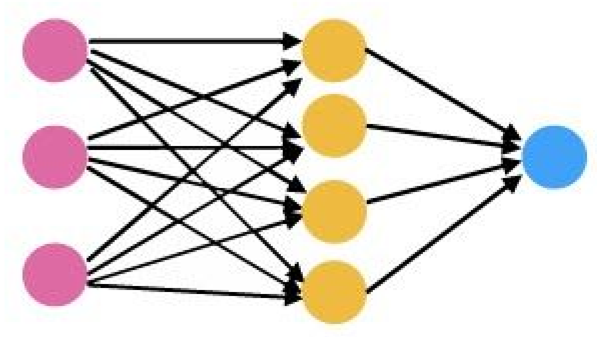
ADV-C108

NEURAL NETWORKS



What is our GOAL for this MODULE?

We completed the facial emotion detection web application by writing codes for the **check()** and **gotResult()** functions.

What did we ACHIEVE in the class TODAY?

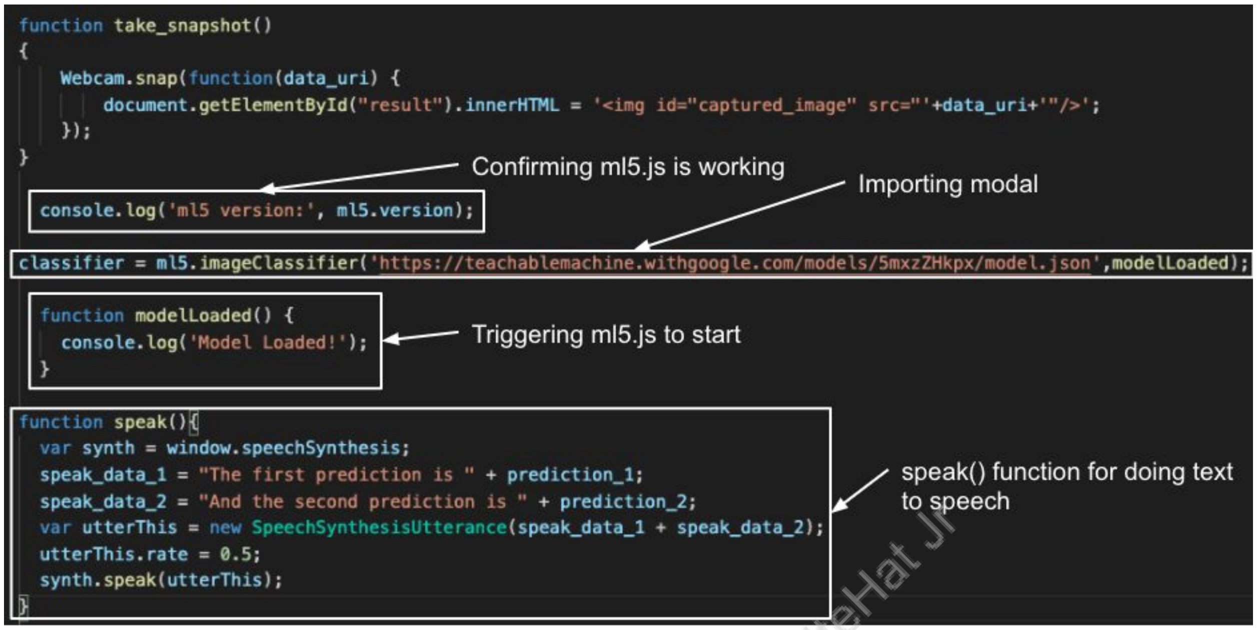
- Added **ml5.js** code for identifying the captured image using the model.

Which CONCEPTS/ CODING did we cover today?

- Wrote code for the check() function.
- Wrote code for the gotResult() function.

