

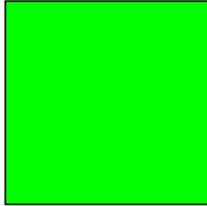
## Hide and Seek

### Your Tasks (Mark these off as you go)

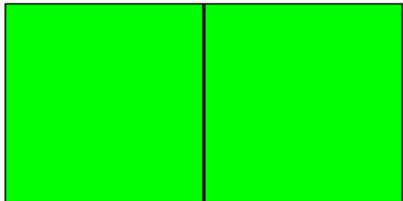
- Write code to create a 2D grid of buttons
- Write code to make your 2D grid of buttons interactive
- Write code to add a monster to a random location on your grid
- Write code to keep track of the game score
- Receive credit for this lab guide

### Write code to create a 2D grid of buttons

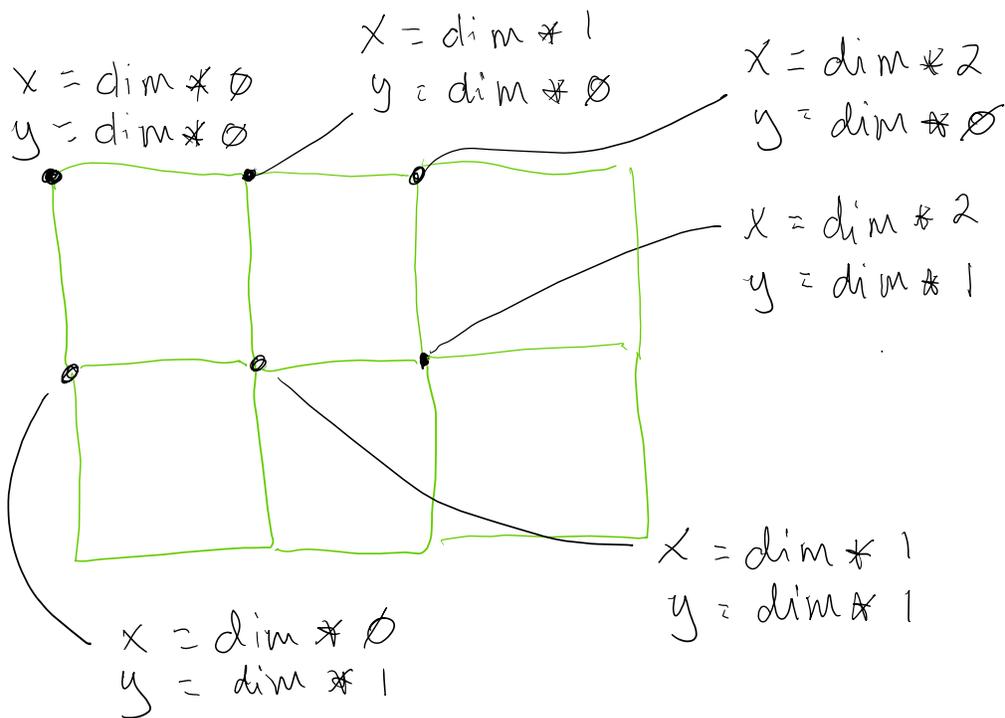
In our previous lab you wrote functions to create various shapes on the screen. The below code for example, creates a rectangle,

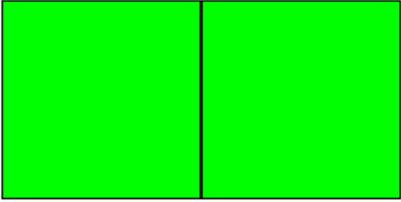
Code	Output
<pre>b = document.createElement("button"); b.style.border = "black solid thin"; b.style.width = "100px"; b.style.height = "100px"; b.style.backgroundColor = "lime"; b.style.position = "absolute"; b.style.left = "0px"; b.style.top = "0px"; document.body.append(b);</pre>	

To create many buttons, we can repurpose our code into a function. In the below function we have created three parameters, `d`, `xPos` and `yPos` and assigned them to `b.style.width`, `b.style.height`, `b.style.left`, and `b.style.right`, respectively.

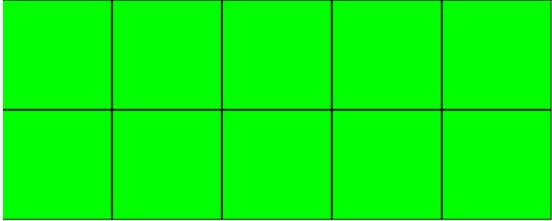
Code	Output
<pre>function makeButton(d, xPos, yPos){   var b = document.createElement("button");   b.style.border = "black solid thin";   b.style.width = d+"px";   b.style.height = d+"px";   b.style.backgroundColor = "lime";   b.style.position = "absolute";   b.style.left = xPos+"px";   b.style.top = yPos+"px";   document.body.append(b);   return b; }  var b0 = makeButton(100,10,10); var b1 = makeButton(100,110,10);</pre>	

We can make one more abstraction, by creating a new variable to represent the buttons dimensions and define the x and y position of the button in terms of it.



