Código JS da última aula

```
img = "";
objects = [];
status = "";

function preload(){
  img = loadImage('dog_cat.jpg');
}


function setup() {
  canvas = createCanvas(640, 420);
  canvas.center();
  objectDetector = ml5.objectDetector('cocossd', modelLoaded);
  document.getElementById("status").innerHTML = "Status: Detectando Objetos";
}

function modelLoaded() {
  console.log("Modelo Carregado!")
  status = true;
  objectDetector.detect(img, gotResult);
}

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

```
function draw() {
  image(img, 0, 0, 640, 420);

      if(status != "")
      {
        for (i = 0; i < objects.length; i++) {
          document.getElementById("status").innerHTML = "Status: Objeto Detectado";

          fill("#FF0000");
          percent = floor(objects[i].confidence * 100);
          text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
          noFill();
          stroke("#FF0000");
          rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
        }
      }
}
```

1. **Defina uma matriz vazia no início do arquivo main.js.**

```
img = "";
status = "";
objects = [];

function preload(){
  img = loadImage('dog_cat.jpg');
}
```

2. **Atribua a matriz "results" a matriz "objects".**

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

3. **Adiciona uma condição "if" para verificar se o status da variável não é vazio.**

```
function draw() {
  image(img, 0, 0, 640, 420);