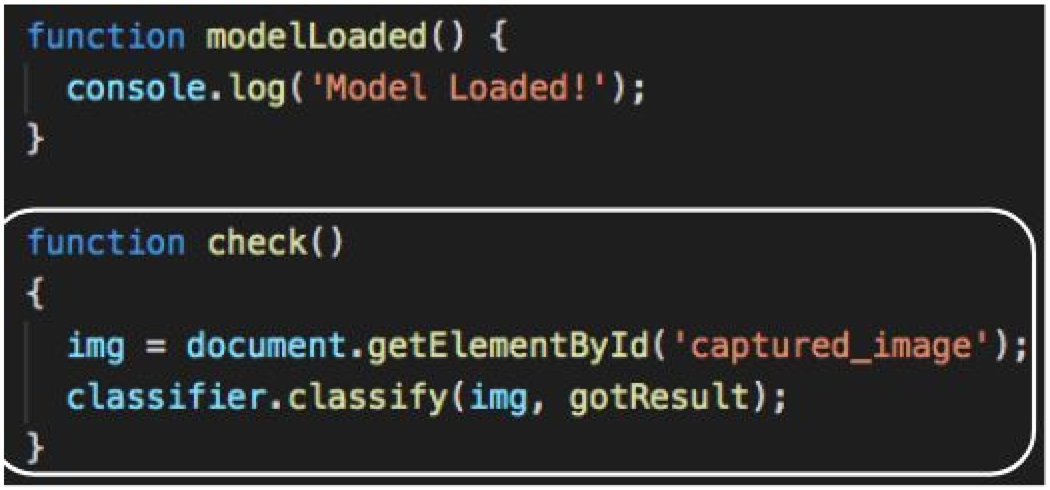
How did we DO the activities?

In the last class regarding **ml5.js** we had added the below code in **main.js**


```
function take_snapshot()
{
    Webcam.snap(function(data_uri) {
        document.getElementById("result").innerHTML = '<img id="captured_image" src="'+data_uri+'"/>';
    });
}

  console.log('ml5 version:', ml5.version);

classifier = ml5.imageClassifier('https://teachablemachine.withgoogle.com/models/5mxzZHkpx/model.json',modelLoaded);

  function modelLoaded() {
    console.log('Model Loaded!');
  }

function speak(){
  var synth = window.speechSynthesis;
  speak_data_1 = "The first prediction is " + prediction_1;
  speak_data_2 = "And the second prediction is " + prediction_2;
  var utterThis = new SpeechSynthesisUtterance(speak_data_1 + speak_data_2);
  utterThis.rate = 0.5;
  synth.speak(utterThis);
}
```
Confirming ml5.js is working
Importing modal
Triggering ml5.js to start
speak() function for doing text to speech

Now we will add a function **check()**, which is called on click of the **Predict Emotion** button.

The purpose of the **check()** function is to get the captured image, pass it to **ml5.js** and do a comparison, and call the result function.



```
function modelLoaded() {
  console.log('Model Loaded!');
}

function check()
{
  img = document.getElementById('captured_image');
  classifier.classify(img, gotResult);
}
```

Explaining the above code:

1. First, define the function.

```
function check()
```

2. Then we get the captured image, and store it inside a variable.

```
img = document.getElementById('captured_image');
```

- `captured_image`, this **id** we had given to the **img** tag which was holding the captured image in the previous class, so using this **id** we are getting the captured image and storing it inside a variable.

