animoi

PROGRAMACIÓ EN ANDROID STUDIO

Núria Murgui Gómez
Facultat de Belles Arts de Sant Carles
Grau en Disseny i Tecnologies Creatives
Curs 2019-2020

La programació en Android Studio es duu a terme en tres nivells: la programació de la interfície i disseny de tots els elements de cada pantalla, dut a terme en llenguatge XML[1]; la programació de la funcionalitat, és a dir, el pas d'informació entre pantalles, les accions que s'executen en clicar un element o qualsevol acció que requereix interactivitat, dut a terme en llenguatge Java[2]; i l'organització de les pantalles i la concessió de permisos, dut a terme en llenguatge XML al fitxer AndroidManifest.

Aquest document és una anàlisis i explicació dels diferents fitxers, la seua organització i la seua funcionalitat, organitzat per pantalles de l'aplicació. De tal manera, es portarà a terme una explicació dels fitxers generals que afecten a tota l'aplicació, i una explicació més concreta dels diferents fitxers, tant Java com XML, necessaris per tal que cada pantalla de l'aplicació funcione.

FITXERS GENERALS

1. AndoridManifest.xml

El AndoridManifest és un arxiu que defineix informació sobre l'aplicació per a les ferrametes de creació d'Android, el sistema operatiu Andoid i Google Play.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.animoi">

    <uses-permission android:name="android.permission.CAMERA"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

    <supports-screens
        android:anyDensity="true"
        android:largeScreens="true"
        android:normalScreens="true"
        android:smallScreens="true"
        android:xlargeScreens="true" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Animoi"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:screenOrientation="landscape"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".PopUpPublicar"
            android:theme="@style/AppTheme.PopMe"></activity>
        <activity android:name=".PuntosConseguidos" />
        <activity android:name=".ExplicacionProyecto" />
        <activity android:name=".IntroProyectos" />
        <activity android:name=".Usuario" />
```

1. XML (eXtensible Markup Language) és un meta-llenguatge creat en 1998 que permet marcar o etiquetar elements per a un determinat ús.

2. Java és un llenguatge de programació orientat a objectes creat en 1990 per James Gosling.

El primer grup d'informació correspon al nom del paquet de l'aplicació, utilitzat per les ferramentes de compilació d'Android i que més avant s'utilitzarà a Google Play.

El bloc *uses-permission* fa referència als permisos que l'aplicació ha de demanar per tal de dur a terme certes activitats. En aquest cas, utilitzar la càmera del telèfon i emmagatzemar les fotografies dutes a terme amb la càmera.

A continuació, el bloc *suports-screens* permet l'adaptació del disseny a tots els tipus de dimensions de pantalla, ja que l'aplicació està plantejada tant per a telèfons mòbils com per a tabletes.

Per últim, l'apartat *application* fa referència a les pantalles que té l'aplicació (definides a la imatge com *activity*), defineix quina d'aquestes pantalles serà la principal, l'orientació de la pantalla en executar l'aplicació (vertical o horitzontal) i l'estil general.

2. Fitxers d'estil

Quant a l'aparença i contingut de l'aplicació, hi ha tres fitxers que permeten definir un estil general que més avant s'aplicarà a les diferents pantalles. Aquests fitxers són *colors.xml, strings.xml* i *styles.xml*, agrupats a la carpeta *values*.

Dits arxius permeten crear variables de colors, text i estils, respectivament. Açò resulta molt útil quant a que, a l'hora de definir el color d'un element, s'utilitza la variable i no el color directament, de manera que si aquest color s'ha utilitzat moltes vegades i es vol canviar, sols haja de canviar-se una vegada a la variable i no totes les vegades que s'ha utilitzat.