    if(status != "")
    {
    }
```

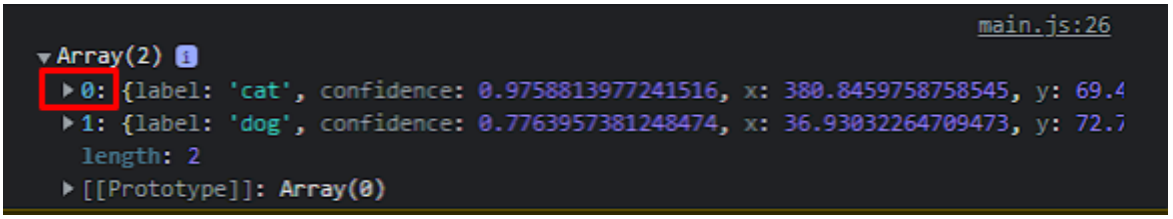
Logo, na condição IF verificamos "if" [`if(`] variável status [`status`] não é [`!=`] igual [`=`] a vazio [`"")`], então ela deve entrar nesta "condição" e iniciar o desenho dos retângulos.

PARTE 1

Como buscar a label da primeira matriz da matriz "objects"

1. **Queremos ler os valores de objects, portanto, primeiro, escreveremos** `objects`**.**

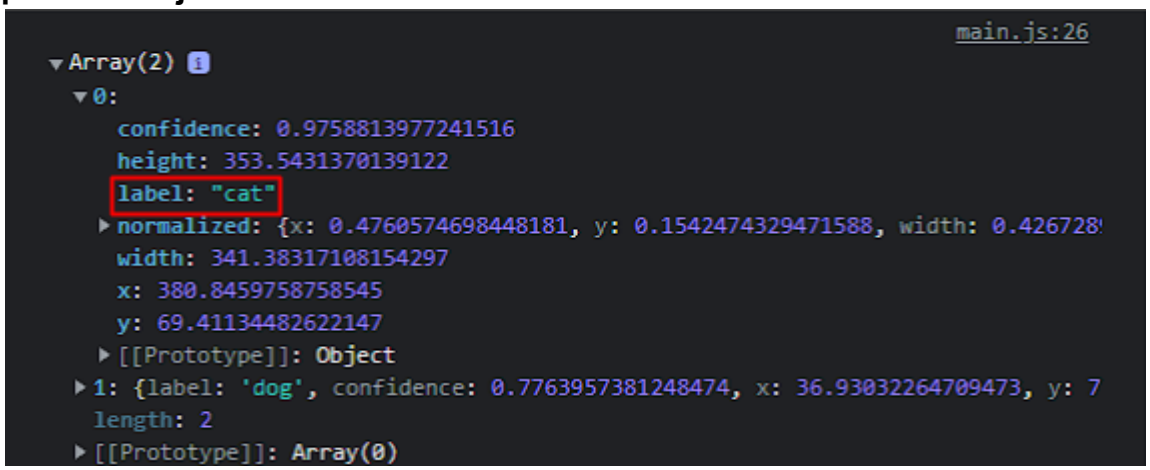
2. **Dentro desta matriz objects queremos obter a primeira matriz, que está no index (índice) 0. Portanto, expanda `0:` clicando no ícone ao lado de `0:` .**



Este modelo é atualizado constantemente, portanto, talvez ele não detecte algum dos objetos mencionados acima, podemos informar 1 OU 2 matrizes. Continue o código para a quantidade de matrizes retornada.

Clicamos no index 0 dentro da matriz objects, logo o código será `objects[0]` **.**

1. **Dentro dessa primeira matriz temos uma label, esta é relacionada ao primeiro objeto detectado.**



Portanto, precisamos clicar em objects- > em seguida index 0 -> depois encontramos a label.

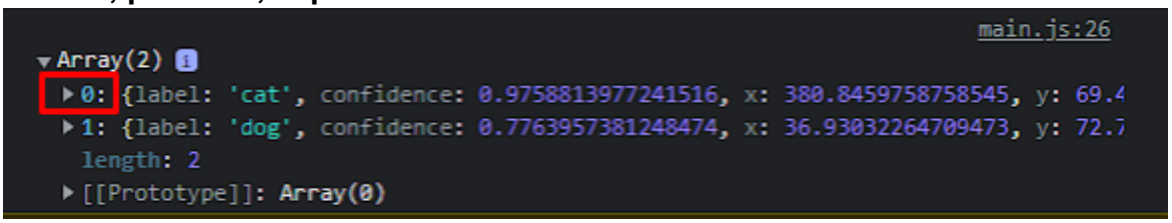
O código será: `objects[0].label` **.**

Como buscar a width (largura) da primeira matriz da matriz objects

1. **Queremos ler os valores de objects, portanto, primeiro, escreveremos** `objects` **.**

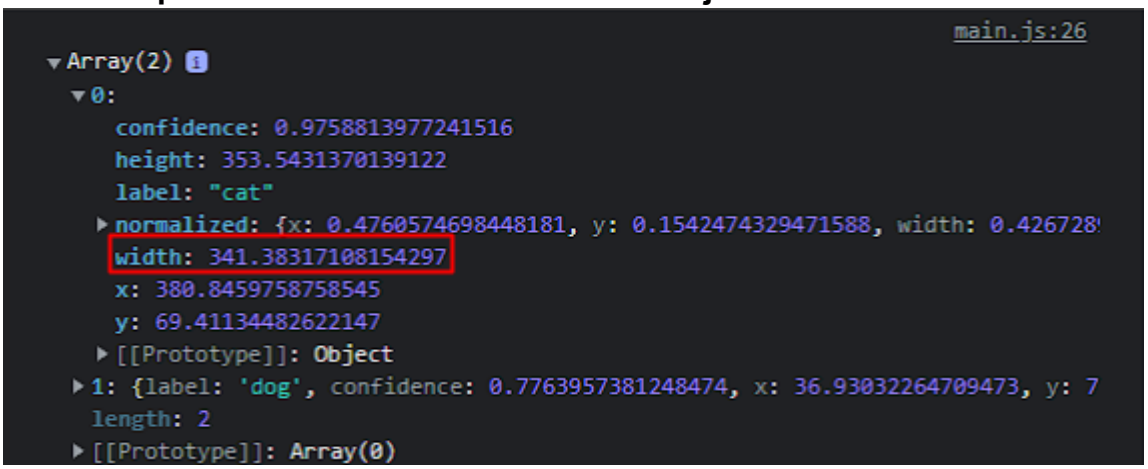
2. **Dentro da matriz objects queremos obter a primeira matriz, que está no**

index 0, portanto, expandimos `0:` clicando no ícone ao lado de `0:`



Clicamos no index 0 que está dentro da matriz objects, logo o código será `objects[0]`.

3. Dentro da primeira matriz temos a width do objeto.



Precisamos clicar em objects > em seguida no index 0 -> depois encontramos a width.

Logo, o código será `objects[0].width`.

Código para o loop for

