There should be an optional "Delete #" button, followed by a text input. This will delete the item # inputted by the user.

**Also**, if the user clicks in the text input under "enter item to add", you should delete any existing text value that may already be there. Think carefully about how to do this. It is pretty easy, but we haven't done an example like this in any of the prior labs.

## Todo List++

Enter item to add:

This has always been a test

Add item

Delete last

Delete item #:

1. This is a test
2. This is only a test
3. This has always been a test

The user has added three items to the list.

## Todo List++

Enter item to add:

This has always been a test

Add item

Delete last

Delete item #:

1. This is a test
2. This is only a test

If "delete item" is pressed, the value in the text field is retrieved (and converted to an integer) and used to delete the item with that number from the list. For example, above item #2 was deleted.

## Todo List++

Enter item to add:

Add item

Delete last

Delete item #:

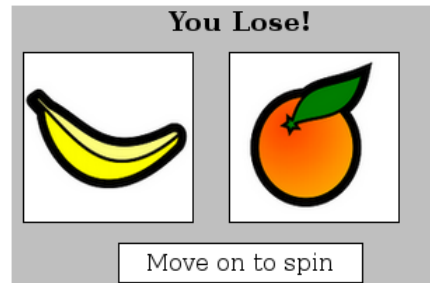
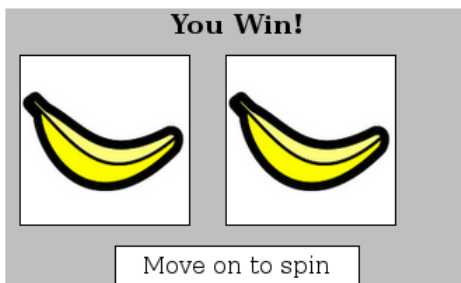
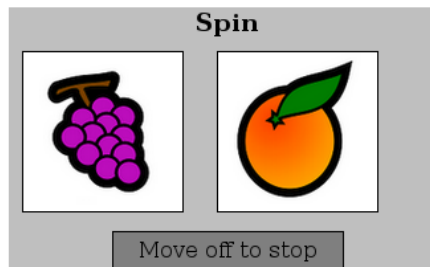
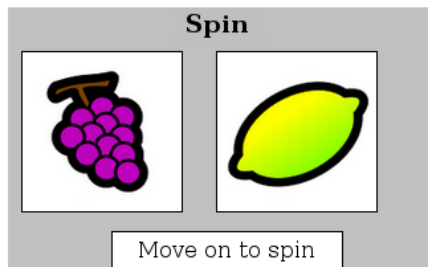
1. This is a test
2. This is only a test

If the user "clicks" on the text field to add, any existing text should be deleted.

**Task 4:** Javascript (Events) - **task4.html** and **task4.js** (and images 01.png – 08.png).

Your task is to write a very simple “slot” machine that loops through 8 images. When the user moves their mouse over the “bar” div, the images should start to change. When the user moves the mouse off the “bar”, the images should stop. If the images are the same, you should let the user know they “won” (in the results header), otherwise “they lost”.

```
<div id="machine">
  <h3 id="result">Spin</h3>
  
   <br>
  <div id="bar">Move on spin</div>
</div>
```



See examples in the Additional Resources for the JS Event's lecture.

If the user moves on the “spin” bar, the images should start to change.

First: set the “result” text to “spin”

Also, set two different intervalTimers with slightly different times (e.g. 50ms and 75ms).

One timer changes `img1`, the other timer changes `img2`.

Change the bar’s background to “grey” and change the text as shown above.

The interval should call functions that changes each image. Keep two global index variables (`i` and `j`) to keep track of what image is being displaced (e.g. `i` for the left `img1`, `j` for `img2`)

When the mouse is moved off the bar, cancel the interval timers. This will stop the images.

Check to see if the two images are the same (you can do this by checking to see if the index variables `i` and `j` are the same).

If they are the same, change the `<h3>` innerHTML to be “You Win”, otherwise to “You Lose”.