```
<color name="colorPaleta1">#28154A</color>
<color name="colorPaleta2">#f9cb12</color>
<color name="colorPaleta3">#c61867</color>
<color name="colorPaleta4">#b47ae0</color>
<color name="white">#ffffff</color>
<color name="black">#000000</color>
<color name="gris">#E2E2E2</color>
<color name="textoGris">#9A9A9A</color>
```

```
<string name="lection_1">Lección I</string>
<string name="atomos">LOS ÁTOMOS </string>

<string name="lection_2">Lección II</string>
<string name="electricidad">ELECTRICIDAD</string>

<string name="lection_3">Lección III</string>
<string name="imanes">IMANES</string>

<string name="lection_4">Lección IV</string>
<string name="motores">MOTORES ELÉCTRICOS</string>
```

El fitxer styles és un poc més complex, ja que permet definir un estil general d'un element. D'aquesta manera es poden definir les dimensions, el color, els marges o la tipografia d'un element, per exemple. Així, si aquest element es repeteix, no és necessari definir-lo cada vegada.

```
<style name="botonBack">
    <item name="android:layout_width">15dp</item>
    <item name="android:layout_height">15dp</item>
    <item name="android:layout_marginStart">10dp</item>
    <item name="android:layout_marginTop">5dp</item>
    <item name="android:clickable">true</item>
    <item name="android:src">@drawable/flecha</item>
    <item name="android:layout_weight">1</item>
    <item name="android:gravity">center</item>
    <item name="android:layout_margin">10dp</item>
</style>
```

3. Toolbar

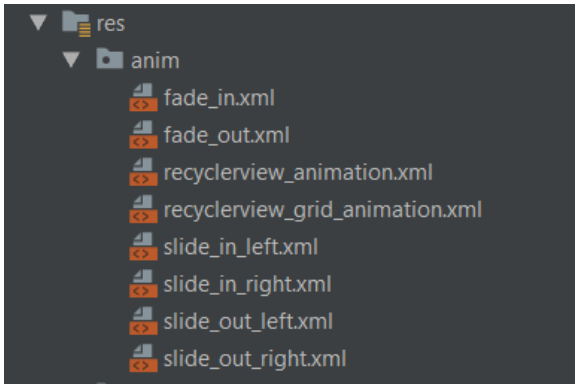
La Toolbar és la barra de menú superior que permet indicar en quin apartat es troba l'usuari. Per tal d'incloure-la als diferents apartats, ha de definir-se el seu disseny a un document XML, amb el nom *toolbar.xml*, i afegir-la a la resta de pàgines mitjançant una etiqueta *<include>*.

```
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".LeccionAprende">

<include
    android:id="@+id/include"
    layout="@layout/toolbar"
    tools:layout_editor_absoluteX="0dp"
    tools:layout_editor_absoluteY="0dp"
    android:elevation="10dp">