```
function draw() {
  image(img, 0, 0, 640, 420);

    if(status != "")
    {
      for (i = 0; i < objects.length; i++) {
        document.getElementById("status").innerHTML = "Status: Objeto Detectado";
        fill("#FF0000");
        percent = floor(objects[i].confidence * 100);
        text(objects[i].label + " " + percent + "%", objects[i].x, objects[i].y);
        noFill();
        stroke("#FF0000");
        rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
      }
    }
}
```

Decomposição do código acima:

1. **Defina o loop for:** `for (i = 0; i < objects.length; i++)`
   - **Defina o ponto de início do loop for:** `i = 0`
   - **Defina o ponto de parada do loop for:** `i < objects.length`
   - **Defina o intervalo entre cada loop:** `i++`
2. **Atualize a tag h3 para "Status: Objeto Detectado"**

```
for (i = 0; i < objects.length; i++) {
  document.getElementById("status").innerHTML = "Status: Objeto Detectado";
}
```

3. **Defina a cor do texto/label com a função fill()**

```
for (i = 0; i < objects.length; i++) {
  document.getElementById("status").innerHTML = "Status: Objeto Detectado";
  fill("#FF0000");
}
```

4. **Busque a confidence (precisão) da matriz "objects" no loop for (para) e converta esse valor para porcentagem; remova os decimais e a armazene em uma variável**

```
for (i = 0; i < objects.length; i++) {
  document.getElementById("status").innerHTML = "Status: Objeto Detectado";
  fill("#FF0000");
  percent = floor(objects[i].confidence * 100);
}
```

- **Crie o código para buscar a confidence na matriz "objects" no loop for**

```
objects[0].confidence
```

Explicação do trecho acima:
Considere que exista duas matrizes dentro da matriz objects. Ou seja, a length (comprimento) dessa matriz e "2".

- **Quando o loop iniciar: i = 0**
  - **objects[i].confidence se tornará objects[0].confidence //isso significa que obtivemos a confidence do primeiro objeto.**
- **Em seguida, i será acrescido: i = 1**
  - **objects[i].confidence se tornará objects[1].confidence //isso significa que obtivemos a confidence do segundo objeto.**
- **Mais uma vez, i será acrescido: i = 2**
  - **A length da matriz é "2", isso significa que o loop foi encerrado.**

- **Converta a confidence para porcentagem**

```
object[0].confidence * 100
```

- **Remova todos os decimais** `floor(objects[i].confidence * 100);`
- **Armazene a confidence em uma variável**

```
percent = floor(objects[i].confidence * 100);
```

5. **Busque a label da matriz "objects" e a exiba com a confidence usando a função text()**

```
for (i = 0; i < objects.length; i++) {
  document.getElementById("status").innerHTML = "Status: Objeto Detectado";
  fill("#FF0000");
  percent = floor(objects[i].confidence * 100);
  text(objects[i].label + " " + percent + "%", objects[i].x, objects[i].y);
}
```

- **Escreva o código para buscar a label da matriz "objects" no loop for**

```
objects[i].label
```

Explicação do trecho acima:
Considere que exista duas matrizes dentro da matriz objects. Isso significa que a length dessa matriz é "2".

- **Quando o loop inicia: i = 0**
  - **objects[i].label se tornará objects[0].label //isso significa que obtivemos a label do primeiro objeto.**

- Em seguida, i é acrescido: i = 1
    - objects[i].label se tornará objects[1].label //isso significa que obtivemos a label do segundo objeto.
- Então, i é acrescido: i =2
    - A length da matriz é “2”, ou seja, o loop é encerrado.

Precisamos que o resultado esteja neste formato: 

label > espaço > confidence > símbolo de porcentagem

- Para a label do objeto, o código será: `text(objects[i].label`
- Para adicionar espaço, o código será:
  `text(objects[i].label + " "`
- Para adicionar a porcentagem, o código será:
  `text(objects[i].label + " " + percent`
- Para adicionar o símbolo de porcentagem, o código será:

```
text(objects[i].label + " " + percent + "%",
```

Buscando as coordenadas x para passá-las dentro da função text()

- Escreva o código para buscar a coordenada x da matriz “objects” no loop for `objects[i].x`
-

Explicação do trecho acima
Considere que exista duas matrizes dentro da matriz objects. Ou seja, a length dessa matriz é “2”.

- Quando o loop inicia: i = 0
    - objects[i].x se tornará objects[0].x //isso significa que obtivemos a coordenada x do primeiro objeto.
- Em seguida, i é acrescido: i = 1
    - objects[i].x se tornará objects[1].x //isso significa que obtivemos a coordenada x do segundo objeto.

- Então, i é acrescido: i =2
    - A length dessa matriz é “2”, ou seja, o loop é encerrado.

Agora, passamos essas coordenadas x dentro da função text()

