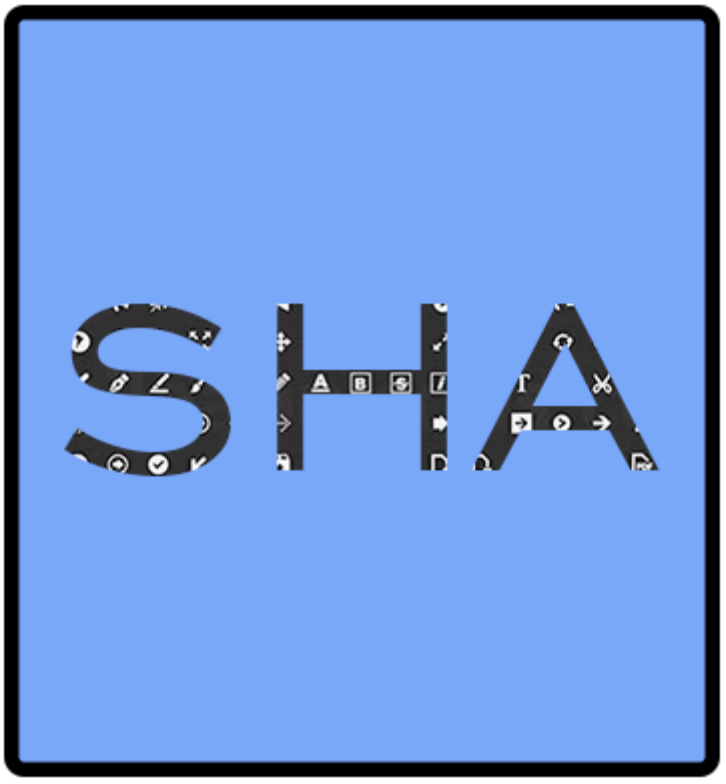
Institut Puig Castellar

Santa Coloma de Gramenet



SHA(Sensor Hardware App)

CFGS Desenvolupament d'Aplicacions Multiplataforma

Javier Ramírez Viñau

Guillermo Pulido Fernández

2n DAM A

01/06/2018

Resum del projecte (màxim 250 paraules):

El nostre projecte és una aplicació Android on la seva funció és analitzar que els sensors i components d'un determinat dispositiu mòbil funcionin correctament.

Cada sensor i component serà avaluat pel propi usuari, que és el que ha de verificar que tot funcioni bé fent les proves de l'app.

En cas que el telèfon mòbil no incorpori un sensor determinat, se li informarà a l'usuari.

Tots els resultats es guardaran en una base de dades per posteriorment poder classificar-los per diferents models de mòbils i així mostrar els resultats en una pàgina web.

Cada vegada que es faci alguna prova, la base de dades s'actualitzarà.

L'app també tindrà un apartat on l'usuari podrà comprovar el resultat dels tests anteriors.

Abstract (in English, 250 words or less):

Our project is an Android application whose function is to analyze that the sensors and components of a certain mobile device work correctly.

Each sensor and component will be evaluated by the user, who is the one who has to verify that everything works well by doing the tests of the app.

In case the mobile phone does not incorporate a certain sensor, the user will be informed.

All the results will be stored in a database to later be able to classify them by different mobile models and thus show the results in a web page.

Each time a test is done, the database will be updated.

The app will also have a section where the user can check the result of the previous tests.

Paraules clau (entre 4 i 8):

Sensor, Hardware, Android Studio, Activity, Firebase, dispositiu mòbil

Índex

1. Introducció

1.1 Context i justificació del Treball

Realitzar una aplicació android que la seva funció sigui analitzar el funcionament de tots els sensors del hardware d'un respectiu telèfon mòbil.

Els resultats passaran a una base de dades i posteriorment es mostraran en una pàgina web on es classificaran per diferents models de mòbil.

Cada vegada que es faci una prova completa dels sensors, es mostraran en un apartat dins de l'app i s'actualitzarà a la base de dades.

1.2 Objectius del Treball

- Realitzar una recerca dels recursos per crear la nostra aplicació Android.
- Desenvolupar una aplicació per provar les funcions del telèfon mòbil.
- Emmagatzemar en una base de dades dels resultats obtinguts.
- Visualitzar els diferents resultats per cada mòbil registrat en una pàgina web.

1.3 Enfocament i mètode seguit

Adaptar programes que serveixen per a l'anàlisi de les funcions del telèfon mòbil i fer una pàgina web on podrem observar les característiques dels telèfons en concret.

1.4 Planificació del projecte

Planificació

Activitat	**Descripció**
Selecció del projecte	Pensar el tema del projecte.
Selecció de materials	Pensar quins requisits i tecnologies haurem d'emprar en el nostre projecte.
Definició del projecte	Escriure una breu descripció del projecte mencionant les tecnologies que utilitzarem.
Planificació temporal	Planificació de la gestió del temps en relació al projecte.

Disseny

Activitat	**Descripció**
Disseny BBDD	Dissenyar com serà la base de dades en Firebase.
Disseny aplicació android	Realització de l'esquema funcional de l'aplicació per conèixer l'estructura i els mòduls que podria tenir
Disseny interficie APP	Pensar quin tipus de disseny tindrà l'aplicació.
Disseny pàgina Web	Realitzar un esquema dels apartats que tindrà la pàgina web.

Implementació

Activitat	**Descripció**
Creació BBDD	Crear la base de dades a partir del disseny realitzat anteriorment en Firebase.
Investigació de la API	Investigar i agafar els coneixements per fer la aplicació.
Desenvolupament aplicació Android	Desenvolupament de l'aplicació en Android Studio amb tots els apartats que tindrà.
Desenvolupament pagina web	Crear una pàgina web a partir del disseny.
Mostrar dades en pagina web	Una vegada feta la pàgina web, implementar les dades de la BBDD i mostrarles.

Finalització

Activitat	**Descripció**
Redacció final de la memòria	Recopilació de documents i confecció de la memòria final
Presentació	Preparació de la presentació amb diapositives.
Revisió	Revisió final del projecte

Diagrama de Gantt

1.5 Breu sumari de productes obtinguts

Obtindrem:

- L'aplicació Android on analitzarem les funcions.
- Una base de dades Firebase on registrarem els resultats obtinguts
- Una pagina web on mostrarem els resultats de la base de dades.

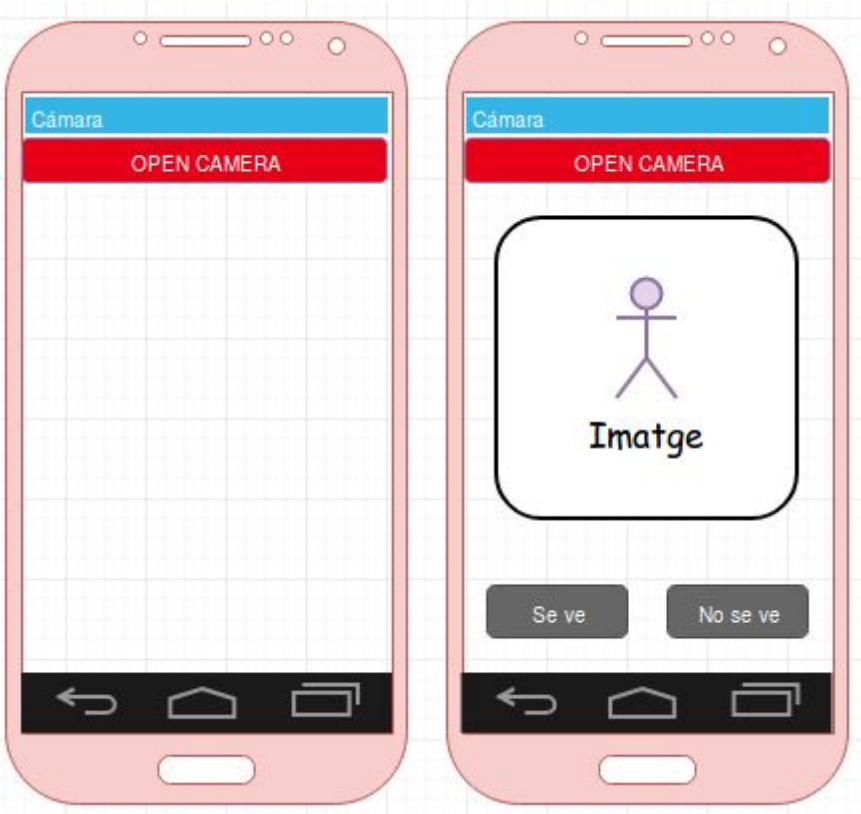
1.6 Breu descripció dels altres capítols de la memòria

En els següents capítols explicarem la manera en la que fem la nostra aplicació, passant per una serie de tasques com per exemple la creació de la base de dades amb la pagina web i el desenvolupament de la aplicació.

2. Resta de capítols

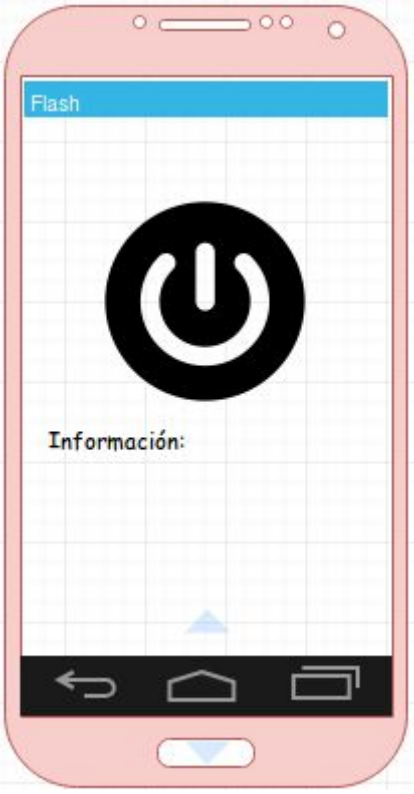
2.1 Disseny de la interfície app

CameraActivity: Quan s'obre aquesta activity trobarem un botó que posa obrir la càmera. Quan es prem el botó, podrem fer una foto i quan estigui feta es mostrarà a la mateixa activity. Després sortirà dos botons més que servirà per confirmar si es veu la fotografia o no. El resultat passarà al base de dades.

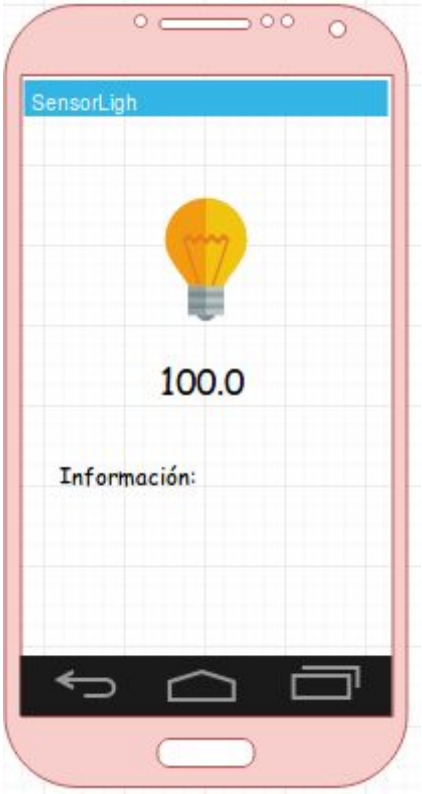


ProximityActivity: Aquí haurà un botó per iniciar el sensor, quan fas clic i el sensor et detecta la pantalla canviarà de color. També es mostrarà el model del sensor a la mateixa activity. En cas de que no tingui aquest sensor, es mostrarà un missatge informant a l'usuari.

FlashlightActivity: Aquí es comprovarà si el flash del mòbil funciona correctament. Si l'usuari veu que funciona o a la inversa, ho tindrà que indicar polsant als botons que hi ha per que l'app o sàpiga.

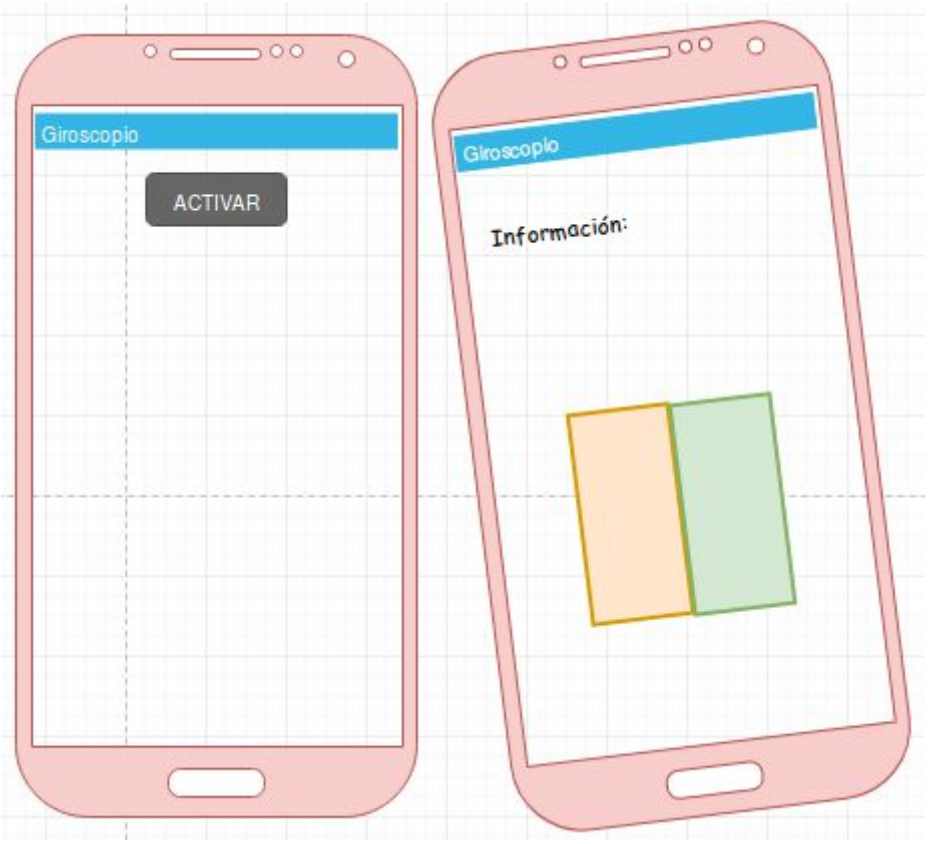


SensorLightActivity: Aquesta activity activa el sensor de llum. La activity mostra un textview, i el valor d'aquest text es la quantitat de llum que hi arriba al telefon mòbil. En cas de que no tingui el sensor, el text canviarà. També té una imatge que canvia segons la quantitat de llum que li arribi.

