Artificial Intelligence Module B Unit #2 pre-trained faceMesh model

Content

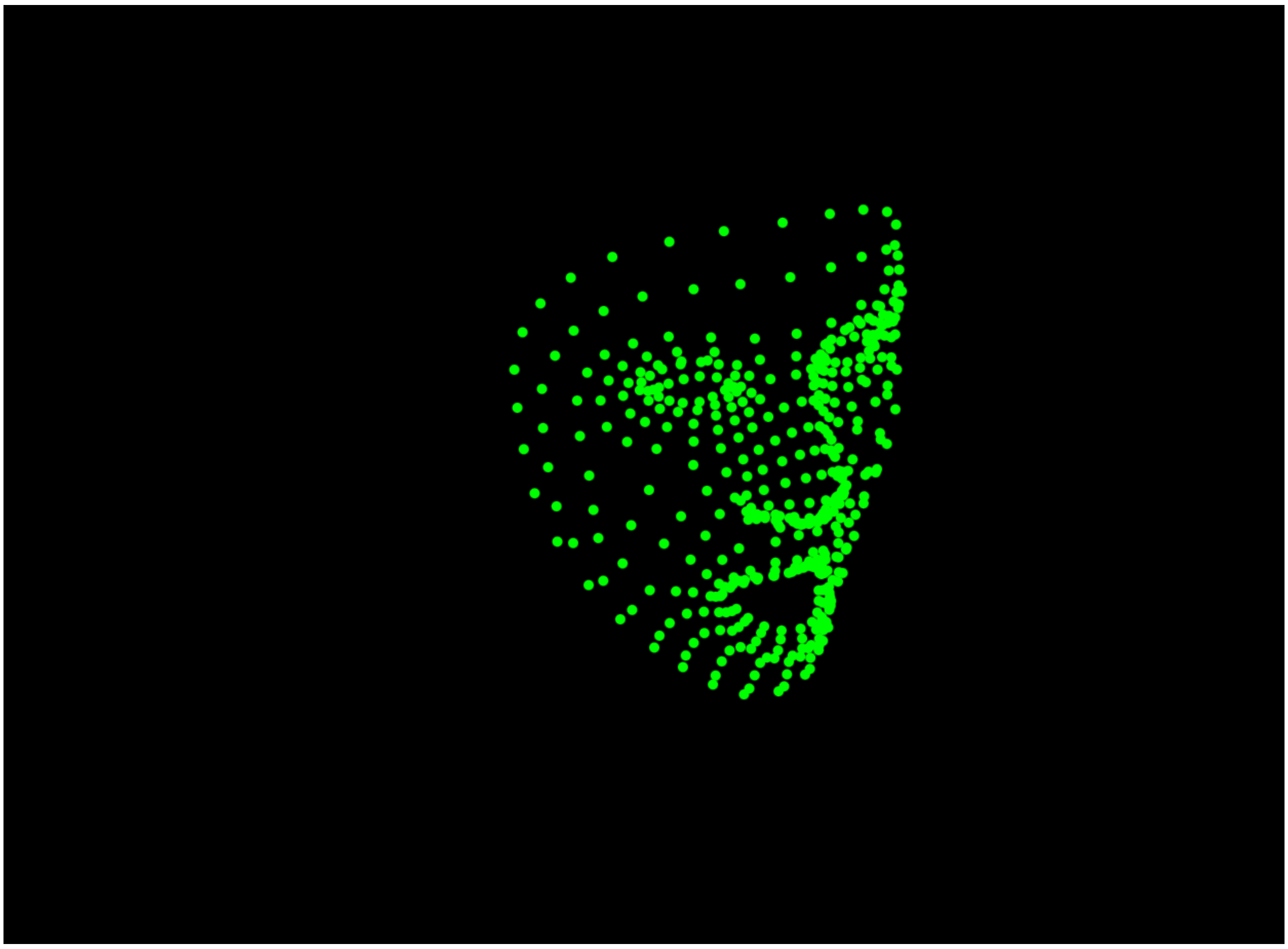
Module B Unit #2 faceMesh

Introduction to pretrained faceMesh with ml5.js

You will need a webcam for this unit. Much of what we are going to do is covered in previous units and so will not spend time elaborating on how to create the video input.

Embedded within the ml5.js library there is a model called faceMesh which has been trained to identify over 400 points on any face. These points can identify the mouth, the eyes, eyebrows, etc. We can access these points if we want to focus on those particular features. The data points of the faceMesh have 3D values but we will just use the 2D co-ordinates. In this example, we are going to draw a box around the mouth using faceMesh, add circles for eyes and draw all the points.

Figure 1: all of the data points of my face



FaceMesh points

The points on the faceMesh are numbered. This means you can reference specific points on the mesh. You can see the reference points and their associated numbers. I will include a button link to this on the website.

