NEURAL NETWORKS

What is our GOAL for this MODULE?

We learned to write the JS code for the emotion detection web app.

What did we ACHIEVE in the class TODAY?

- Set webcam properties.
- Wrote trigger code for ml5.js
- Wrote code for text t o speech

Which CONCEPTS/ CODING did we cover today?

- Added code for webcam properties.
- Imported the **ml5.js** script.
- Added triggering code for ml5.js.
- Added code of speak() function,

How did we DO the activities?

JS code from main.js file.

First we will create 2 variables

```
prediction_1 = ""
prediction_2 = ""
```

and set the value as empty. This variables will be used to store the result gets from the modal and the this variables will be passed to the system to speak out the results

CODE for setting the webcam properties, and triggering the webcam.

```
Webcam.set({
  width:350,
  height:300,
  image_format : 'png',
  png_quality:90
});

camera = document.getElementById("camera");

Webcam.attach( '#camera' );
```

Explaining the above code:

1. Add JS code to set webcam properties.

```
Webcam.set({
  width:350,
  height:300,
  image_format : 'png',
  png_quality:90
});
```

- **Webcam.set** - is a function of **webcam.js** to see the properties for the live view of the webcam.
    - **width** - set the width you want for the webcam view, you can give any value as per your choice. Here we have given 350 (means 350px).
    - **height** - set the height you want for the webcam view, you can give any value as per your choice. Here we have given 300 (means 300px).
    - **image_format** - We have given png.
    - **png_quality** - means the quality of the live view of a webcam.

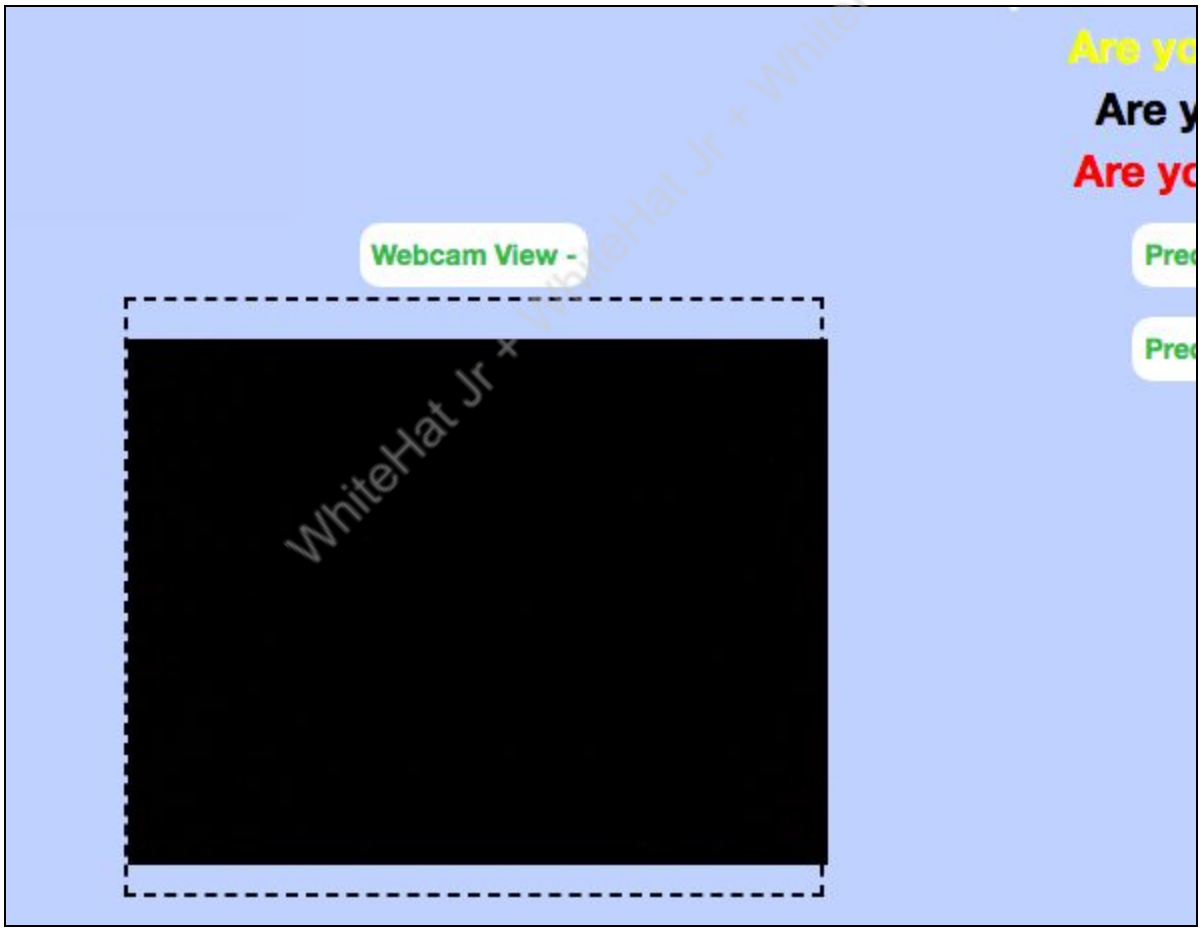
2. Then get the HTML element in which we want to show the live view of the webcam and store it inside a variable.

```
camera = document.getElementById("camera");
```