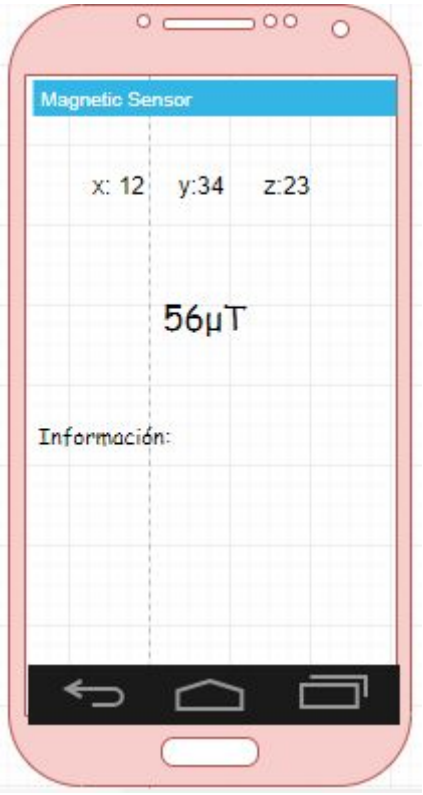


SensorTestFinal: Quan es faci un test sencer de totes sortirà tots els resultats en una activity.

GiroscopeActivity: Aquest activity tindrà un botó per activar el sensor, quan es mogui el telefon la pantalla canviarà de color respecte la seva posició. També apareixerà informació del sensor.

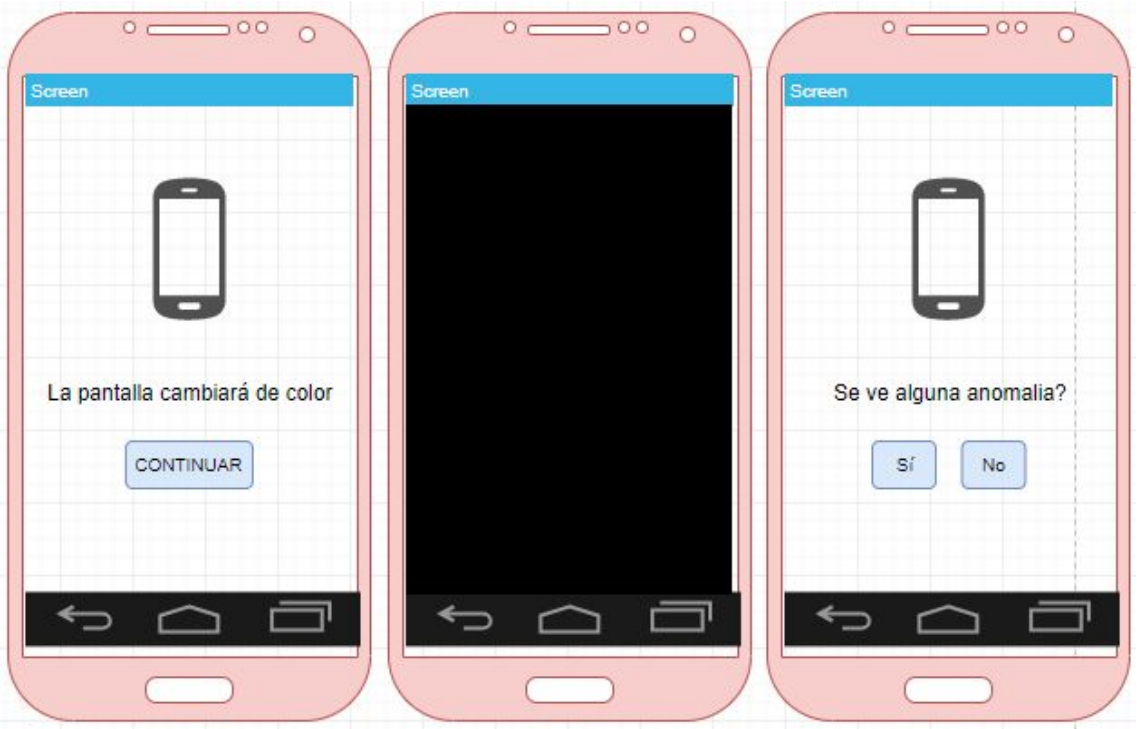


MagneticActivity: Això utilitza el sensor magnètic del mòbil, i mostrarà quants microTeslas hi ha en l'entorn. També mostrarà tota la informació relacionada.

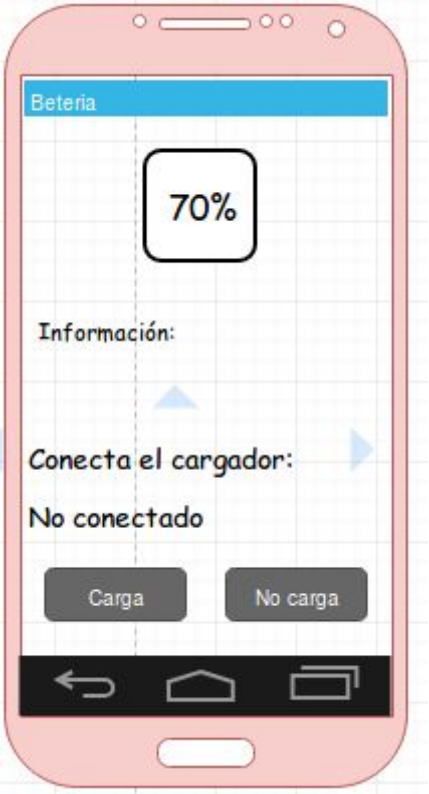


ScreenActivity: Això no es un sensor, però la nostra app està per verificar que estigui tot correcta, per això aquí el que s'intenta fer es detectar que la pantalla no tingui píxels morts o els píxels encallats. Un píxel mort es un píxel que està apagat de color negre. Un píxel encallat son aquells que no canvien de color, sol ser un color diferent als dels seu voltant.

El que fa la activity es primer canviar de color negre a un color blanc, i així es podrà observar si hi ha algun defecte.



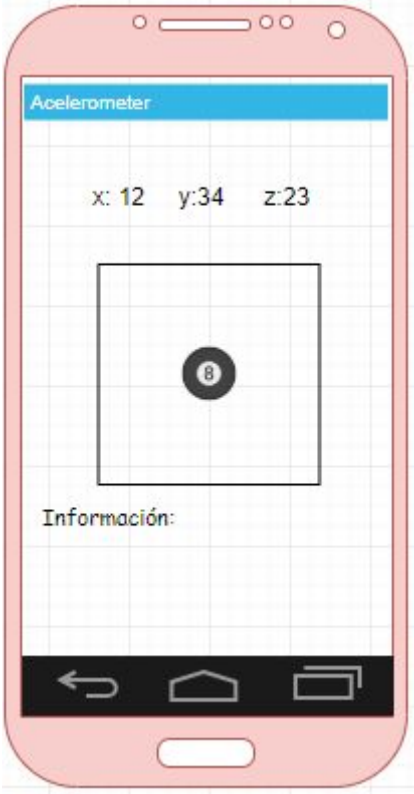
BatteryActivity: Quan s'obre el primer que es veu es el percentatge de bateria actual. Després sortirà quin es el model de bateria i més informació. També et mostrarà un missatge perquè es connecti el carregador, i un cop posat es mostrarà si s'està carregant el mòbil o no.



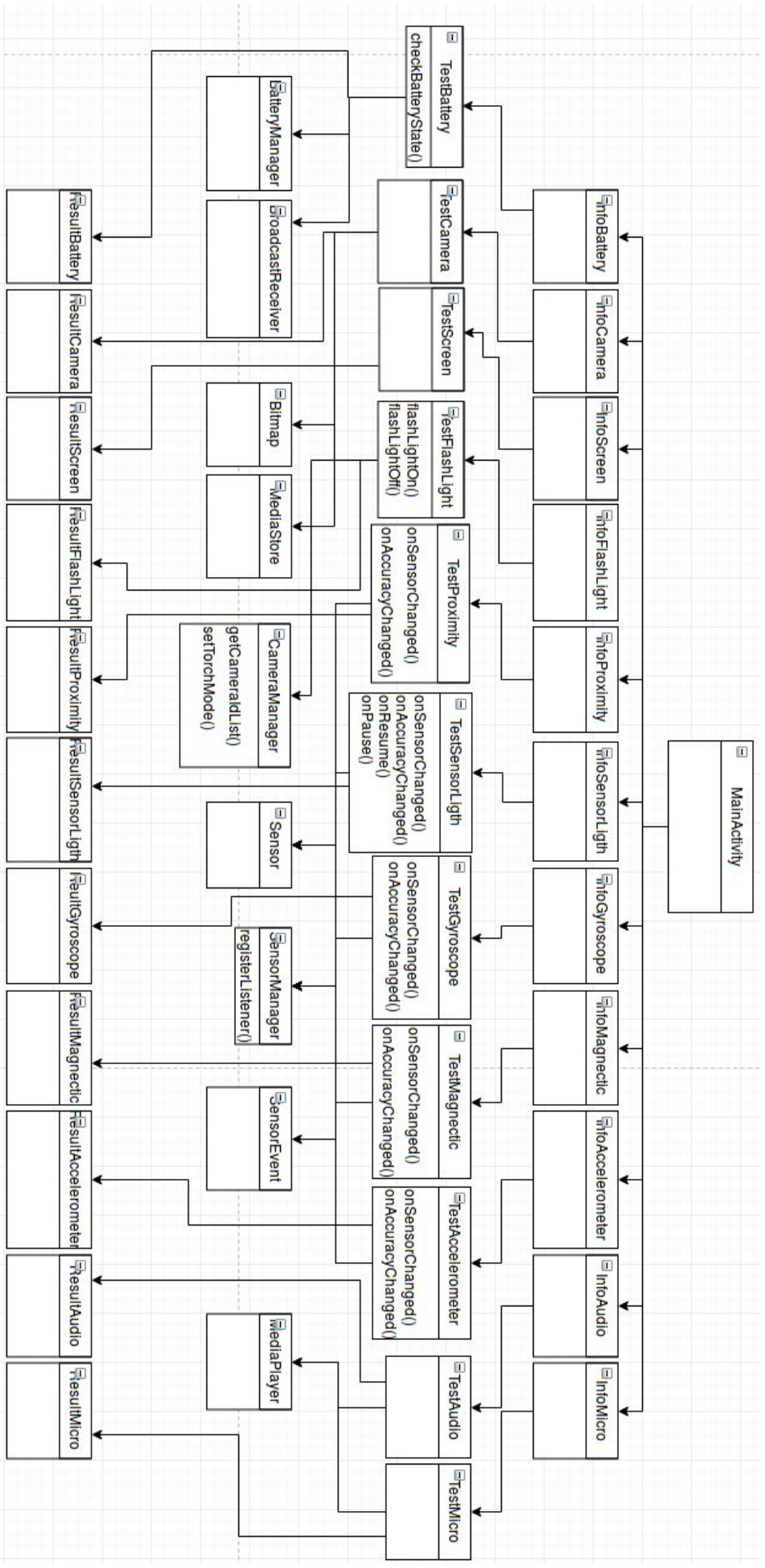
MicroActivity: Quan s'obre hi haurà un botó per començar a gravar la teva veu. El micròfon estarà actiu durant 10 segons. Després començara a reproduir-se el que has dit.

AudioActivity: Aquí quan s'obre hi haurà un botó per començar a reproduir-se un àudio. Després hi haurà dos botons per que l'usuari indiqui si funciona o no.

AccerelometerActivity: Aquí es prova el funcionament d'aquest sensor. Es mostrarà una imatge que canvi de posició segons la posició de la pantalla. També es mostrarà la informació del sensor.



2.2 Diagrama de classes / Disseny App android:



2.3 Implementació de l'app

Al final hem fet una cosa diferent al que teníem pensat. Quan iniciem l'app el primer que veiem es la icona de la aplicació, i quan es clica al icona surt el menú principal. Quan es clica a un sensor es passa a una activity on es mostra informació del sensor seleccionat. Després es passa al test i per últim la pantalla del resultat.

Pantalla d'inici:

Aquesta pantalla es mostra quan inicies l'aplicació. Mostra la icona de l'aplicació i un text on dona la instrucció per començar a utilitzar l'app.

Com diu el text, té que mantenir pressionat la imatge per avançar.