</include>
```

4. Transicions

Quant a les transicions, Andorid Studio compta amb algunes transicions ja existents. A l'hora de canviar la transició per defecte, es pot recórrer a les transicions ja existents, com ara *Fade*, *Slide*, o *Explode*; o bé es poden programar noves transicions.

En el cas d'aquesta segona opció, s'utilitzen fitxers xml amb la funcionalitat determinada i s'assignen a la funció *overridePendingTransition()* després del *Intent* per canviar de pantalla. Aquestes transicions que, en el fons, són animacions de les pantalles, es troben a la carpeta *anims*.



*Quant als fitxers específics que van a definir-se a continuació, cada “activity” o pantalla té assignats un fitxer Java i un fitxer XML, com a mínim. La majoria d’aquests fitxers XML es troben a l’interior de carpetes amb el mateix nom del document i on hi ha dos fitxers: un per a pantalles de telèfon mòbil i altre per a pantalles de tableta (amb la indicació que són fitxers per a dimensions *large*). D’aquesta manera es poden adaptar els dissenys a les diferents dimensions de pantalla.

FITXERS ESPECÍFICS

1. Splash Screen

Java	XML
SplashScreen.java	activity_splash_screen.xml

Activity_splash_screen.xml determina el disseny de la primera pantalla que apareix en obrir l'aplicació, que consta simplement d'una imatge. *SplashScreen.java* fa servir un comptador que manté aquesta pantalla durant 4 segons i, a continuació, canvia automàticament a la pantalla següent.

2. Main Activity

Java	XML
MainActivity.java	activity_main.xml
SliderAdapter.java	slide_layout.xml
PopUpUsuario.java	activity_pop_up_usuario.xml

La pantalla principal, on es selecciona a quin apartat es vol entrar, funciona a partir d'un slider. Això implica que al fitxer *slide_layout.xml* han de definir-se quins elements es troben al slider (una imatge i un text), i el *SliderAdapter.java* determina quines imatges i quins texts van a aparèixer. Per altra part, al fitxer *activity_main* es dissenya tota la pantalla, que inclou tant el slider com els botons per a accedir a les pantalles de informació i usuari/configuració. Mentre que *MainActivity.java* controla la funcionalitat de tota aquesta pantalla.

Destacar a aquest apartat l'ús del *ViewPager* als documents xml i de la classe[3] *PageAdapter*[4] proporcionats a les biblioteques d'Android i necessaris per tal de crear el slider.

Per últim, els fitxers *PopUpUsuario.java* i *activity_pop_up_usuario.xml* corresponen a la finestra emergent que apareix la primera vegada que l'usuari entra a l'aplicació, per tal que registre el seu nom i seleccione una icona que servirà com a imatge de perfil de l'usuari. Allò que l'usuari seleccione es guardarà a *SharedPreferences*, un apartat que permet guardar informació en format clau-valor i que és accessible des de totes les pantalles de l'aplicació. D'aquesta manera, en qualsevol pantalla es pot obtindre el nom que l'usuari ha introduït.

3. Una classe en Java és una plantilla virtual que permet crear objectes amb les mateixes característiques (atributs) o funcionalitats (mètodes).

4. El PagerAdapter és un *Adapter*, és a dir, permet implementar el slider de forma predeterminada.

3. Usuari - Configuració

Java	XML
Usuario.java	activity_usuario.xml
	spinner_item.xml
	bk_spinner.xml

Aquesta pàgina permet a l'usuari canviar el seu nom a l'aplicació i dona informació sobre els projectes completats. El disseny d'aquesta pàgina es duu a terme al fitxer *activity_usuario.xml*, mentre que a *Usuario.java* s'assigna la funcionalitat de cada element.

