Objectives

- Javascript Strikes Back

Preparation

Before the lab you should re-read the relevant lecture slides and their accompanying examples.

Getting Started

Create a new directory for this lab called `lab11` by typing:

```
$ mkdir lab11
```

Change to this directory by typing:

```
$ cd lab11
```

Exercise: Counter

Similar to last week clone the repo again. This time it'll have the week 11 materials.

```
$ git clone https://github.com/COMP2041UNSW/js
...
```

Now run the Node Package Manager (NPM)

```
$ cd js
$ 2041 npm install
...
```

This time we will be using the server, by using the below you will start a simple server which will serve all the files in the repo.

```
$ 2041 npm start
...
```

The command will make the web-server available at http://127.0.0.1:8080

You can stop the server by pressing `control-c`

The Lab

Now that you are all set up you can see the first lab exercise by going to http://127.0.0.1:8080/labs/lab11/counter/

Go to the code and edit the code to show the time in the `output` element. Element must update the time every second.

```
$ cd labs
$ cd lab11
$ cd counter
$ vi counter.js # or any text editor that is not notpad++
```

Remember to refresh the html page when you make a change to the js so it gets loaded, good luck!

Working from Home

You can complete all the Javascript exercises in your own computer.

You'll first need to install **node** on your computer. You can find installation instructions here: https://nodejs.org/en/download/

If you have problem installing **node** ask in the course forum.

You run the same commands on your own computer but without the 2041 prefix

```
$ cd js
$ npm install
....
$ npm start
```

There is no autotest and no automarking of this question.
When you have completed demonstrate your work to another student in your lab and ask them to enter a peer assessment here

It is prefered you do this during your lab, but if this is not possible you may demonstrate your work to any other COMP[29]041 student before Sunday midnight. Note, you must also submit the work with give.

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 lab11_js_counter counter.js
```

Sample solution for `counter.js`

```
(function() {
   'use strict';
   const output = document.getElementById('output');

   void setInterval(() => {
      const d = new Date();
      output.innerText = `${d.getHours()}:${d.getMinutes()}:${d.getSeconds()}`;
   }, 1000);

}());
```

Exercise: Toggle

Navigate to http://127.0.0.1:8080/labs/lab11/toggle/

Code is available at `labs/lab11/toggle/toggle.js`

Write a script that adds and removes the class 'hide' from the `output` element every 2 seconds.

There is no autotest and no automarking of this question.
When you have completed demonstrate your work to another student in your lab and ask them to enter a peer assessment here

It is prefered you do this during your lab, but if this is not possible you may demonstrate your work to any other COMP[29]041 student before Sunday midnight. Note, you must also submit the work with give.

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 lab11_js_toggle toggle.js
```

Sample solution for `toggle.js`

```
(function() {
   'use strict';
   // write your js here.
   const d = document;
   const output = d.getElementById('output');

   setInterval(() => {
      output.classList.toggle('hide');
   }, 2000);
}());
```

Exercise: Collapse

Navigate to http://127.0.0.1:8080/labs/lab11/collapse/

Code is available at `labs/lab11/collapse/collapse.js`

I frankly don't like UNSW handbook so I've made my own to outline all the COMP courses there are for me. But the page is a bit cluttered.

Write some JS that makes the "extra info" section of each information element disappear on the click of the "up" arrow button on the top right of each element.

Once collapsed the item doesn't need to reexpand just yet.

Optional Challange: Finish this exercise using only a single click event listener and 0 changes to the HTML.

There is no autotest and no automarking of this question.
When you have completed demonstrate your work to another student in your lab and ask them to enter a peer assessment here

It is prefered you do this during your lab, but if this is not possible you may demonstrate your work to any other COMP[29]041 student before Sunday midnight. Note, you must also submit the work with give.

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 lab11_js_collapse collapse.js
```

Sample solution for `collapse.js`

```
// Challange Solution
//     This is pretty unmaintable code, if we were to add anything
//     to the info cards the code would break but
//     this is one of the cases where you can leverage
//     event bubbling to utilise Js and avoid HTML id tags
//     Take with a grain of salt though
(function() {
  'use strict';
  const main = document.getElementById('main');
  main.addEventListener('click', e => {
    if (e.target.tagName == 'I')
      e.target.parentNode.parentNode.parentNode.children[1].children[2].style.display = 'none';
  });
}());

// This is a standard Solution - relies on editing html
//    Note this code doesn't run, the function is just delcared
//    The code above is a Immediately Invoked Function Expression
//    Which declares a function and then runs it immediately
//    To run it remove the brackets and function invoke from above and
//    shift it down here
function standard() { /* eslint-disable-line no-unused-vars */
  'use strict';
  const NUM_CARDS = 2;
  const cards = Array(NUM_CARDS).fill(0).map((_,i) =>`item-${i+1}`);
  cards.map(card =>
    document.getElementById(card).addEventListener('click', toggle)
  );

  function toggle(e) {
    const content = document.getElementById(`${e.target.id}-content`);
    content.style.display = 'none';
  }
}
```

Exercise: Expand

Navigate to http://127.0.0.1:8080/labs/lab11/expand/

Code is available at `labs/lab11/expand/expand.js`

Extend your collapse code to change the icon from a "up" arrow to a down arrow once the `div` is collapsed. The button should then expand the `div` when clicked on and revert the icon.

Optional Challange: Finish this exercise using only a single click event listener and 0 changes to the HTML.

There is no autotest and no automarking of this question.
When you have completed demonstrate your work to another student in your lab and ask them to enter a peer assessment here

It is prefered you do this during your lab, but if this is not possible you may demonstrate your work to any other COMP[29]041 student before Sunday midnight. Note, you must also submit the work with give.

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 lab11_js_expand expand.js
```