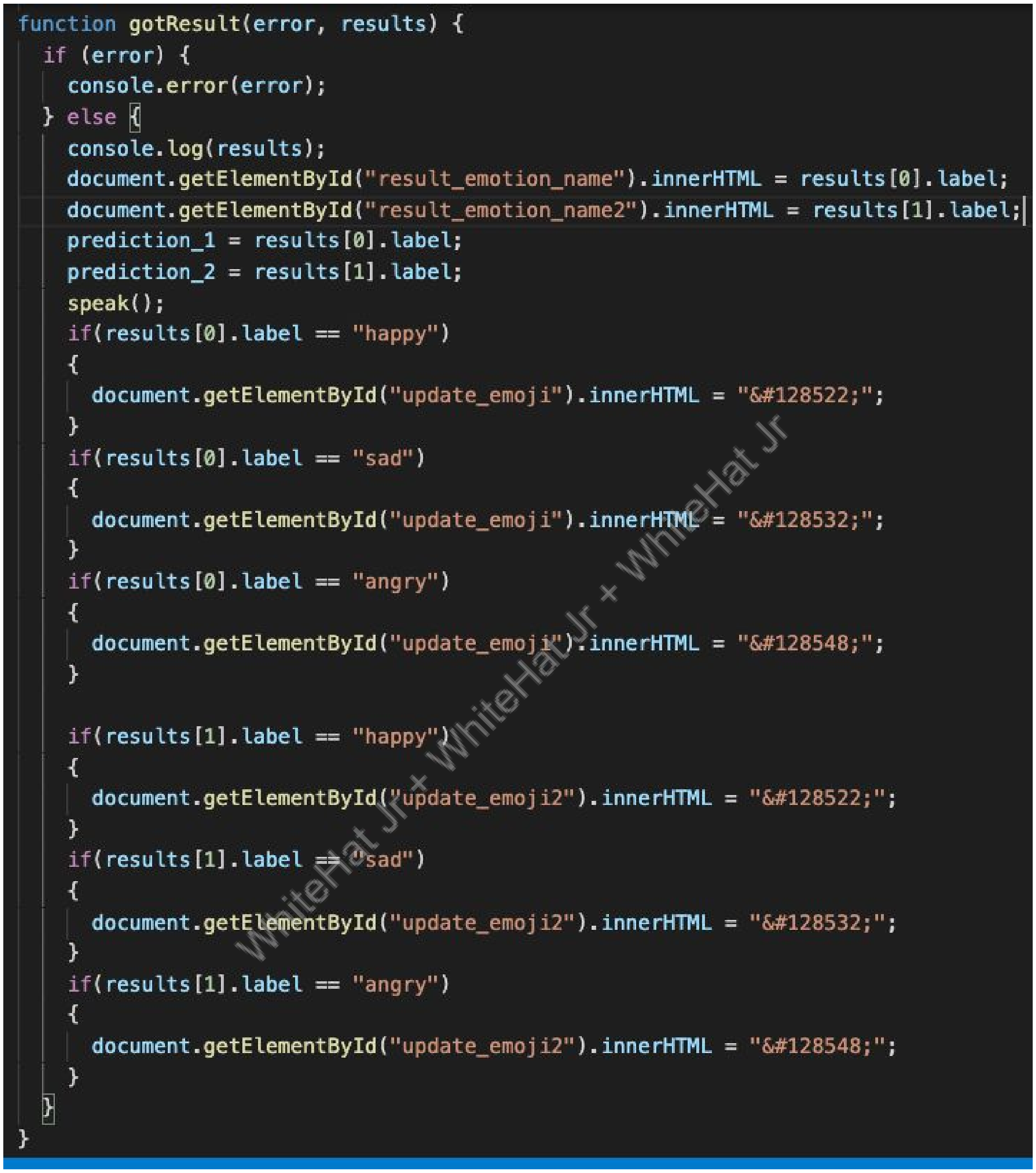
3. Now we will call the predefined function of **ml5.js** which is used to compare the captured image with the model, and pass this image inside it.

```
classifier.classify(img, gotResult);
```

- `classifier` is the variable that holds the model which we had imported in the starting of **ml5.js** coding in the previous class.
- `classify` is a predefined function of **ml5.js** that is used to compare the captured image with the model, and get the results.
- In `classify` function we are required to pass two things:
    - The captured image.

        - `classifier.classify(img,` where **img** variable contains the captured image.
    - A function, which will hold the result of the comparison.
        - `classifier.classify(img, gotResult);` where **gotResult** function will hold the result of the comparison, this function we will define in the next step.

Now we will write the code of the **gotResult** function.
The purpose of this function is to show the result which is achieved after the comparison between captured image and the model in the **check()** function.

