PRÁCTICA 3 - ANDROID GYMKANA

APLICACIONES REALIZADAS:

- **4. PuntoMovimientoSonido**: se pedirá una appMovimientoSonido que reconozca un patrón de movimientos de vuestra elección usando el giroscopio y/o el acelerómetro, una vez detectado el patrón, se reproducirá un sonido.

- **1. BrújulaVoz:** con la appBrujulaVoz se debe identificar mediante interfaz vocal un punto cardinal y el porcentaje de error. Una vez reconocido, se debe mostrar una brújula para que el usuario se oriente en la dirección indicada.

- **5. PuntoSorpresa**: la appSorpresa hará uso del sensor de luz.

- **3. PuntoGestosFoto**: se pedirá una appGestosFoto que reconozca un patrón de gestos sobre pantalla de vuestra elección. Una vez detectado el patrón se debe ejecutar la aplicación de cámara y realizar automáticamente una foto transcurridos 3 segundos.

Autores:

- Miguel Sánchez Maldonado
  Correo: miguesanchez181@gmail.com
  Github: https://github.com/msmaldonado
- Cristina Zuheros Montes.
  Correo: zuhe18@gmail.com
  Github: https://github.com/cristinazuhe

Fecha de realización: 10/02/2016

● 4. PuntoMovimientoSonido:

En esta aplicación vamos a pedir al usuario que mueva el dispositivo enérgeticamente una vez hacia abajo y otra vez hacia arriba. Es decir, tendremos que mover el móvil en el eje Y. Una vez que el movimiento sea correcto sonará un sonido.

Veamos cómo actúa nuestro programa:
Comenzamos en nuestro MainActivity.java, al ejecutar OnCreate se establece el contexto que creamos en activity_main.xml. Aquí tenemos la pantalla principal de nuestra aplicación. En ella creamos varios TextView para explicar el funcionamiento de nuestra aplicación al usuario, además de un button_start que nos permite comenzar con la aplicación en sí. En el mismo OnCreate hacemos que este botón pase a estar escuchando.

Al hacer click sobre el botón de comienzo, nos vamos a la actividad 1, lo que viene siendo nuestro `MovementSoundActivity.` Al ejercutarse OnCreate se establece el contexto que creamos en activity_movement_sound.xml. Este sería nuestro entorno donde realizamos el trabajo que nos pide la aplicación. Se nos irá indicado mediante un mensaje de texto los movimientos que tenemos que realizar. Primero mover el dispositivo hacia abajo y luego hacia arriba. Cuando lo hagamos correctamente se usará el mismo texto para indicar que el movimiento ha sido correcto y sonará el tono que hemos establecido.

Además en dicho método se establecen los sensores, vamos a trabajar con el acelerómetro. Pasamos al método onSensorChanged, donde se lleva realmente la cabo la funcionalidad interna de nuestra aplicación. Veámoslo:

```
public final void onSensorChanged(SensorEvent event) {
    double Y = event.values[1];

    if (first) {
        cY = Y;
        first = false;
        result.setText("Mueva el dispositivo hacia abajo enérgicamente.");
    } else if (!movement_ok) {
        if (Y > cY + 20) {
            result.setText("Mueva el dispositivo hacia arriba enérgicamente.");
            arriba = true;
        } else if (arriba && (Y < cY - 20)) {
            movement_ok = true;
        }
        cY = Y;

        if (movement_ok)
            playSound();
    }
```

Como hemos comentado anteriormente nos movemos en el eje Y. Tenemos que ver que la posición inicial (cY) y la posición que tenemos en el momento (Y), difieran en un valor de 20, primero en positivo, para que el movimiento se haya hecho hacia abajo y posteriormente en negativo para que el movimiento se haga hacia arriba. Una vez que se mueve hacia abajo y arriba, indicamos que el movimiento es correcto y pasamos al método playSound() en el que se provoca la reproducción del sonido que indica el final de la aplicación.

Veamos como queda:

● 1. BrújulaVoz:

En esta aplicación vamos a pedir al usuario que indique un punto cardinal por voz (Norte, Sur, Este, Oeste) seguido de un número entero que será la tolerancia que el mismo usuario permitirá para estar en la dirección (punto cardinal) que desea.
Una vez que reconoce el patrón vocal , se nos permite cambiar de nuevo el punto cardinal y tolerancia, sino comienza la navegación. Para ello aparece una brújula la cual nos ayudará a orientarnos hacia la dirección elegida, una vez que estamos en la posición correcta se nos indica que ya lo hemos realizado.