Aquesta pàgina també presenta dues opcions de configuració que permeten adaptar el contingut a les necessitats visuals de l'usuari. Aquestes dues opcions es presenten en forma de *spinner*, element que ha de ser definit gràficament en un fitxer per separat. Aquest fitxer és *spinner_item.xml*. A més, per tal que aquests elements apareguin subratllats, ha de definir-se un fons (background) al fitxer *bk_spinner.xml* i assignar-lo al *spinner*.

4. Aprèn – Selecció de la lliçó

Java	XML
Apren.java	activity_apren.xml
ItemData.java	layout_item.xml
MyAdapter.java	bk_layout_item.xml
GridRecyclerView.java	

En aquest cas, la selecció de l'element del drone sobre el qual es vol aprendre està dut a terme a través d'un mateix element que es repeteix diverses vegades. D'aquesta manera, l'element de selecció és sempre igual però canvien la imatge i el títol. *Layout_item.xml* és el fitxer que permet dissenyar aquest element. A més, per tal que aquest element compte amb un fons arrodonit, s'ha de definir una forma geomètrica amb les puntes arrodonides al fitxer *bk_layout_item.xml*, i assignar aquesta configuració al fons (background) de l'element.

Pel que respecta a *ItemData.java* i *MyAdapter.java*, són els fitxers que creen les propietats de l'element de selecció que es repetirà, i determinen quines de les seues propietats es mostren a pantalla i quines s'utilitzen per a passar informació a pantalles següents.

Una vegada fet açò, a *activity_apren.xml* es dissenya la pantalla completa, mentre que a *Apren.java* es defineix quina imatge, quin text i quina informació correspon a cadascun

d'aquests elements. A més, a *Apren.java* també es defineix informació que correspon a cada element però que no es mostrarà a aquesta pantalla sinó a la pantalla següent.

Per últim, els diferents elements estan organitzats en forma d'una quadrícula irregular. El fitxer *GridRecyclerView.java* permet que s'aplique una animació sobre dita quadrícula quan s'accedeix a la pantalla.

4. Aprèn – Introducció de la lliçó

Java	XML
IntroLeccion.java	activity_intro_leccion.xml

Destacar que aquest apartat recopila la informació enviada des d'*Apren.java*, de manera que la pregunta i el títol, canviaran segons l'element que seleccione l'usuari. A més, *IntroLeccion.java* també rep un *id*, és a dir, un identificador (en aquest cas un número) procedent d' *Apren.java*, que al mateix temps envia a la següent pantalla. Aquest número determinarà el contingut de la pantalla explicatòria de l'element seleccionat per l'usuari.

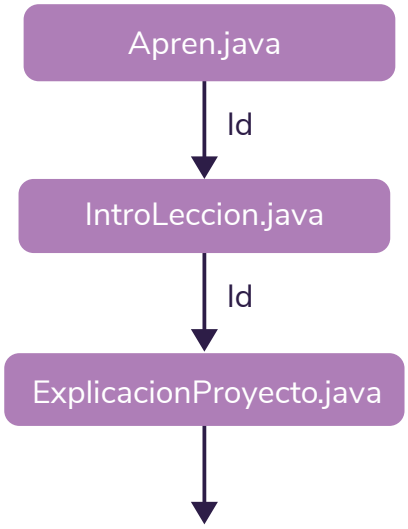
4. Aprèn – Explicació

Java	XML
ExplicacionProyecto.java	activity_explicacion_proyecto.xml
ItemExplicacion.java	layout_item_explicacion.xml
SliderAdapterApren.java	

Aquesta pantalla funciona de manera similar a la pantalla principal. A *layout_item_explicacion.xml* es dissenya l'element del slider, format per una imatge o animació, un títol i un text. A *ItemExplicacion.java* i *SliderAdapterApren.java* es defineixen les propietats i la funcionalitat del slider.

A continuació, a *activity_explicacion_proyecto.xml* es dissenya la pantalla completa, incloent el slider, el botó per donar per finalitzada la lliçó i la *toolbar*. Mentre, a *ExplicacionProyecto.java* es defineixen tots els elements del slider, és a dir, s'indica quin text, quina animació i quin títol correspon a cada element. I segons el id que haja rebut des de *IntroLeccion.java*, es mostraran uns elements o altres.

Destacar que les animacions utilitzades a aquest apartat es troben a una carpeta específica anomenada assets. Aquestes animacions es guarden en format JSON.



Id determina quines imatges i quins texts apareixen en pantalla