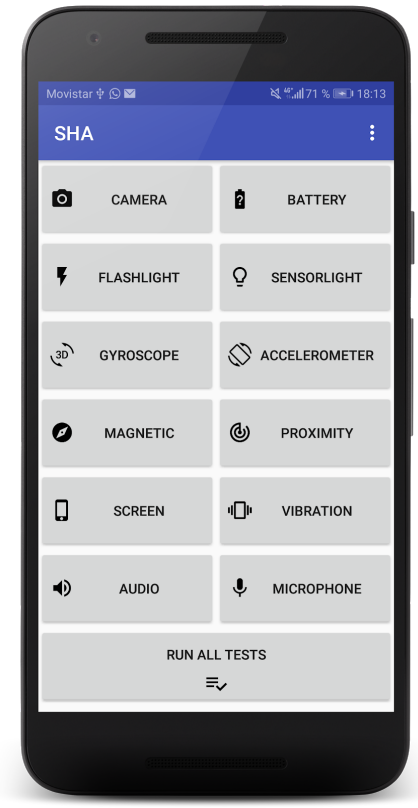
XML: Aquesta activity conté dos 'LinearLayout'. El primer per mostrar una 'ImageView' i el segon mostra un text d'instrucció.

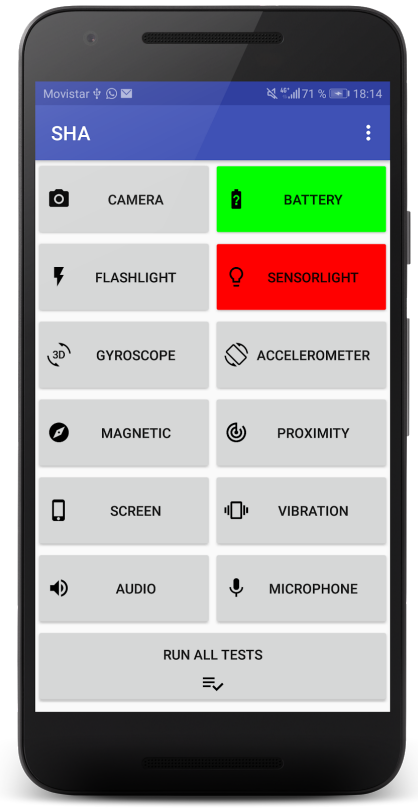
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="center_horizontal">
    <Button
        android:id="@+id/start"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_marginHorizontal="@dimen/marginGeneral"
        android:layout_marginTop="100dp"
        android:background="@drawable/logo"/>

</LinearLayout>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">
    <TextView
        android:id="@+id/text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="20dp"
        android:layout_marginHorizontal="@dimen/marginGeneral"
        android:layout_marginTop="@dimen/marginInfo"
        android:textColor="@android:color/black"
        android:textAlignment="center"
        android:text="Mantenga pulsado para empezar"/>

</LinearLayout>
```

Codi: El codi d'aquesta activitat es molt sencill. Conté un botó on es referència amb l'id del XML (start). Després, es crea un Listener per que quan es manté polsat el botó, avanci. Això es fa amb un OnLongClickListener().

```
Button start;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_start);

    start = findViewById(R.id.start);
    start.setOnLongClickListener(new View.OnLongClickListener() {
        @Override
        public boolean onLongClick(View view) {
            startActivity(new Intent(StartActivity.this, MainActivity.class));
            return false;
        }
    });
}
```

Menú: Aquí podem observar 12 botons, aquest botons son els sensor que podem testejar. Després hi ha un botó més gran que serveix per iniciar totes les proves seguides dels sensors.





Això es un exemple del funcionament del botons, quan fem clic en ell, s'inicia una nova activitat.

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);

    //----------------------------------------

    camButton = (Button) findViewById(R.id.camera);
    camButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            startActivity(new Intent(MainActivity.this, CameraInfoActivity.class));
        }
    });
```

Una vegada estan implementats tots els botons, cridem al mètode **loadResults()**. Aquest mètode busca dintre de la base de dades Firebase i comprova el valor.
Si es “null”, vol dir que es la primera vegada que s’inicia aplicació en aquest dispositiu i no s’ha fet cap test d’un sensor. En aquest cas el color del botó no varia (per defecte es gris).

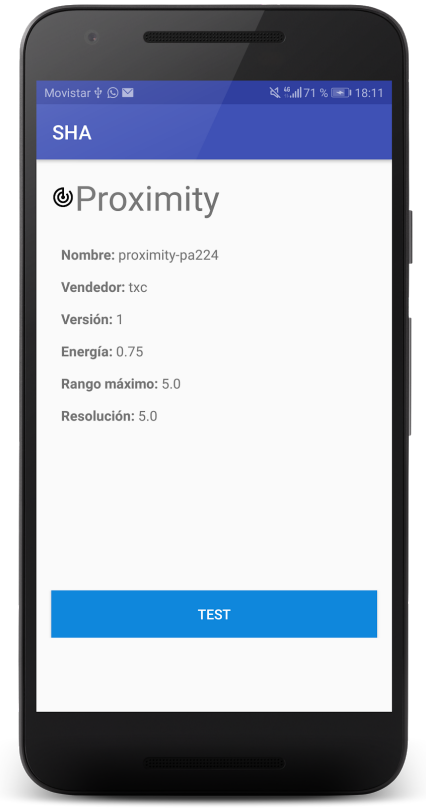
En cas que no sigui “null”, el botó del sensor que s’hagi testejat canviarà a verd(si funciona correctament) o a vermell (si no funciona bé).

```
void loadResults(){
    String iid = SHAUtils.getAndroidId(this);

    FirebaseDatabase.getInstance().getReference().child(iid).addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            SHAResult shaResult = dataSnapshot.getValue(SHAResult.class);

            if(shaResult == null){
                setButtonColor(batteryButton, null);
                setButtonColor(camButton, null);
                setButtonColor(screenButton, null);
                setButtonColor(vibrateButton, null);
                setButtonColor(sensorlightButton, null);
                setButtonColor(flashlightButton, null);
                setButtonColor(gyroscopeButton, null);
                setButtonColor(audioButton, null);
                setButtonColor(microButton, null);
                setButtonColor(proximityButton, null);
                setButtonColor(magneticButton, null);
                setButtonColor(accelerometerButton, null);
            }else {
                setButtonColor(batteryButton, shaResult.battery);
                setButtonColor(camButton, shaResult.camera);
                setButtonColor(screenButton, shaResult.screen);
                setButtonColor(vibrateButton, shaResult.vibrate);
                setButtonColor(sensorlightButton, shaResult.sensorlight);
                setButtonColor(flashlightButton, shaResult.flashlight);
                setButtonColor(gyroscopeButton, shaResult.gyroscope);
                setButtonColor(audioButton, shaResult.audio);
                setButtonColor(microButton, shaResult.microphone);
                setButtonColor(proximityButton, shaResult.proximity);
                setButtonColor(magneticButton, shaResult.magnetic);
                setButtonColor(accelerometerButton, shaResult.accelerometer);
                setButtonColor(alltestButton, shaResult.runAllTest);
            }
        }
```

Aquest mètode analitza un booleà per canviar el color dels botons.

```
void setButtonColor(Button button, Boolean ok){
    if(ok == null){
        return;
    }else if(ok){
        button.setBackgroundTintList(ColorStateList.valueOf(Color.GREEN));
    }else{
        button.setBackgroundTintList(ColorStateList.valueOf(Color.RED));
    }
}
```

InfoActivity: Aquesta activitat es molt semblant per a tots els sensors. Per això farem l'exemple amb una activitat aleatòria, en aquest cas es el sensor de proximitat.



Per les activitats d'informació haurem d'agafar les dades d'un sensor que agafi les dades dels serveis dels sensors que hi han al sistema (SensorManager) .

També necessitem un sensor per a dir-li que volem el sensor d'un tipus en concret, en aquest cas agafem el sensor "Proximity".

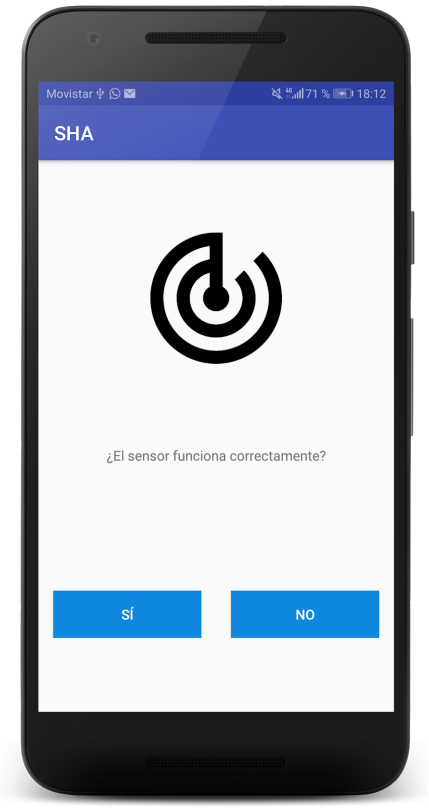
Per mostrar la informació per pantalla utilitzem un TextView i apliquem el text en format Html agafant els valors del sensor, com el nom i la resolució.
Finalment tenim el botó per iniciar el test on quan cliquem, s'inicia l'activitat del test en concret.

```
Sensor s;
SensorManager sensorM;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_proximity_info);

    Button test = (Button) findViewById(R.id.proximityTest);
    TextView tvInfo = (TextView) findViewById(R.id.info);

    sensorM = (SensorManager) getSystemService(SENSOR_SERVICE);
    s = sensorM.getDefaultSensor(Sensor.TYPE_PROXIMITY);

    tvInfo.setText(Html.fromHtml("<p><b>Nombre: </b>"+ s.getName() +"</p><p><b>Vendedor: </b>" + s.getVendor() +
            "</p><p><b>Versión: </b>"+ s.getVersion() + "</p><p><b>Energía: </b>" + s.getPower() + "</p><p><b>Rango máximo: </b>" +
            s.getMaximumRange() + "</p><p><b>Resolución: </b>" + s.getResolution()+"</p>"));

    test.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            startActivity(new Intent(ProximityInfoActivity.this, ProximityTestActivity.class));
        }
    });
}
```

L'xml de l'activitat de la informació té dos LinearLayout on el primer conté dos TextView, un amb el titòl de la activity, amb un drawable que porta l'icona del sensor, i tota la informació. A l'altre LinearLayout hi ha un botó per anar al test. Té un marge horitzontal on s'assigna amb el nom "marginGeneral" un marge de 15dp.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <TextView
        android:layout_marginTop="@dimen/marginGeneral"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Proximity"
        android:textSize="@dimen/info_title"
        android:drawableLeft="@drawable/proximity"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>

    <TextView
        android:id="@+id/info"
        android:layout_marginTop="@dimen/marginInfo"
        android:layout_marginHorizontal="@dimen/marginInfo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/proximityTest"
        android:text="Test"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

ResultActivity: Aquesta activitat es molt semblant per a tots els sensors. Per això farem l'exemple amb una activitat aleatòria, en aquest cas es el sensor de proximitat.



En el cas de l'activitat dels resultats, al codi necessitem dos botons, per indicar si funciona correctament o no el sensor al dispositiu. Tant si funciona correctament com no, en fer clic s'inicia el mètode saveResults() però depenent del botó clicat, enviarà "true" o "false" com paràmetre.

```
Button butSi, butNo;
boolean alltest;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_proximity_result);

    butSi = (Button) findViewById(R.id.butSi);
    butNo = (Button) findViewById(R.id.butNo);

    butSi.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            saveResults(true);
        }
    });

    butNo.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            saveResults(false);
        }
    });
}
```

El mètode saveResults() emmagatzema el paràmetre a la base de dades Firebase. Aquest resultat ho guarda dintre de la BBDD amb tots els altres valors dels sensors amb referència de l'id d'Android.

L'activity del test envia un paràmetre anomenat "alltest" on si es fals, torna a l'activity principal (MainActivity). En canvi, si es vertadera, vol dir que aquesta activitat s'ha iniciat pel botó de comprovar tots els sensors, aleshores l'activitat que s'inicia es el test del següent sensor.

```
void saveResults(boolean ok){
    String iid = SHAUtils.getAndroidId(this);

    FirebaseDatabase.getInstance().getReference().child(iid).child("proximity").setValue(ok);
    alltest =  getIntent().getBooleanExtra("alltest",false);

    if (alltest != true) {
        Intent main = new Intent(ProximityResultActivity.this, MainActivity.class);
        startActivity(main);
    }else {
        Intent main = new Intent(ProximityResultActivity.this, ScreenTestActivity.class);
        main.putExtra("alltest",alltest);
        startActivity(main);
    }
}
```

L’xml té dos LinearLayout, on el primer té un ImageView amb la imatge del sensor, i un TextView amb la pregunta si el sensor es correcte o no.
L’altre LinearLayout conté els dos botons amb la resposta “Si” o “No”.

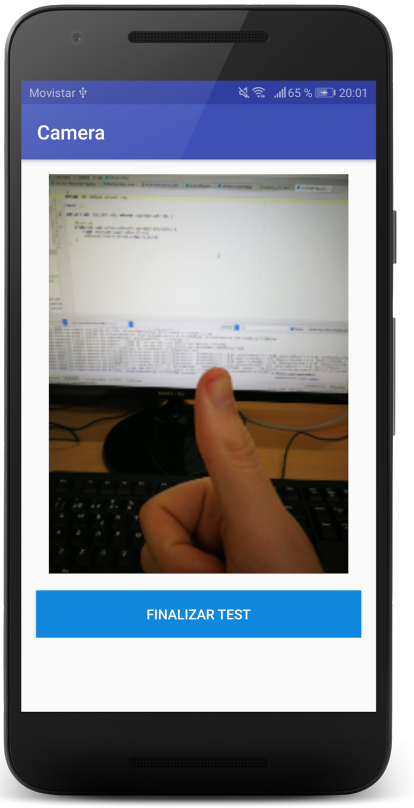
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center_horizontal"
    android:id="@+id/fondo">

    <ImageView
        android:layout_marginTop="50dp"
        android:layout_width="160dp"
        android:layout_height="180dp"
        android:src="@drawable/proximity" />

    <TextView
        android:layout_marginTop="60dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAlignment="center"
        android:text="@string/sensorResult"/>

</LinearLayout>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/butSi"
        android:text="Sí"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>

    <Button
        android:id="@+id/butNo"
        android:text="No"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

2.3.1 TestActivities

Aquí mostrarem el resultat final de cada sensor, part del codi per veure el resultat final i algun XML que sigui diferent, perquè gairebé tots tenen una estructura molt semblant .