Veamos cómo actúa nuestro programa:
Comenzamos en nuestro MainActivity.java, al ejecutar OnCreate se establece el contexto que creamos en activity_main.xml. Aquí tenemos la pantalla principal de nuestra aplicación. En ella creamos varios TextView para explicar el funcionamiento de nuestra aplicación al usuario, además de un button_start que nos permite comenzar con la aplicación en sí. Al pulsar el botón nos redirige a la actividad VoiceRecognitionActivity. Al ejercutarse OnCreate se establece el contexto que creamos en activity_voice.xml. En esta pantalla aparecerá un texto indicando que tenemos que hacer y un button_speak el cual al ser pulsado hará `startRecognition();` para que el usuario diga el punto cardinal con el error.
Si tras el reconocimiento de voz se detecta un punto cardinal acompañado de su tolerancia, entonces aparecerá en la pantalla debajo de speak_button el punto cardinal con su tolerancia.

```
private void checkData(ArrayList<String> data, TextView text) {
    float aux = search(data, "norte");
    if (aux != 0F) {
        error = aux;
        cardinal = "norte";
        text.setText("Norte" + " : " + aux);
    }
     aux = search(data, "sur");
    if (aux != 0F) {
        error = aux;
        cardinal = "sur";
        text.setText("Sur" + " : " + aux);
    }
     aux = search(data, "este");
    if (aux != 0F) {
        error = aux;
        cardinal = "este";
        text.setText("Este" + " : " + aux);
    }
     aux = search(data, "oeste");
    if (aux != 0F) {
        error = aux;
        cardinal = "oeste";
        text.setText("Oeste" + " : " + aux);
    }
}
```

Una vez que nos aparece nuestro punto cardinal y error, se activa el boton de navegación así como un texto para indicarnos que ya podemos comenzar la navegación o introducir otro punto cardinal y su tolerancia. Esto se realiza en la función CheckResults la cuál nos activa el botón de navegación y el texto y además nos permite seguir pulsando el speak_button (por si queremos introducir un nuevo punto cardinal) . Veamos el código:

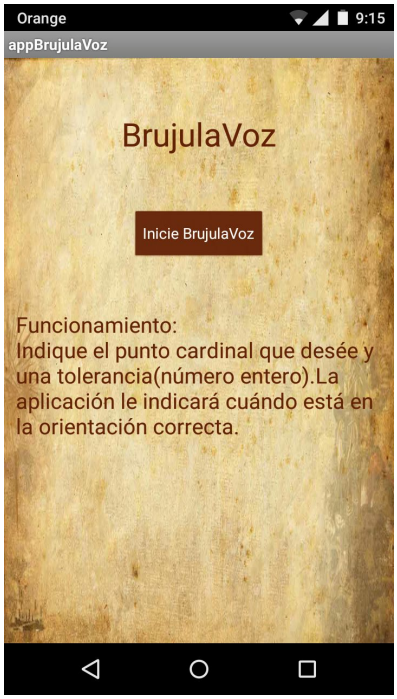
```
private void checkResults(ArrayList<String> matches) {
    checkData(matches, card);
    speakButton.setText(getResources().getString(R.string.button_speak_again));
    if (error != 0F) {
        navigationButton.setEnabled(true);
        naveg = (TextView) findViewById(R.id.naveg);
        naveg.setText("Puede comenzar navegación o indicar otro punto cardinal y su tolerancia.");
    }
    else
        navigationButton.setEnabled(false);
}
```