```
//Intent que recupera el id enviat des de la pantalla anterior
Intent i = getIntent();
id = i.getIntExtra( name: "id", defaultValue);

//Definició de tots els items dels sliders. Tots definits a la vegada
itemExplicacions = new ArrayList<>();
ItemExplicacion Uno = new ItemExplicacion( imgExplicacion: "atomo_lapis_color.json", titleExplicacion: "LOS ÁTOMOS", getString(R.string.atomos_1));
ItemExplicacion Dos = new ItemExplicacion( imgExplicacion: "atomo_nucleo_color.json", titleExplicacion: "LOS ÁTOMOS", getString(R.string.atomos_2));
ItemExplicacion Tres = new ItemExplicacion( imgExplicacion: "atomo_color.json", titleExplicacion: "LOS ÁTOMOS", getString(R.string.atomos_3));
ItemExplicacion Cuatro = new ItemExplicacion( imgExplicacion: "bateria_color.json", titleExplicacion: "ELECTRICIDAD", getString(R.string.electricidad_1));
ItemExplicacion Seis = new ItemExplicacion( imgExplicacion: "iman_estatico_color.json", titleExplicacion: "LOS IMANES", getString(R.string.iman_1));
ItemExplicacion Siete = new ItemExplicacion( imgExplicacion: "imanes_brillos_color.json", titleExplicacion: "LOS IMANES", getString(R.string.iman_2));
ItemExplicacion Ocho = new ItemExplicacion( imgExplicacion: "motor_ilustracion.json", titleExplicacion: "MOTOR DE CORRIENTE CONTINUA", getString(R.string.motor_1));
ItemExplicacion Nueve = new ItemExplicacion( imgExplicacion: "motor_imanes.json", titleExplicacion: "EL ESTATOR", getString(R.string.motor_2));
ItemExplicacion Diez = new ItemExplicacion( imgExplicacion: "motor_rotor.json", titleExplicacion: "EL ROTOR", getString(R.string.motor_3));
ItemExplicacion Once = new ItemExplicacion( imgExplicacion: "helice_eje_color.json", titleExplicacion: "HÉLICES", getString(R.string.helices_1));
ItemExplicacion Doce = new ItemExplicacion( imgExplicacion: "helices_color.json", titleExplicacion: "HÉLICES", getString(R.string.helices_2));
ItemExplicacion Trece = new ItemExplicacion( imgExplicacion: "helices_rotando_color.json", titleExplicacion: "HÉLICES", getString(R.string.helices_3));
ItemExplicacion Diecisiete = new ItemExplicacion( imgExplicacion: "ultrasonido_color.json", titleExplicacion: "ULTRASONIDO", getString(R.string.ultrasonido_1));
ItemExplicacion Dieciocho = new ItemExplicacion( imgExplicacion: "ultrasonido_color.json", titleExplicacion: "ULTRASONIDO", getString(R.string.ultrasonido_2));
```

```
//Col·locació dels items que corresponen a cada lliçó, segons el que seleccione l'usuari
switch (id) {
    case 1:
        itemExplicacions.add(Uno);
        itemExplicacions.add(Dos);
        itemExplicacions.add(Tres);
        numPuntos = 3;
        indicador.setText("Lección I - Los átomos");
        break;

    case 2:
        itemExplicacions.add(Cuatro);
        numPuntos = 1;
        indicador.setText("Lección II - Electricidad");
        bAcabar2.setEnabled(true);
        bAcabar2.setAlpha(1f);
        break;