CameraTestActivity: Quan passem de la pantalla d'informació a la del test, l'usuari el primer que veu és que s'inicia la càmera del seu mòbil automàticament on podrà fer una foto. Un cop treta la foto, la càmera es tancarà i l'usuari podrà observar la seva fotografia.



XML: Aquesta activity conté dos 'LinearLayout'. El primer per mostrar una 'ImageView' i el segon mostra el botó de finalitzar test, on passarà a la pantalla result. El segon 'LinearLayout' s'anirà repetint al llarg de totes les pantalles de test, perquè així estaran totes a la mateixa posició.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginBottom="140dp"
        android:layout_marginTop="@dimen/marginGeneral"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/testEnd"
        android:text="Finalizar test"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

Codi: Primer l'usuari necessita donar-li permisos a l'app per poder capturar una imatge. Per obrir la càmera automàticament, hem posat dintre del mètode 'onCreate()' un 'Intent' que obrís la 'MediaStore'.

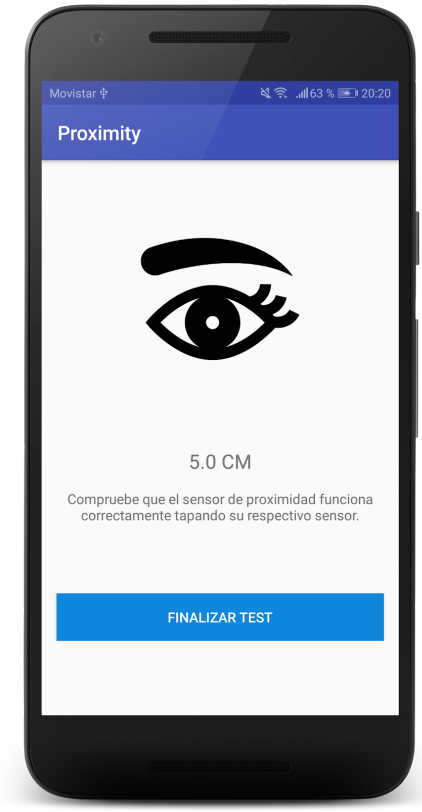
El 'CAMERA_REQUEST' s'utilitza per rebre el resultat del intent de la càmera.
El 'boolean alltest', serveix per saber si s'ha obert aquesta activity des de el 'RunAllTest' o des de la 'CameraInfo', el seu valor per defecte es false.

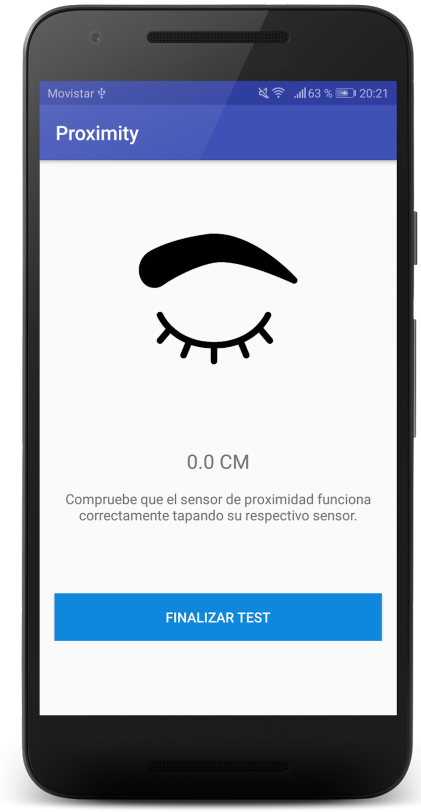
```
    ImageView imageView;
    TextView textView;
    private static final int CAMERA_REQUEST = 50;
    Button testEnd;
    boolean alltest;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_camera_test);

        textView = (TextView)findViewById(R.id.text);
        testEnd = (Button)findViewById(R.id.testEnd);
        imageView = (ImageView)findViewById(R.id.imageView);

        ActivityCompat.requestPermissions(CameraTestActivity.this, new String[] {Manifest.permission.CAMERA}, CAMERA_REQUEST);


        Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        startActivityForResult(intent,0);

        final Intent resultactivity = new Intent(CameraTestActivity.this, CameraResultActivity.class);

        alltest = getIntent().getBooleanExtra("alltest",false);

        testEnd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                resultactivity.putExtra("alltest",alltest);
                startActivity(resultactivity);

            }
        });


    }
```

Per mostrar la imatge tenim un altre mètode anomenat ‘onActivityResult()’, on rep de paràmetres ‘requestCode’ de tipus numèric, ‘resultCode’ de tipus numèric i ‘data’ de tipus intent. Aquest mètode conté el ‘Bitmap’ que serveix per generar la fotografia, i després aquest bitmap se li passa al ‘ImageView’ per mostrar-la.

```
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        Bitmap bitmap = (Bitmap)data.getExtras().get("data");
        imageView.setImageBitmap(bitmap);
    }
}
```

ProximityTestActivity: L'usuari el primer que veu es una imatge d'un ull obert, un text on se li indica el que ha de fer, un altre text per indicar la longitud de proximitat que te el sensor i un botó per finalitzar el test. Cada cop que es tapa o arriba a longitud indicada del sensor del dispositiu mòbil, la imatge del ull canvia per un ull tancat i el text de la longitud passa a ser 0 cm.





XML: Aquest xml conté dos 'LinearLayout'. El primer té un 'ImageView' i dos 'TextViews'. El segon té el botó de finalitzar test. A partir d'aquí gairebé totes les pantalles tenen un XML semblant amb algun canvi minúscul.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center_horizontal">

    <ImageView
        android:id="@+id/fondo"
        android:layout_marginTop="50dp"
        android:layout_width="160dp"
        android:layout_height="180dp"
        android:src="@drawable/eye_open"/>

    <TextView
        android:id="@+id/text1"
        android:layout_marginTop="60dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="20dp"
        android:text="exemple"
        android:textAlignment="center"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/marginGeneral"
        android:textAlignment="center"
        android:text="@string/proximityTest"/>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/testEnd"
        android:text="Finalizar test"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

Codi: Aquesta classe implementa de ‘SensorEventListener’, això fa que necessiti certs mètodes per funcionar, com el 'onAccuracyChanged' i el 'onSensorChanged'.

Per fer funcionar el sensor, necessitarem l’ajuda del ‘Sensor’ i del ‘SensorManager’.

El ‘SensorManager’ serveix per obtenir els serveis dels sensors que hi han al sistema. I amb això, li podem dir al ‘Sensor’ que sigui d’un tipus en concret, en aquest cas que sigui de tipus proximitat.

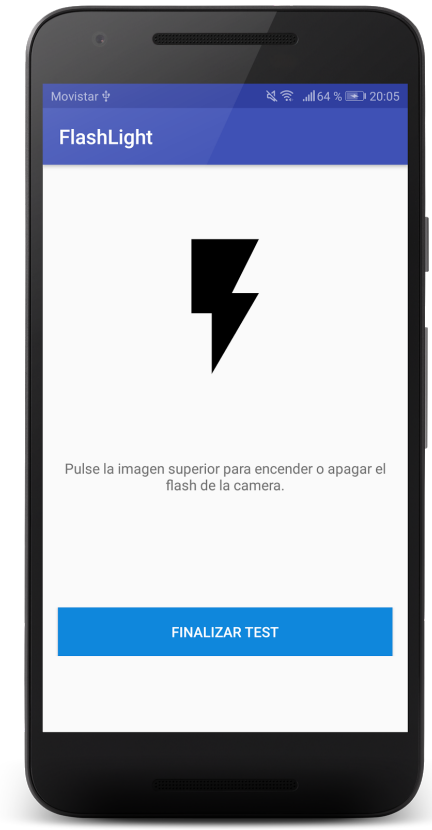
```
public class ProximityTestActivity extends AppCompatActivity implements SensorEventListener{

    ImageView fondo;
    Sensor s;
    SensorManager sensorM;
    TextView prox;
    Button testEnd;
    boolean alltest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_proximity_test);

        fondo = (ImageView) findViewById(R.id.fondo);
        prox = (TextView) findViewById(R.id.text1);
        testEnd = (Button) findViewById(R.id.testEnd);

     sensorM = (SensorManager) getSystemService(SENSOR_SERVICE);
     s = sensorM.getDefaultSensor(Sensor.TYPE_PROXIMITY);
     sensorM.registerListener(ProximityTestActivity.this,s,
             SensorManager.SENSOR_DELAY_NORMAL);

         final Intent resultactivity = new Intent(ProximityTestActivity.this, ProximityResultActivity.class);

         alltest = getIntent().getBooleanExtra("alltest",false);

         testEnd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                resultactivity.putExtra("alltest",alltest);
                startActivity(new Intent(resultactivity));
            }
        });

     }
```

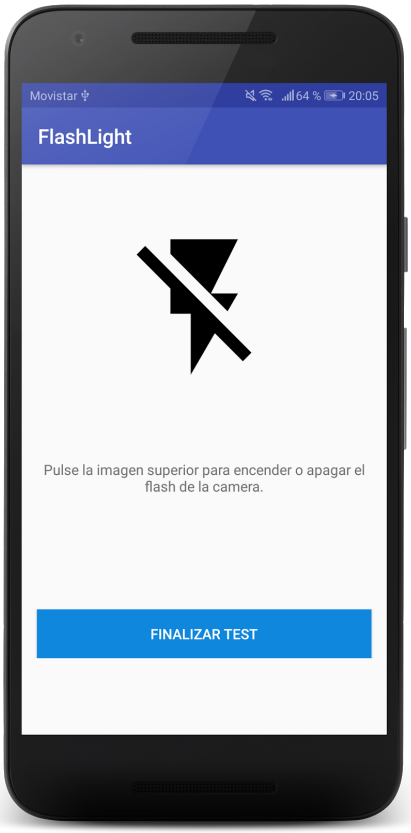
A partir d’aquí entra en acció el ‘onSensorChanged’, que serveix per detectar si el sensor ha canviat de valor o no. Aquí dins fiquem un condicional on si el valor del sensor es menor del valor màxim, el text canvia per 0.0 CM i la imatge canvia per la de un ull tancat. Si això no es compleix, la imatge seguirà sent la del ull obert i el text serà la del valor màxim per a cada dispositiu mòbil.

El ‘onAccuracyChanged’ s'invoca cada vegada que un sensor informa amb diferent precisió. Però això a nosaltres no ens feia falta i per això està buit.

```
@Override
public void onAccuracyChanged(Sensor arg0, int arg1) {

}

@Override
public void onSensorChanged(SensorEvent evento) {

    if (evento.values[0] < s.getMaximumRange()) {
        fondo.setImageResource(R.drawable.eye_close);
        prox.setText("0.0 CM");
    }else {
        fondo.setImageResource(R.drawable.eye_open);
        prox.setText(s.getMaximumRange() + " CM");
    }
}
```

FlashLightTestActivity: Aquesta activity l'usuari el que veu es una imatge d'un llampeg, representat el flash de la càmera, que quan cliques a sobre, la imatge canvia i el flash del mòbil s'activa. També te un text on indica al usuari el que ha de fer.





XML: Aquí no mostraré el codi perquè es quasi idèntic al anterior. Son dos 'LinearLayout', el primer conté un 'ImageView' i un 'TextView'. El segon conté el botó de finalitzar.