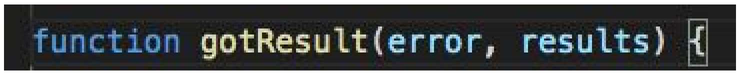


```
function gotResult(error, results) {
  if (error) {
    console.error(error);
  } else {
    console.log(results);
    document.getElementById("result_emotion_name").innerHTML = results[0].label;
    document.getElementById("result_emotion_name2").innerHTML = results[1].label;
    prediction_1 = results[0].label;
    prediction_2 = results[1].label;
    speak();
    if(results[0].label == "happy")
    {
      document.getElementById("update_emoji").innerHTML = "&#128522;";
    }
    if(results[0].label == "sad")
    {
      document.getElementById("update_emoji").innerHTML = "&#128532;";
    }
    if(results[0].label == "angry")
    {
      document.getElementById("update_emoji").innerHTML = "&#128548;";
    }

    if(results[1].label == "happy")
    {
      document.getElementById("update_emoji2").innerHTML = "&#128522;";
    }
    if(results[1].label == "sad")
    {
      document.getElementById("update_emoji2").innerHTML = "&#128532;";
    }
    if(results[1].label == "angry")
    {
      document.getElementById("update_emoji2").innerHTML = "&#128548;";
    }
  }
}
```

Explaining the above code:

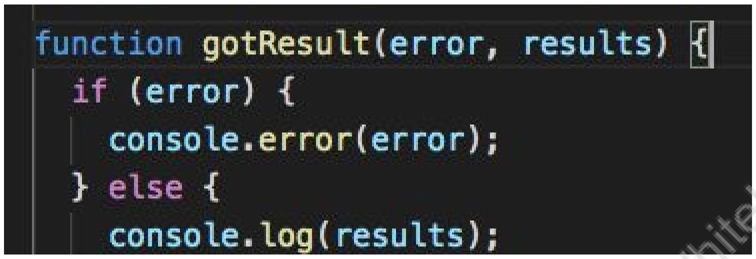
This **gotResult** function which holds the result of the comparison, has two things inside it one is error and second is results.

4. So when we define the **gotResult** function we need to pass error and results inside the function.



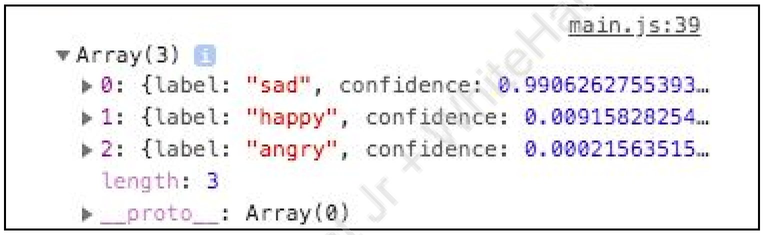
```
function gotResult(error, results) {
```

5. Now we will check if there is error, if yes then console the error, else console the results, and see what are the things in results:



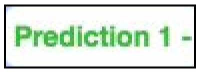
```
function gotResult(error, results) {
  if (error) {
    console.error(error);
  } else {
    console.log(results);
```

- Output on console screen:

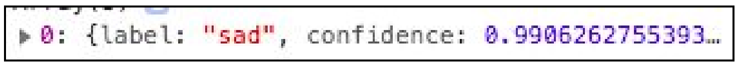


```
                                        main.js:39
▼Array(3) ℹ
  ▶0: {label: "sad", confidence: 0.9906262755393…
  ▶1: {label: "happy", confidence: 0.00915828254…
  ▶2: {label: "angry", confidence: 0.00021563515…
   length: 3
  ▶__proto__: Array(0)
```

- In the beginning of the class when I showed you the demo at that time we discussed that always the first result is the most accurate one, so we will fetch the first result label and confidence and print it on the HTML page.

Code for updating Prediction 1 -:

6. **So to get the label from the first result:**
   - The array which we expanded in the console screen is result, which means first we will write **results.**
   - We want the first label, and it is inside a 0 index:

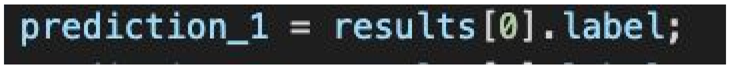


```
▸0: {label: "sad", confidence: 0.9906262755393…
```

- It means we want the 0 index which is inside the results, so we will write **results[0].**
- We want the **label** which is inside the 0 index, so we will write **results[0].label.**

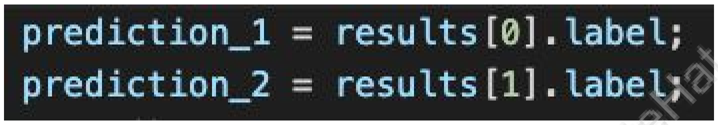
Now we know how to get the value of the first label which is inside the result.