Figure 2: faceMesh reference points

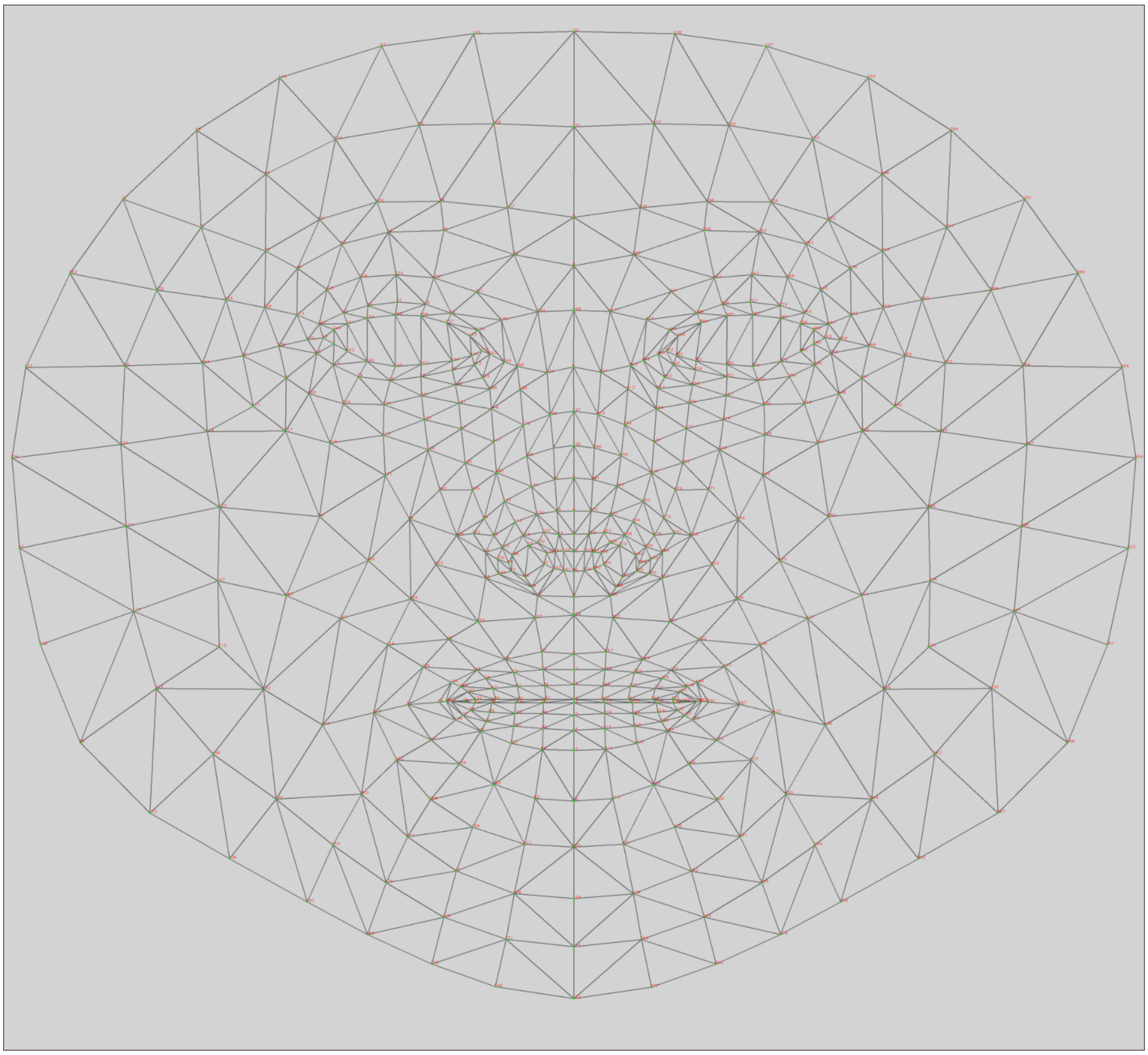
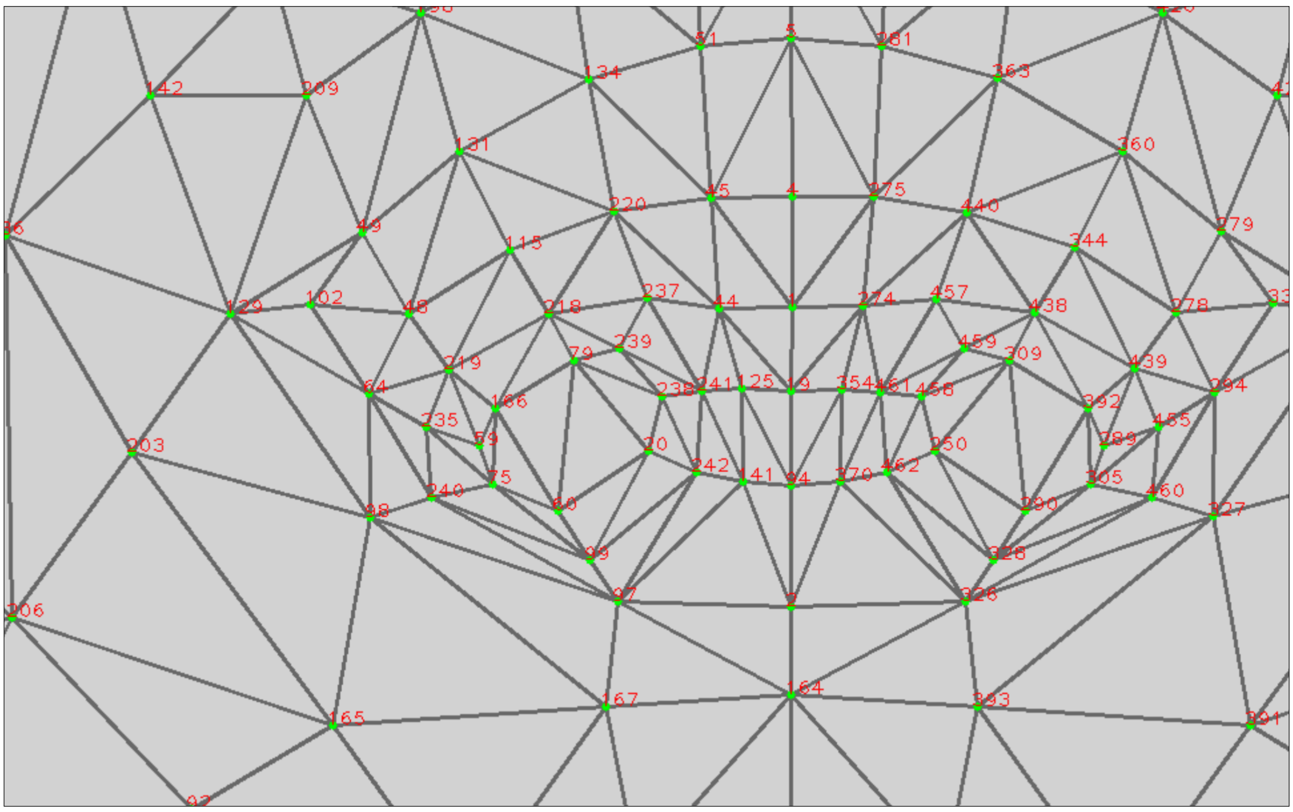


Figure 3: zooming in



The index.html file

Adding the `ml5.js` to the `index.html` file, just as you have done before.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/
1.11.1/p5.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/
1.11.1/addons/p5.sound.min.js"></script>
    <script src="https://unpkg.com/ml5@1/dist/ml5.min.js"></
script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8" />


  </head>
  <body>
    <main>
    </main>
    <script src="sketch.js"></script>
  </body>
</html>
```

Sketch B2.1 our starting sketch

Starting sketch as per normal, except note the canvas size.

```
function setup()
{
  createCanvas(640, 480)
}

function draw()
{
  background(220)
}
```

Notes

You should have a grey canvas 640 by 480. We will use this size of canvas because we will be using a video feed. You might need to slide the canvas, code divide by clicking and dragging it so you can see the whole canvas.

Sketch B2.2 creating a video from your webcam

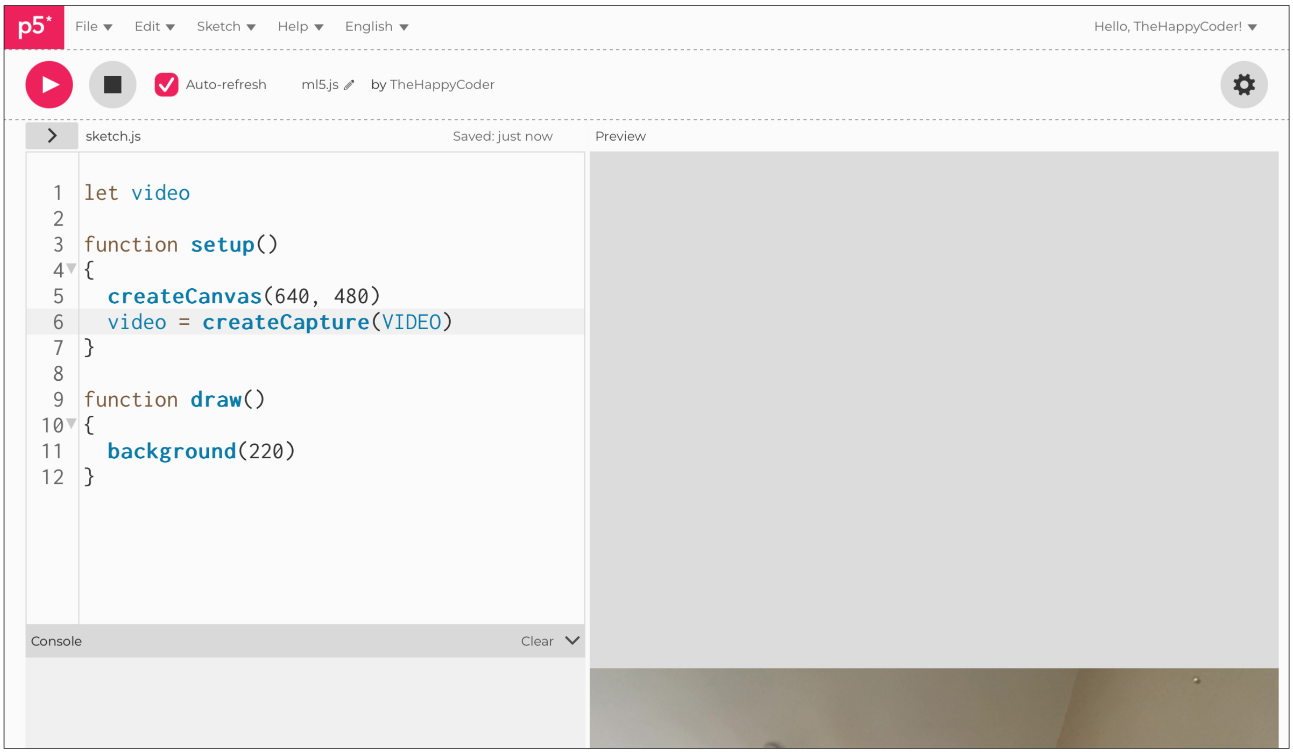
We added a video capture function to our basic sketch.