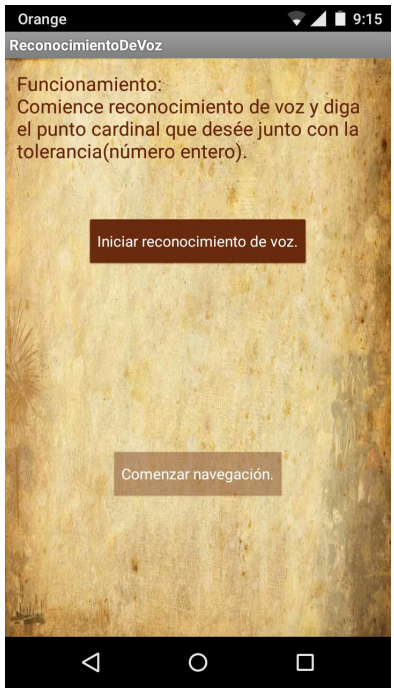
Al pulsar el botón de navegación nos vamos a la última actividad, CompassActivity la cual nos muestra la pantalla activity_gps.xml . Esta actividad nos almacenará el punto cardinal con su tolerancia y por tanto nos mostrará el ángulo actual de nuestra brújula así como el punto cardinal que deseábamos dirigirnos. Por último hará comprobaciones para determinar en qué momento estamos en la dirección indicada. Para la comprobación tendremos en cuenta que cada punto cardinal tiene unos grados asociados, siendo el norte el grado 0 y van aumentando en sentido antihorario hasta dar la vuelta completa (360º), por tanto el oeste serían 90º, el sur 180º y el este 270º. La tolerancia nos permitirá que el rango de valores válidos del punto cardinal seán
 [gradosPuntoCardinal - tolerancia, gradosPuntoCardinal +tolerancia] destacando que como el norte comienza en 0 y no hay números negativos, el otro lado del intervalo sería el 360º - tolerancia. Veámos el código.

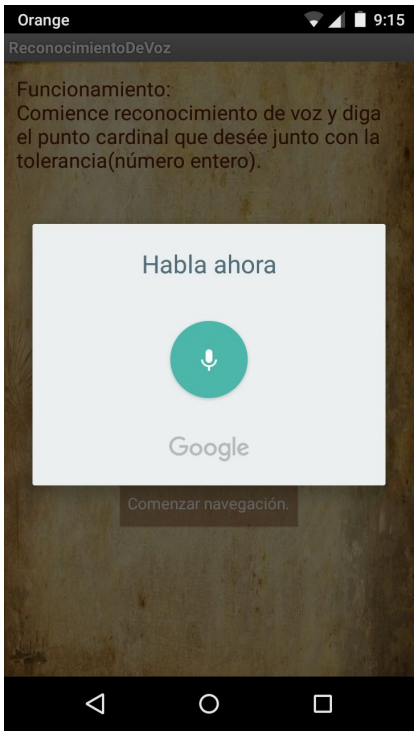
```
float degree = Math.round(event.values[0]);
```

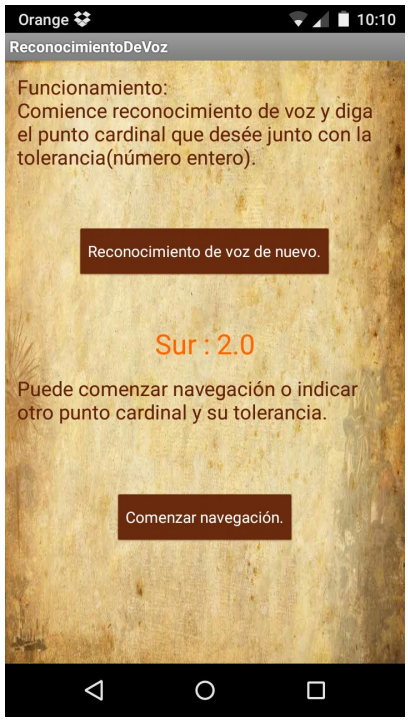
```
if(( cardinal.equals("norte") && (degree < tolerancia || degree > 360.0 - tolerancia )) ||
    ( cardinal.equals("sur") && degree > 180.0 - tolerancia && degree < 180.0 + tolerancia ) |
    ( cardinal.equals("oeste") && degree > 90.0 - tolerancia && degree < 90.0 + tolerancia ) |
    ( cardinal.equals("este") && degree > 270.0 - tolerancia && degree < 270.0 + tolerancia ))
        Intent returnIntent = new Intent();
        setResult(RESULT_OK, returnIntent);
        finish();
}
```

