CFGS: DESENVOLUPAMENT D'APLICACIONS MULTIPLATAFORMA

Mòdul: Programació multimèdia i dispositius mòbils

TEMA 5: INTERFÍCIES D'USUARI. FRAGMENTS

Germán Gascón Grau
ggascon@gmail.com

Introducció

- Quan començaren a aparèixer dispositius de gran grandària tipus tablet, l'equip d'Android va haver de solucionar el **problema de l'adaptació de la interfície gràfica** de les aplicacions a aquest **nou tipus de pantalles**.
- Una interfície d'usuari dissenyada per a un telèfon mòbil no s'adaptava fàcilment a una pantalla diverses polzades major. La solució va vindre en forma d'un nou tipus de component anomenat **Fragment**.
- Un fragment és una porció de la interfície d'usuari que pot **afegir-se o eliminar-se** de la interfície **de forma independent a la resta d'elements** de l'Activity, i que per descomptat pot reutilitzar-se en altres Activitats.
- D'aquesta forma, podem dividir la nostra interfície en diverses porcions de manera que s'adapte a diferents configuracions de pantalla, depenent de la seua grandària i orientació, sense haver de duplicar codi en cap moment, tan sols utilitzant els diferents fragments per a cadascuna de les possibles configuracions.

Exemple - Enunciat

- Suposem una **aplicació de correu electrònic**, en la qual d'una banda hem de mostrar la llista de correus disponibles, amb els seus camps clàssics *De* i *Assumpte*, i d'altra banda hem de mostrar el *contingut* complet del correu seleccionat.
- En un telèfon mòbil, l'habitual serà tindre una primera Activity que mostre el llistat de correus, i quan l'usuari seleccione un d'ells, es navegue a una nova Activity que mostre el contingut d'aquest correu.
- No obstant això, en una tablet pot existir espai suficient per a tenir ambdues parts de la interfície en la mateixa pantalla, per exemple en una tablet en posició horitzontal podríem tenir una columna a l'esquerra amb el llistat de correus i dedicar la zona dreta a mostrar el detall del correu seleccionat, tot això sense haver de canviar d'Activity.

Exemple – Primer fragment

- Definirem **dos Fragments**: un per al llistat i un altre per a la vista de detall. Al igual que a una Activity, cada Fragment es composa d'un fitxer xml (**layout**) per a la interfície i d'una **classe java** per a la lògica associada.
- El layout del **primer Fragment** a definir contindrà un control ListView, per al qual definirem un adaptador personalitzat per a mostrar dos camps per fila ("De" i "Assumpte"). El layout XML (l'anomenarem fragment_listado.xml) podria quedar de la següent forma:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <ListView
        android:id="@+id/LstListado"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>
</LinearLayout>
```

Exemple – Llibreria de suport

- Tot fragment ha de tenir associat, a més del layout, la seua pròpia classe java, que en aquest cas ha d'estendre de la classe **Fragment**.
- Els **Fragment** van aparèixer amb la versió 3 d'Android pel que si volem donar suport a versions anteriors hem d'utilitzar la llibreria de suport appcompat-v7.

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:27.1.1'
    implementation 'com.android.support.constraint:constraint-layout:1.1.3'
}
```

Exemple – Model Correu

- El primer que farem és **a definir el model** que emmagatzemarà cada correu.

```
public class Correo {
    private String de;
    private String asunto;
    private String texto;
    public Correo(String de, String asunto, String texto) {
        this.de = de;
        this.asunto = asunto;
        this.texto = texto;
    }
    public String getDe() {
        return de;
    }
    public String getAsunto() {
        return asunto;
    }
    public String getTexto() {
        return texto;
    }
}
```

Exemple – listitem_correo.xml

- En segon lloc creem el layout **listitem_correo** que contindrà cadascun dels elements del ListView.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/tvDe"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Persona"
        android:textSize="18sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>
    <TextView
        android:id="@+id/tvAsunto"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Asunto"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/tvDe"/>
</android.support.constraint.ConstraintLayout>
```

Exemple – AdaptadorCorreos

- En tercer lloc, definim l'adaptador del **ListView**.

```
public class AdaptadorCorreos extends ArrayAdapter<Correo> {
    private Context context;
    private Correo[] datos;
    public AdaptadorCorreos(Fragment context, Correo[] datos) {
        super(context.getActivity(), R.layout.listitem_correo, datos);
        this.datos = datos;
        this.context = context.getActivity();
    }
    @NonNull
    @Override
    public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {
        LayoutInflater inflater = LayoutInflater.from(context);
        View item = inflater.inflate(R.layout.listitem_correo, null);
        TextView tvDe = item.findViewById(R.id.tvDe);
        tvDe.setText(datos[position].getDe());
        TextView tvAsunto = item.findViewById(R.id.tvAsunto);
        tvAsunto.setText(datos[position].getAsunto());
        return item;
    }
}
```

Exemple – FragmentListado (I)

- En quart lloc, creem la **classe per al Fragment** estenent a la classe **Fragment**.

