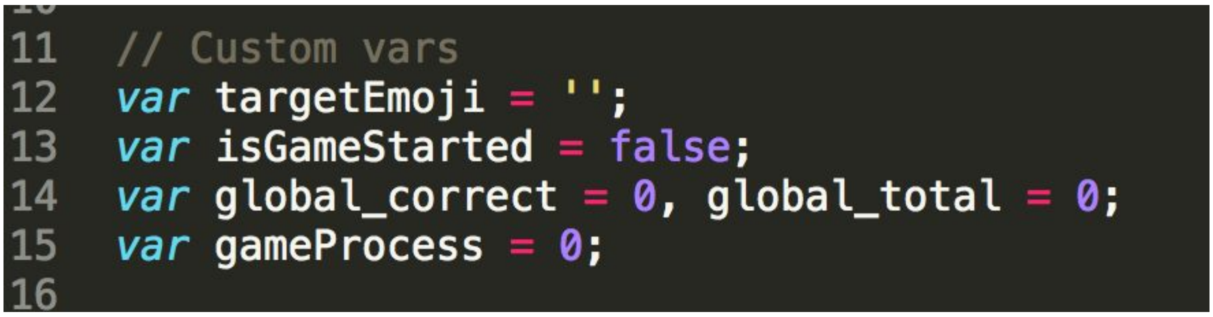
AIND Term 3 Project Mimic Me

1. To Start with, we initialize the necessary variabels:

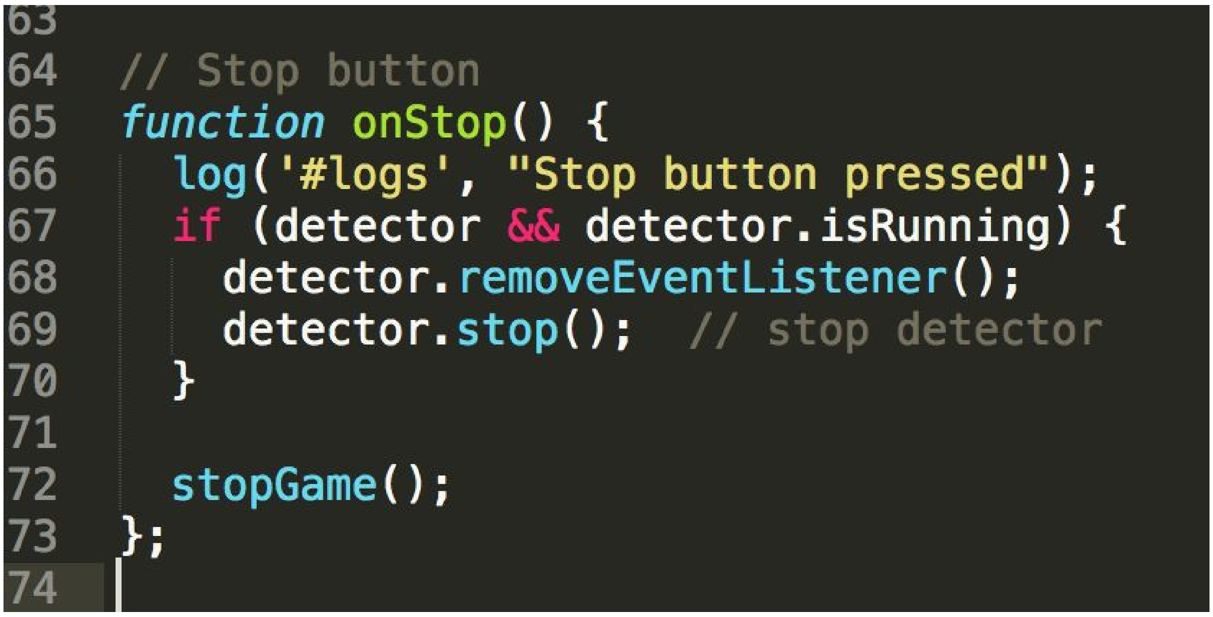


```
// Custom vars
var targetEmoji = '';
var isGameStarted = false;
var global_correct = 0, global_total = 0;
var gameProcess = 0;
```

Here are some special variables descriptions:

- isGameStarted - when game is started (by pressing a start button), this variable will be set to True. This is a flag to recognize if a game is started or not.
- global_correct/global_total - store the overall score: correctness and total number of emoji shown and matched.
- gameProcess - this is a time interval ID used to control the process of a game