Code	Output
<pre> var dim = 100;  function makeButton(d, xPos, yPos){   var b = document.createElement("button");   b.style.border = "black solid thin";   b.style.width = d+"px";   b.style.height = d+"px";   b.style.backgroundColor = "lime";   b.style.position = "absolute";   b.style.left = xPos+"px";   b.style.top = yPos+"px";   document.body.append(b);   return b; }  var b0 = makeButton(dim, dim*0, dim*0); var b1 = makeButton(dim, dim*1, dim*0); </pre>	

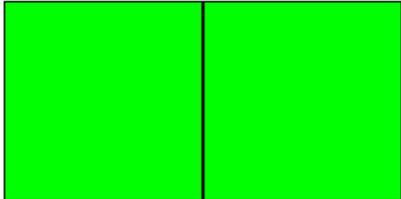
Refer to the function above. Write code that calls the function to create the grid shown below. The position of each button should be defined in terms of `dim`.

Code	Output
	

### □ Write code to make your 2D grid of buttons interactive

To make our grid interactive requires that we add action listeners to each of the buttons. We also need a way to tell which button is clicked. To do this, we will associate an id with each of our buttons. As we create our buttons we will assign an id to each button and an action listener.

In the below example, we have added a fourth parameter to represent the id. In the body of the function we assign the id to the id of the button and we also create an event listener to listen for a click. `check` is the function called when a button is clicked.

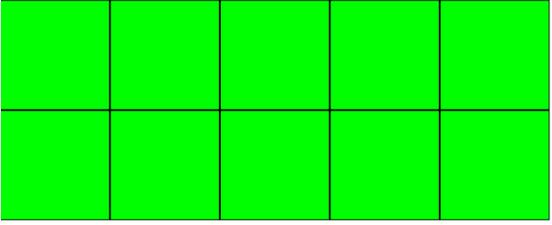
Code	Output
<pre>var dim = 100;  function makeButton(d, xPos, yPos, id){   var b = document.createElement("button");   b.style.border = "black solid thin";   b.style.width = d+"px";   b.style.height = d+"px";   b.style.backgroundColor = "lime";   b.style.position = "absolute";   b.style.left = xPos+"px";   b.style.top = yPos+"px";   b.id = id;   b.addEventListener("click", check);   document.body.append(b);   return b; }  var b0 = makeButton(dim, dim*0, dim*0, 0); var b1 = makeButton(dim, dim*1, dim*0, 1);</pre>	

The `check` function below is called when a button is clicked. Notice in the function that we pass a parameter. The parameter allows us to capture the object (or button in our case) that was clicked. Once we have the button, we can get its id log it to the console.

```
function check(e){
  console.log(e.target.id);
}
```

Handwritten annotations:  
- Red circle around `e` with arrow pointing to "the event"  
- Red arrow from `e.target.id` pointing to "the ID of the target"  
- Red arrow from `e.target` pointing to "the target clicked"

Modify your code calls from the previous example to include an id parameter. Each button you create using the parameter should have a unique id.

Code	Output
	

**□ Write code to add a monster to a random location on your grid**

Now that we have a grid of clickable buttons, we will need to assign one button to the "monster".

In the body of the `makeMonster` function below,

- Create a random number that corresponds to one of the id's of the buttons above
- Register a click event listener to the button that calls the `found` function which clicked.

Inside the body of the `found` function,

- Write code to change the background color of the random button to red. Recall that the following command can be used to capture the id of the button clicked,

```
e.target.id;
```

```
function makeMonster(){
}
function found(e){
}
```

## □ Write code to keep track of the game score

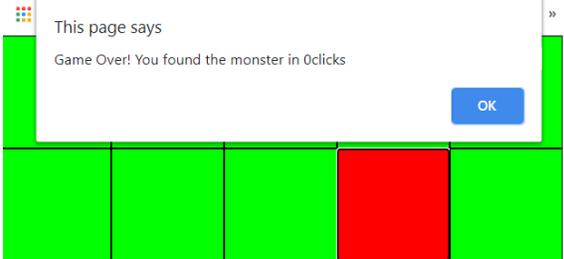
To finish our game we will need a way to keep track of how many clicks it took to find our monster. We will need a variable called score to keep track of this. We will initialize score to 0,

```
var score = 0;
```

Each time a button is clicked that does not have the monster, you will need to add one to the score,

The check function is called each time a button is clicked. Inside the body of the check function add to the score.

In the body of the found function alert the user "Game Over! You found the monster in" + score + "clicks".

Code	Output
<pre>function check(e){  } function found(e){  }</pre>	

## □ Receive Credit for this lab guide

Submit this portion of the lab to Pluska to receive credit for the lab guide. Once received, your completed code challenges will also be graded and will count towards your final lab grade.