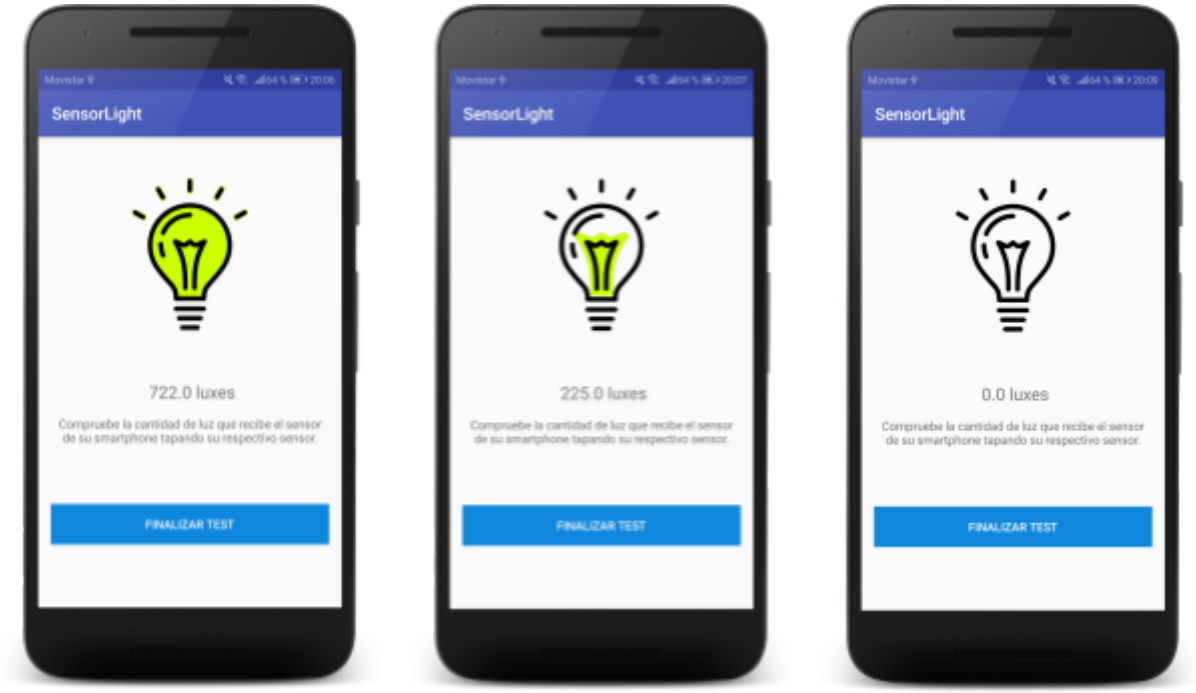
Codi: Tenim dos mètodes anomenats 'flashLightOn()' i flashLightOff()'. El primer mètode serveix per encendre el flash i el segon per apagar-ho. Els dos mètodes utilitzen el 'CameraManager' que serveix per utilitzar el serveis de la càmera. I dins dels dos mètodes es on es diu que canviï la imatge.

```
private void flashLightOn() {
    CameraManager cameraManager = (CameraManager) getSystemService(Context.CAMERA_SERVICE);

    try {
        String cameraId = cameraManager.getCameraIdList()[0];
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            cameraManager.setTorchMode(cameraId, true);
        }
        flashLightStatus = true;
        imageFlashlight.setImageResource(R.drawable.flash);
    } catch (CameraAccessException e) {

    }
}

private void flashLightOff() {
    CameraManager cameraManager = (CameraManager) getSystemService(Context.CAMERA_SERVICE);

    try {
        String cameraId = cameraManager.getCameraIdList()[0];
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            cameraManager.setTorchMode(cameraId, false);
        }
        flashLightStatus = false;
        imageFlashlight.setImageResource(R.drawable.flash_off);
    } catch (CameraAccessException e) {
        //res de res
    }
}
```

Dins del ‘onCreate()’, tenim un booleà per saber si el dispositiu té flash o no. Després es comprova si té o no amb un condicional. Si té, primer el que fa es demanar permisos i després executa el ‘flashLightOn()’, i si no té, surt un missatge avisant al usuari.

```
private ImageView imageFlashlight;
private static final int CAMERA_REQUEST = 50;
private boolean flashLightStatus = false;
Button testEnd;
boolean alltest;



@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_flashlight_test);
    imageFlashlight = (ImageView) findViewById(R.id.imageFlashlight);
    testEnd = (Button) findViewById(R.id.testEnd);

    final boolean hasCameraFlash = getPackageManager().
            hasSystemFeature(PackageManager.FEATURE_CAMERA_FLASH);


    ActivityCompat.requestPermissions(FlashlightTestActivity.this, new String[] {android.Manifest.permission.CAME

    imageFlashlight.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (hasCameraFlash) {
                if (flashLightStatus)
                    flashLightOff();
                else
                    flashLightOn();
            } else {
                Toast.makeText(FlashlightTestActivity.this, "No flash available on your device",
                        Toast.LENGTH_SHORT).show();
            }
        }
    });

    final Intent resultactivity = new Intent(FlashlightTestActivity.this, FlashLightResultActivity.class);

    alltest = getIntent().getBooleanExtra("alltest",false);

    testEnd.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            resultactivity.putExtra("alltest",alltest);
            startActivity(resultactivity);

        }
    });
}
```

SensorLightTestActivity: Quan s'obre l'activity, l'usuari el que veu es una imatge d'una bombeta encesa del tot, que segons la quantitat de llum que li arribi al sensor, la intensitat de la bombeta canviarà. També veurà la quantitat exacte de llum en luxes i un missatge indicant al usuari el que ha de fer.



XML: Dos 'LinearLayout', el primer per la imatge i els dos 'TextView', el segon per el botó de finalitzar.

Codi: Al ser un sensor, la classe implementa de 'SensorEventListener', així que necessita el mètode 'onAccuracyChanged' i el 'onSensorChanged'. També hem necessitat el mètode 'onResume' i el 'onPause'.

Un altre cop utilitzem el 'Sensor' i el 'SensorManager'. Aquí hem fet que el tipus del 'Sensor' sigui de 'TYPE_LIGHT'.

```
private SensorManager mSensorManager;
private Sensor mLight;
TextView textView;
ImageView imageView;
Button testEnd;
boolean alltest;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_sensor_light_test);
    imageView = (ImageView)findViewById(R.id.imageSensor);
    textView = (TextView)findViewById(R.id.text1);
    // Get an instance of the sensor service, and use that to get an instance of
    // a particular sensor.
    mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
    mLight = mSensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
    testEnd = (Button)findViewById(R.id.testEnd) ;

    final Intent resultactivity = new Intent(SensorLightTestActivity.this, SensorLightResultActivity.class);

    alltest = getIntent().getBooleanExtra("alltest",false);

    testEnd.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            resultactivity.putExtra("alltest",alltest);
            startActivity(resultactivity);
        }
    });
}
```

En el mètode ‘onAccuracyChanged’ esta buit. El ‘onResume’ serveix per registrar un detector per al sensor. En el ‘onPause’, s’assegura d’anular el registre del sensor quan l’activitat es detingui. El ‘onSensorChanged’ es el mètode mes important, detecta el valor que li arriba al sensor i segons el valor del sensor la imatge anirà canviant.

```java
@Override
public final void onAccuracyChanged(Sensor sensor, int accuracy) {
    // Do something here if sensor accuracy changes.
}

@Override
public final void onSensorChanged(SensorEvent event) {
    if(event.sensor.getType() == Sensor.TYPE_LIGHT){
        textView.setText(" " + event.values[0] + " luxes");

        if (event.values[0] <= 1){
            imageView.setImageResource(R.drawable.light1);
        } else if (event.values[0] < 100 && event.values[0] > 1 ){
            imageView.setImageResource(R.drawable.light2);
        } else if (event.values[0] < 200 && event.values[0] > 100 ){
            imageView.setImageResource(R.drawable.light3);
        } else if (event.values[0] < 300 && event.values[0] > 200 ){
            imageView.setImageResource(R.drawable.light4);
        }else if (event.values[0] < 400 && event.values[0] > 300 ){
            imageView.setImageResource(R.drawable.light5);
        }else if (event.values[0] < 500 && event.values[0] > 400 ){
            imageView.setImageResource(R.drawable.light6);
        }else {
            imageView.setImageResource(R.drawable.light7);
        }
    }
}

@Override
protected void onResume() {
    // Register a listener for the sensor.
    super.onResume();
    mSensorManager.registerListener(this, mLight, SensorManager.SENSOR_DELAY_NORMAL);
}

@Override
protected void onPause() {
    // Be sure to unregister the sensor when the activity pauses.
    super.onPause();
    mSensorManager.unregisterListener(this);
}
```

GyroscopeTestActivity: Aquí el que es pot veure es una imatge, un text on es diu a l'usuari el que ha de fer i un botó per iniciar el sensor. Un cop es clica, el valor del text canviara per el valor que dona el sensor, X, Y i Z. El botó s'amagarà. I cada cop que el mòbil es mou de dreta a esquerra, el fons de la imatge canvia de color indicant de que el sensor funciona bé.

XML: Com tots els altres XML's, son dos 'LinearLayout'. En el primer es veu la imatge, el 'TextView' i el botó. Al segon només està el botó de finalitzar test.

Codi: Al ser un sensor, la classe implementa de 'SensorEventListener', així que necessita el mètode 'onAccuracyChanged' i el 'onSensorChanged'.

Un altre cop utilitzem el 'Sensor' i aques cop dos 'SensorManager'. Aquí hem fet que el tipus del primer 'Sensor' sigui de 'TYPE_GYROSCOPE' i el segon de 'TYPE_ROTATION'. Quan es clica al botó el sensor es posarà en marxa.

```
private SensorManager sensorManager;
private Sensor gyroscopeSensor, rotationVectorSensor;
Button activador, testEnd;
TextView textGyroscope;
ImageView fondo;
boolean alltest;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_gyroscope_test);

    fondo = (ImageView) findViewById(R.id.fondo);
    textGyroscope =  (TextView) findViewById(R.id.textGyroscope);
    activador = (Button) findViewById(R.id.activador);
    testEnd = (Button) findViewById(R.id.testEnd);
    sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);

    activador.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View arg0) {

            gyroscopeSensor = sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
            rotationVectorSensor = sensorManager.getDefaultSensor(Sensor.TYPE_ROTATION_VECTOR);

            sensorManager.registerListener(GyroscopeTestActivity.this, gyroscopeSensor,
                    SensorManager.SENSOR_DELAY_NORMAL);
```

També tenim dos mètodes ‘onSensorChanged’, un dins del ‘onCreate’ per administrar el sensor de rotació i treure els valors del sensor que son els que es mostren. Es creen tres arrays de floats, i el que ens interessa es l’array ‘orientation’, que es el que dona els valors de la posició de la pantalla.

```
        SensorEventListener rvListener = new SensorEventListener() {
            @Override
            public void onSensorChanged(SensorEvent sensorEvent) {
                // More code goes here
                float[] rotationMatrix = new float[16];
                SensorManager.getRotationMatrixFromVector(
                        rotationMatrix, sensorEvent.values);

                // Remap coordinate system
                float[] remappedRotationMatrix = new float[16];
                SensorManager.remapCoordinateSystem(rotationMatrix,
                        SensorManager.AXIS_X,
                        SensorManager.AXIS_Z,
                        remappedRotationMatrix);

                // Convert to orientations
                float[] orientations = new float[3];
                SensorManager.getOrientation(remappedRotationMatrix, orientations);
                for(int i = 0; i < 3; i++) {
                    orientations[i] = (float)(Math.toDegrees(orientations[i]));
                }

                int px = Math.round(orientations[0]);
                int py = Math.round(orientations[1]);
                int pz = Math.round(orientations[2]);

                textGyroscope.setText("X: " + px + ", Y: " + py + ", Z: " + pz);

            }
            @Override
            public void onAccuracyChanged(Sensor sensor, int i) {
            }
        };
        sensorManager.registerListener(rvListener,
                rotationVectorSensor, SensorManager.SENSOR_DELAY_NORMAL);
    }
});
```

L''altre mètode està fora i serveix per canviar el fons de la imatge segons el valor que dona el sensor del giroscopi.

```
            @Override
            public void onSensorChanged(SensorEvent sensorEvent) {
                // More code goes here
                if(sensorEvent.values[2] > 0.5f) { // anticlockwise
                    fondo.setBackgroundColor(Color.MAGENTA);
                } else if(sensorEvent.values[2] < -0.5f) { // clockwise
                    fondo.setBackgroundColor(Color.CYAN);
                }
            }
```

MagneticTestActivity: Quan s'obre l'activity, l'usuari el que veu es una imatge d'una brújula i dos textes. Un d'ells porta el compte dels microTesla(μT) i a l'altre text observem el valor X,Y i Z.

XML: Aquest xml porta cuatra 'LinearLayout'. En el primer es veu la imatge, el 'TextView'. Al segon només està el valor del camp magnetic del dispositiu. El tercer mostra el valor tridimensional (X,Y i Z). Porta tres 'TextView' per mostrar en cada un els valors.

Entre el tercer 'LinearLayout' y el quart, hi ha un 'TextView' que mostra com pots fer la comprovació que el sensor funciona correctament.
Per ultim, el quart 'LinearLayout' mostra el botó per avançar y finalitzar el test.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center_horizontal">

    <ImageView
        android:id="@+id/battery"
        android:layout_marginTop="50dp"
        android:layout_width="160dp"
        android:layout_height="180dp"
        android:src="@drawable/magnetic" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="@dimen/marginGeneral"
    android:orientation="vertical">

    <TextView
        android:id="@+id/teslaTxt"
        android:layout_width="wrap_content"
        android:text="Tesla"
        android:layout_height="wrap_content"
        android:textSize="20dp"
        android:layout_gravity="center"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center"
        android:layout_marginTop="5dp">

        <TextView
            android:id="@+id/x"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="16dp"
            android:text="1"
            android:layout_gravity="center"/>

        <TextView
            android:id="@+id/y"
            android:layout_width="wrap_content"
            android:layout_marginHorizontal="@dimen/marginGeneral"
            android:layout_height="wrap_content"
            android:textSize="16dp"
            android:text="1"
            android:layout_gravity="center"/>

        <TextView
            android:id="@+id/z"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:text="1"
            android:textSize="16dp" />

    </LinearLayout>

</LinearLayout>
```

```
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAlignment="center"
        android:text="@string/magneticTest"
        android:layout_marginHorizontal="@dimen/marginGeneral" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/testEnd"
        android:text="Finalizar test"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