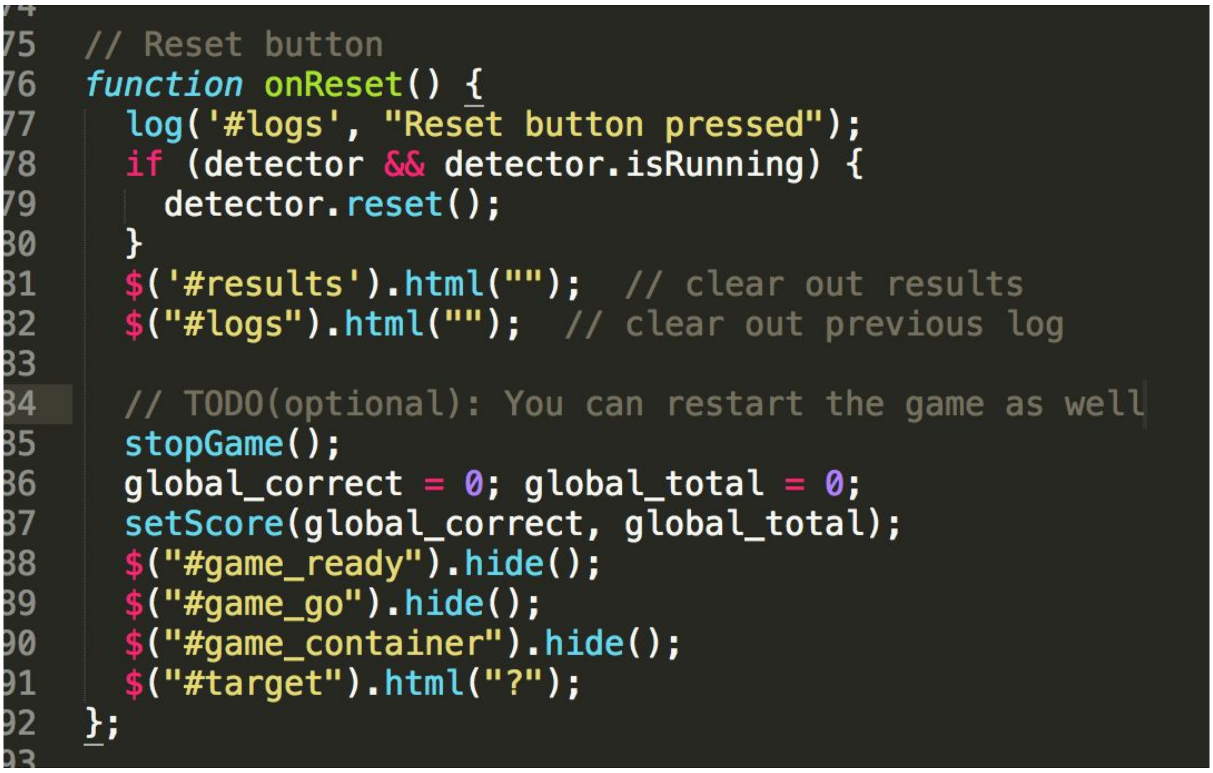
2. Then we defined a list of functions. Some functions have been modified as below:



```
// Stop button
function onStop() {
  log('#logs', "Stop button pressed");
  if (detector && detector.isRunning) {
    detector.removeEventListener();
    detector.stop();  // stop detector
  }

  stopGame();
};
```

The onStop() function have a new stopGame() function called when the stop button is pressed. Not only will the program stop the camera sensor but also stop the game if it is started.