7. We will update that variable created to store the prediction 1 value with the first results.



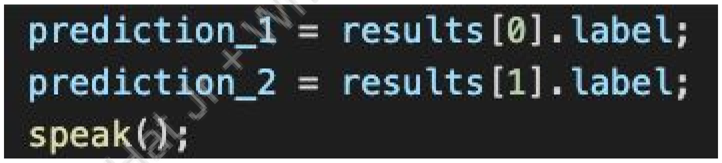
```
prediction_1 = results[0].label;
```

8. We will update the variable created to store the prediction 2 value with the second results.



```
prediction_1 = results[0].label;
prediction_2 = results[1].label;
```

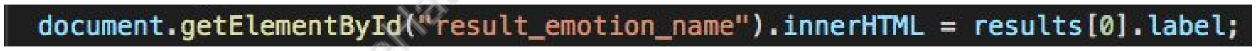
9. Now we will call the speak() function which we had defined in the previous class, which will speak out the result



```
prediction_1 = results[0].label;
prediction_2 = results[1].label;
speak();
```

10. So we will update the HTML element which we defined to hold the object name in **emotion_to_emoji.html** with **results[0].label.**

Code:

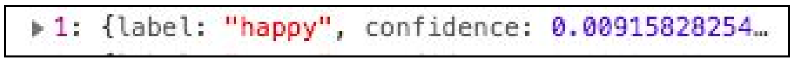


```
document.getElementById("result_emotion_name").innerHTML = results[0].label;
```

- As we have two predictions so we will take the second result as well.

Code for updating Prediction 2 - :

11. **So to get the second label from the first result:**
    - The array which we expanded in the console screen is result, which means first we will write **results.**
    - We want the first label, and it is inside a 1 index:



```
▸1: {label: "happy", confidence: 0.00915828254…
```

- It means we want the 1 index which is inside the results, so we will write **results[1].**

- ○ We want the **label** which is inside the 1 index, so we will write **results[1].label.**

Now we know how to get the value of the first label which is inside the result.

12. So we will update the HTML element which we defined to hold the object name in **emotion_to_emoji.html** with **results[1].label.**

Code:

```
document.getElementById("result_emotion_name2").innerHTML = results[1].label;
```

13. Now we want to update the emoji as per the emotion for `Prediction 1 -`. As we know `results[0].label` contains the first result. So we will use this to compare against all the emotions one by one. So for that we will write **if** conditions to check:

- If the first result is happy then update the HTML element which is defined to hold the emoji with a **happy emoji**.

```
if(results[0].label == "happy")
{
  document.getElementById("update_emoji").innerHTML = "&#128522;";
}
```

- ○ We have used hexadecimal values to display emojis.

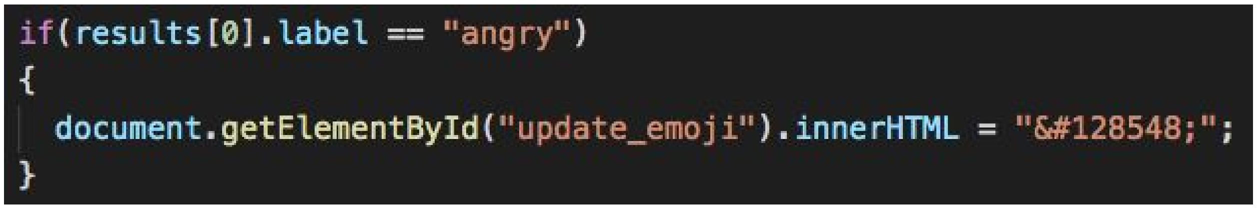
At the end of this document you will find the hexadecimal values for the emojis used.

- If the first result is sad then update the HTML element which is defined to hold the emoji with a **sad emoji.**

```
if(results[0].label == "sad")
{
  document.getElementById("update_emoji").innerHTML = "&#128532;";
}
```

- ○ We have used hexadecimal values to display emojis.