```
let video


function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO)
}


function draw()
{
  background(220)
}
```

Notes

You should have the grey canvas, and underneath it there will be a video feed from your webcam. You will probably need to give your computer permission to allow p5.js to access the webcam.

Figure B2.2



```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO)
}

function draw()
{
  background(220)
}
```

Sketch B2.3 video on the canvas

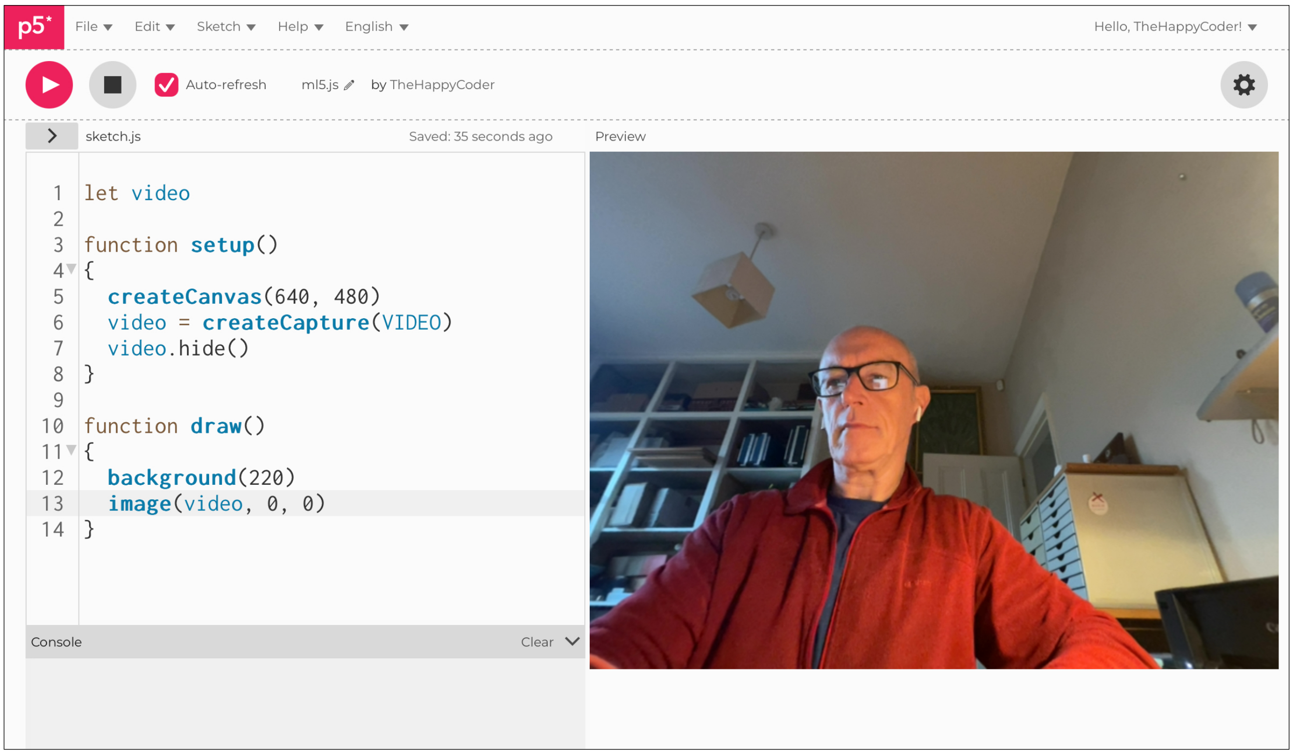
As we want the video in the canvas, we will hide the live stream and create an `image()` of the video on the canvas.

```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO)
  video.hide()
}

function draw()
{
  background(220)
  image(video, 0, 0)
}
```

Notes

The video stream has disappeared from below the canvas and should now appear on the canvas in the dimensions we are using. The full size is `(640, 480)`.

Figure B2.3



```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO)
  video.hide()
}

function draw()
{
  background(220)
  image(video, 0, 0)
}
```

Sketch B2.4 video flipped

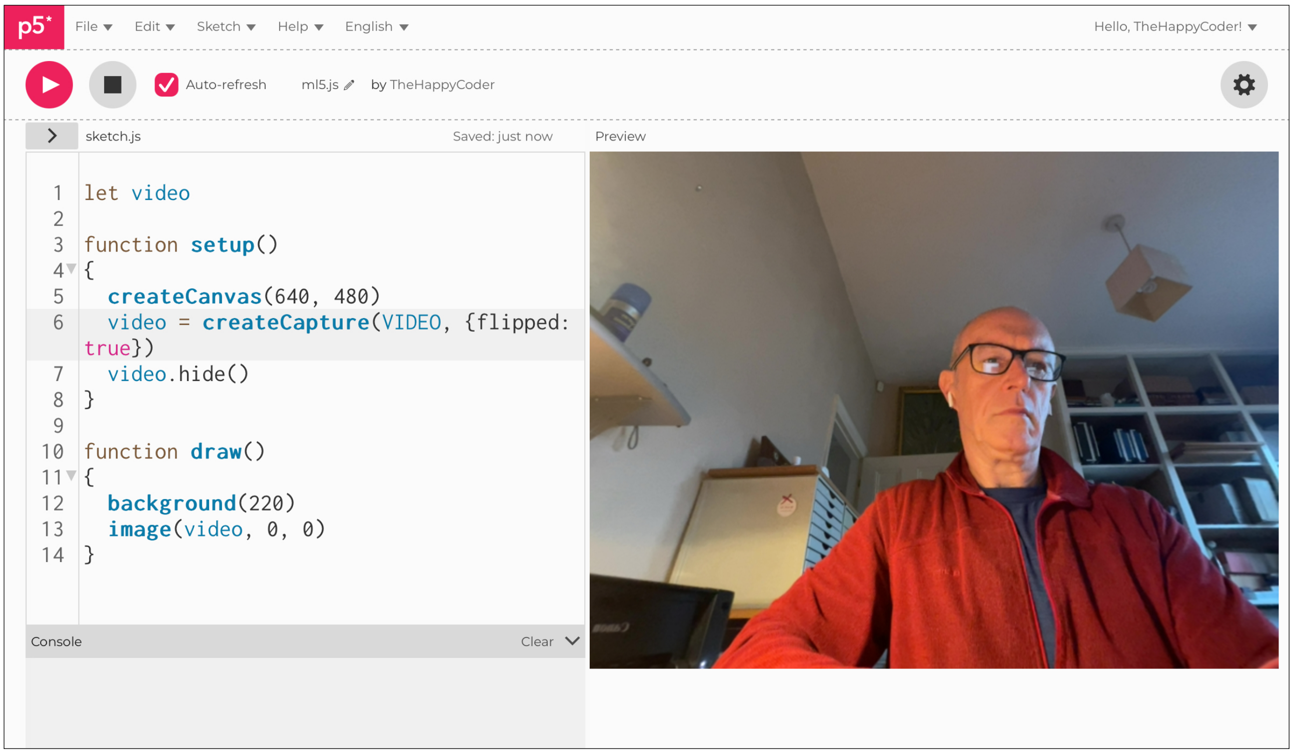
To mirror the image, we will flip the video.

```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
}

function draw()
{
  background(220)
  image(video, 0, 0)
}
```

Notes

The image should now be reversed or mirrored.

Figure B2.4



```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
}

function draw()
{
  background(220)
  image(video, 0, 0)
}
```

Sketch B2.5 faceMesh

Now we want to use the faceMesh model, so we need to upload the model asynchronously that is built into ml5.js.