Codi: Aquesta classe implementa de ‘SensorEventListener’, això fa que necessiti certs mètodes per funcionar, com el 'onAccuracyChanged' i el 'onSensorChanged'. Per fer funcionar el sensor, necessitarem l’ajuda del ‘Sensor’ i del ‘SensorManager’.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_magnetic_test);

    x = (TextView) findViewById(R.id.x);
    y = (TextView) findViewById(R.id.y);
    z = (TextView) findViewById(R.id.z);
    teslatxt = (TextView) findViewById(R.id.teslaTxt);
    testEnd = (Button) findViewById(R.id.testEnd);

    sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
    sensor = sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
    sensorManager.registerListener(MagneticTestActivity.this,sensor,SensorManager.SENSOR_DELAY_NORMAL);

    final Intent resultactivity = new Intent(MagneticTestActivity.this, MagneticResultActivity.class);

    alltest = getIntent().getBooleanExtra("alltest",false);

    testEnd.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            resultactivity.putExtra("alltest",alltest);
            startActivity(new Intent(resultactivity));
        }
    });
}
```

El ‘onSensorChanged’, que serveix per detectar si el sensor ha canviat de valor o no. Dintre d’aquest mètode utilitzem el paràmetre ‘SensorEvent’, que agafa els valors del sensor.

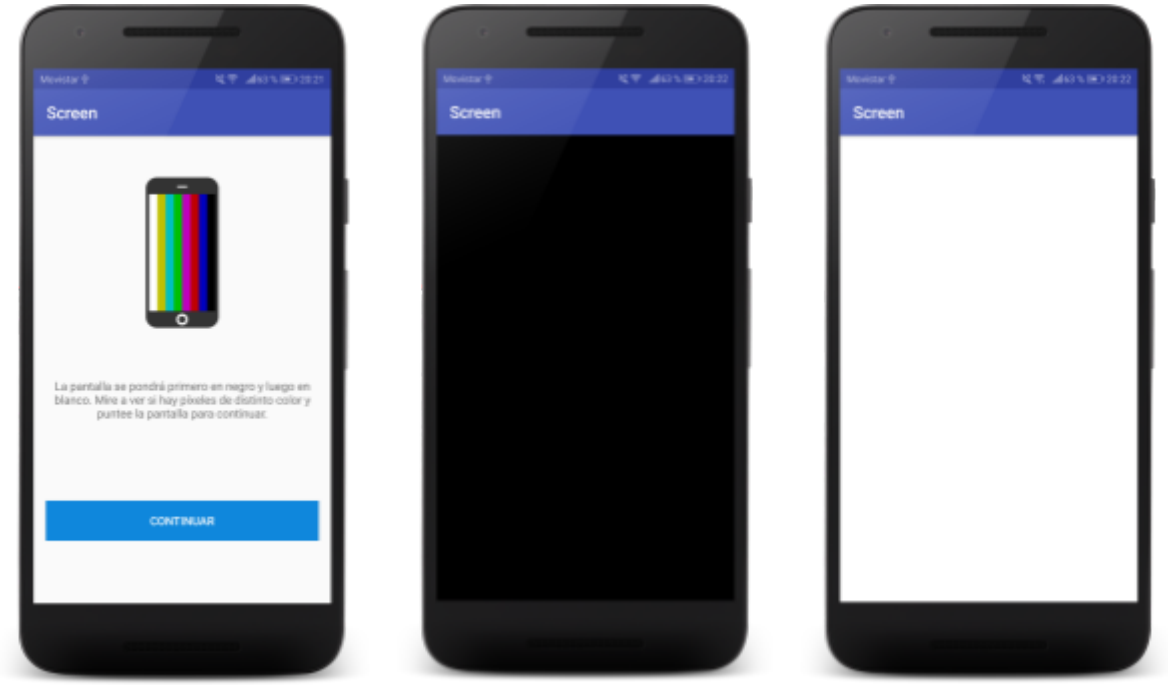
En aquest cas, es un ‘Array’ on el primer valor es la variable X, el segon es la variable Y i per ultim, el tercer valor es la Z.

Per donar-li el valor correcte al text que controla els µT que rep el sensor, utilitzem una formula matemàtica per calcular-la.

Finalment, el resultat d’aquesta formula la assignem al ‘TextView’.

```
@Override
public void onSensorChanged(SensorEvent sensorEvent) {

    float px = Math.round(sensorEvent.values[0]);
    float py = Math.round(sensorEvent.values[1]);
    float pz = Math.round(sensorEvent.values[2]);

    x.setText("X: " + String.valueOf(px));
    y.setText("Y: " + String.valueOf(py));
    z.setText("Z: " + String.valueOf(pz));

    double tesla = Math.round(Math.sqrt((px * px)+(py * py) + (pz * pz)));
    teslatxt.setText(String.format(tesla + " µT"));


}
```

ScreenTestActivity: Això no es un sensor, però serveix per garantir que la pantalla no tingui cap píxel mort. L'usuari haurà de seguir les instruccions indicades a un text que hi ha. El que ha de fer es clicar al botó per començar i tota la pantalla es posarà en negre, després quan cliqui a qualsevol lloc de la pantalla el fons canviarà a blanc, i després clicant un altre cop el test finalitzarà.

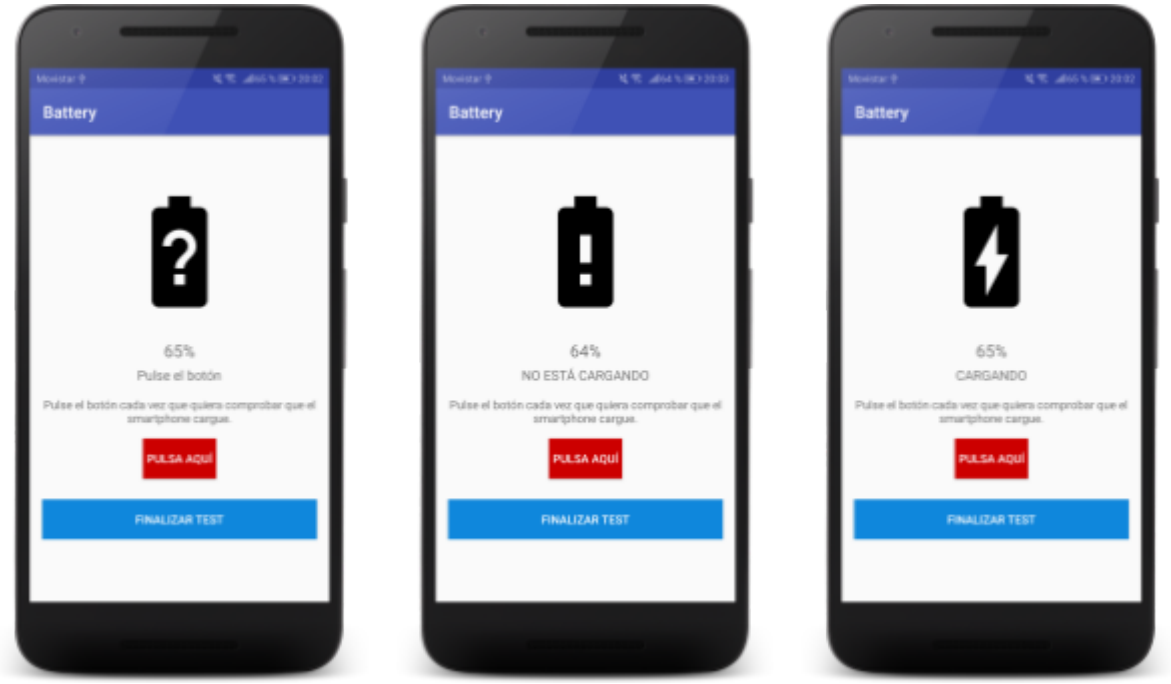


XML: Dos 'LinearLayouts' dins d'un 'ConstraintLayout', el primer conté una imatge i un 'TextView', el segon conté el botó de continuar. Quan es va clicant la pantalla el 'ConstraintLayout' canvia el fons i els altres elements anteriors s'oculten.

Codi: El codi es molt senzill, hi ha quatre 'setOnClickListeners'. El primer oculta tots els elements i posa el fons del 'ConstraintLayout' en negre. El segon posa el fons de color blanc. El tercer mostra la imatge, el text i el botó que es veia a la primera screen. El quart serveix per canviar al resultActivity.

```java
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_screen_test);
    phone = (ImageView) findViewById(R.id.phone);
    text1 = (TextView) findViewById(R.id.text1);
    continuar = (Button) findViewById(R.id.continuar);
    fondo = (LinearLayout) findViewById(R.id.fondo);

    continuar.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            phone.setVisibility(View.INVISIBLE);
            text1.setVisibility(View.INVISIBLE);
            continuar.setVisibility(View.INVISIBLE);

            fondo.setBackgroundColor(Color.BLACK);

            fondo.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    fondo.setBackgroundColor(Color.WHITE);

                    fondo.setOnClickListener(new View.OnClickListener() {
                        @Override
                        public void onClick(View view) {
                            phone.setVisibility(View.VISIBLE);
                            text1.setText("Pulse el botón para finalizar el test.");
                            text1.setVisibility(View.VISIBLE);

                            continuar.setText("Finalizar test");
                            continuar.setVisibility(View.VISIBLE);

                            final Intent resultactivity = new Intent(ScreenTestActivity.this, ScreenResultActivity.class);

                            alltest = getIntent().getBooleanExtra("alltest",false);

                            continuar.setOnClickListener(new View.OnClickListener() {
                                @Override
                                public void onClick(View view) {
                                    resultactivity.putExtra("alltest",alltest);
                                    startActivity(new Intent(resultactivity));
                                }
                            });
                        }
                    });
                }
            });
        }
    });
}
```

BatteryTestActivity: Quan s'obre l'activity, l'usuari el que veu es una imatge d'una bateria,tres text i dos botons. La imatge va canviant depenent si està conectat o no el dispositiu a l'electricitat.



El primer text es el percentatge de carga del dispositiu. El segon text mostra si se està carregant o no. L'ultim dona la instrucció per comprovar si funciona bé el sensor.

El botó principal es el de 'Pulsa aquí'. Fent clic a aquest botó comprova si carrega o no el dispositiu.

Finalment hi es el botó per finalitzar el text.

XML: Conté tres 'LinearLayout', on el primer mostra la imatge de la bateria, que canvia depenent del estat de carga.

El segon conté el percentatge de carga i el text que varia mostrant si carrega o no.

Entre el segon y el tercer layout hi ha un text i un botó. El text mostra la instrucció i el botó es el comprovant del sensor.

L'ultim 'LinearLayout' mostra el botó per avançar a la següent activitat.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center_horizontal">

    <ImageView
        android:id="@+id/battery"
        android:layout_marginTop="50dp"
        android:layout_width="160dp"
        android:layout_height="180dp"
        android:src="@drawable/battery" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/marginGeneral"
        android:orientation="vertical">

        <TextView
            android:id="@+id/textBattery"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="20dp"
            android:text="1"
            android:layout_gravity="center"/>

        <TextView
            android:id="@+id/textCharge"
            android:layout_width="wrap_content"
            android:text="Pulse el botón"
            android:layout_height="wrap_content"
            android:textSize="16dp"
            android:layout_gravity="center"
            android:layout_marginTop="5dp"/>

    </LinearLayout>



    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAlignment="center"
        android:text="@string/batteryTest"
        android:layout_marginHorizontal="@dimen/marginGeneral" />

    <Button
        android:id="@+id/checkBattery"
        android:text="Pulsa aquí"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/marginGeneral"
        android:background="@android:color/holo_red_dark"
        android:textColor="#ffffff"/>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/testEnd"
        android:text="Finalizar test"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

Codi: Com tots els tests anteriors, identifiquem els botons i texts amb la id del xml. Després, tenim els dos botons on ‘checkBattery’ canvia el text segons la carrega del dispositiu quan pulsem el botó. L’altre botó es per finalitzar el test i anar cap a la activitat del resultat.

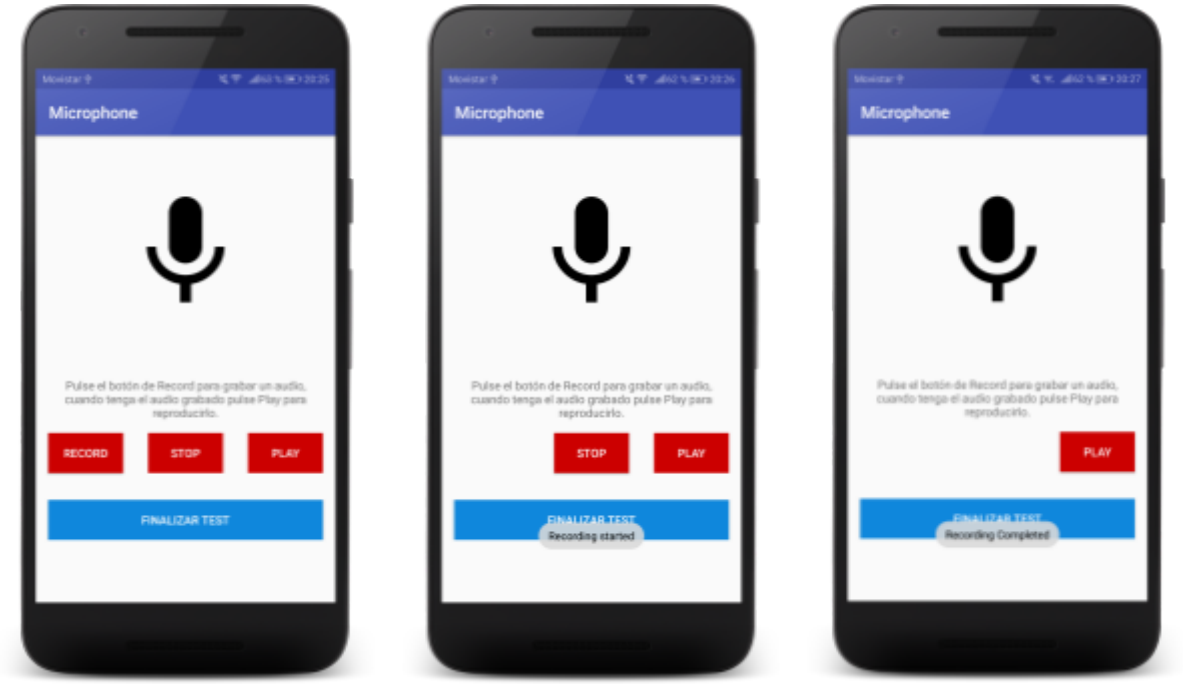
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_battery_test);

    checkBattery = (Button) this.findViewById(R.id.checkBattery);
    textBattery = (TextView) this.findViewById(R.id.textBattery);
    textCharge = (TextView) this.findViewById(R.id.textCharge);
    battery = (ImageView) this.findViewById(R.id.battery);
    testEnd = (Button) this.findViewById(R.id.testEnd);

    this.registerReceiver(this.mBatInfoReceiver, new IntentFilter(Intent.ACTION_BATTERY_CHANGED));

    checkBattery.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            checkBatteryState(textCharge);
        }
    });

    final Intent resultactivity = new Intent(BatteryTestActivity.this, BatteryResultActivity.class);

    alltest = getIntent().getBooleanExtra("alltest",false);

    testEnd.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            resultactivity.putExtra("alltest",alltest);
            startActivity(resultactivity);
        }
    });

}
```

Tenim una clase de tipus ‘BroadcastReceiver’ on calcula el percentatge de bateria i el posa al ‘TextView’.

