Term 2 - Project 3: Mimic Me!

By Paula Martínez

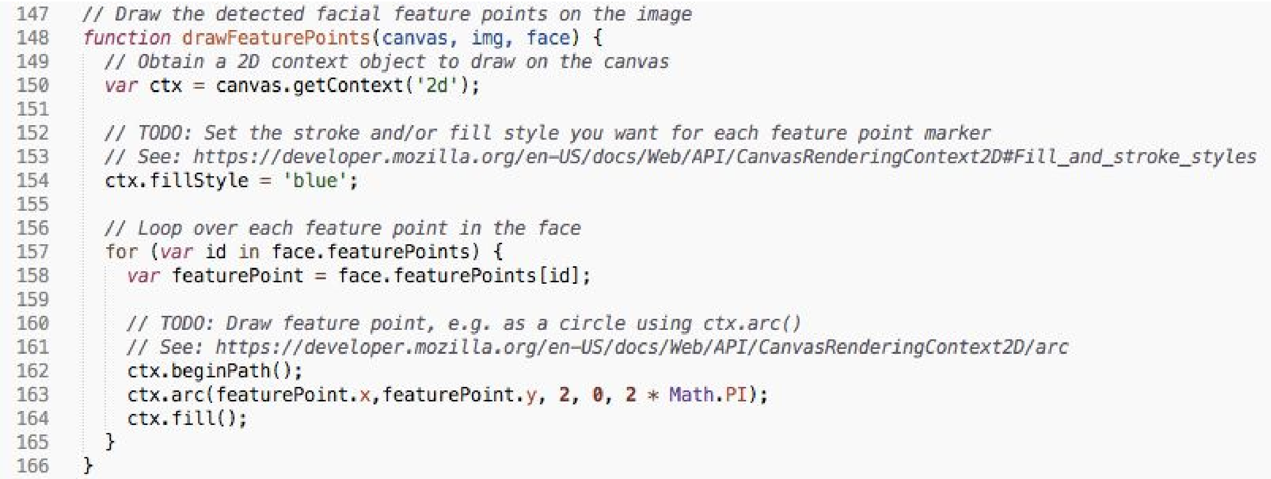
Introduction

In this project, you will learn to track faces in a video and identify facial expressions using Affectiva. As a fun visualization, you will tag each face with an appropriate emoji next to it. You will then turn this into a game where the player needs to mimic a random emoji displayed by the computer.

Feature Points

The function *drawFeaturePoints* should draw the feature points recognized on the face. The inputs for this function are the canvas (where the points should be drawn), the image and the first face detected on the image.

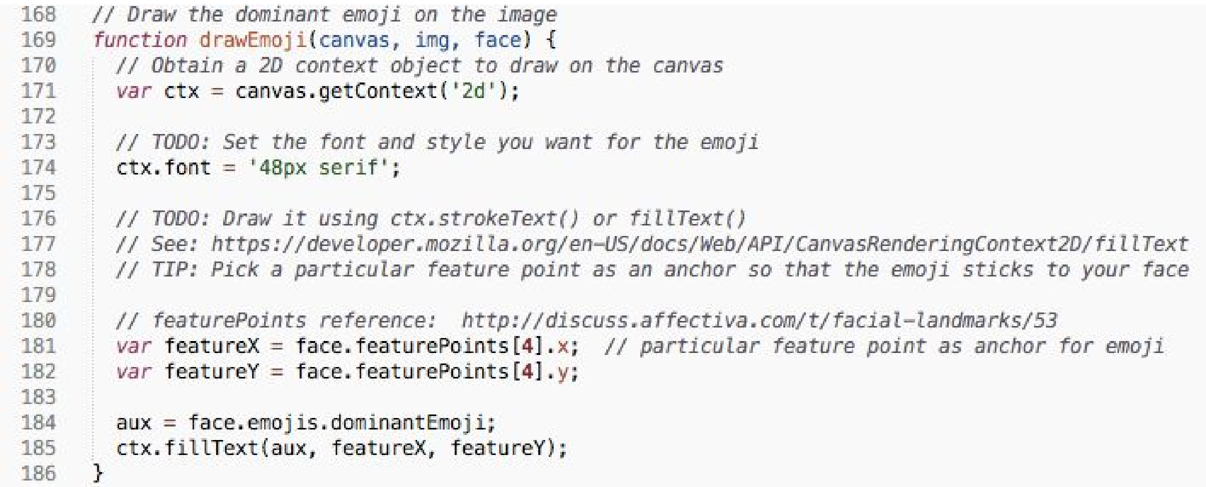
First, the fill style for drawing the features points is set to blue.
Then, all features points are drawn and fill calling the functions arc and fill. To determine where to draw the points, are used the attributes x and y of face.featurePoints.



```
// Draw the detected facial feature points on the image
function drawFeaturePoints(canvas, img, face) {
  // Obtain a 2D context object to draw on the canvas
  var ctx = canvas.getContext('2d');

  // TODO: Set the stroke and/or fill style you want for each feature point marker
  // See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D#Fill_and_stroke_styles
  ctx.fillStyle = 'blue';

  // 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(featurePoint.x,featurePoint.y, 2, 0, 2 * Math.PI);
    ctx.fill();
  }
}
```