3. Additional function added on onReset() function. Here is the description:

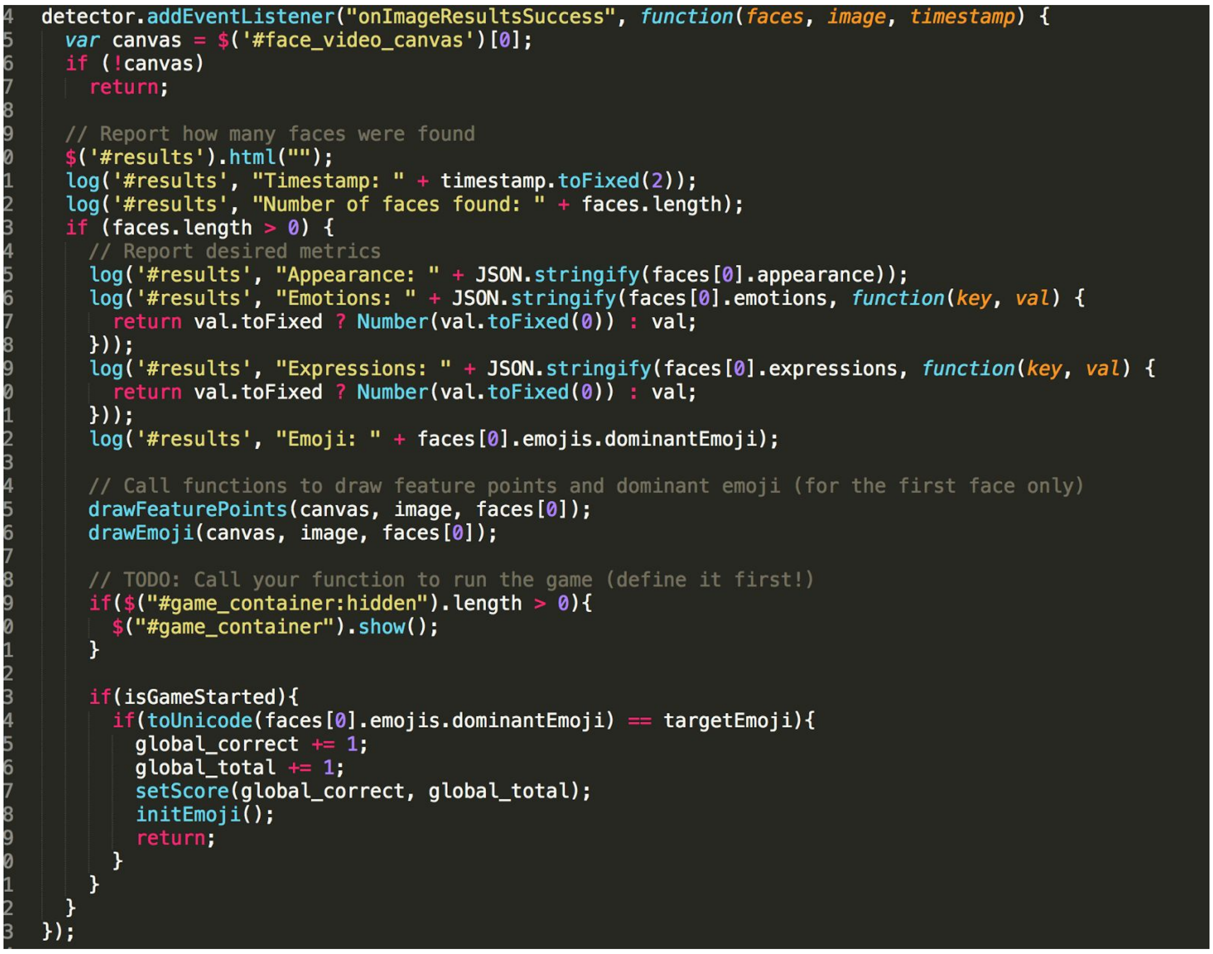


```
// Reset button
function onReset() {
  log('#logs', "Reset button pressed");
  if (detector && detector.isRunning) {
    detector.reset();
  }
  $('#results').html("");  // clear out results
  $("#logs").html("");  // clear out previous log

  // TODO(optional): You can restart the game as well
  stopGame();
  global_correct = 0; global_total = 0;
  setScore(global_correct, global_total);
  $("#game_ready").hide();
  $("#game_go").hide();
  $("#game_container").hide();
  $("#target").html("?");
};
```

When the “Reset” button is called, not only will the game stopped, but also to reset all necessary variables and on screen components (like buttons) back to the beginning number.

Please note that there is a new “Start game” button added to the page and, there is also a “Ready” / “Go” status change to get people ready for the game.