```
let video
let faceMesh

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh()
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
}

function draw()
{
  background(220)
  image(video, 0, 0)
}
```

Notes

Nothing will happen on the canvas just yet; we need to connect the faceMesh to the video.

Code Explanation

`let faceMesh`	Create a variable to hold the face mesh
`faceMesh = ml5.faceMesh()`	Call the pretrained model faceMesh in ml5.js

Sketch B2.6 callback

We can use a built-in function for detecting a video and any face(s) in it called detectStart(). We can specify how many faces to detect; for a single face, use the function maxFaces and make it equal to 1. We will also need a callback function; we will give it the name gotFaces(), which will hold the data (results) from the faceMesh. We will console log the results, which will give us all the data points.

```
let video
let faceMesh

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
}

function gotFaces(results)
{
  console.log(results)
}
```

🗒️ Notes

This only works for one face. If you want multiple faces, you need to specify the number. For example, five faces would be:

Notice also that it gives an x, y, and z value; this means it is a three-dimensional model of your face.

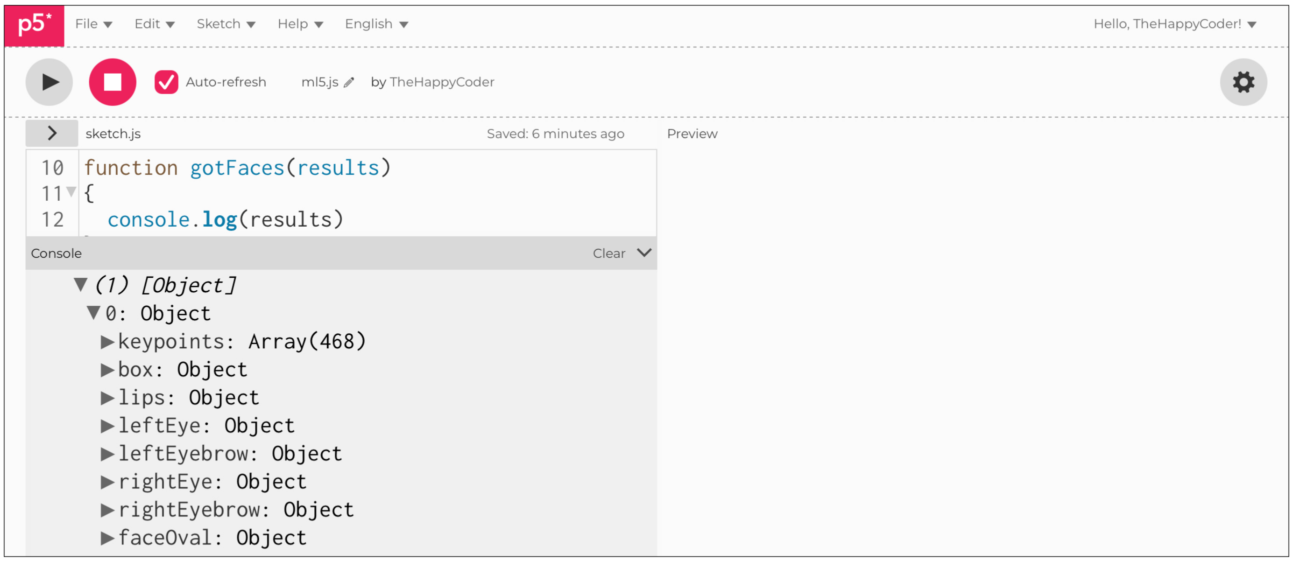
🌻 Challenge

Have a drill down into the results in the console log. You will need to stop the code running to open it up.

🛠️ Code Explanation

`faceMesh = ml5.faceMesh({maxFaces: 1})`	Option to only pick one face at a time or multiple faces
`faceMesh.detectStart(video, gotFaces)`	A function that detects whether there is a video and if so is there a face in it, then calls the callback function with the data points
`function gotFaces(results)`	The data points are sent to the callback function gotFaces()

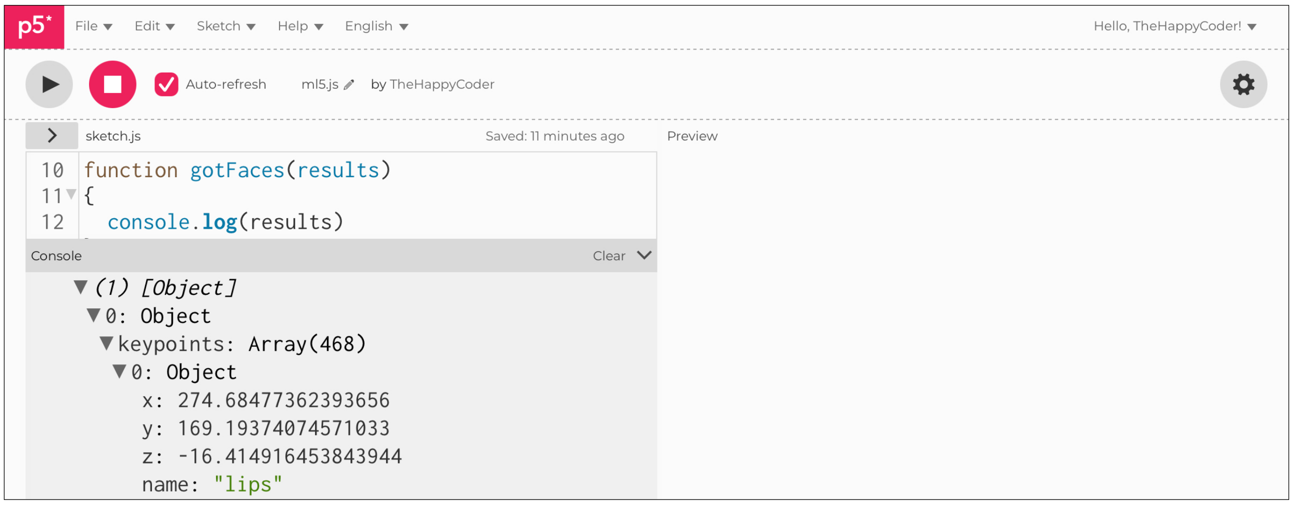
Figure B2.6a: console log of data points



```
function gotFaces(results)
{
  console.log(results)
```

```
▼(1) [Object]
  ▼0: Object
    ▶keypoints: Array(468)
    ▶box: Object
    ▶lips: Object
    ▶leftEye: Object
    ▶leftEyebrow: Object
    ▶rightEye: Object
    ▶rightEyebrow: Object
    ▶faceOval: Object
```

Figure B2.6b: drilling down a bit more



```
function gotFaces(results)
{
  console.log(results)
```

```
▼(1) [Object]
  ▼0: Object
    ▼keypoints: Array(468)
      ▼0: Object
         x: 274.68477362393656
         y: 169.19374074571033
         z: -16.414916453843944
         name: "lips"
```

Sketch B2.7 the faces array

We want to save the relevant data in an empty array called faces.
❗ Remove: console.log() and replace it by filling the faces[] empty array with the data points (results).

```
let video
let faceMesh
let faces = []

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
}

function gotFaces(results)
{
  // console.log(results)
    faces = results
}
```

🗒️ Notes

We are just collecting the data points first before we draw anything on the canvas.

🌻 Challenge

Try `console.log(faces)` to see that it now holds the data points, but remember to move that line of code to after this line of code: `faces = results`; otherwise, you might get an error.

🛠️ Code Explanation

`let faces = []`	Empty array called faces
`faces = results`	Filling the array with those data points in the results

Sketch B2.8 collecting the lips

If you open an object in the console, you will see that there is a keypoints array with 468 points. These are the total number of points on your face. We only want the lips, and to do that, we create an if() loop to collect them and create a bounding box.