- In **image_recognizer.html** we had defined a div with **id="camera"**. This div was defined for the purpose of displaying a webcam live view in it.

- So now the variable **camera** has the HTML div.

3. Now pass the variable **camera** (which has the HTML div) inside `Webcam.attach()`. So as a result the webcam live view will be displayed in the HTML div like this:



4. Now trigger the webcam code.

```
Webcam.attach( '#camera' );
```

- This time we haven't triggered the webcam inside any function, we have just written it, so as a result as soon as the page is loaded the webcam will get triggered, and you will get a popup asking for the permission.

CODE for capturing the image.

```
Webcam.set({
    width:350,
    height:300,
    image_format : 'png',
    png_quality:90
});

camera = document.getElementById("camera");

Webcam.attach( '#camera' );

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

Explaining the above code:

1. Define the function.

```
function take_snapshot()
{
```

- `Webcam.snap()` is a predefined function of **webcam.js** used to take images from a webcam, this function contains `data_uri` that can be used to show preview of the image which generates after taking a snapshot.

2. So first define `Webcam.snap()`.

```
function take_snapshot()
{
    Webcam.snap( );
```

3. Now write a function inside `Webcam.snap()`, and pass `data_uri` inside it. And use this `data_uri` to display the image.

```
function take_snapshot()
{
    Webcam.snap( function(data_uri) { } );
```

4. Now let's update the div which we had made for the purpose of holding the image in **emotion_to_emoji.html**, with this `data_uri` which has the image taken.

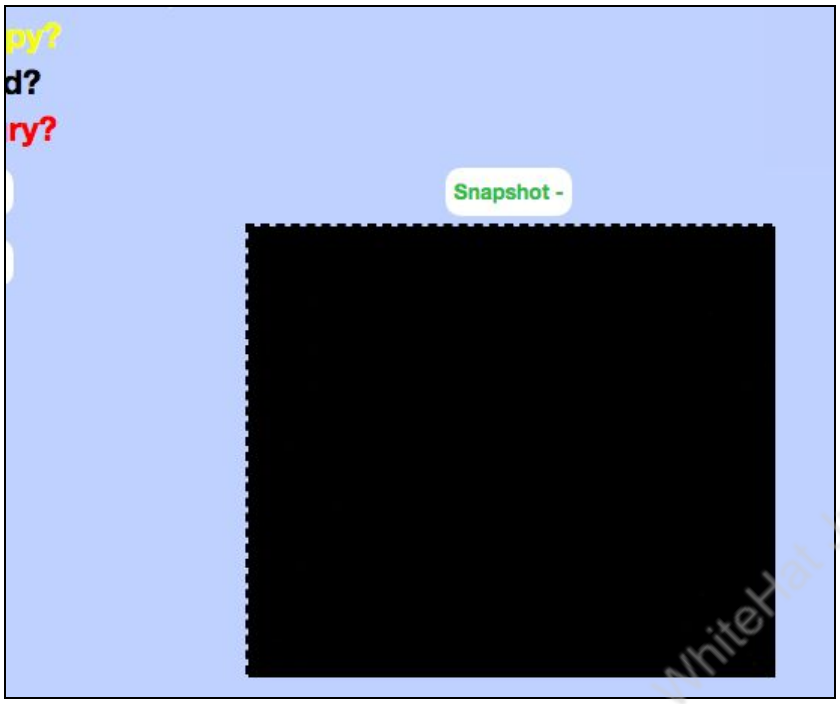
```
Webcam.snap(function(data_uri) {
    document.getElementById("result").innerHTML = '<img id="captured_image" src="'+data_uri+'"/>';
});
```

- `document.getElementById("result").innerHTML =` - this we already know.
- Define the **img** tag `= '<img`
- Then we give an id to this img tag `'<img id="captured_image"`. We are giving id to img tag, so that later we can take this image from this img tag and use it for comparing it with the model.
- Now in src of the img tag pass `data_uri`. So that this image gets updated with the selfie taken and gets displayed.