    case 3:
        itemExplicacions.add(Seis);
        itemExplicacions.add(Siete);
        numPuntos = 2;
        indicador.setText("Lección III - Los imanes");
        break;
```

5. Monta – Selecció del projecte

Java	XML
Montar.java	activity_monta.xml

En aquest cas, no van a incloure's molts projectes a este primer prototip de l'aplicació, sinó que en una versió més desenvolupada ja se'n afegirien més. És per això que a *activity_montar.xml* es defineixen dos elements amb el mateix disseny però sense utilitzar ningun adapter. A *Monta.java* s'assigna un *id* per a cada element segons el que seleccione l'usuari, i aquest *id* determinarà el contingut de les pantalles següents. En el cas del primer projecte d'ampliació, simplement s'arriba a una pantalla que indica que el projecte encara està en desenvolupament. En canvi, en punjar sobre el dron s'accedeix a les instruccions.

Activity_montar.xml és el fitxer que determina el disseny de la pantalla.

5. Monta – Introducció

Java	XML
IntroProyectos.java	activity_intro_proyectos.xml

Aquesta és una pantalla de transició entre la selecció del projecte i la seua explicació. El *id* rebut des de *Monta.java* determinarà el contingut d'aquesta pantalla i s'enviarà també a la pantalla següent. Açò es gestiona des del fitxer *IntroProyectos.java*.

A més, un mateix projecte està dividit en diferents fases de construcció. D'aquesta manera, l'usuari ha de decidir quina fase seleccionar i, segons això, s'enviarà un altre número a la pantalla següent a través de la variable *instruct*, que determina el número de la instrucció seleccionada per l'usuari.

5. Monta – Instruccions sobre les peces

Java	XML
InstructPiezas.java	activity_instruc_piezas.xml

Abans d'iniciar la construcció del projecte seleccionat, ha de indicar-se quines són les peces necessàries. *Activity_instruc_piezas.xml* determina el disseny de la pantalla, mentre que InstructPiezas.java assigna unes imatges i uns textos diferents segons els valors de *id* i *instruct* rebuts, que indiquen de quin projecte i quina instrucció procedeix l'usuari.

5. Monta – Instruccions sobre el projecte

Java	XML
InstructProyecto.java	activity_instruc_proyecto.xml

A aquesta pantalla ja s'expliquen els passos a seguir per a la construcció. I*nstructProyecto.java* rep els valors del *id* i *instruct* per tal de determinar el contingut. Per altra part, una mateixa instrucció pot tindre diferents passos de construcció. D'aquesta manera, es defineix una plantilla comú a *activity_instruc_proyecto.xml*, i segons l'usuari vaja passant de pas a pas de la instrucció, l'animació i el text aniran canviant. Açò és possible gràcies a un comptador, és a dir, un número que augmenta el seu valor de un en un cada vegada que l'usuari selecciona el botó per a accedir al pas següent.

Destacar que les animacions utilitzades, en fromat MP4, es troben a la carpera *raw*.

5. Monta – Rebre punts

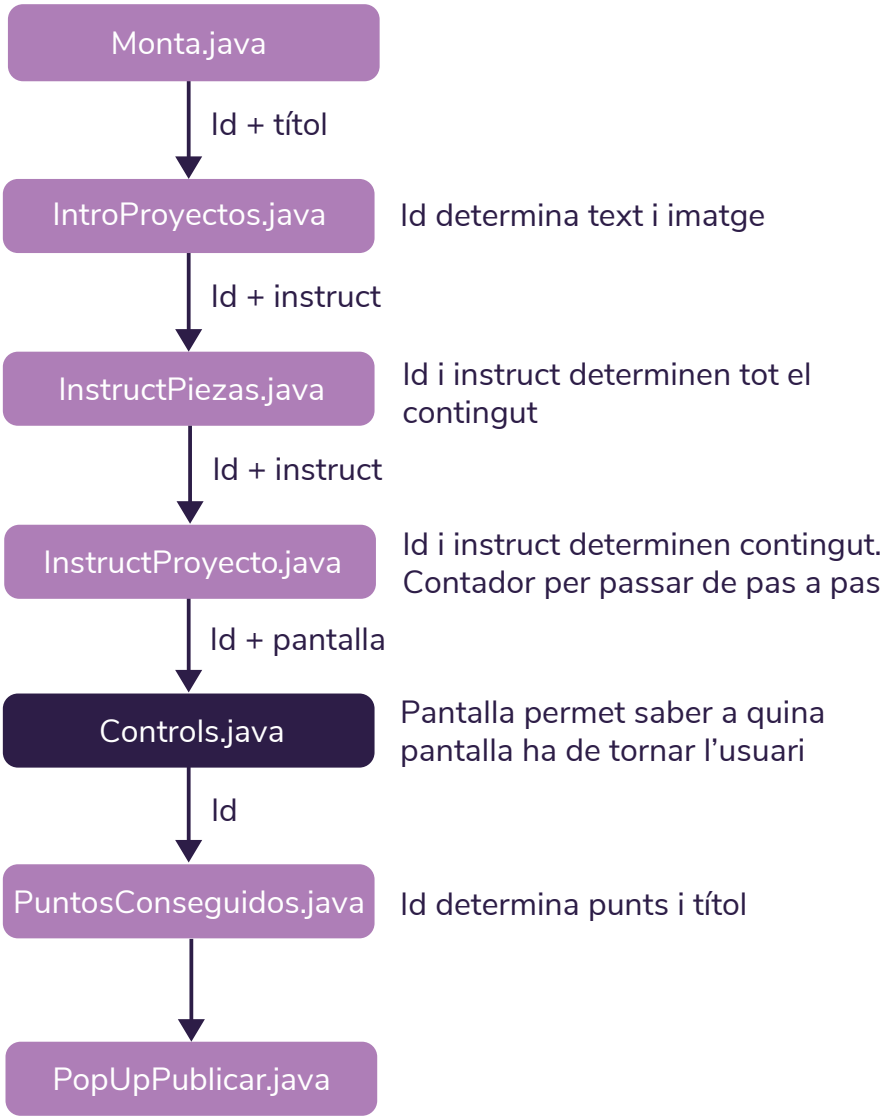
Java	XML
PuntosConseguidos.java	activity_puntos_conseguidos.xml

A *PuntosConseguidos.java* es recuperen el *id* i el títol enviats des de les pantalles anteriors. Segons el *id*, que indica quin projecte s'ha completat, l'usuari rebrà uns punts determinats. Una vegada rebuts els punts, es dona l'opció de finalitzar el projecte i tornar al menú incial, o publicar el projecte a la pàgina web d'Animoi.

5. Monta – Publicar projecte

Java	XML
PopUpPublicar.java	activity_pop_up_publicar.xml

A *activity_pop_up_publicar.xml* es dissenya com serà la pantalla emergent, la qual conté camps que permeten a l'usuari introduir el títol, la descripció del projecte i una imatge. A *PopUpPublicar.java*, aquests textos es guarden a variables, que permetrien la publicació de dites dades en una fase més avançada del projecte. A més, aquesta pantalla conté els permisos necessaris per tal de poder fer una fotografia amb el dispositiu que estiga utilitzant l'usuari i utilitzar aquesta fotografia per a la publicació del projecte. Una vegada finalitzat el procés, l'usuari torna al menú inicial.