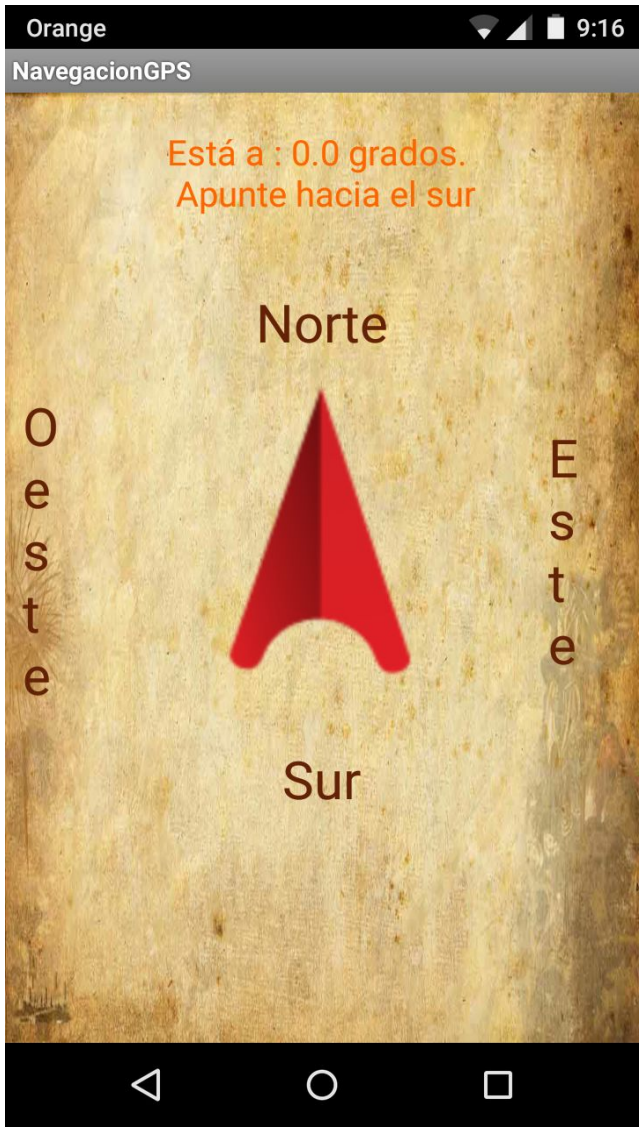
Veamos cómo quedaría el funcionamiento de la aplicación










Orange
9:16
NavegacionGPS
Está a : 0.0 grados.
Apunte hacia el sur
Norte
Oeste
Este
Sur

● 5. PuntoSorpresa:

En esta aplicación vamos a activar el sensor de luz del móvil, de modo que al dependiendo de la intensidad de luz que recoja irá cambiando una imagen mostrada. A mayor número de luz recibida aparece una imagen con más estrellas.

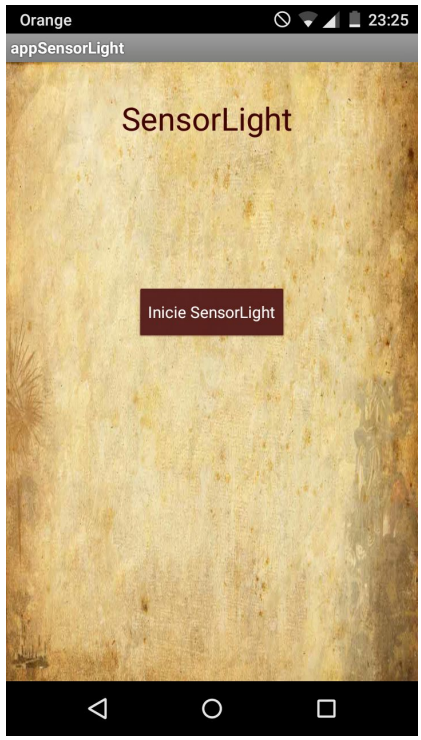
Veamos cómo actúa nuestro programa:
Comenzamos en nuestro MainActivity.java, al ejecutar OnCreate se establece el contexto que creamos en activity_main.xml. Aquí tenemos la pantalla principal de nuestra aplicación. En ella creamos principalmente tenemos el button_start que nos permite comenzar con la aplicación en sí. Al pulsar el botón nos redirige a la actividad SensorLightActivity. Al ejercutarse OnCreate se establece el contexto que creamos en activity_light.xml.Esta aplicación será la encargada de activar los sensores de luz y de su uso. En esta pantalla se nos muestra un textView para indicarnos que tenemos que variar la intensidad de luz. Debajo tenemos otro TextView que nos mostrará la intensidad de luz captada por el sensor en todo momento .

