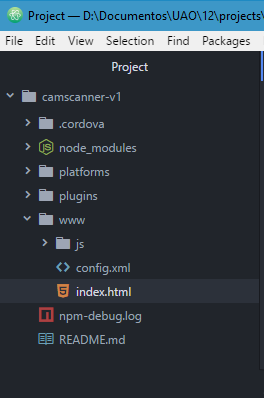
**DESCRIPCIÓN DEL CÓDIGO DE LA APLICACIÓN**

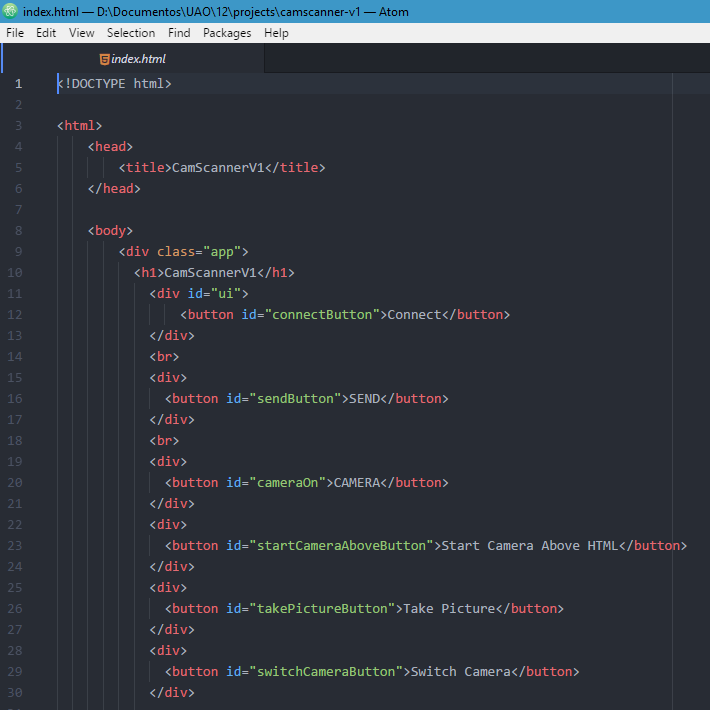
**CAM-LN (Versión 1.0)**

1. Instalar Apache Cordova
   1. Se debe instalar previamente Node.js
   2. Abrir la consola de node.js e instalar npm
   3. Desde la misma consola de node instalar cordova, usando el comando $ npm install -g cordova
   4. Ahora se debe crear un proyecto nuevo, usando el comando $ cordova create “NombreDeLaAplicacion”
   5. Entramos a la carpeta del proyecto y añadimos la plataforma para la que queremos que corra la aplicación, $ cordova platform add android
2. Cordova creará y configurará el proyecto por sí solo, dándonos la posibilidad de entrar directamente a crear nuestro proyecto, sin altercados de configuración o compatibilidad.
3. Para que la aplicación funcione de manera correcta se deben instalar diferentes plugins
   1. Usamos el comando $ cordova plugin add “NombreDelPlugin”
   2. A continuación instalaremos los plugins correspondientes:
      1. cordova plugin add http://github.com/phoenixtwister/cordova-juleskelly-tesseract.git API encargada del reconocimiento de caracteres.
      2. $ cordova plugin add cordova-plugin-bluetooth-serial API encargada de la comunicación serial mediante el bluetooth, entre el celular y el módulo bt del arduino.
      3. $ cordova plugin add cordova-plugin-camera API para el uso de la cámara del celular.
      4. $ cordova plugin add https://github.com/cordova-plugin-camera-preview/cordova-plugin-camera-preview.git API para la camara enbebida en la aplicacion.
      5. $ cordova plugin add cordova-plugin-canvas2image API para convertir imágenes del canvas en imágenes .jpg o .png para ser almacenadas en el celular.

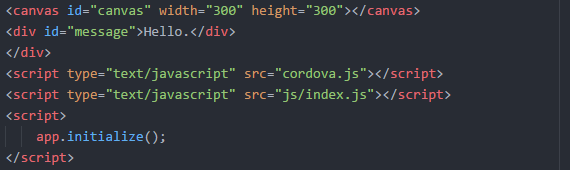
1. Explicacion del codigo HTML
   1. Dentro del proyecto se creará una carpeta llamada /www la cual contendrá el código raiz de nuestra aplicación, accederemos a esta carpeta.