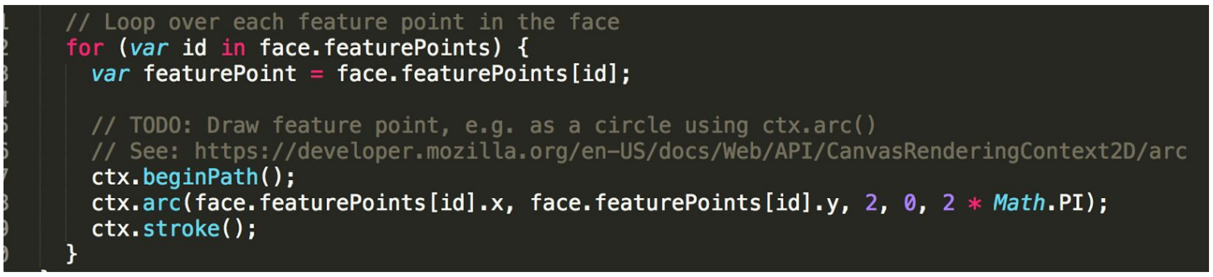
4. The main function: here is how the main function works and what are added to this function.



```
detector.addEventListener("onImageResultsSuccess", function(faces, image, timestamp) {
  var canvas = $('#face_video_canvas')[0];
  if (!canvas)
    return;

  // Report how many faces were found
  $('#results').html("");
  log('#results', "Timestamp: " + timestamp.toFixed(2));
  log('#results', "Number of faces found: " + faces.length);
  if (faces.length > 0) {
    // Report desired metrics
    log('#results', "Appearance: " + JSON.stringify(faces[0].appearance));
    log('#results', "Emotions: " + JSON.stringify(faces[0].emotions, function(key, val) {
      return val.toFixed ? Number(val.toFixed(0)) : val;
    }));
    log('#results', "Expressions: " + JSON.stringify(faces[0].expressions, function(key, val) {
      return val.toFixed ? Number(val.toFixed(0)) : val;
    }));
    log('#results', "Emoji: " + faces[0].emojis.dominantEmoji);

    // Call functions to draw feature points and dominant emoji (for the first face only)
    drawFeaturePoints(canvas, image, faces[0]);
    drawEmoji(canvas, image, faces[0]);

    // TODO: Call your function to run the game (define it first!)
    if($("#game_container:hidden").length > 0){
      $("#game_container").show();
    }

    if(isGameStarted){
      if(toUnicode(faces[0].emojis.dominantEmoji) == targetEmoji){
        global_correct += 1;
        global_total += 1;
        setScore(global_correct, global_total);
        initEmoji();
        return;
      }
    }
  }
});
```

- The first part is to check if a face is detected.
- Interestingly enough that when I wear my glasses, although the program recognize that I have a glasses but, the face detection anchors were not shown. Probably because my face may still be “not recognizable” according to the library
- If a face is detected, a series of log will be shown on the page, and then it will start drawing feature points on the face and recognize the emoji currently on the face.
- Then we initialize the game: first if a face is found, show the “Start game” button. Otherwise the game cannot start if there is no feature points on a face
- Now check if a game is started, if yes, check if currently detected emoji is the same as targetEmoji, which is picked when the game started (more on this later).
- If match, increment the correct and total score, then use setScore to update the score on the screen, then use initEmoji() to load another emoji.

5. Now let’s go into the drawFeaturePoints() function, which draw some small circle points on a detected faces.

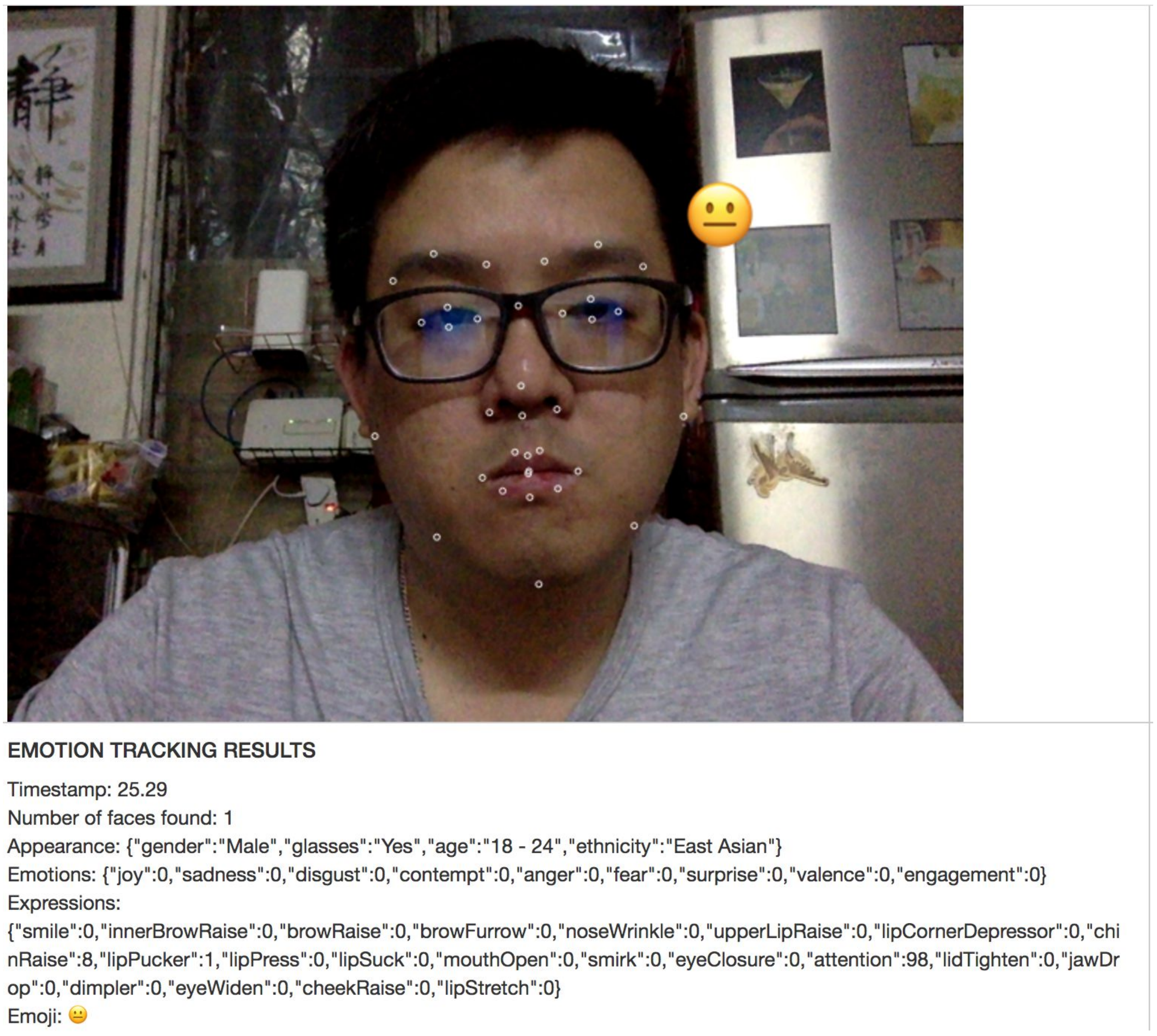


```
  // Loop over each feature point in the face
  for (var id in face.featurePoints) {
    var featurePoint = face.featurePoints[id];

    // TODO: Draw feature point, e.g. as a circle using ctx.arc()
    // See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/arc
    ctx.beginPath();
    ctx.arc(face.featurePoints[id].x, face.featurePoints[id].y, 2, 0, 2 * Math.PI);
    ctx.stroke();
  }
```