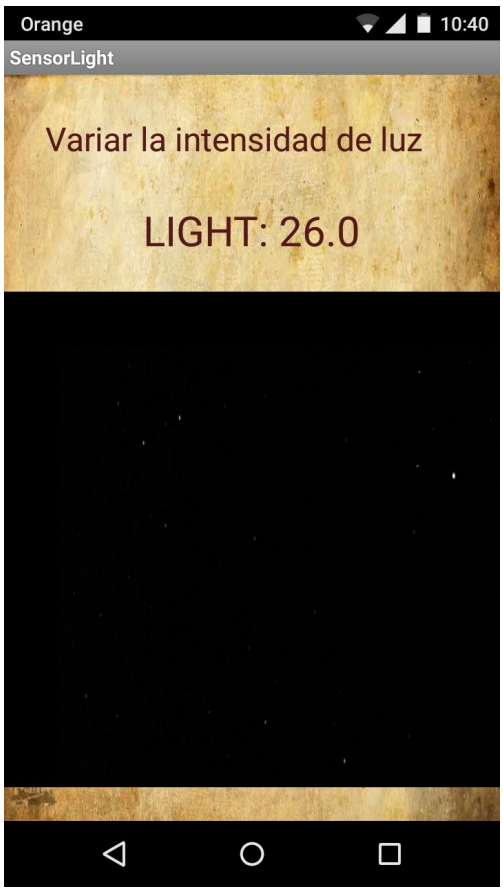
```
public final void onSensorChanged(SensorEvent event) {
    if(event.sensor.getType() == Sensor.TYPE_LIGHT)
        txtLight.setText("LIGHT: " + event.values[0]);
```

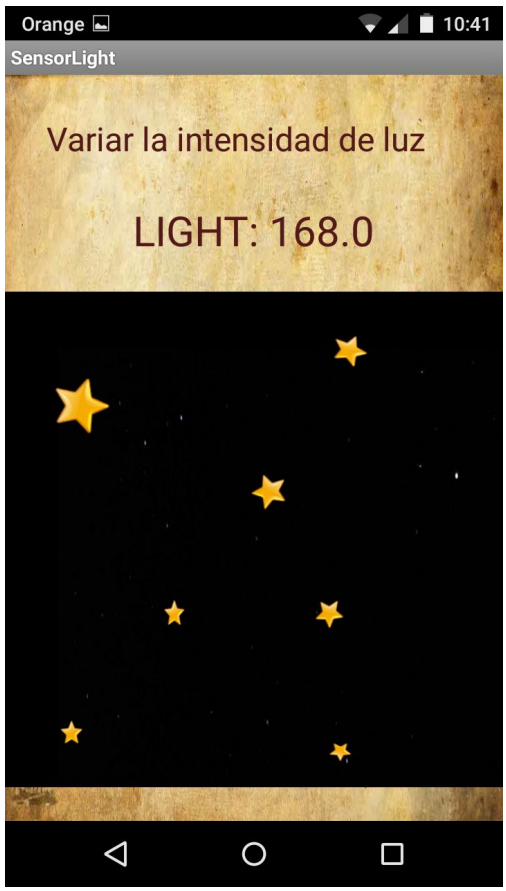
Debajo tenemos una imagen que se irá cambiando dependiendo de la intensidad de luz captada por el sensor. Para ello tenemos varias imágenes dentro del proyecto en drawable y lo que haremos para cada rango de intensidad de luz activar que en la imageView se nos muestre uno u otra. Veamos el código

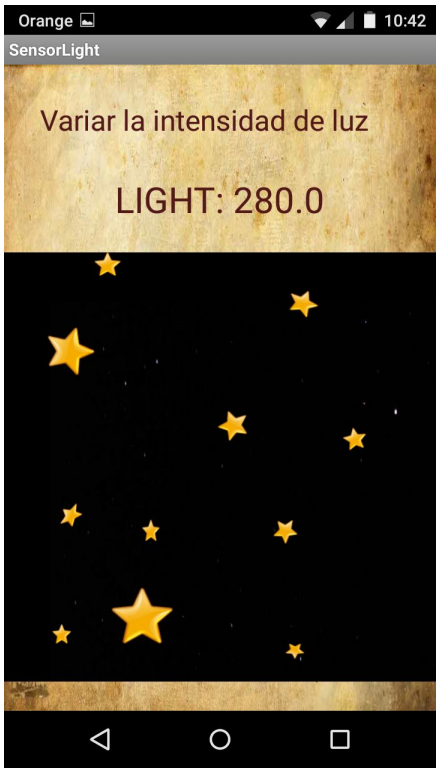
```
if(event.values[0] < 100){
    image.setImageDrawable(getResources().getDrawable(R.drawable.fondo));
}
else if(event.values[0] < 200){
    image.setImageDrawable(getResources().getDrawable(R.drawable.fondo1));
}
else if(event.values[0] < 300){
    image.setImageDrawable(getResources().getDrawable(R.drawable.fondo2));
}
else if(event.values[0] < 400){
    image.setImageDrawable(getResources().getDrawable(R.drawable.fondo3));
}
else if(event.values[0] > 400){
    image.setImageDrawable(getResources().getDrawable(R.drawable.fondo4));
}
```