Dominant Emoji

The function *drawEmoji* should draw the dominant emoji detected on the image. The inputs to the function are the same as for the previous function.
Fists is set the font and style for drwaing the emoji with the function ctx.font. Then I pick the feature point number 4 as an anchor for the emoji. Using again the attributes x and y of the face.featurePoints (from the fourth feature point in this case) are find the coordinates to draw the emoji in the image. To determine the dominant emoji found is used the attribute dominantEmoji from face.emojis, and the emoji is drawn using fillText.

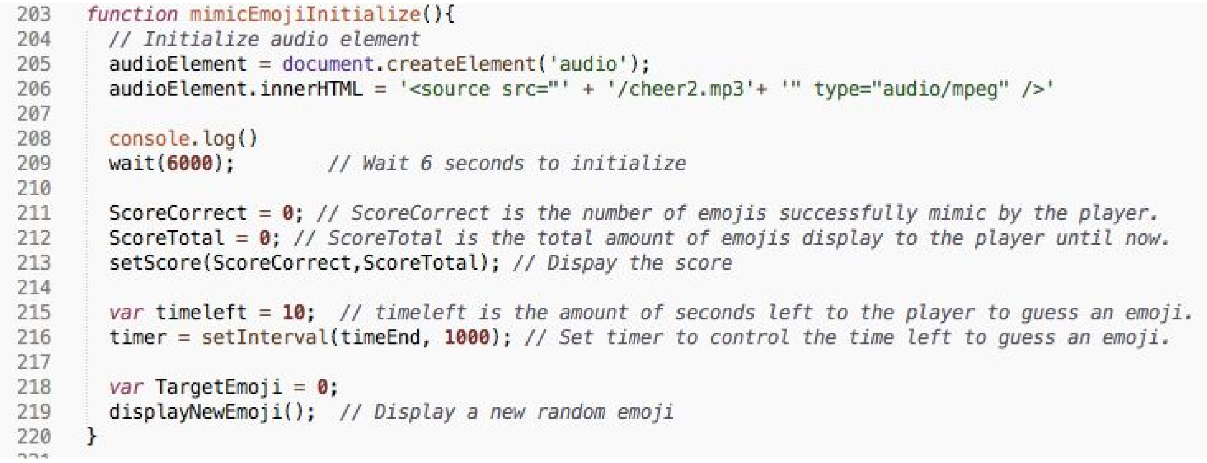


```
// Draw the dominant emoji on the image
function drawEmoji(canvas, img, face) {
  // Obtain a 2D context object to draw on the canvas
  var ctx = canvas.getContext('2d');

  // 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

  // featurePoints reference:  http://discuss.affectiva.com/t/facial-landmarks/53
  var featureX = face.featurePoints[4].x;  // particular feature point as anchor for emoji
  var featureY = face.featurePoints[4].y;

  aux = face.emojis.dominantEmoji;
  ctx.fillText(aux, featureX, featureY);
}
```

Mimic Game

To initialize the game is called the function ***mimicEmojiInitialize()***. This function does the following:

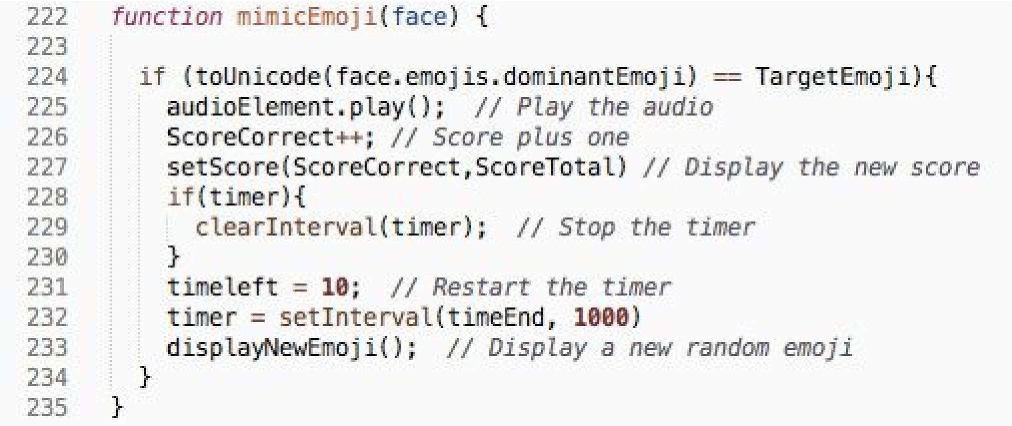
- Initialize an audio track that is going to be played when the player mimics the right emoji.
- Waits six seconds, so the camera and the rest of the software can set up. For this is used the auxiliary function wait(ms).
- Sets the variable ScoreCorrect to zero. ScoreCorrect is the number of emojis successfully mimic by the player.
- Sets the variable ScoreTotal to zero. ScoreTotal is the total amount of emojis display to the player until now.
- Displays the score with the function setScore().
- Sets to zero the variable timeleft. timeleft is the amount of seconds left to the player to guess an emoji.
- Start the timer to control the time left to guess an emoji using the function setInterval. To do it is used the auxiliary function timeEnd().
- Defines the variable TargetEmoji (the emoji being display that the player has to mimic).
- Calls the function displayNewEmoji() to display a new emoji to be mimic by the player.



```
function mimicEmojiInitialize(){
  // Initialize audio element
  audioElement = document.createElement('audio');
  audioElement.innerHTML = '<source src="' + '/cheer2.mp3'+ '" type="audio/mpeg" />'

  console.log()
  wait(6000);       // Wait 6 seconds to initialize

  ScoreCorrect = 0; // ScoreCorrect is the number of emojis successfully mimic by the player.
  ScoreTotal = 0; // ScoreTotal is the total amount of emojis display to the player until now.
  setScore(ScoreCorrect,ScoreTotal); // Dispay the score

  var timeleft = 10;  // timeleft is the amount of seconds left to the player to guess an emoji.
  timer = setInterval(timeEnd, 1000); // Set timer to control the time left to guess an emoji.

  var TargetEmoji = 0;
  displayNewEmoji();  // Display a new random emoji
}
```

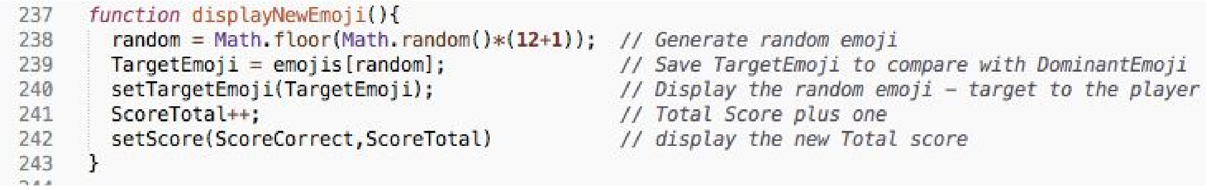
The function **mimicEmoji()** is the one that runs the game. The function continually evaluates if the dominant emoji detected from the face of the player is the same as the target emoji displayed. If so, the function does the following:

- Plays the audio to indicates that the player correctly mimic the emoji displayed.
- Add one point to the score and displays the new score.
- Clear and reset the timer
- Calls the funcition displayNewEmoji() to display a new emoji and start all over again.



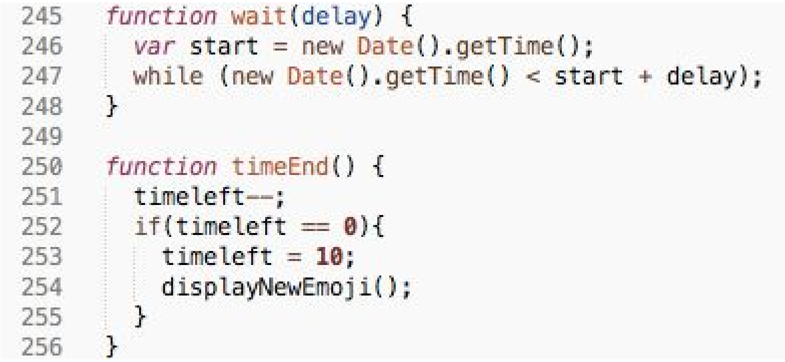
```
function mimicEmoji(face) {

  if (toUnicode(face.emojis.dominantEmoji) == TargetEmoji){
    audioElement.play();  // Play the audio
    ScoreCorrect++; // Score plus one
    setScore(ScoreCorrect,ScoreTotal) // Display the new score
    if(timer){
      clearInterval(timer);  // Stop the timer
    }
    timeleft = 10;  // Restart the timer
    timer = setInterval(timeEnd, 1000)
    displayNewEmoji();  // Display a new random emoji
  }
}
```