```
//Intent que recupera el id i el número de la instrucció enviats en la pantalla IntroProyectos.java
Intent i = getIntent();
id = i.getIntExtra( name: "id", defaultValue);
instruct = i.getIntExtra( name: "instruct", defaultValue);

switch (id){
    case 0:
        if(instruct==1) {
            pieza6.setAlpha(0f);
            txPieza6.setAlpha(0f);
        }

        if(instruct==2){
            pieza2.setImageResource(R.drawable.motor_render);
            txPieza2.setText("Motor");
            pieza3.setImageResource(R.drawable.motor_render);
            txPieza3.setText("Motor");
            pieza6.setAlpha(1f);
            pieza6.setImageResource(R.drawable.placa);
            txPieza6.setAlpha(1f);
            txPieza6.setText("Placa");
        }
    break;
```

6. Controls

Java	XML
Controles.java	activity_controles.xml

La pantalla dels controls no és funcional en aquest moment, sinó que podria controlar el dron en versions futures del projecte. No obstant això, conté tots el elements que apareixerien en les versions posteriors. Activity_controles.xml és el fitxer dedicat al disseny de la pantalla. Mentre que Controles.java rep informació sobre la pantalla de la qual procedeix l'usuari a través d'una variable. Això es deu a que es pot accedir als controls de vol directament, o en finalitzar un projecte o bé en acabar algunes de les lliçons de l'apartat Aprèn. Segons el valor de la variable i per tant, d'on vinga l'usuari, en acabar tornarà a una pantalla o una altra.

A més, la pantalla de controls compta amb un botó que activa o desactiva cartells per indicar a l'usuari quina és la funció de cada un dels elements que apareixen en pantalla i a quina funció correspon cada control.

```
//Mètode per establir a quina pàgina es torna quan se ix de la pantalla de Controls
public void volverControles (View v){
    //Si ve de Vola, torna al Menú principal
    if(pantalla == 0){
        Intent intentVolverControles = new Intent( packageContext: Controles.this, MainActivity.class);
        startActivity(intentVolverControles);
    }
    //Si ve de Monta, torna a la pantalla explicatòria del projecte corresponent
    else if (pantalla ==1){
        switch (id){
            case 0:
                break;
            case 1:
                break;
        }
        Intent intentVolverMonta = new Intent( packageContext: Controles.this, ExplicacionProyecto.class);
        intentVolverMonta.putExtra( name: "id", id);
        startActivity(intentVolverMonta);
    }
    //Si ve d'Aprèn, torna al menú d'Aprèn
    else if (pantalla == 2){
        Intent intentVolverApren = new Intent( packageContext: Controles.this, Apren.class);
        startActivity(intentVolverApren);
    }
}
```