```
= '<img id="captured_image" src="'+data_uri+'"/>';
```

Be careful while closing and opening double quotes and single quotes.

- **Output:**



Now let's put the code to start **ml5.js**. This time we don't have to import the link of **ml5.js** in the HTML file because it's already given to you. So let's start code in the **main.js** code.

5. Write a console message to test that ml5.js is working. Like this:

```
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);
```

- Here we are consoling the **ml5.js** version, so if we get the version number in console screen without any error it means we are ready to use **ml5.js** library.
- **Output:**

```
ml5 version: 0.4.3                                                    main.js:20
```

- Now import our model in the file.

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

classifier = ml5.imageClassifier('https://teachablemachine.withgoogle.com/models/v_sl95BzE/model.json',modelLoaded);
```

6. First define a variable `classifier =`
7. Then write the library name ml5.js `= ml5.`
8. Then write **imageClassifier**, like this - `= ml5.imageClassifier();`
    - **imageClassifier** is a predefined function of ml5.js that is used to trigger the ml5.js image classification function.
9. We have to pass two things inside this function.
    - First, the model link.

```
= ml5.imageClassifier('https://teachablemachine.withgoogle.com/models/v_sl95BzE/model.json'
```

        - **So copy and paste that link here and at the end of the link add `/model.json`.**
        - **model.json:**
            - **model** is the model which we created in the teachable machine.
            - **json** - JavaScript Object Notation is an open standard file format that is used to hold data in an object format. Do you remember while doing API classes like the weather app we had viewed the JSON file in **Json Viewer**.
            - So we are adding this at the end of the link because we just want to access the model created in a teachable machine and nothing else from the model which has been created.
    - Second, a function, this function will start the ml5 image classification.

```
classifier = ml5.imageClassifier('https://teachablemachine.withgoogle.com/models/v_sl95BzE/model.json' modelLoaded);
```

    - If we don't pass this function, then ml5 image classification won't start.

Now write the **modelLoaded** function:

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

classifier = ml5.imageClassifier('https://teachablemachine.withgoogle.com/models/v_sl95BzE/model.json',modelLoaded);

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

Zoom in look:

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

10. First define the function `function modelLoaded() {`

11. Now console a message just to check that ml5 image classification has started.

```
  console.log('Model Loaded!');
```

- Output:

```
ml5 version: 0.4.3                                    main.js:21
Model Loaded!                                         main.js:26
```

Now we will define speak() function
This function will help to do text to speech of the results and speak out the results got from the modal

```
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);
  synth.speak(utterThis);

}
```

Explaining the code:

1. Define the function.

```
function speak(){
```

2. We will define an API and store it inside a variable for converting the result of which is in text into speech.

```
function speak(){
    var synth = window.speechSynthesis;
```

3. If we want the system to speak the same, we need to give that particular text to the system to speak. So for that -

4. First we will define a variable for prediction 1 `speak_data_1 =` and assign a write up that says “The first prediction is”

```
speak_data_1 = "The first prediction is "
```

Then concat the variable which will be holding the result of prediction 1

```
function speak(){
  var synth = window.speechSynthesis;
  speak_data_1 = "The first prediction is " + prediction_1;
```

5. Now we will define another variable for prediction 2 `speak_data_2` and assign a write up that says “And the second prediction is”

```
speak_data_2 = "And the second prediction is "
```

Then concat the variable which will be holding the result of prediction 2

```
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;
```

6. Now we will convert this text to speech.

```
var utterThis = new SpeechSynthesisUtterance(speak_data_1 + speak_data_2);
```

- **utterThis** - is a variable in which we will store the converted text to speech.
- **SpeechSynthesisUtterance** - is the function of an **API** that will convert text to speech.
- We are using a **new** keyword because, for every next result, we want to

convert that text to speech.
- **speak_data_1** - contains the text which is the result of the first prediction
- **speak_data_2** - contains the text which is the result of the second prediction

7. Now we have converted text to speech which is stored variable, so we will pass this variable to the **speak()** function of the **API**.

```
synth.speak(utterThis);
```

- **synth** - because in this we have stored the **API** in **point 2**
- **speak()** - **speak()** function is a predefined function of the **API**.
- **utterThis** - has the converted value of text to speech that we want to the system to speak

The functionality of this speak() function is to trigger the system to speak whatever is passed inside this speak function

What's NEXT?

In the next class we will complete making the facial emotion detection web application by writing codes for the check() and gotResult() functions.