```
let video
let faceMesh
let faces = []
let lips

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
  if (faces.length > 0)
  {
    lips = faces[0].lips
    rect(lips.x, lips.y, lips.width, lips.height)
  }
}
```

```
function gotFaces(results)
{
    faces = results
}
```

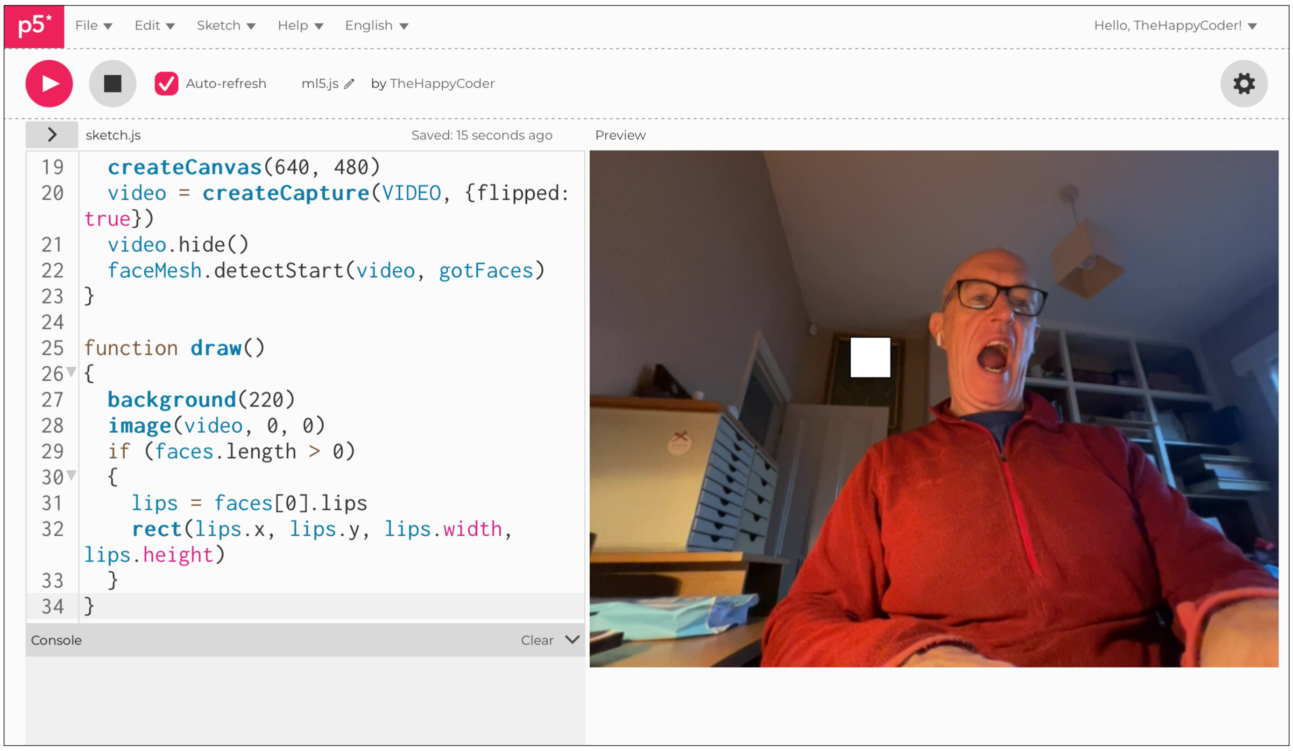
🗒️ Notes

You get a white box that responds to you opening and closing your mouth. You will notice that the box is not mirrored. We will resolve that in the next sketch.

🛠️ Code Explanation

`if (faces.length > 0)`	Checks to see if a face has been detected
`lips = faces[0].lips`	The lips are the first element in the array
`rect(lips.x, lips.y, lips.width, lips.height)`	Draws a rectangle with dimensions width and height of lips

Figure B2.8



```
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped:
true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
  if (faces.length > 0)
  {
    lips = faces[0].lips
    rect(lips.x, lips.y, lips.width,
lips.height)
  }
}
```

Sketch B2.9 face mirrored

Here we will be resolving the not mirrored issue and have a rectangle instead of a box.

```
let video
let faceMesh
let faces = []
let lips

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1, flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
  if (faces.length > 0)
  {
    lips = faces[0].lips
    noFill()
    strokeWeight(3)
    stroke(255)
    rect(lips.x, lips.y, lips.width, lips.height)
  }
}
```

```
function gotFaces(results)
{
    faces = results
}
```

🗒️ Notes

You should have a white rectangle following the shape and movement of your mouth.

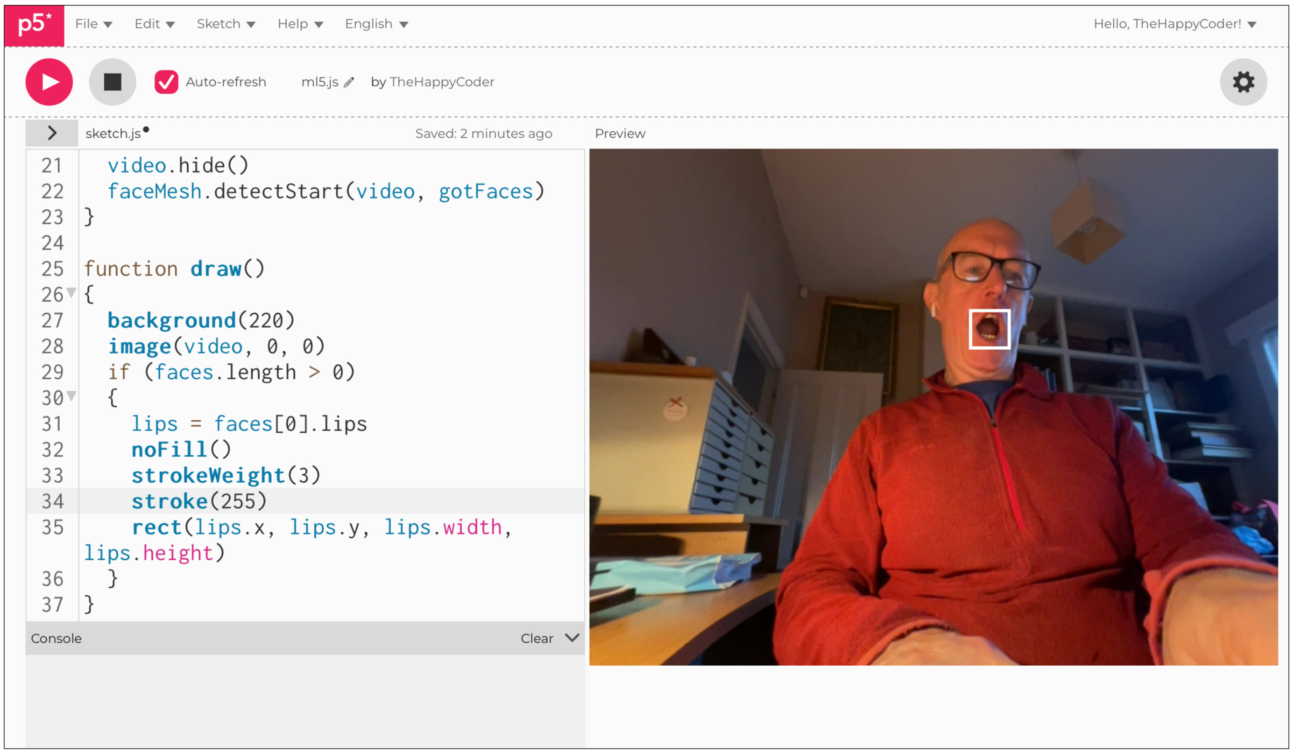
🌻 Challenge

Fancy trying some other shapes?

🛠️ Code Explanation

`faceMesh = ml5.faceMesh({maxFaces: 1, flipped: true})`	faceMesh is mirrored

Figure B2.9



```
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
  if (faces.length > 0)
  {
    lips = faces[0].lips
    noFill()
    strokeWeight(3)
    stroke(255)
    rect(lips.x, lips.y, lips.width,
lips.height)
  }
}
```

Sketch B2.10 adding eyes

Let's add the eyes; there are labels for the left and right eye called leftEye and rightEye, respectively. If we look at a console.log() of the results, we notice that it gives us a centreX and a centreY value, which are the centres of the respective eyes. If you look at figure B2.10a below, you can see their reference. We can add them to our sketch and draw circles to those data points.