```
text(objects[i].label + " " + percent + "%", objects[i].x
```

Buscando as coordenadas y para passá-las dentro da função text()

- **Escreva o código para buscar a coordenada y da matriz “objects” no loop for** `objects[i].y`

Explicação do trecho acima
Considere que exista duas matrizes dentro da matriz objects. Ou seja, a length dessa matriz é “2”.

- **Quando o loop inicia: i = 0**
  - **objects[i].y se tornará objects[0].y //isso significa que obtivemos a coordenada x do primeiro objeto.**
- **Em seguida, i é acrescido: i = 1**
  - **objects[i].y se tornará objects[1].y //isso significa que obtivemos a coordenada x do segundo objeto.**

- **Então, i é acrescido: i =2**
  - **A length dessa matriz é “2”, ou seja, o loop é encerrado.**

Agora, passamos essas coordenadas y dentro da função text()

```
text(objects[i].label + " " + percent + "%", objects[i].x, objects[i].y);
```

6. **Cancele a cor definida pela função fill() com a função p5.js noFill()**

Se não fizermos isso, o resultado será este:

Código para cancelar a cor definida pela função fill()

```
for (i = 0; i < objects.length; i++) {
  document.getElementById("status").innerHTML = "Status: Objeto Detectado";
  fill("#FF0000");
  percent = floor(objects[i].confidence * 100);
  text(objects[i].label + " " + percent + "%", objects[i].x, objects[i].y);
  noFill();
}
```

7. **Defina a cor da borda do retângulo com a função stroke()**

```
for (i = 0; i < objects.length; i++) {
  document.getElementById("status").innerHTML = "Status: Objeto Detectado";
  fill("#FF0000");
  percent = floor(objects[i].confidence * 100);
  text(objects[i].label + " " + percent + "%", objects[i].x, objects[i].y);
  noFill();
  stroke("#FF0000");
}
```

8. **Desenhe um retângulo para todos os objetos com a função rect()**

```
for (i = 0; i < objects.length; i++) {
  document.getElementById("status").innerHTML = "Status: Objeto Detectado";
  fill("#FF0000");
  percent = floor(objects[i].confidence * 100);
  text(objects[i].label + " " + percent + "%", objects[i].x, objects[i].y);
  noFill();
  stroke("#FF0000");
  rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
}
```

O código para buscar as coordenadas x da matriz "objects" no loop for será:

```
objects[i].x
```

O código para buscar as coordenadas y da matriz "objects" no loop for será:

```
objects[i].y
```

O código para buscar a width da matriz "objects" no loop for será:

```
objects[i].width
```

Considere que exista duas matrizes dentro da matriz objects. Ou seja, a length dessa matriz é "2".

- **Quando o loop inicia: i = 0**
  - **objects[i].width se tornará objects[0].width //isso significa que obtivemos a width do primeiro objeto.**
- **Em seguida, i é acrescido: i = 1**
  - **objects[i].width se tornará objects[1].width //isso significa que obtivemos a width do segundo objeto.**
- **Então, i é acrescido: i =2**
  - **A length da matriz é "2", ou seja, o loop é encerrado.**

O código para buscar a height (altura) da matriz "objects" no loop for será:

```
objects[i].height
```

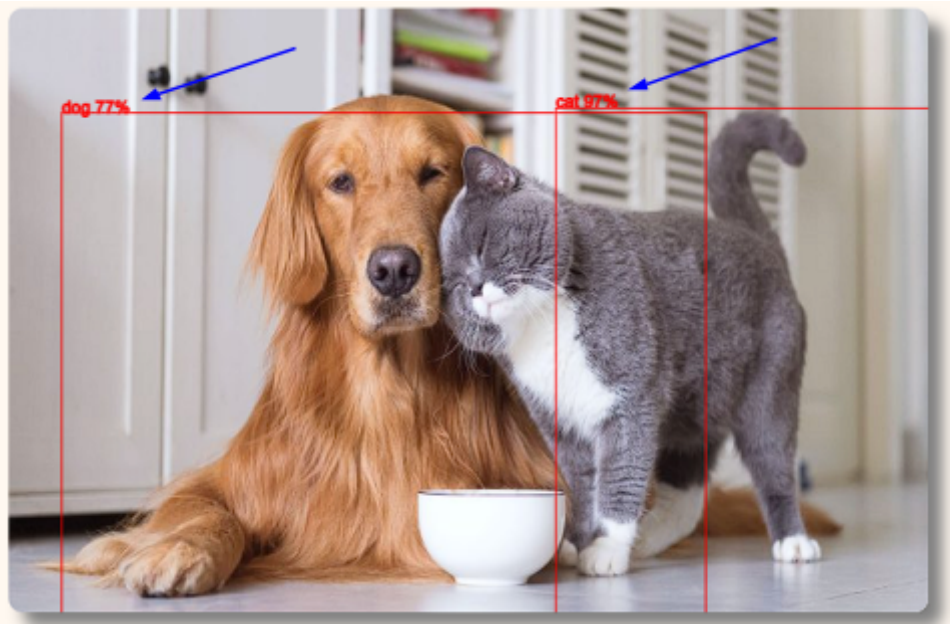
Considere que exista duas matrizes dentro da matriz objects. Ou seja, a length dessa matriz é "2".

- **Quando o loop inicia: i = 0**
  - **objects[i].height se tornará objects[0].height //isso significa que obtivemos a height do primeiro objeto.**
- **Em seguida, i é acrescido: i = 1**
  - **objects[i].height se tornará objects[1].height //isso significa que obtivemos a height do objet.**
- **Então, i é acrescido: i =2**
  - **A length da matriz é "2", ou seja, o loop foi encerrado.**

Logo, o código completo de rect() será:

```
rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
```

Código para modificar as coordenadas x e y de text() para evitar o resultado abaixo:



Este modelo é atualizado constantemente, logo, talvez ele não detecte algum dos objetos mencionados acima, como, por exemplo, ele pode não detectar a "bowl" (tigela), ou qualquer outro objeto da imagem, não se preocupe. Você pode continuar com o código.

Código:

```
function draw() {
  image(img, 0, 0, 640, 420);

    if(status != "")
    {
      for (i = 0; i < objects.length; i++) {
        document.getElementById("status").innerHTML = "Status: Objeto Detectado";
        fill("#FF0000");
        percent = floor(objects[i].confidence * 100);
        text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
        noFill();
        stroke("#FF0000");
        rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
      }
    }
}
```

Modifique as coordenadas x e y de text() adicionando 15 pixels.

```
text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
```

PARTE 2

CÓDIGO HTML e CSS

1. **Adicionar uma tag h3 para conter a quantidade de objetos detectados.**

```
<body background="background.jpg">
<center>
  <h1 class="btn btn-info heading">Detecção de Objetos</h1>

  <h3 id="status" class="btn btn-danger"></h3>
  <h3 id="numberOfObjects" class="btn btn-warning"></h3>
```

Adicione duas tags br após a tag h3

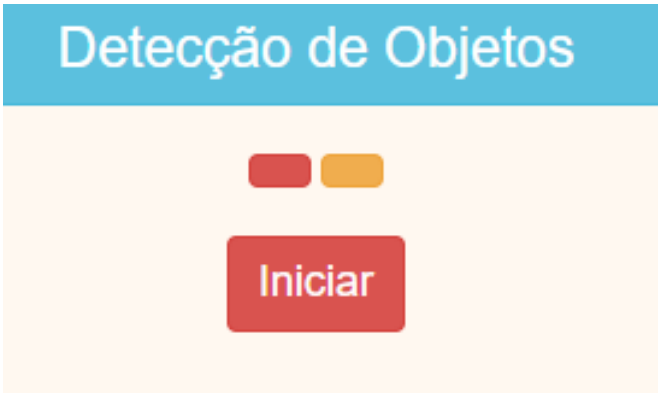
```
<body background="background.jpg">
<center>
  <h1 class="btn btn-info heading">Detecção de Objetos</h1>

