Homework

Homework

- On today's homework, you will write JavaScript to enable the form's submit button. Currently the button is disabled from the previous homework.
- The idea is to use the random code that was generated, where the user is required to enter the displayed code correctly to enable the button. This is a common feature seen on many websites.
- To start, launch VS Code, and select the **m1-homework** folder.
- Open **contact.html** in VS Code.

> **Reminder:**
> Make at least 2 commits and push to your remote Github repository during and also at end of coding.

Homework

- The first thing you will do is group all variables in one area and at same time, add a couple more new ones for the button. In the JavaScript section at the bottom of the page (below partners), move the variables *code* and *str* from the *generateCode function* and also declared new variables **getCode** and **btnvalue**:

```
//Display all six image codes stored in the array
  document.getElementById("partners").innerHTML = imageList;
```

```
/* ------ RANDOM CODES ------ */

//NOTE: submit button is initially disabled upon loading of this page - see <body> in html

  var code = ' '; //to store generated codes and initialize to empty value
  var getCode = ' '; //to store entered code
  var btnvalue; //for button boolean value
  //create variable to hold the type of characters we want to show as codes
  var str = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789@#$';
```

Homework

- The next script is to change the Boolean value that enable or disable the button, to a variable to make it more dynamic. On the previous slide, we added this new variable:

```
var btnvalue; //for button boolean value
```

- Modify the *disableButton* function as follows:

```
//determine when to able or disable button
function disableButton(btnvalue) {
  document.getElementById("submit").disabled = btnvalue; //able/disable button
}
```

This is refer to as ***parameter*** that can pass data (a boolean in this case) into the function

The hard coded value is replaced with this new variable that will contain the data pass in from the function. This makes it dynamic as the value is no longer fix or static.

Homework

- Within the disableButton function, you will create an *if-else* statement to evaluate whether to enable or disable the submit button. The return from this statement is basically set to change the appearance of the button. On the first condition ie. if disabled, the button will appear grayed out. Add the following scripts for this scenario:

```
//determine when to able or disable button
  function disableButton(btnvalue) {
    document.getElementById("submit").disabled = btnvalue; //able/disable button
    if (btnvalue == true) { //test if button is disabled or enabled
      //set button and label color translucent
      document.getElementById("submit").style.backgroundColor = "rgba(73, 119, 209, 0.3)";
      document.getElementById("submit").style.color = "rgba(255, 255, 255, 0.5)";
    }
  }
```

Homework

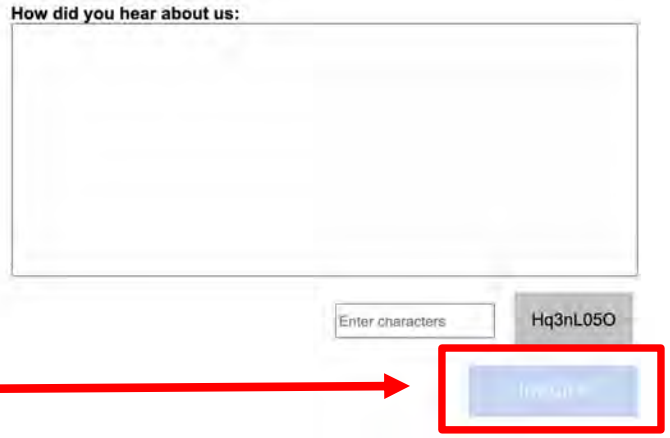
- Still within the *if-else* statement, you will create the else scenario ie. if enabled, the button will have a full color active appearance. Add the following scripts for this scenario:

```
function disableButton(btnvalue) {
  document.getElementById("submit").disabled = btnvalue; //able/disable button
  if (btnvalue == true) { //test if button is disabled or enabled
    //set button and label color translucent
    document.getElementById("submit").style.backgroundColor = "rgba(73, 119, 209, 0.3)";
    document.getElementById("submit").style.color = "rgba(255, 255, 255, 0.5)";
  } else {
    //set button and label color with no transparency
    document.getElementById("submit").style.backgroundColor = "rgba(73, 119, 209, 1)";
    document.getElementById("submit").style.color = "rgba(255, 255, 255, 1)";
  }
}
```

Homework

- The submit button appearance at this time still has the original grayish color from last class. To apply the custom style to make it appear disabled upon loading of the page, you will pass a Boolean value of true to the parameter (button value) of the disableButton function. If you recall, this button value is the determination factor for the disabled property that you'd applied to the button. If the disabled property is set to true, that will disabled the button. If set to false, then it enables the button. Enter the onload function in the open <body> tag:

```
<body id="contact" onLoad="disableButton(true)">
```



When preview on the browser:

Homework

- The final step is to turn the submit button active when the codes entered matches the codes generated. To determine if codes are entered in the input box, you will need to detect for activities on the text box. The event listener will help do that. Enter the following script:

```
    } else {
      //set button and label color with no transparency
      document.getElementById("submit").style.backgroundColor = "rgba(73, 119, 209, 1)";
      document.getElementById("submit").style.color = "rgba(255, 255, 255, 1)";
    }
}

//listen to user input code
var codebox = document.getElementById("codeentered"); //get textbox
codebox.addEventListener("input", evaluateCode); //listen to code entered in textbox
```

- If activity is detected, the listener will run the function **evaluateCode** in which you will create next.

Homework

- The **evaluateCode** function will be called each time a character is entered. And within this function will contain scripts to get the character entered as well as test if they match the generated codes. First you will be tasked to create the function and then the script to get the character. Add the following scripts:

```
//listen to user input code
var codebox = document.getElementById("codeentered"); //get textbox
codebox.addEventListener("input", evaluateCode); //listen to code entered in textbox
```

```
//run function if detected user had entered a character in textbox
function evaluateCode() {
    getCode = document.getElementById("codeentered").value; //get character entered
}
```

Homework

- Before testing the codes to see if they match, the following script's purpose is to ensure there are no hidden characters on both codes entered and generated. The trim() method will help on that. With that removed, the "cleaned" codes will be stored in to their respective new variables.

```
//run function if detected user had entered a character in textbox
function evaluateCode() {
    getCode = document.getElementById("codeentered").value; //get character entered
    var charset1 = getCode.trim(); //remove any hidden characters entered
    var charset2 = code.trim(); //remove any hidden characters generated
}
```

Homework

- You will now create an if statement to test if the characters entered (including total number) matches what's generated. Add the following scripts within the function:

```
//run function if detected user had entered a character in textbox
function evaluateCode() {
    getCode = document.getElementById("codeentered").value; //get character entered
    var charset1 = getCode.trim(); //remove any hidden characters entered
    var charset2 = code.trim(); //remove any hidden characters generated
    //test if code entered matches the number of generated characters
    if (charset1.length == charset2.length && charset1 == charset2) {
      disableButton(false); //if match, run the function to enable button
    }
}
```

- If the conditions are met (true), the disableButton function will be called and pass the Boolean value to it at the same time. This Boolean value shall reverse the current status of the submit button, ie. from *true* (disabled) to *false* (enable).

Homework

- **Due Date:**
  - This Friday 10.30PM PT.

- **Submission**
  - Post Github URL link on GAP Week 4 Day 2 Homework dropbox

Questions?

Connect with Us (WEB301/501/801)

Rich Loke

richloke@westcliff.edu

Christopher Paul

christopherpaul@westcliff.edu

Ilya Valasov

ilyavalasov@westcliff.edu

Elizabeth Kipp

elizabethkipp@westcliff.edu

End of Presentation