```
let video
let faceMesh
let faces = []
let lips
let leftEye
let rightEye

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1, flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
  if (faces.length > 0)
  {
    lips = faces[0].lips
    leftEye = faces[0].leftEye
    rightEye = faces[0].rightEye
    noFill()
    strokeWeight(3)
    stroke(255)
    rect(lips.x, lips.y, lips.width, lips.height)
    circle(leftEye.centerX, leftEye.centerY, leftEye.width)
    circle(rightEye.centerX, rightEye.centerY, rightEye.width)
  }
}

function gotFaces(results)
{
    faces = results
}
```

🗒️ Notes

The console log comes in very useful for navigating through all this information. It is often a good idea to comment it out afterwards (or delete it) as it can slow it down.

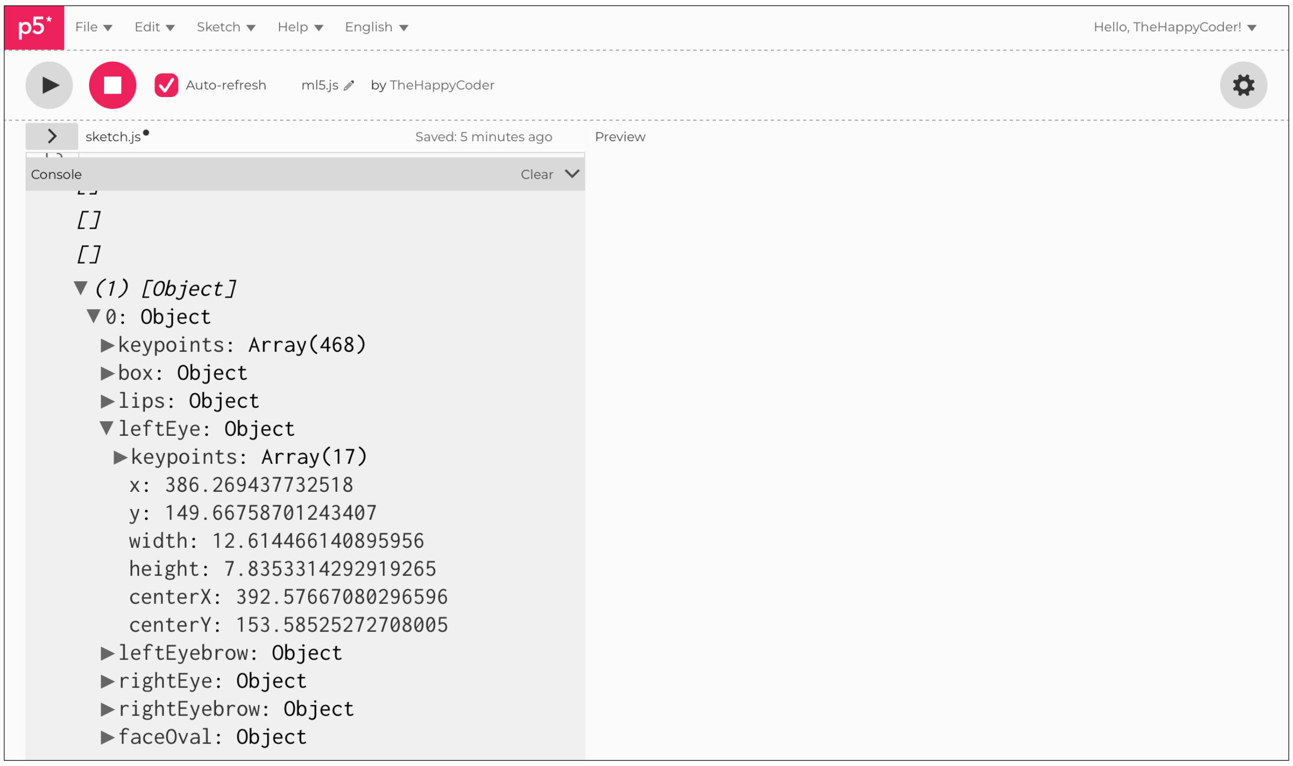
🌻 Challenges

1. Add other features to the image.
2. Can you remove the background image and have just a plain background?

🛠️ Code Explanation

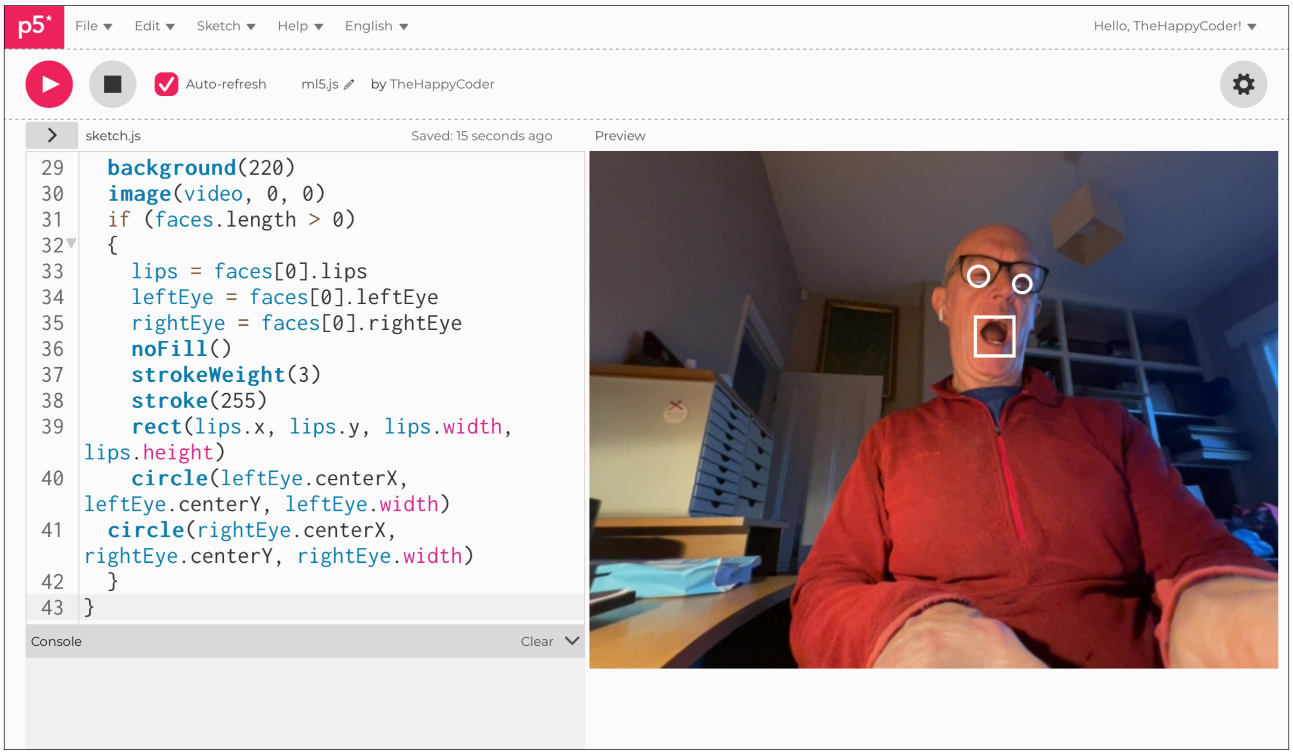
`leftEye = faces[0].leftEye`	The variable leftEye takes all the data for that eye
`circle(leftEye.centerX, leftEye.centerY, leftEye.width)`	We pull out the data for the centerX and centerY for the left eye and draw the circle.

Figure B2.10a: looking at the console.log for the eyes