The key is that we first locate all the (x,y) coordinate on a detected face coming from the library, and then use the arc(x,y) function to draw a very small circle on this point.

Since when the face in the camera moves, this function is called, which means the x,y and the “circle” positions will be updated continuously.

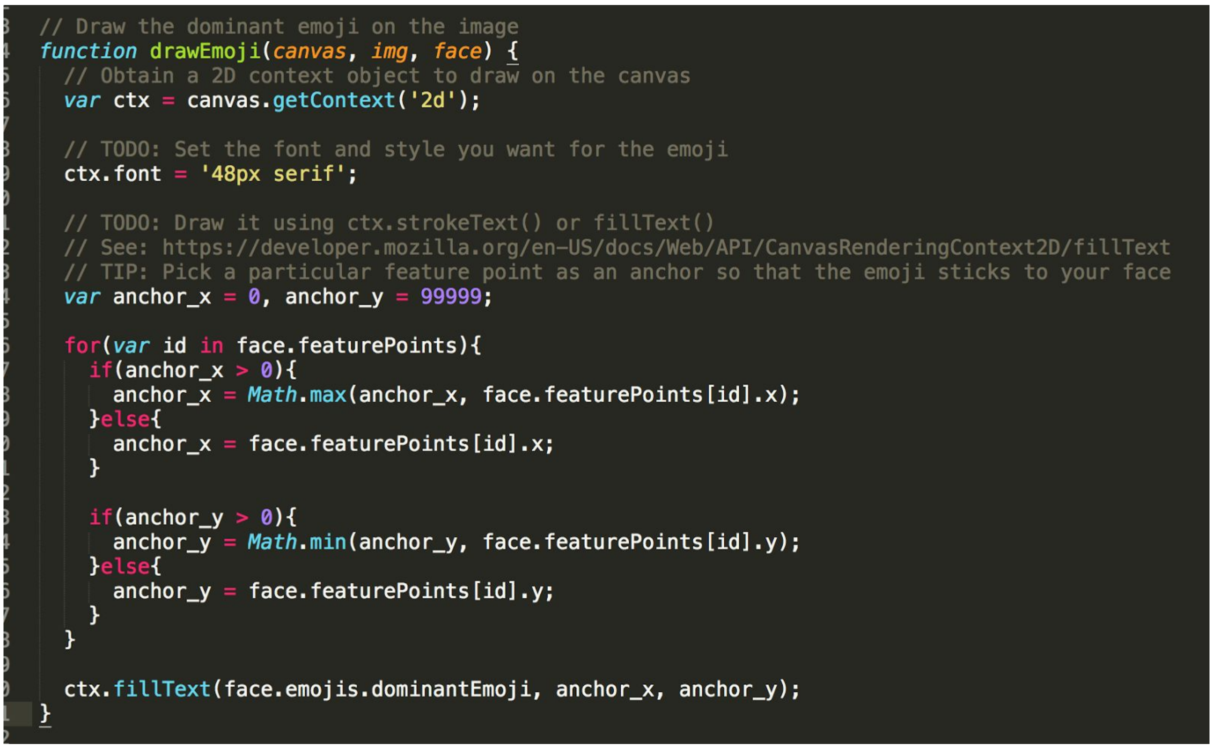
And the final result will be like this:



EMOTION TRACKING RESULTS

Timestamp: 25.29
Number of faces found: 1
Appearance: {"gender":"Male","glasses":"Yes","age":"18 - 24","ethnicity":"East Asian"}
Emotions: {"joy":0,"sadness":0,"disgust":0,"contempt":0,"anger":0,"fear":0,"surprise":0,"valence":0,"engagement":0}
Expressions:
{"smile":0,"innerBrowRaise":0,"browRaise":0,"browFurrow":0,"noseWrinkle":0,"upperLipRaise":0,"lipCornerDepressor":0,"chinRaise":8,"lipPucker":1,"lipPress":0,"lipSuck":0,"mouthOpen":0,"smirk":0,"eyeClosure":0,"attention":98,"lidTighten":0,"jawDrop":0,"dimpler":0,"eyeWiden":0,"cheekRaise":0,"lipStretch":0}
Emoji: 😐

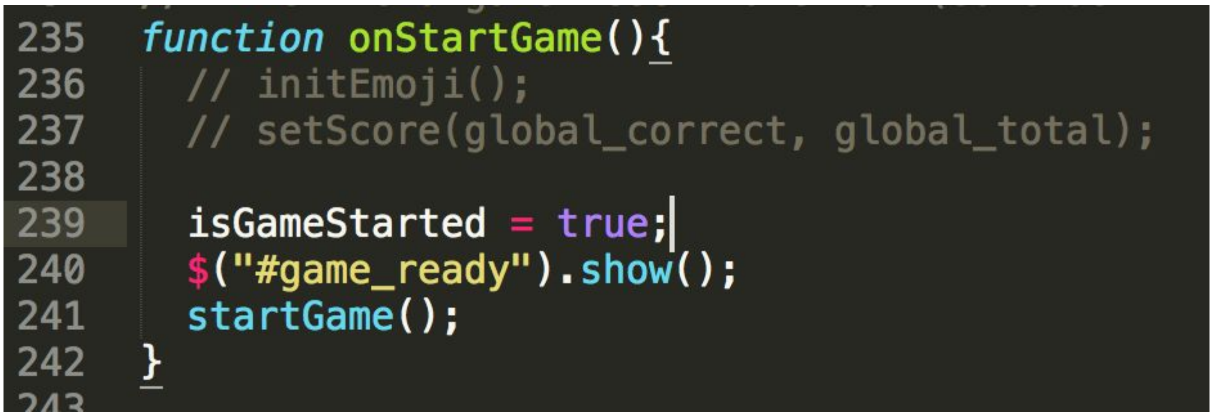
6. Now we draw the emoji on the nearest position of my face.



```
// Draw the dominant emoji on the image
function drawEmoji(canvas, img, face) {
  // Obtain a 2D context object to draw on the canvas
  var ctx = canvas.getContext('2d');

  // TODO: Set the font and style you want for the emoji
  ctx.font = '48px serif';

  // TODO: Draw it using ctx.strokeText() or fillText()
  // See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/fillText
  // TIP: Pick a particular feature point as an anchor so that the emoji sticks to your face
  var anchor_x = 0, anchor_y = 99999;

  for(var id in face.featurePoints){
    if(anchor_x > 0){
      anchor_x = Math.max(anchor_x, face.featurePoints[id].x);
    }else{
      anchor_x = face.featurePoints[id].x;
    }

    if(anchor_y > 0){
      anchor_y = Math.min(anchor_y, face.featurePoints[id].y);
    }else{
      anchor_y = face.featurePoints[id].y;
    }
  }

  ctx.fillText(face.emojis.dominantEmoji, anchor_x, anchor_y);
}
```