  <h3 id="status" class="btn btn-danger"></h3>
  <h3 id="numberOfObjects" class="btn btn-warning"></h3>
  <br><br>
  <button onclick="start()" class="btn btn-danger" id="startModel">Iniciar</button>
</center>
```

Adicionar estilos em style.css

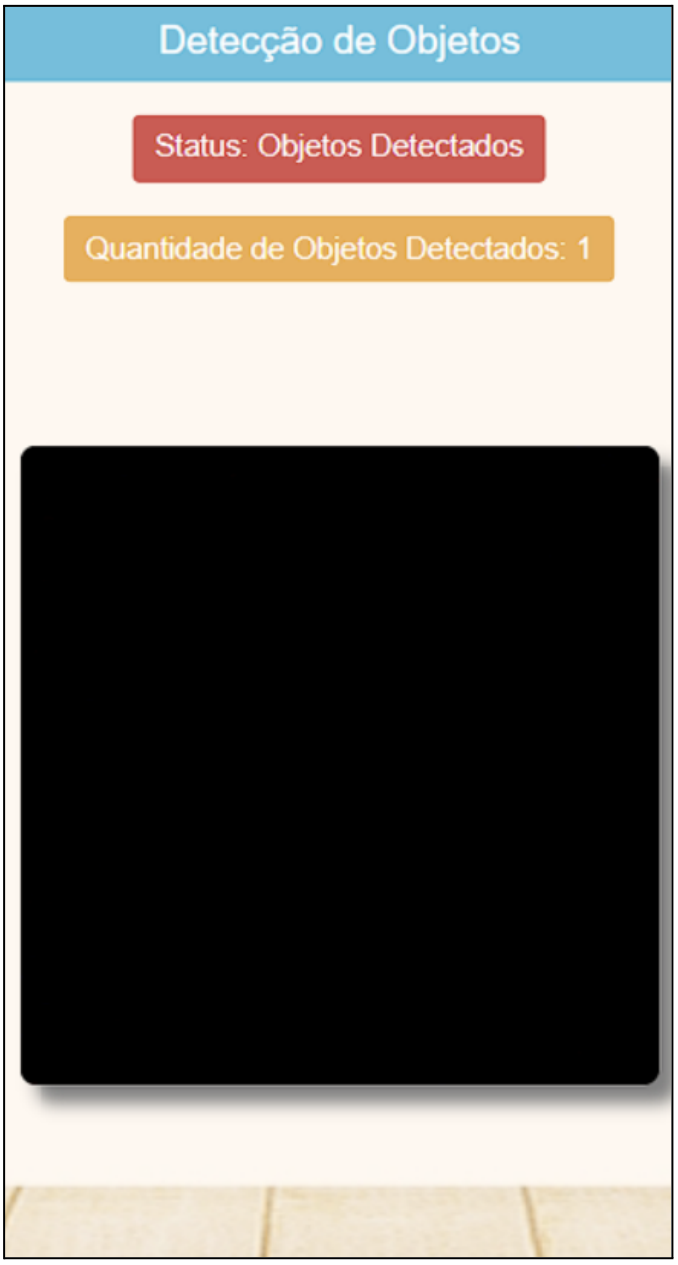
```
#status , #numberOfObjects , #startModel
{
font-size: 18px;
}
```

Saída



2. **Adicionar margin-top (margem superior) ao elemento canvas**

Em dispositivos móveis, o app web deve ter esta aparência:



Após adicionar margin-top ao canvas

```
canvas
{
box-shadow: 10px 10px 10px grey;
border-radius: 10px;
margin-top: 30px;
}
```

CÓDIGO JS

1. **Reduzimos o tamanho do elemento canvas.**

De:

```
function setup() {
  canvas = createCanvas(640, 420);
  canvas.center();
  objectDetector = ml5.objectDetector('cocossd', modelLoaded);
  document.getElementById("status").innerHTML = "Status: Detectando Objetos";
}
```

Para:

```
function setup() {
  canvas = createCanvas(380, 380);
  canvas.center();
  objectDetector = ml5.objectDetector('cocossd', modelLoaded);
  document.getElementById("status").innerHTML = "Status: Detectando Objetos";
}
```

2. **Código para acessar a webcam:**

```
function setup() {
  canvas = createCanvas(380, 380);
  canvas.center();
  video = createCapture(VIDEO);
  video.hide();
}
```

3. **Atualizar o código JS para posicionar a visualização da webcam na tela:**

De:

```
function draw() {
  image(img, 0, 0, 640, 420);

    if(status != "")
    {
      for (i = 0; i < objects.length; i++) {
        document.getElementById("status").innerHTML = "Status: Objeto Detectado";

        fill("#FF0000");
        percent = floor(objects[i].confidence * 100);
        text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
        noFill();
        stroke("#FF0000");
        rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
      }
    }
}
```

Para:

```
function draw() {
  image(video, 0, 0, 640, 420);

      if(status != "")
      {
        for (i = 0; i < objects.length; i++) {
          document.getElementById("status").innerHTML = "Status: Objeto Detectado";

          fill("#FF0000");
          percent = floor(objects[i].confidence * 100);
          text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
          noFill();
          stroke("#FF0000");
          rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
        }
      }
}
```

4. Atualizar o tamanho da visualização da webcam na tela:

De:

```
function draw() {
  image(video, 0, 0, 640, 420);

      if(status != "")
      {
        for (i = 0; i < objects.length; i++) {
          document.getElementById("status").innerHTML = "Status: Objeto Detectado";

          fill("#FF0000");
          percent = floor(objects[i].confidence * 100);
          text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
          noFill();
          stroke("#FF0000");
          rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
        }
      }
}
```

Para:

```
function draw() {
  image(video, 0, 0, 380, 380);

    if(status != "")
    {
      for (i = 0; i < objects.length; i++) {
        document.getElementById("status").innerHTML = "Status: Objeto Detectado";

        fill("#FF0000");
        percent = floor(objects[i].confidence * 100);
        text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
        noFill();
        stroke("#FF0000");
        rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
      }
    }
}
```

5. Modificar a entrada fornecida na função detect():

De:

```
function modelLoaded() {
  console.log("Modelo Carregado!")
  status = true;
  objectDetector.detect(img, gotResult);
}
```

Para:

```
function modelLoaded() {
  console.log("Modelo Carregado!")
  status = true;
  objectDetector.detect(video, gotResult);
}
```

6. Mover o código de execução do modelo CocoSSD para dentro da função draw():

```
function draw() {
  image(video, 0, 0, 380, 380);

    if(status != "")
    {
      objectDetector.detect(video, gotResult);
      for (i = 0; i < objects.length; i++) {
        document.getElementById("status").innerHTML = "Status: Objeto Detectado";

        fill("#FF0000");
        percent = floor(objects[i].confidence * 100);
        text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
        noFill();
        stroke("#FF0000");
        rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
      }
    }
}
```

7. Adicionar o código para gerar números aleatórios para “RGB” e armazená-lo em variáveis:

```
function draw() {
  image(video, 0, 0, 380, 380);

      if(status != "")
      {
        r = random(255);
        g = random(255);
        b = random(255);
        objectDetector.detect(video, gotResult);
        for (i = 0; i < objects.length; i++) {
          document.getElementById("status").innerHTML = "Status: Objeto Detectado";

          fill("#FF0000");
          percent = floor(objects[i].confidence * 100);
          text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
          noFill();
          stroke("#FF0000");
          rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
        }
      }
}
```

8. Atualizar as funções fill() e stroke():

```
function draw() {
  image(video, 0, 0, 380, 380);
      if(status != "")
      {
        r =  random(255);
        g =  random(255);
        b =  random(255);
        objectDetector.detect(video, gotResult);
        for (i = 0; i < objects.length; i++) {
          document.getElementById("status").innerHTML = "Status: Objetos Detectados";


          fill(r,g,b);
          percent = floor(objects[i].confidence * 100);
          text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
          noFill();
          stroke(r,g,b);
          rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
        }
      }
}
```

9. Atualizar a tag h3 que utilizamos para conter a quantidade de objetos detectados:

```
function draw() {
  image(video, 0, 0, 380, 380);
      if(status != "")
      {
        r =  random(255);
        g =  random(255);
        b =  random(255);
        objectDetector.detect(video, gotResult);
        for (i = 0; i < objects.length; i++) {
          document.getElementById("status").innerHTML = "Status: Objetos Detectados";
          document.getElementById("numberOfObjects").innerHTML = "Quantidade de Objetos Detectados: "+ objects.length;

          fill(r,g,b);
          percent = floor(objects[i].confidence * 100);
          text(objects[i].label + " " + percent + "%", objects[i].x + 15, objects[i].y + 15);
          noFill();
          stroke(r,g,b);
          rect(objects[i].x, objects[i].y, objects[i].width, objects[i].height);
        }
      }
}
```

10. Adicionar a função size() para o desenho dos retângulos e o posicionamento das labels (etiquetas) serem mais precisos:

```
function setup() {
  canvas = createCanvas(380, 380);
  canvas.center();
  video = createCapture(VIDEO);
  video.size(380,380);
  video.hide();
}
```