* 1. Dentro de /www abriremos el index.html el cual contiene todo el front de nuestra aplicación (botones, texto, canvas, etc).
  2. Creación del front:
     1. Dentro de las etiquetas <title> se coloca el nombre de la aplicación.
     2. Dentro del body que es el cuerpo del html se hace un contenedor div que tendrá todos los elementos en cuestión.
     3. Entre <h1> colocaremos el nombre como título para saber qué aplicación estás usando y la versión.
     4. Crearemos diferentes contenedores div para agrupar nuestros botones.
     5. Crearemos primeramente el boton de conexion para que se aparee con el arduino, usando la etiqueta <button>
     6. Usaremos la etiqueta <br> para dar una separación entre cada botón.
     7. Un nuevo contenedor div y dentro el botón de prueba que enviará una cadena al arduino para testear la conexión.
     8. Un nuevo contenedor div y dentro el botón de cámara, el cual activará la cámara predeterminada del celular.
     9. Un nuevo contenedor div y dentro el botón de start camera, el cual activará la cámara embebida en el html para ser usada de manera directa.
     10. Un nuevo contenedor div y dentro el botón de captura, el cual tomara la foto de la cámara embebida en el html.
     11. Un nuevo contenedor div y dentro el botón de switch camera, el cual cambiará entre la cámara frontal y la principal de la cámara embebida en el html.



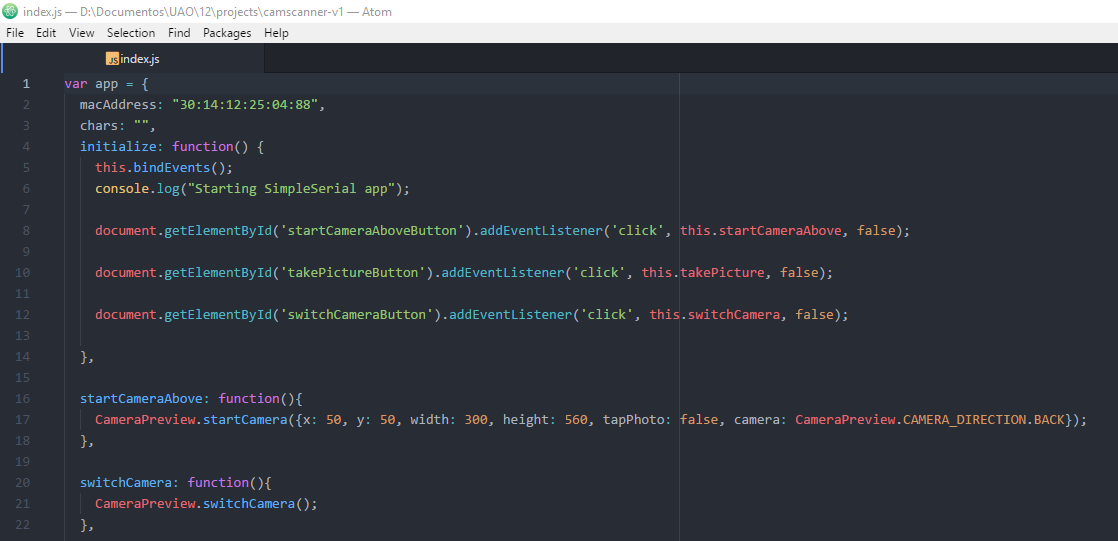
* + 1. Crearemos también un lienzo con la etiqueta <canvas> con un alto y ancho determinado, un id único y un mensaje de prueba, el cual mostrará la imagen que acabamos de tomar
    2. Finalmente esta la seccion de los scripts, los dos primeros son creados automáticamente por cordova, mientras que el último servirá para enlazar el back de la aplicación, que será donde pondremos toda la lógica y el procesamiento.