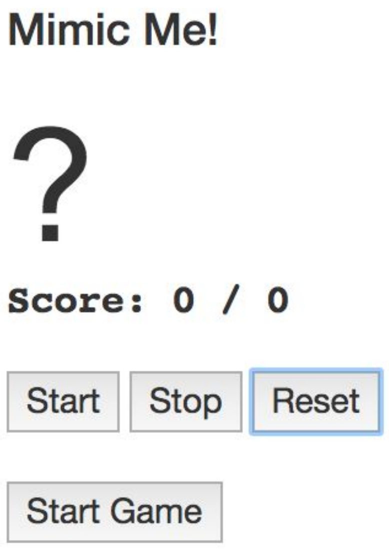
- First we define the size of the emoji. Now you can think of this emoji icon as a normal HTML display text, we can define font-size and font-type like normal “font” css here, which is “48px serif”.
- Second, we initialize a x,y anchors as 0,0
- Then, loop through all the faces featurePoints, and fine the rightmost (max of anchor x) and top-most (min of anchor y) of the anchor, and set the value of it.
- Finally, display the detected emoji using anchor_x and anchor_y

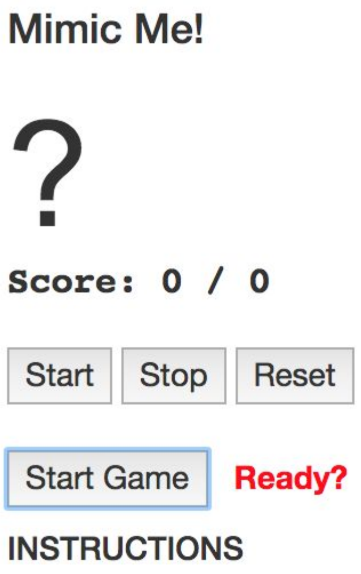
7. There are some support functions for the game as follow:



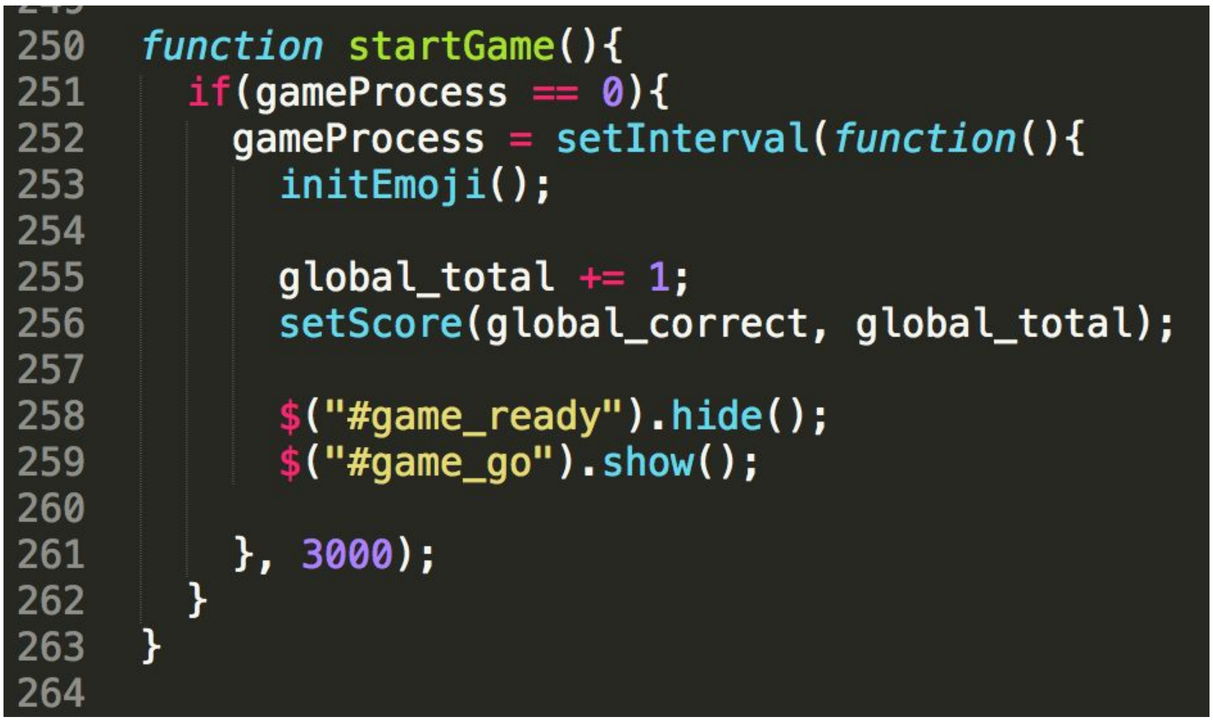
```
function onStartGame(){
  // initEmoji();
  // setScore(global_correct, global_total);

  isGameStarted = true;
  $("#game_ready").show();
  startGame();
}
```

onStartGame() is mainly used to initialize the game by setting the isGameStarted to true. Then run the startGame() function. Like below:



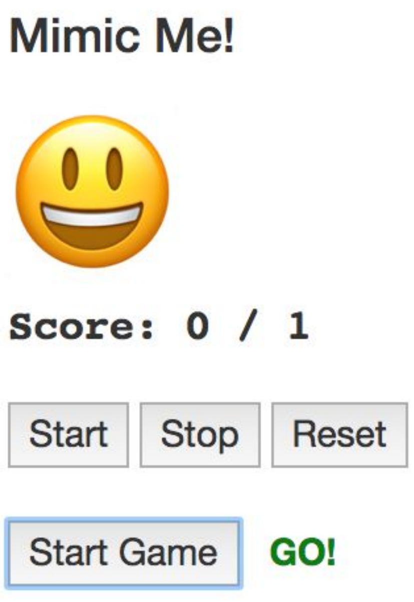


8. When the game started, the following function will be called:

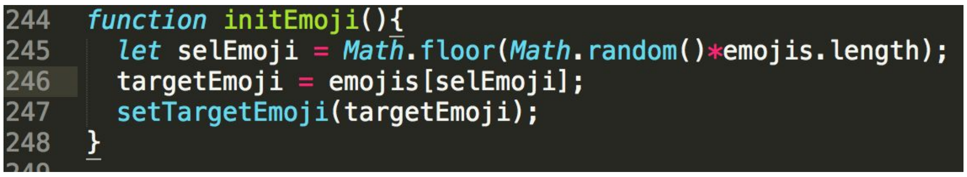


```
function startGame(){
  if(gameProcess == 0){
    gameProcess = setInterval(function(){
      initEmoji();

      global_total += 1;
      setScore(global_correct, global_total);

      $("#game_ready").hide();
      $("#game_go").show();

    }, 3000);
  }
}
```

- First, check if a game is currently running by seeing if an ID is assigned to gameProcess. If it is 0 (zero), which means there is no game started.
- Now we know that the game is not started. We use javascript setInterval() function to continousuly:
    - Update the emoji
    - Increment the total count
    - Show game status
- And this process will be re-run (which means the emoji will be automatically updated) every 3 seconds.
- The final outcome is as follow:



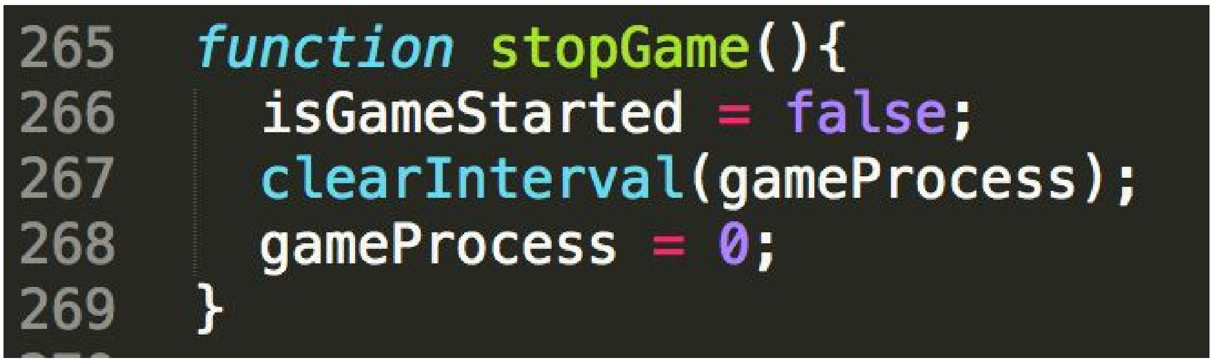
9. Finally, there are two more functions to be included here:



```
function initEmoji(){
  let selEmoji = Math.floor(Math.random()*emojis.length);
  targetEmoji = emojis[selEmoji];
  setTargetEmoji(targetEmoji);
}
```

- initEmoji() is used to pick a random emoji from current emojis list, and then use setTargetEmoji() to display the selected emoji, while update the global variable targetEmoji to be used by the game.

10. stopGame()



```
function stopGame(){
  isGameStarted = false;
  clearInterval(gameProcess);
  gameProcess = 0;
}
```

- This function is to stop the game by setting the isGameStarted flag to false.
- Then clear the process interval using clearInterval() function
- To play save, we reset the gameProcess by assiging a 0 (zero) to it.