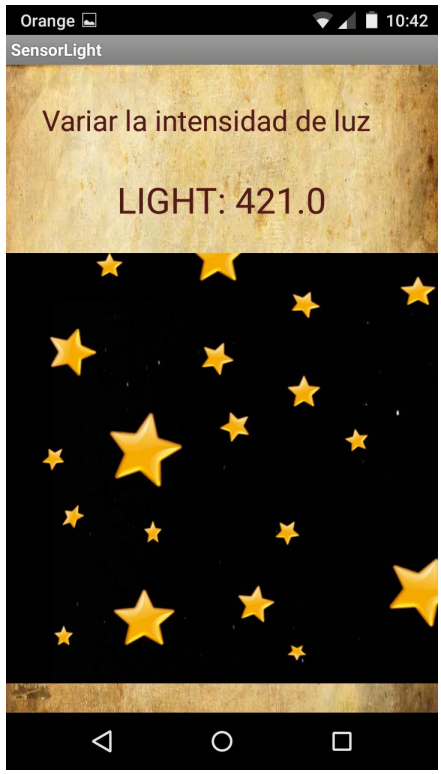
Veamos a continuación el funcionamiento de la aplicación


Orange
23:25
appSensorLight
SensorLight
Inicie SensorLight


Orange
10:40
SensorLight
Variar la intensidad de luz
LIGHT: 26.0


Orange
10:41
SensorLight
Variar la intensidad de luz
LIGHT: 168.0

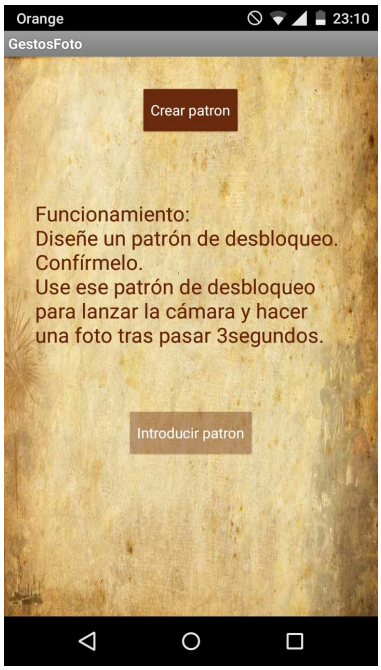

Orange
10:42
SensorLight
Variar la intensidad de luz
LIGHT: 280.0

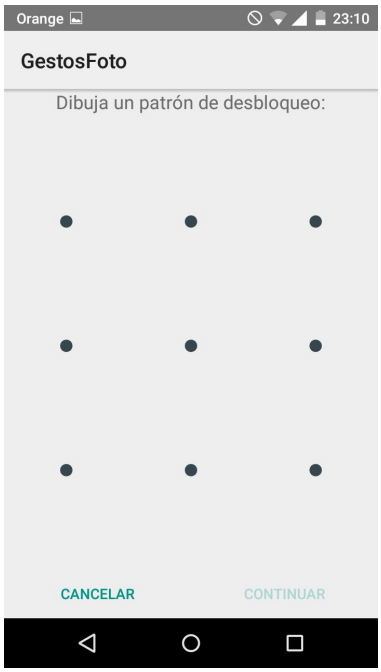

Orange
10:42
SensorLight
Variar la intensidad de luz
LIGHT: 421.0

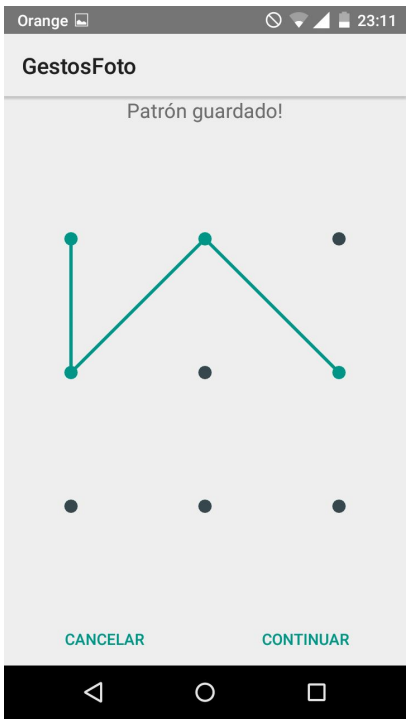
● 3: GestosFoto:

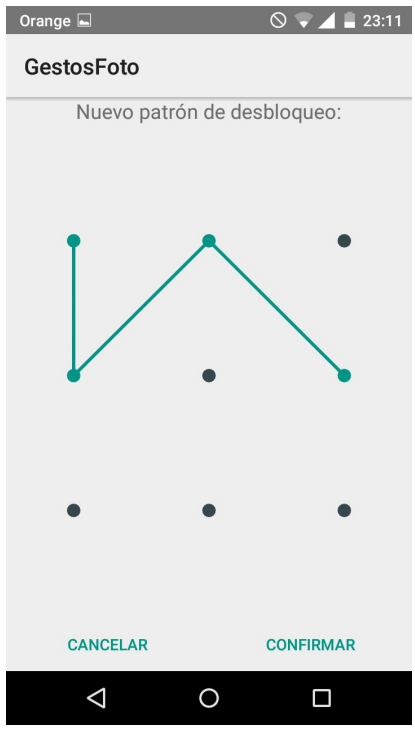
En esta aplicación se trata de pedir al usuario que cree que un patrón para posteriormente al introducir ese patrón correctamente, se nos activa la cámara y pasados 3 segundos efectuará una foto. Para asegurar que el patrón que se crea es correcto y el que el usuario deseaba le pedimos que se introduzca dos veces a modo de comprobación. Una vez que tenemos un patrón guardado ya podemos darle al botón de introducir el patrón , una vez que el patrón sea correcto se nos activará la cámara.

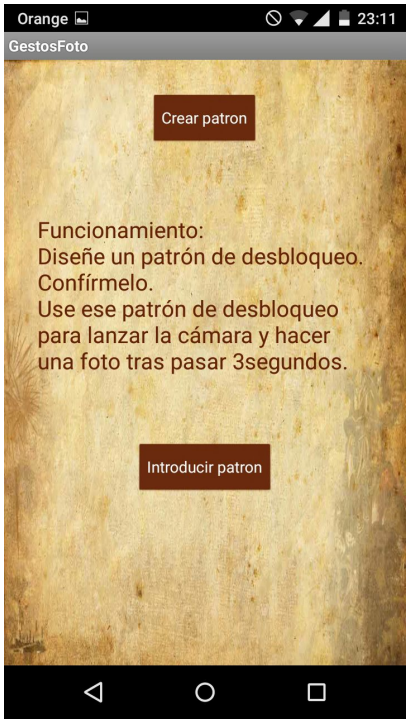
Veamos más detalladamente el programa.
Nuestra aplicación consta de 3 actividades, MainActivity, CameraPreview y CameraActivity.
MainActivity será la actividad principal, la cual nos lleva a la pantalla principal donde se encuentra una breve descrpción de lo que realiza la aplicación, el botón para crear el patrón e introducir patrón. En ella se tiene en cuenta el patrón que se introduce para almacenarlo, así como comprobarlo cuando introducimos el patrón para activar la foto.
Las actividades CameraPreview y CameraActivity será las encargadas de activar los sensores de la cámara. Una vez activada se nos irá mostrando la vista previa de la cámara permitiendo al usuario enfocar a que quiere realizar la foto, y pasados 3 segundos realizará una foto. Esta aplicación guarda la foto en nuestra galería en un nuevo albúm con el nombre de la aplicación. Si es la primera vez que descargamos y realizamos una foto, creará el albúm.

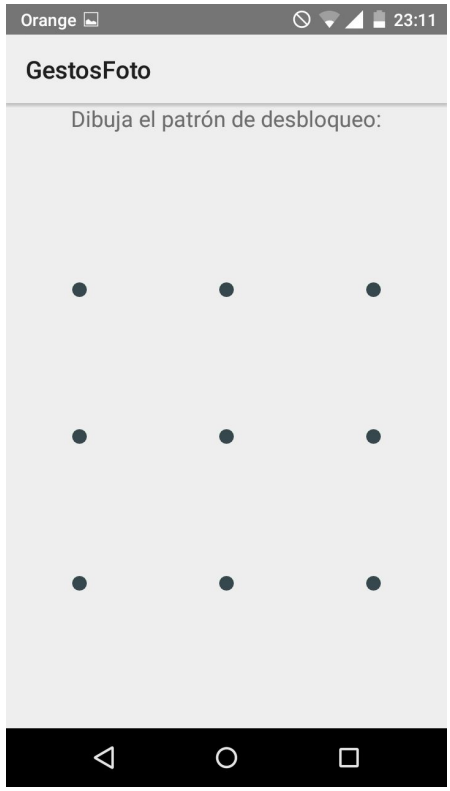

Orange
23:10
GestosFoto
Crear patron
Funcionamiento:
Diseñe un patrón de desbloqueo.
Confírmelo.
Use ese patrón de desbloqueo
para lanzar la cámara y hacer
una foto tras pasar 3segundos.
Introducir patron