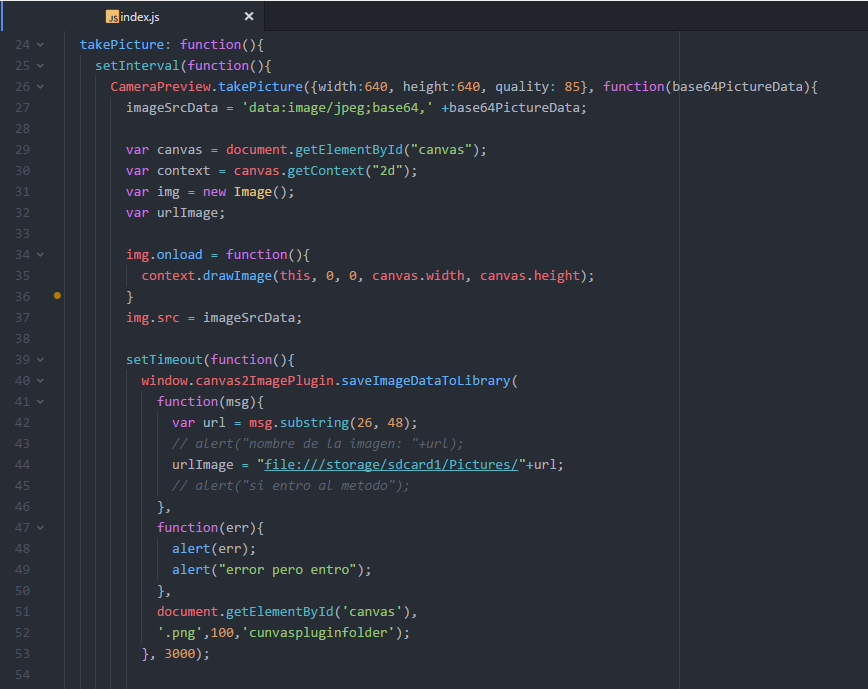
1. Explicación del código JavaScript
   1. Creación del back:
      1. En esta sección se inicializa la aplicación, además de que se ingresa la dirección mac del módulo bluetooth del arduino para realizar la conexión. Se crea una lista de los elementos que están en el front, para que cuando sea presionado un botón determinado accione una función específica.

startCameraAbove: Define el comportamiento y las dimensiones de la cámara embebida, como el tamaño y la direccion(cámara frontal o predeterminada).

switchCamera: Permite hacer el cambio entre la cámara frontal y la predeterminada del celular.



1. takePicture: Es la función encargada de tomar las fotos periódicamente de la cámara embebida, se define el tamaño de la imagen y la calidad, al igual que se le adjunta la propiedad de convertir la imagen en base64 para ser guardada la data en el celular. Luego se especifica dónde se va a mostrar la foto que será tomada, de esta manera se define el canvas que se encuentra en el front y se carga en el mismo. A continuación se emplea el plugin de canvas2image que lo que hace es convertir la imagen que está en el canvas .png o .jpg para ser almacenada en el celular.



1. Finalmente se pasa la ruta de donde fue almacenada la imagen anteriormente a la función encargada de hacer el reconocimiento, tessereactOCR recibe el url de una imagen y retorna lo que interprete por el texto como una cadena de caracteres. Cuando se obtenga el resultado, la cadena de respuesta es enviada al arduino por blutoothSerial.



1. onDeviceReady: Se encarga de preparar el celular para establecer la conexión con el arduino.



1. manageConnection: Se encarga gestionar la conexion y desconexion del celular con el modulo bluetooth del arduino, creando así, un puente seguro para el envío de datos.

openPort: Se encarga de limpiar las conexiones posibles que tenga el celular para vincular estrictamente la dirección mac del módulo bluetooth que definimos al comienzo, de esta manera no se conectará a otros dispositivos visibles, sino, únicamente al que nosotros deseamos.



1. closePort: Todo lo contrario al openPort, hace la desconexión(cierra el puente) entre el celular y el módulo bluetooth



* + 1. senData: Encargado de hacer la prueba de conexión básica entre el arduino y el celular, envía una cadena de prueba “Hola Arduino!”, para saber si el puente se ha hecho de manera correcta.

onCamera: Cumple el papel de abrir la cámara nativa del celular, correcto para hacer pruebas pero no influye en el desarrollo del sistema, al igual que la cámara embebida, esta cumple el papel de tomar la foto, pasarla al reconocedor de imágenes y enviar al arduino lo que entienda de del texto capturado.