```
[]
[]
▼(1) [Object]
 ▼0: Object
  ▶keypoints: Array(468)
  ▶box: Object
  ▶lips: Object
  ▼leftEye: Object
   ▶keypoints: Array(17)
    x: 386.269437732518
    y: 149.66758701243407
    width: 12.614466140895956
    height: 7.8353314292919265
    centerX: 392.57667080296596
    centerY: 153.58525272708005
  ▶leftEyebrow: Object
  ▶rightEye: Object
  ▶rightEyebrow: Object
  ▶faceOval: Object
```

Figure B2.10b



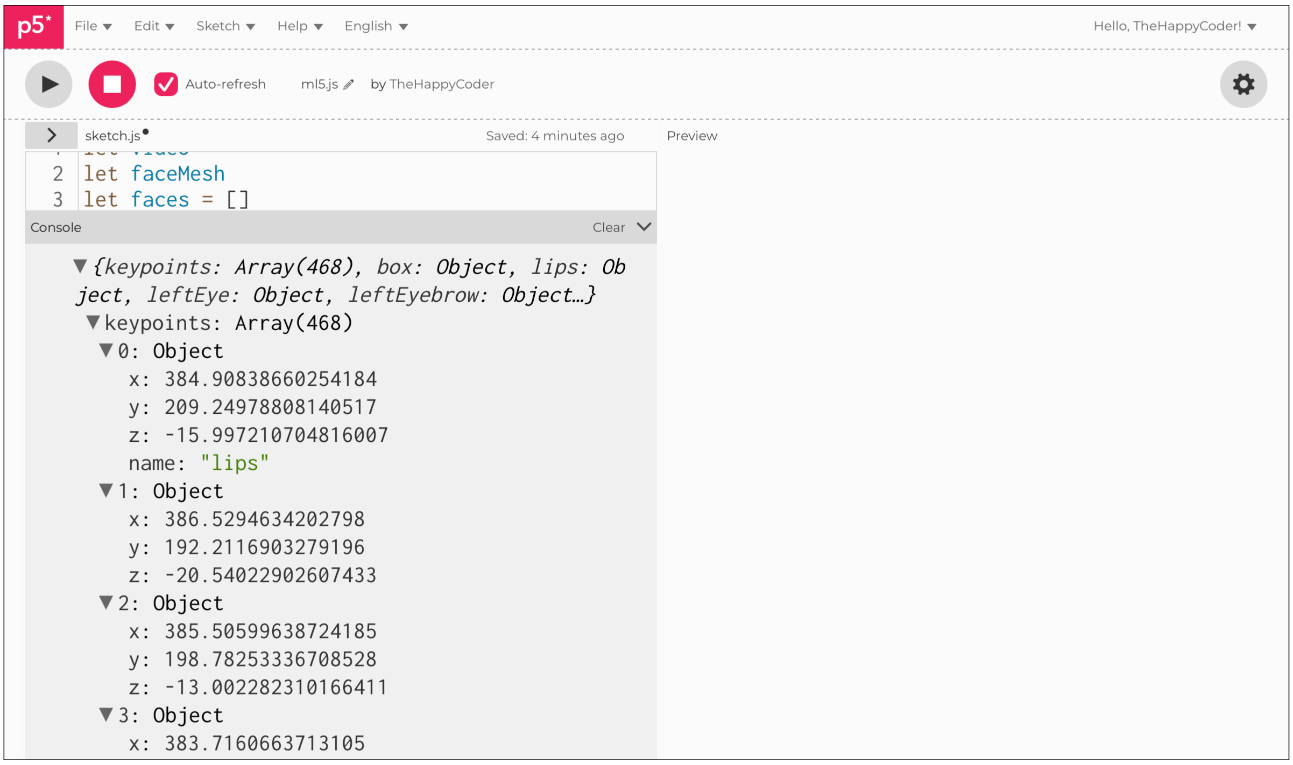
```
  background(220)
  image(video, 0, 0)
  if (faces.length > 0)
  {
    lips = faces[0].lips
    leftEye = faces[0].leftEye
    rightEye = faces[0].rightEye
    noFill()
    strokeWeight(3)
    stroke(255)
    rect(lips.x, lips.y, lips.width,
lips.height)
    circle(leftEye.centerX,
leftEye.centerY, leftEye.width)
  circle(rightEye.centerX,
rightEye.centerY, rightEye.width)
  }
}
```

Drawing the data points

In this final part of the unit, we will draw all the data points for the face mesh. We will use the above sketch but first remove much of the code in the `draw()` function. We will be drawing `points()`, not circles; in case you have forgotten, points are the size of pixels, so we will enlarge them slightly.

When you do a `console.log()` of the results, you can see there are `468 keypoints`; these are the values we are after. See `figure 4` below:

Figure 4: console.log() of the keypoints



```
let faceMesh
let faces = []
```

```
▼{keypoints: Array(468), box: Object, lips: Ob
ject, leftEye: Object, leftEyebrow: Object…}
  ▼keypoints: Array(468)
    ▼0: Object
        x: 384.90838660254184
        y: 209.24978808140517
        z: -15.997210704816007
        name: "lips"
    ▼1: Object
        x: 386.5294634202798
        y: 192.2116903279196
        z: -20.54022902607433
    ▼2: Object
        x: 385.50599638724185
        y: 198.78253336708528
        z: -13.002282310166411
    ▼3: Object
        x: 383.7160663713105
```

Sketch B2.11 a slimmed down version

Delete the lines of code not used in the draw() function below.

```
let video
let faceMesh
let faces = []
let lips
let leftEye
let rightEye

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1, flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
}

function gotFaces(results)
{
    faces = results
}
```

🗒️ Notes

We have an almost empty `draw()` function.

Sketch B2.12 a new array of objects

We are going to loop through all the data points in the faces array. This will give us an array of objects; each object carries with it a bunch of data.

```
let video
let faceMesh
let faces = []
let lips
let leftEye
let rightEye

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1, flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
  for (let i = 0; i < faces.length; i++)
  {
    let face = faces[i]
  }
}
```

```
function gotFaces(results)
{
    faces = results
}
```

🗒️ Notes

We have created another array of objects called face.

🛠️ Code Explanation

`for (let i = 0; i < faces.length; i++)`	Loop through the faces array of objects
`let face = faces[i]`	Move the objects into a new array called face

Sketch B2.13 keypoint array

Next, we want another loop to go through the face array and pull out all the keypoints one by one and draw them as points.