Si observem el codi anterior, veiem que utilitza el métode ‘checkBatteryState()’, que depenent si está conectat al carregador, mostrará un text o un altre.

```
private BroadcastReceiver mBatInfoReceiver = new BroadcastReceiver(){
    @Override
    public void onReceive(Context ctxt, Intent intent) {
        int level = intent.getIntExtra(BatteryManager.EXTRA_LEVEL, 0);
        textBattery.setText(String.valueOf(level) + "%");

        if(intent.getExtras() != null){
            String tec = intent.getExtras().getString(BatteryManager.EXTRA_TECHNOLOGY);
        }
    }
};

public void checkBatteryState(View sender) {
    IntentFilter filter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
    Intent batteryStatus = registerReceiver(null, filter);

    int chargeState = batteryStatus.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
    String strState;

    switch (chargeState) {
        case BatteryManager.BATTERY_STATUS_CHARGING:
        case BatteryManager.BATTERY_STATUS_FULL:
            strState = "CARGANDO";
            battery.setImageResource(R.drawable.battery_charge);
            break;
        default:
            strState = "NO ESTÁ CARGANDO";
            battery.setImageResource(R.drawable.battery_alert );
    }

    textCharge.setText(strState);
}
```

MicrophoneTestActivity: L'usuari quan entra el primer que veu es una imatge, un text on diu el que ha de fer i tres botons, un per gravar, un per parar la gravació i un altre per escoltar l'àudio. En cas de que vulgui clicar al de parar gravació o la de reproduir, no podrà perquè els botons estan desactivats. I quan clica al de gravar, el botó desapareix. Igual que quan clica al de parar.



XML: Aquest XML conté tres 'LinearLayout', el primer conté la imatge, el text i un altre 'LinearLayout', aquest es horitzontal i té els tres botons.
El tercer conté el botó de finalitzar test. Aquest es el codi:

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center_horizontal">

    <ImageView
        android:layout_marginTop="50dp"
        android:layout_width="160dp"
        android:layout_height="180dp"
        android:src="@drawable/mic" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp"
        android:textAlignment="center"
        android:text="@string/microTest"
        android:layout_marginHorizontal="@dimen/marginGeneral" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal"
        android:layout_marginBottom="@dimen/bottom_layout">

        <Button
            android:id="@+id/record"
            android:text="Record"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:layout_margin="@dimen/marginGeneral"
            android:background="@android:color/holo_red_dark"
            android:textColor="#ffffff"/>

        <Button
            android:id="@+id/record1"
            android:text="Stop"
            android:layout_width="0dp"
            android:layout_weight="1"
            android:layout_height="wrap_content"
            android:layout_margin="@dimen/marginGeneral"
            android:background="@android:color/holo_red_dark"
            android:textColor="#ffffff"/>

   <Button
       android:id="@+id/record2"
       android:text="Play"
       android:layout_width="0dp"
       android:layout_weight="1"
       android:layout_height="wrap_content"
       android:layout_margin="@dimen/marginGeneral"
       android:background="@android:color/holo_red_dark"
       android:textColor="#ffffff"/>

</LinearLayout>
```

Codi: Aquest es dels codis mes extensos que tenim. Tenim els mètode ‘onCreate’, ‘MediaRecorderReady’, ‘CreateRandomAudioFileName’, ‘requestPermission’, ‘onRequestPermissionsResult’ i el ‘checkPermission’.

En el onCreate tenim el codi per iniciar els botons. El primer que veiem es el de finalitzar test. El més important d’aquesta classe es el MediaRecorder, que serveix per gravar.

```
Button buttonStart, buttonStop, buttonPlayLastRecordAudio, testEnd;
String AudioSavePathInDevice = null;
MediaRecorder mediaRecorder ;
Random random ;
String RandomAudioFileName = "ABCDEFGHIJKLMNOP";
public static final int RequestPermissionCode = 1;
MediaPlayer mediaPlayer ;
boolean alltest;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_microphone_test);

    testEnd = (Button) findViewById(R.id.testEnd);

    final Intent resultactivity = new Intent(MicrophoneTestActivity.this, MicrophoneResultActivity.class);

    alltest = getIntent().getBooleanExtra("alltest",false);

    testEnd.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(mediaPlayer == null){

            }else {
                mediaPlayer.pause();
            }

            resultactivity.putExtra("alltest",alltest);
            startActivity(new Intent(resultactivity));
        }
    });

    buttonStart = (Button) findViewById(R.id.record);
    buttonStop = (Button) findViewById(R.id.record1);
    buttonPlayLastRecordAudio = (Button) findViewById(R.id.record2);

    buttonStop.setEnabled(false);
    buttonPlayLastRecordAudio.setEnabled(false);

    random = new Random();
```

- ‘buttonStart’: El primer que fa es comprovar si estan els permisos donats, si ho estan se li donara una ruta on guardar l’audio, i després començarà a grabar.

```
buttonStart.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        if(checkPermission()) {

            AudioSavePathInDevice =
                    Environment.getExternalStorageDirectory().getAbsolutePath() + "/" +
                            CreateRandomAudioFileName(5) + "AudioRecording.3gp";

            MediaRecorderReady();

            try {
                mediaRecorder.prepare();
                mediaRecorder.start();
            } catch (IllegalStateException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }

            buttonStart.setEnabled(false);
            buttonStop.setEnabled(true);

            buttonStart.setVisibility(View.INVISIBLE);

            Toast.makeText(MicrophoneTestActivity.this, "Recording started",
                    Toast.LENGTH_LONG).show();
        } else {
            requestPermission();
        }

    }
});
```

-'buttonStop': Quan es clica es pararà la gravació amb el media.stop.

-'buttonPlay': Utilitza el mediaPlayer que serveix per escoltar el audio. I com es veu agafa el audio de la ruta abans guardada.

```
buttonStop.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        mediaRecorder.stop();
        buttonStop.setEnabled(false);
        buttonPlayLastRecordAudio.setEnabled(true);
        buttonStart.setEnabled(true);

        Toast.makeText(MicrophoneTestActivity.this, "Recording Completed",
                Toast.LENGTH_LONG).show();

        buttonStop.setVisibility(View.INVISIBLE);
    }
});

buttonPlayLastRecordAudio.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) throws IllegalArgumentException,
            SecurityException, IllegalStateException {

        buttonStop.setEnabled(false);
        buttonStart.setEnabled(false);

        mediaPlayer = new MediaPlayer();
        try {
            mediaPlayer.setDataSource(AudioSavePathInDevice);
            mediaPlayer.prepare();
        } catch (IOException e) {
            e.printStackTrace();
        }

        mediaPlayer.start();
        Toast.makeText(MicrophoneTestActivity.this, "Recording Playing",
                Toast.LENGTH_LONG).show();
    }
});
```

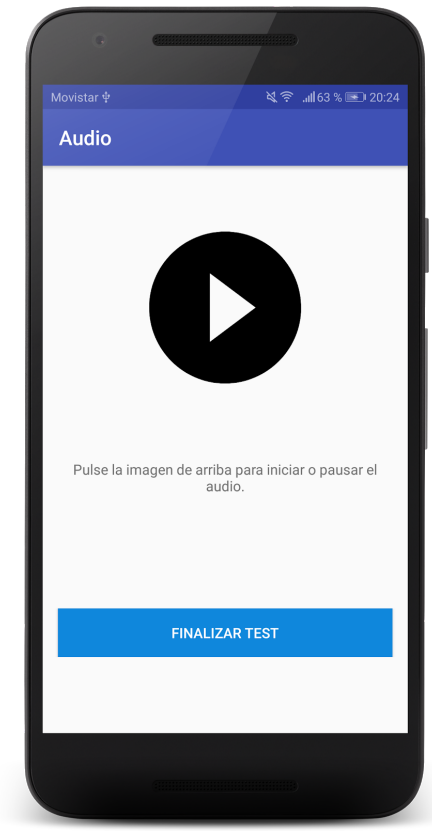
El ‘MediaRecorderReady’, es el gravador multimèdia. Agafa el micròfon i li dona format.

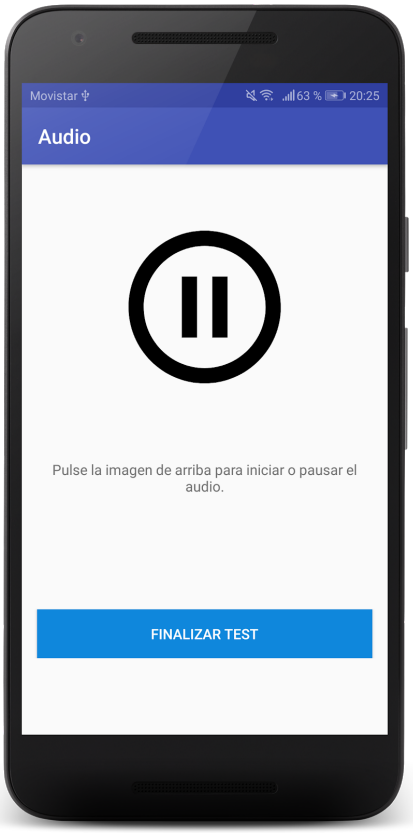
El ‘CreateRandomAudioFileName’, genera un nom aleatori a partir d’un numero i unes lletres ja escrites a dalt. Fa un random.

```
public void MediaRecorderReady(){
    mediaRecorder=new MediaRecorder();
    mediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
    mediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
    mediaRecorder.setAudioEncoder(MediaRecorder.OutputFormat.AMR_NB);
    mediaRecorder.setOutputFile(AudioSavePathInDevice);
}

public String CreateRandomAudioFileName(int string){
    StringBuilder stringBuilder = new StringBuilder( string );
    int i = 0 ;
    while(i < string ) {
        stringBuilder.append(RandomAudioFileName.
                charAt(random.nextInt(RandomAudioFileName.length())));

        i++ ;
    }
    return stringBuilder.toString();
}

private void requestPermission() {
    ActivityCompat.requestPermissions(MicrophoneTestActivity.this, new
            String[]{WRITE_EXTERNAL_STORAGE, RECORD_AUDIO}, RequestPermissionCode);
}

@Override
public void onRequestPermissionsResult(int requestCode,
                                       String permissions[], int[] grantResults) {
    switch (requestCode) {
        case RequestPermissionCode:
            if (grantResults.length> 0) {
                boolean StoragePermission = grantResults[0] ==
                        PackageManager.PERMISSION_GRANTED;
                boolean RecordPermission = grantResults[1] ==
                        PackageManager.PERMISSION_GRANTED;

                if (StoragePermission && RecordPermission) {
                    Toast.makeText(MicrophoneTestActivity.this, "Permission Granted",
                            Toast.LENGTH_LONG).show();
                } else {
                    Toast.makeText(MicrophoneTestActivity.this,"Permission Denied",Toast.LENGTH_LONG).show();
                }
            }
            break;
    }
}
```

El ‘requestPermission’ serveix per demanar permisos a l’usuari. I el ‘onRequestPermissionsResult’ serveix per poder guardar l’àudio dins del telèfon.

AudioTestActivity: Aquesta activity l'usuari el que veu es una imatge d'un play, representat el inici de la cançó, que quan cliques a sobre, la imatge canvia i la canció para. També te un text on indica al usuari el que ha de fer.





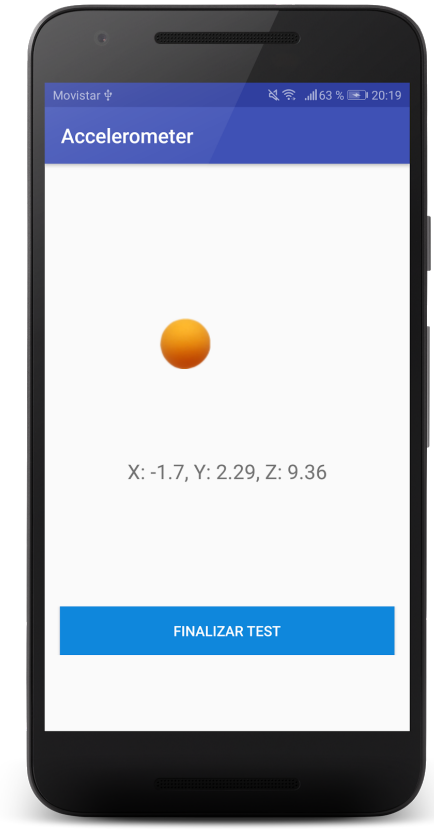
XML: Son dos ‘LinearLayout’, el primer conté un ‘ImageView’ i un ‘TextView’. El segon conté el botó de finalitzar.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center_horizontal">

    <Button
        android:id="@+id/play_pause"
        android:layout_marginTop="50dp"
        android:layout_width="180dp"
        android:layout_height="180dp"
        android:background="@drawable/play" />

    <TextView
        android:layout_marginTop="60dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAlignment="center"
        android:text="@string/audioTest"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/testEnd"
        android:text="Finalizar test"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

Codi: En aquest codi ens centrem en el 'MediaPlayer', ja que aquesta classe fa que puguis iniciar una cançó. Com sempre, iniciem les variables amb els valors de l'xml. Tenim un listener al botó per iniciar i pausar la cançó.
El mètode onBackPressed() l'utilitzem per que si volem tirar enrere, es pari la cançó.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_audio_test);

    play_pause = (Button) findViewById(R.id.play_pause);
    testEnd = (Button) findViewById(R.id.testEnd);

    mp = MediaPlayer.create(this, R.raw.song);


    play_pause.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            if(mp.isPlaying()){
                mp.pause();
                play_pause.setBackgroundResource(R.drawable.play);
            }else {
                mp.start();
                play_pause.setBackgroundResource(R.drawable.pause);
            }

        }
    });

    final Intent resultactivity = new Intent(AudioTestActivity.this, AudioResultActivity.class);

    alltest = getIntent().getBooleanExtra("alltest",false);

    testEnd.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            mp.pause();
            play_pause.setBackgroundResource(R.drawable.play);
            resultactivity.putExtra("alltest",alltest);
            startActivity(new Intent(resultactivity));
        }
    });
}

@Override
public void onBackPressed() {
    super.onBackPressed();
    play_pause.setBackgroundResource(R.drawable.play);
    mp.pause();
}
```