Sample solution for `expand.js`

```
// Standard Solution
//    this relies on adding id's into the html

function standard() { /* eslint-disable-line */
  'use strict';
  const NUM_CARDS = 2;
  const cards = Array(NUM_CARDS).fill(0).map((x,i)=>`item-${i+1}`);

  cards.map((x)=>document.getElementById(x).addEventListener('click',toggle));

  function toggle(e) {
    const content = document.getElementById(`${e.target.id}-content`);
    content.style.display = (content.style.display == 'none') ? 'block' : 'none';
    e.target.innerHTML = (e.target.innerHTML == 'expand_more') ? 'expand_less' : 'expand_more';
  }
}

// Challenge Solution
//     Again , take with a grain of salt though
function challenge() { /* eslint-disable-line */
  'use strict';
  const main = document.getElementById('main');
  main.addEventListener('click',(e)=>{
    // see if the event originated from the button
    if (e.target.tagName == 'I') {
      let state = null;
      if(e.target.innerHTML == 'expand_more'){
        e.target.innerHTML = 'expand_less';
        state = 'block';
      } else {
        e.target.innerHTML = 'expand_more';
        state = 'none';
      }
      // traverse the dom tree to find the content section and toggle it's display
      e.target.parentNode.parentNode.parentNode.children[1].children[2].style.display = state;
    }
  });
}

// Since we didn't wrap these functions and execute them immedietly i.e
//        (function(){console.log('hello workd')})();
// we need to ask the browser to run one of them once the page is loaded.
// the bottom acheives the same effect as window.addEventListener('load', standard)

document.addEventListener('DOMContentLoaded', challenge);
// swap 'standard' to 'challange' above to run the challange Solution
```

Challenge Exercise: js_adventure_script

Navigate to http://127.0.0.1:8080/labs/lab11/adventure-script/

Code is available at `labs/lab11/adventure-script/adventure.js`

There is the basic framework for a simple game, fill out the following functions:

1. Make the player (the ghost) move left and right via the left and right arrow keys.
2. Have the 'z' key toggle the player between walk and sprint mode (slow and fast movement).
3. Have the 'x' key shoot a fireball towards the right of the screen. There is a provided image in `imgs/`. To make things simple the player can only have 1 fireball on the screen at any one time.

There is no autotest and no automarking of this question.
When you have completed demonstrate your work to another student in your lab and ask them to enter a peer assessment here

It is prefered you do this during your lab, but if this is not possible you may demonstrate your work to any other COMP[29]041 student before Sunday midnight. Note, you must also submit the work with give.

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 lab11_js_adventure_script adventure.js
```

Sample solution for `adventure.js`

```
(function() {
   'use strict';
   const player = {
     dom: document.getElementById("player"),
     fb: document.getElementById("fb"),
     fbp: -1,
     x: 0,
     y: 0,
     speed: 10,
     paint: function() {
       this.dom.style.transform = `translateX(${this.x}px) translateY(${this.y}px)`;
       if(this.fbp === -1)
        this.fb.style.transform = `translateX(${this.x+64}px) translateY(${this.y}px)`;
     },
     shoot: function() {
       console.log(this.fbp);
       if(this.fbp !== -1) return;
       this.fb.style.display = "block";
       this.fbp = this.x+64;
       let i = setInterval(function(){
         this.fb.style.transform = `translateX(${this.fbp}px) translateY(${this.y}px)`;
         this.fbp+=5;
       }.bind(this), 5);
       setTimeout(function(){
         clearInterval(i);
         this.fbp = -1;
         this.fb.style.display = "none";
       }.bind(this),1000);
     }
   }
   window.addEventListener('keydown',(e)=>{
     if(e.key === 'ArrowLeft') player.x-=player.speed;
     if(e.key === 'ArrowRight') player.x+=player.speed;
     if(e.key === 'z') player.speed = player.speed === 20 ? 10 : 20;
     if(e.key === 'x') player.shoot();
     player.paint();
   })

}());
```

Submission

When you are finished each exercises make sure you submit your work by running **give**.
You can run **give** multiple times. Only your last submission will be marked.

Don't submit any exercises you haven't attempted.

If you are working at home, you may find it more convenient to upload your work via give's web interface.

Remember you have until **Monday 15 October 23:59:59** to submit your work.

You cannot obtain marks by e-mailing lab work to tutors or lecturers.

You check the files you have submitted here

Automarking will be run by the lecturer several days after the submission deadline for the test, using test cases that you haven't seen: different to the test cases `autotest` runs for you.

(Hint: do your own testing as well as running `autotest`)

After automarking is run by the lecturer you can view it here the resulting mark will also be available via via give's web interface

Lab Marks

When all components of a lab are automarked you should be able to view the the marks via give's web interface or by running this command on a CSE machine:

```
$ 2041 classrun -sturec
```

The lab exercises for each week are worth in total 1 mark.
The best 10 of your 11 lab marks will be summed to give you a mark out of 9. If their sum exceeds 9 - your mark will be capped at 9.

- You can obtain full marks for the labs without completing challenge exercises
- You can miss 1 lab without affecting your mark.