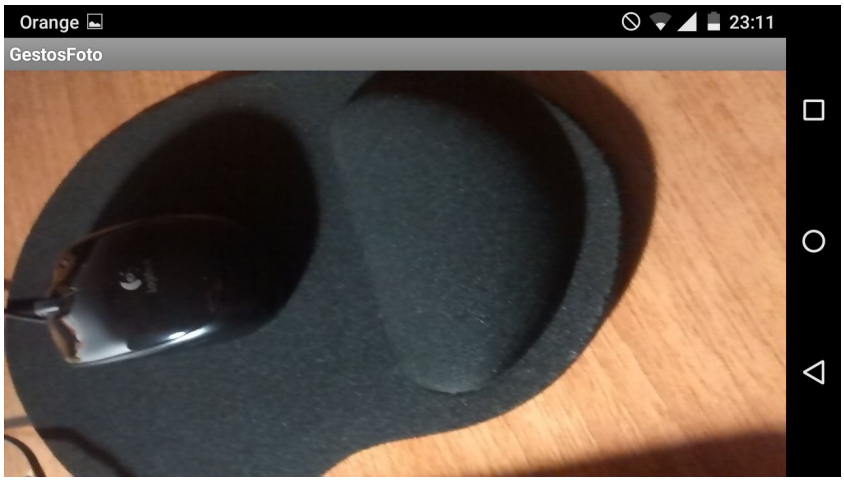

Orange
23:10
GestosFoto
Dibuja un patrón de desbloqueo:
CANCELAR
CONTINUAR


Orange
23:11
GestosFoto
Patrón guardado!
CANCELAR
CONTINUAR


Orange
23:11
GestosFoto
Nuevo patrón de desbloqueo:
CANCELAR
CONFIRMAR


Orange
23:11
GestosFoto
Crear patron
Funcionamiento:
Diseñe un patrón de desbloqueo.
Confírmelo.
Use ese patrón de desbloqueo
para lanzar la cámara y hacer
una foto tras pasar 3segundos.
Introducir patron


Orange
23:11
GestosFoto
Dibuja el patrón de desbloqueo:


Orange
23:11
GestosFoto

Errores frecuentes o aspectos destacados

Casi todos los errores que encontramos fueron con el manejo de los sensores. Sobre todo con el sensor del movimiento ya que recogía información de movimiento en 3D y tuvimos que profundizar más detalladamente en internet para entender y conseguir controlar el movimiento solo en un eje en nuestro caso fue el Y, ya que hacemos movimiento vertical.

En todas las aplicaciones le hemos puesto un icono para identificar la aplicación instalada en el móvil. Para ello simplemente hemos puesto una imagen en drawable con el nombre de ic_launcher.png. La única restricción es que la imagen tiene que ser poco pesada ya que será usada como icono.

Bibliografía

http://developer.android.com/intl/es/guide/index.html
https://github.com/jmorenov/NPI_GYMKANA
http://developer.android.com/intl/es/guide/topics/media/index.html
http://developer.android.com/intl/es/guide/topics/ui/index.html
http://stackoverflow.com/questions/8924683/camera-application-in-android
http://developer.android.com/intl/es/sdk/index.html
http://androidstudiofaqs.com/tutoriales/como-poner-una-imagen-de-fondo-en-una-aplicacion-android
http://www.maestrosdelweb.com/curso-android-trabajando-con-imagenes/
http://www.jc-mouse.net/android/tomar-fotos-con-la-camara-y-guardar-en-la-sdcard
http://www.tutorialeshtml5.com/2013/12/tutorial-android-utilizar-el-intent-de.html
http://code.tutsplus.com/es/tutorials/getting-started-with-android-studio--mobile-22958
http://www.elandroidelibre.com/2015/06/como-programar-la-aplicacion-calculadora-basica-en-android-studio.html