AccelerometerTestActivity: Amb aquest test comprovem el funcionament de l'acceleròmetre. En quant fem clic a continuar, veurem l'eix tridimensional de l'acceleròmetre, i una pilota moven't-se per la pantalla en sentit del moviment del dispositiu.



XML: Mostra un 'ImageView' que la posició va variant segons el moviment del dispositiu.

Tenim un 'TextView' que mostra la X,Y i Z del dispositiu.

Finalment tenim el botó per finalitzar el test

```
<ImageView
    android:id="@+id/ball"
    android:background="@drawable/ball"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center_horizontal">

    <ImageView
        android:id="@+id/acceImg"
        android:layout_marginTop="50dp"
        android:layout_width="160dp"
        android:layout_height="180dp"
        android:src="@drawable/accelerometer" />

    <TextView
        android:id="@+id/text1"
        android:layout_marginTop="60dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAlignment="center"
        android:text="@string/acceTest"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:gravity="bottom"
    android:layout_marginBottom="@dimen/bottom_layout">

    <Button
        android:id="@+id/continuar"
        android:text="Continuar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/button_general"
        android:textColor="#ffffff"
        android:layout_marginHorizontal="@dimen/marginGeneral"/>
</LinearLayout>
```

Codi: Primer de tot inicialitzem les variables amb el valor de l'xml.

La pilota (ball) comença sent invisible, perque quan fem clic a 'Comenzar', es fa visible i el text del botó canvia a 'Finalizar Test'.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_accelerometer_test);

    acceImg = (ImageView) findViewById(R.id.acceImg);
    text1 = (TextView) findViewById(R.id.text1);
    continuar = (Button) findViewById(R.id.continuar);
    pantalla = (ConstraintLayout) findViewById(R.id.pantalla);
    ball = (ImageView) findViewById(R.id.ball);

    ball.setVisibility(View.INVISIBLE);

    continuar.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
            mySensor = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);

            sensorManager.registerListener(AccelerometerTestActivity.this,mySensor,SensorManager.SENSOR_DELAY_NORMAL);

            ball.setVisibility(View.VISIBLE);
            acceImg.setVisibility(View.INVISIBLE);
            continuar.setText("Finalizar test");

            final Intent resultactivity = new Intent(AccelerometerTestActivity.this, AccelerometerResultActivity.class);

            alltest = getIntent().getBooleanExtra("alltest",false);

            continuar.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    resultactivity.putExtra("alltest",alltest);
                    startActivity(new Intent(resultactivity));
                }
            });

        }
    });
}
```

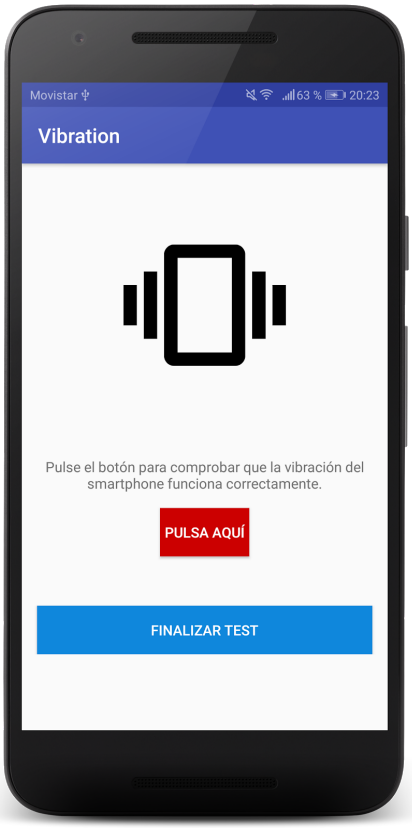
Aquesta classe implementa de 'SensorEventListener', això fa que necessiti certs mètodes per funcionar, com el 'onAccuracyChanged' i el 'onSensorChanged'.

Al 'onSensorChanged()' canviem els valors de l'eix tridimensional segons el moviment del dispositiu.

Tenim condicions perquè si la pilota sobrepassa el minim o maxim de la pantalla, es queda a aquesta alçada/amplada.

```
@Override
public void onSensorChanged(SensorEvent event) {

    text1.setTextSize(20.0f);
    float px = (float) (Math.round(event.values[0]*100.0)/100.0);
    float py = (float) (Math.round(event.values[1]*100.0)/100.0);
    float pz = (float) (Math.round(event.values[2]*100.0)/100.0);

    text1.setText("X: " + px + ", Y: " + py + ", Z: " + pz);

    float valorx = -event.values[0]*200;
    float valory = event.values[1]*200;

    if (valory <= pantalla.getMinHeight()){
        ball.setY(pantalla.getMinHeight());
    }else {
        ball.setY(valory);
    }
    if (valorx <= pantalla.getMinWidth()){
        ball.setX(pantalla.getMinWidth());
    }else {
        ball.setX(valorx);
    }
}

@Override
public void onAccuracyChanged(Sensor sensor, int accuracy) {

}
```

VibrationtTestActivity: Aquí el que encontrem es una imatge, un text on diu el que ha de fer i un botó per iniciar la vibració. Cada cop que cliqui el mòbil vibrarà.



XML: Aquest XML te la mateixa estructura que algun que ja hem explicat. Dos ‘LinearLayouts’, un ‘TextView’, una ‘ImageView’ i dos botons.

Codi: Dins del onCreate es crida a la classe Vibrator, que es la que fa que el mòbil vibri. Després es comprova si el terminal té vibració o no. En cas de que no tingui, sortirà un missatge.

```java
public class VibratorTestActivity extends AppCompatActivity {

    Button checkVibrate, testEnd;
    TextView textVibrate;
    boolean alltest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_vibrator_test);

        checkVibrate = (Button) this.findViewById(R.id.checkVibrate);
        testEnd = (Button) this.findViewById(R.id.testEnd);
        textVibrate = (TextView) this.findViewById(R.id.textVibrate);

        final Vibrator vibrator = (Vibrator) this.getSystemService(Context.VIBRATOR_SERVICE);

        if(vibrator.hasVibrator()){
            textVibrate.setText(R.string.vibratorTest);
            checkVibrate.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    vibrator.vibrate(1000);
                }
            });

        }else{
            textVibrate.setText("Este smarthphone no contiene modo vibratorio.");
        }

        final Intent resultactivity = new Intent(VibratorTestActivity.this, VibratorResultActivity.class);

        alltest = getIntent().getBooleanExtra("alltest",false);

        testEnd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                resultactivity.putExtra("alltest",alltest);
                startActivity(new Intent(resultactivity));
            }
        });
    }
}
```

2.4 Disseny de la base de dades

Hem utilitzat la base de dades de Firebase per guardar els resultats dels test. Només emmagatzema els resultats del últim test fet per poder mostrar els resultats actualitzats.

Com podem veure, el que es guarda a la base de dades es una id única del propi mòbil i després si ha sortit bé o no la prova dels tests.
Així cada vegada que s'encengui l'app, mirarà si la id del mòbil existeix i si existeix es posarà els colors als botons del menú de l'app.

Per guardar la id del mòbil, hem fet una classe anomenada SHAUtils. Se li pasa el context i amb això i el 'Settings' s'obté una id única del mòbil.

```
public class SHAUtils {
    public static String getAndroidId(Context context){
        return Settings.Secure.getString(context.getContentResolver(), Settings.Secure.ANDROID_ID);
    }
}
```

3. Conclusions

Hem aprés a utilitzar l'Android Studio amb mes profunditat, una base de dades Firebase que hem pogut explorar i aconseguir bons resultats amb les dades que emmagatzema. Ens ha faltat alguna cosa, però estem bastant contents amb el resultat final de l'app.

Reflexió crítica:

Com diria qualsevol, la falta de temps ha sigut un factor que ens ha perjudicat. També s'ha afegit la dificultat del Firebase, no teníem molta idea i després donava error a l'hora de mostrar la informació.

Vam descartar la pàgina web, perquè volíem centrar-nos en el que es l'aplicació i el firebase, per així mostrar les dades dins de l'app, però com he dit, ens ha donat bastants errors algunes comandes del Firebase i per això tampoc hem pogut mostrar les dades com voliem.

També hem tingut dificultat a l'hora d'aconseguir informació per alguns apartats de la nostra app. O no hem sabut buscar o no existia cap mètode per agafar informació especifica.

Anàlisi crític:

Vam seguir la planificació fins ha un punt, on ens vam estancar a l'hora de decidir i fer els sensors per fer una aplicació sencera. Vam descartar la pàgina web per motius que hem explicat ja. Hem fet alguns canvis perquè si no no arribarem a la entrega del projecte. La metodologia no ha sigut la millor cosa del món però ha servit per el que voliem.

Treball de futur:

Ha quedat pendent el mostrar el llistat de dispositius que han utilitzat aquesta app i que ens pogués donar una llista dels sensors de cada dispositius. També queda pendent el que es implementar una pàgina web perquè qualsevol usuari pugui mirar la informació de cada mòbil.

4. Glossari

SensorManager: SensorManager us permet accedir als sensors del dispositiu. Alguns dels sensors són basats en maquinari i alguns són sensors basats en programari. Independentment del sensor, Android ens permet obtenir les dades en brut d'aquests sensors i utilitzar-los a la nostra aplicació.

MediaPlayer: El marc multimèdia d'Android inclou compatibilitat per reproduir diversos tipus multimedia, de manera que podeu integrar fàcilment àudio, vídeo i imatges a les vostres aplicacions. Podeu reproduir àudio o vídeo des de fitxers multimèdia emmagatzemats en els recursos de la vostra aplicació (recursos primers), des de fitxers independents del sistema de fitxers o des d'un flux de dades que arriba a través d'una connexió de xarxa, tot utilitzant les API de MediaPlayer.

TextView: Un element d'interfície d'usuari que mostra text a l'usuari.

LinearLayout: Un disseny que organitza altres vistes horitzontalment en una única columna o verticalment en una única fila.

Listener: Un listener és una interfície a la classe Vista que conté un únic mètode de devolució de trucada. Aquests mètodes seran cridats pel framework d'Android quan la Vista a la qual s'hagi registrat l'agent d'escolta es desencadena mitjançant la interacció de l'usuari amb l'element a la UI.

Intent: Un intent és una descripció abstracta d'una operació que es realitzarà. Es pot utilitzar amb startActivity per iniciar una activitat, broadcastIntent per enviar-la a qualsevol component BroadcastReceiver i startService (Intent) o bindService (Intent, ServiceConnection, int) per comunicar-se amb un servei en segon pla.

MediaRecorder: La classe MediaRecorder permet especificar l'origen de l'àudio o vídeo, el format del fitxer de sortida i els codecs a utilitzar.

5. Bibliografia

https://developer.android.com/reference/android/os/Build.html?hl=ja
https://stackoverflow.com/questions/17489518/how-to-detect-a-mobile-device-manufacturer-and-model-programmatically-in-android

Camera: https://www.youtube.com/watch?v=ondCeqlAwEI
http://www.vogella.com/tutorials/AndroidCamera/article.html

Flashlight:
https://android.jlelse.eu/create-a-torch-flashlight-application-for-android-c0b6951855c

Position Sensors:
https://developer.android.com/guide/topics/sensors/sensors_position.html

Giroscope y Proximity Sensor:
https://code.tutsplus.com/es/tutorials/android-sensors-in-depth-proximity-and-gyroscope--cms-28084

Battery:
https://stackoverflow.com/questions/3291655/get-battery-level-and-state-in-android
https://developer.android.com/reference/android/os/BatteryManager

Vibrator info
https://source.android.com/reference/hidl/android/hardware/vibrator/1.0/IVibrator

Id movil
https://developer.android.com/training/articles/user-data-ids

Screen
https://developer.xamarin.com/api/type/Android.Util.DisplayMetrics/

6. Annexos

Codi del projecte des del github:

https://github.com/guillermobdn/SHA