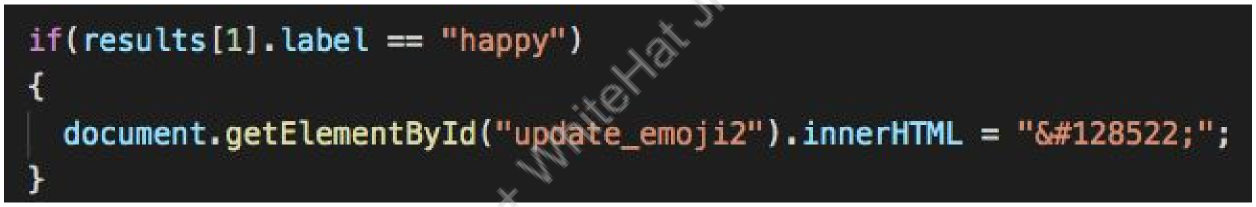
- If the first result is angry then update the HTML element which is defined to hold the emoji with an **angry emoji**.



```
if(results[0].label == "angry")
{
  document.getElementById("update_emoji").innerHTML = "&#128548;";
}
```

- We have used hexadecimal values to display emojis.

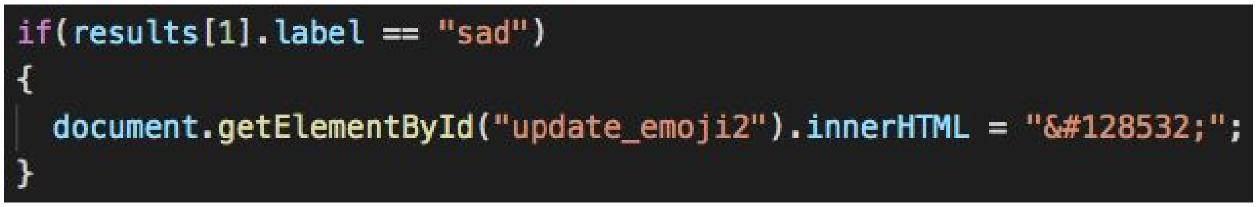
14. Now we want to update the emoji as per the emotion for `Prediction 2 -`. As we know `results[1].label` contains the second result. So we will use this to compare against all the emotions one by one. So for that we will write if conditions to check:
    - If the second result is happy then update the HTML element which is defined to hold the emoji with a **happy emoji**.



```
if(results[1].label == "happy")
{
  document.getElementById("update_emoji2").innerHTML = "&#128522;";
}
```

- We have used hexadecimal values to display emojis.

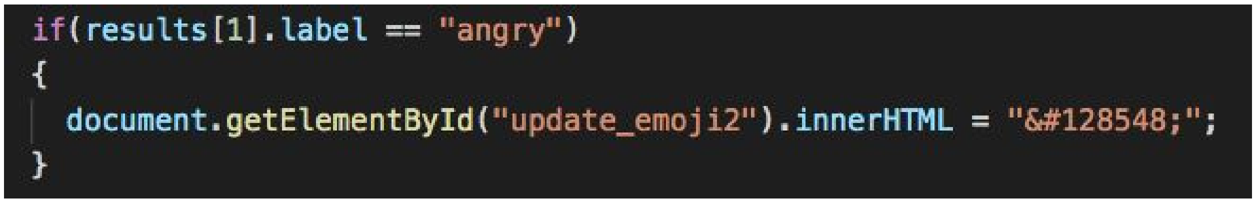
- If the second result is sad then update the HTML element which is defined to hold the emoji with a **sad emoji**.



```
if(results[1].label == "sad")
{
  document.getElementById("update_emoji2").innerHTML = "&#128532;";
}
```

- We have used hexadecimal values to display emojis.

- If the second result is angry then update the HTML element which is defined to hold the emoji with an **angry emoji**.



```
if(results[1].label == "angry")
{
    document.getElementById("update_emoji2").innerHTML = "&#128548;";
}
```

- We have used hexadecimal values to display emojis.

- **Hexadecimal values for emojis**

&#128522;	😊
&#128512;	😀
&#128532;	😔
&#128546;	😢
&#128545;	😡
&#128548;	😤

What's NEXT?

In the next class we will learn to design a 3D Spacecraft.