The function **displayNewEmoji()** chooses randomly an emoji from the array of emojis and display it with the function setTargetEmoji(). Because a new emoji is being displayed, this function also adds one to the ScoreTotal variable and display it.



```
function displayNewEmoji(){
  random = Math.floor(Math.random()*(12+1));  // Generate random emoji
  TargetEmoji = emojis[random];               // Save TargetEmoji to compare with DominantEmoji
  setTargetEmoji(TargetEmoji);                // Display the random emoji - target to the player
  ScoreTotal++;                               // Total Score plus one
  setScore(ScoreCorrect,ScoreTotal)           // display the new Total score
}
```

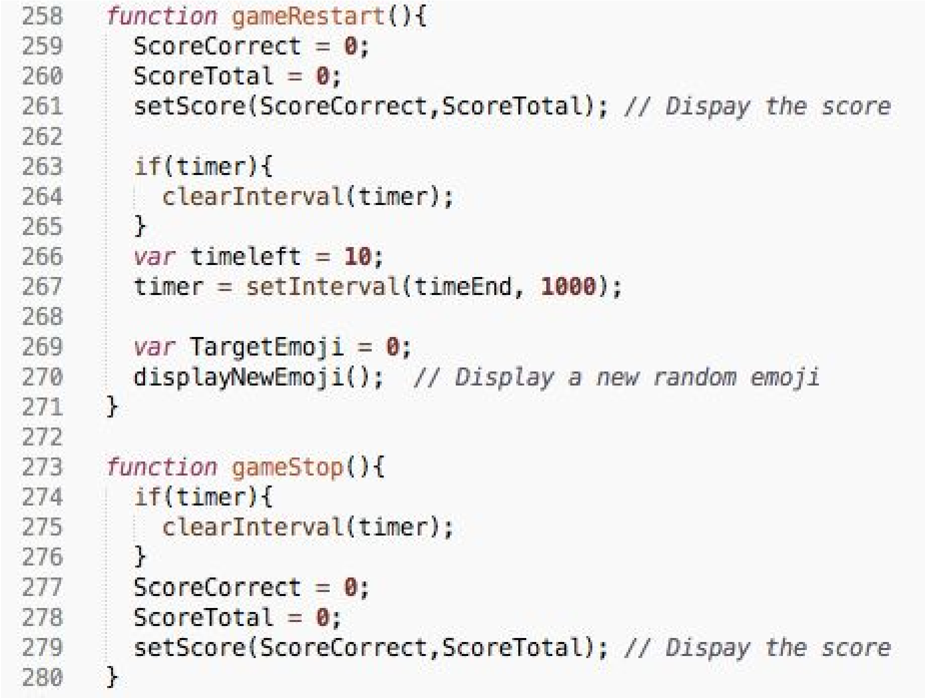
The auxiliary function **wait()** waits a amount of time in milliseconds. The function **timeEnd()** reduces in one the variable timeleft every time is called by the function setInterval; if the remaining time is zero (timeleft is zero), the time to try to imitate the emoji is over and a new emoji is displayed calling the function displayNewEmoji. In this case the timer is reset, setting timeleft to 10 again.



```
function wait(delay) {
  var start = new Date().getTime();
  while (new Date().getTime() < start + delay);
}

function timeEnd() {
  timeleft--;
  if(timeleft == 0){
    timeleft = 10;
    displayNewEmoji();
  }
}
```

The **gameReset()** function is called when the reset button is pressed. It resets the score variables, the timer and displays a new emoji.

The **gameStop()** function is called when the stop button is pressed. It stops the timer and reset the score variables.



```
function gameRestart(){
  ScoreCorrect = 0;
  ScoreTotal = 0;
  setScore(ScoreCorrect,ScoreTotal); // Dispay the score

  if(timer){
    clearInterval(timer);
  }
  var timeleft = 10;
  timer = setInterval(timeEnd, 1000);

  var TargetEmoji = 0;
  displayNewEmoji();  // Display a new random emoji
}

function gameStop(){
  if(timer){
    clearInterval(timer);
  }
  ScoreCorrect = 0;
  ScoreTotal = 0;
  setScore(ScoreCorrect,ScoreTotal); // Dispay the score
}
```