```
let video
let faceMesh
let faces = []
let lips
let leftEye
let rightEye

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1, flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(220)
  image(video, 0, 0)
  for (let i = 0; i < faces.length; i++)
  {
    let face = faces[i]
    for (let j = 0; j < face.keypoints.length; j++)
    {
      let keypoint = face.keypoints[j]
      strokeWeight(2)
      stroke(255, 255, 0)
      point(keypoint.x, keypoint.y)
    }
  }
}

function gotFaces(results)
{
    faces = results
}
```

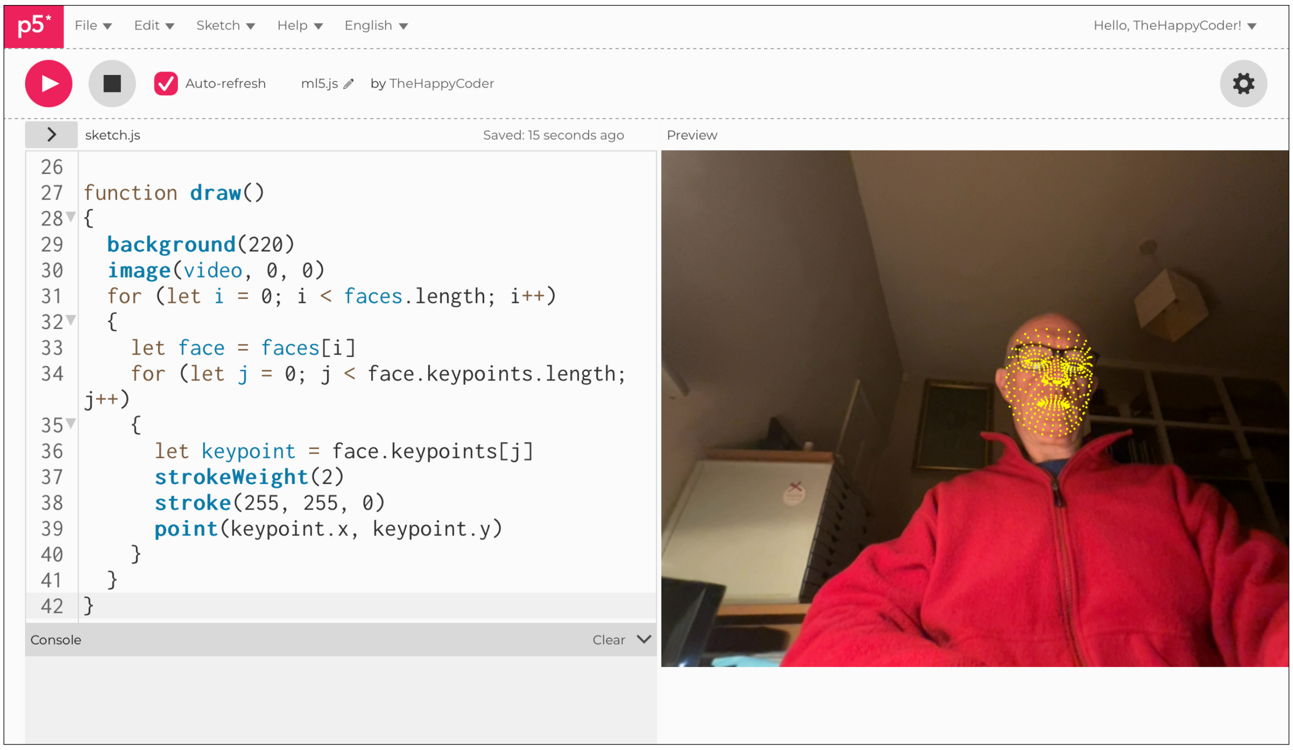
🗒️ Notes

We cycle through all the keypoints and draw a point at each of their x and y co-ordinates.

🛠️ Code Explanation

`for (let j = 0; j < face.keypoints.length; j++)`	Cycle through just the keypoints part of the array
`let keypoint = face.keypoints[j]`	Give those values to a new array called keypoint
`point(keypoint.x, keypoint.y)`	Draw the x and y components of each keypoint as a point (pixel)

Figure B2.13



```

function draw()
{
  background(220)
  image(video, 0, 0)
  for (let i = 0; i < faces.length; i++)
  {
    let face = faces[i]
    for (let j = 0; j < face.keypoints.length;
j++)
    {
      let keypoint = face.keypoints[j]
      strokeWeight(2)
      stroke(255, 255, 0)
      point(keypoint.x, keypoint.y)
    }
  }
}
```

Sketch B2.14 finally remove the background image

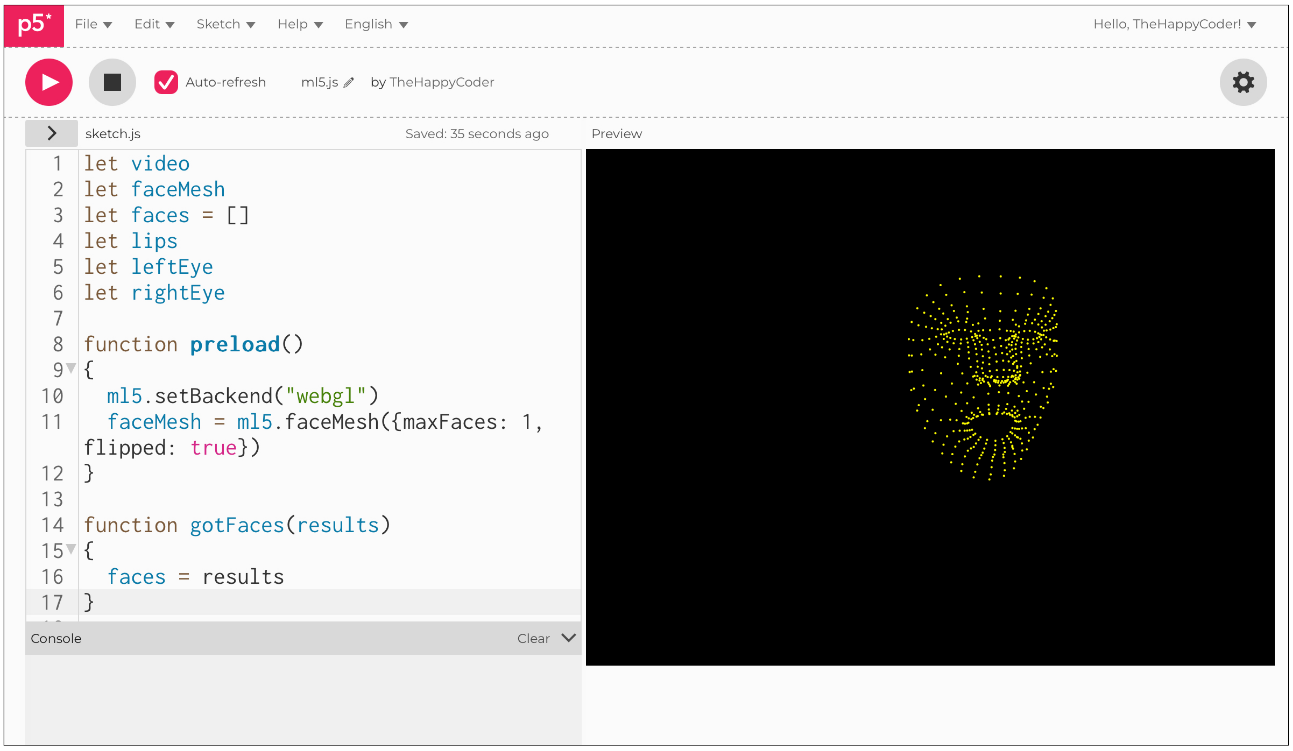
We want a nice, stark contrast.

```
let video
let faceMesh
let faces = []
let lips
let leftEye
let rightEye

async function setup()
{
  ml5.setBackend("webgl")
  faceMesh = await ml5.faceMesh({maxFaces: 1, flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  faceMesh.detectStart(video, gotFaces)
}

function draw()
{
  background(0)
  // image(video, 0, 0)
  for (let i = 0; i < faces.length; i++)
  {
    let face = faces[i]
    for (let j = 0; j < face.keypoints.length; j++)
    {
      let keypoint = face.keypoints[j]
      strokeWeight(2)
      stroke(255, 255, 0)
      point(keypoint.x, keypoint.y)
    }
  }
}

function gotFaces(results)
{
    faces = results
}
```

🗒️ Notes
Our final effect is quite something.

🌻 Challenge
What else could you develop or create? 3D, perhaps.

Figure B2.14



```
let video
let faceMesh
let faces = []
let lips
let leftEye
let rightEye

function preload()
{
  ml5.setBackend("webgl")
  faceMesh = ml5.faceMesh({maxFaces: 1,
flipped: true})
}

function gotFaces(results)
{
  faces = results
}
```