```
public class FragmentListado extends Fragment {
    private Correo[] datos = new Correo[] {
            new Correo("Persona 1", "Asunto del correo 1", "Texto del correo 1"),
            new Correo("Persona 2", "Asunto del correo 2", "Texto del correo 2"),
            new Correo("Persona 3", "Asunto del correo 3", "Texto del correo 3"),
            new Correo("Persona 4", "Asunto del correo 4", "Texto del correo 4"),
            new Correo("Persona 5", "Asunto del correo 5", "Texto del correo 5")
    };
    private ListView lstListado;
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_listado, container, false);
    }
    @Override
    public void onActivityCreated(@Nullable Bundle savedInstanceState) {
        super.onActivityCreated(savedInstanceState);
        lstListado = getView().findViewById(R.id.LstListado);
        lstListado.setAdapter(new AdaptadorCorreos(this, datos));
    }
}
```

Exemple – FragmentListado (II)

- En estendre la classe **Fragment** hi ha dos mètodes interessants que podem **sobreescriure**: onCreateView() i onActivityCreated().
- onCreateView() és l'equivalent a l'onCreate() de les Activities, i ací és on **assignarem** un **layout** determinat **al Fragment**. En aquest cas, haurem de “inflar-lo” mitjançant el mètode inflate() passant-li com a paràmetre l'ID del layout corresponent, en el nostre cas fragment_listado.
- onActivityCreate() s'executarà quan **l'Activity contenidora del Fragment estiga completament creada**. Aprofitem aquest esdeveniment per a obtenir la referència al control **ListView** i associar-li el seu adaptador.

Exemple – Segon fragment

- El segon **Fragment** s'encarregarà de mostrar la vista detall.
- En primer lloc definim el layout associat que anomenarem fragment_detalle.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/tvDetalle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="16dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:text="Detalle"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>
</android.support.constraint.ConstraintLayout>
```

Exemple – FragmentDetalle

- En aquest cas la classe Java associada al Fragment detall es llimitarà a **carregar el layout de la interfície** i definir un mètode públic anomenat mostrarDetalle() que ens ajude a assignar el contingut a mostrar.

```
public class FragmentDetalle extends Fragment {

    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, Bundle
savedInstanceState) {

        return inflater.inflate(R.layout.fragment_detalle, container, false);

    }

    public void mostrarDetalle(String texto) {

        TextView tvDetalle = getView().findViewById(R.id.tvDetalle);

        tvDetalle.setText(texto);

    }

}
```

Exemple – MainActivity

- Una vegada definits els **dos Fragments**, ens queda definir les Activities de la nostra aplicació, amb els seus respectius layouts que faran ús dels fragments que acabem d'implementar.
- Per a l'Activity **principal** definirem **3 layouts diferents**:
  - El primer d'ells per als casos en els quals l'aplicació s'execute en una pantalla “normal” (per exemple un telèfon mòbil)
  - Els **dos layouts restants** per a pantalles grans (per exemple una tablet, un pensat per a **orientació horitzontal** i un altre per a **orientació vertical**).
- Tots es diran activity_main.xml, i el que **determinarà quin s'utilitzarà** serà **la carpeta en la qual col·locarem cadascun**.
- El **primer layout** el col·locarem **en la carpeta** per defecte /res/layout.
- Els altres dos en les carpetes /res/layout-large (pantalla gran amb orientació horitzontal) i /res/layout-large-port (pantalla gran amb orientació vertical) respectivament.
- D'aquesta forma, segons la grandària i orientació de la pantalla **Android utilitzarà un layout o un altre de forma automàtica** sense que nosaltres tinguem que fer res més.

Exemple – MainActivity layout mòbil

- Per al cas de pantalla “normal”, l'Activity principal **mostrarà només el llistat de correus**, per la qual cosa el layout inclourà tan sols el fragment **FragmentListado**.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <fragment
        android:id="@+id/FrgListado"
        android:name="com.germangascon.fragments.FragmentListado"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>
</android.support.constraint.ConstraintLayout>
```

Exemple – DetalleActivity layout mòbil

- Per a la **vista de detall** també haurem de crear el seu layout, que anomenarem activity_detalle.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:layout_editor_absoluteY="81dp">
    <fragment
        android:id="@+id/FrgDetalle"
        android:name="com.germangascon.fragments.FragmentDetalle"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>
</android.support.constraint.ConstraintLayout>
```

Exemple – MainActivity layout tablet horitzontal

- El layout per al cas de **pantalla gran horitzontal**, serà de la següent forma:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:orientation="horizontal"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <fragment
      android:id="@+id/FrgListado"
      android:name="com.germangascon.fragments.FragmentListado"
      android:layout_width="0px"
      android:layout_height="match_parent"
      android:layout_weight="30"/>
   <fragment
      android:id="@+id/FrgDetalle"
      android:name="com.germangascon.fragments.FragmentDetalle"
      android:layout_width="0dp"
      android:layout_height="match_parent"
      android:layout_weight="70"/>
</LinearLayout>
```

- Incloem els dos fragment en la mateixa pantalla, tots dos dins d'un LinearLayout horitzontal, assignant al primer d'ells un pes de **30** i al segon de **70** perquè la columna de llistat ocupe un 30% de la pantalla a l'esquerra i la de detall ocupe la resta.

Exemple – MainActivity layout tablet vertical

- Per a la pantalla gran vertical serà practicament igual, només que usarem un **LinearLayout vertical**.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <fragment
      android:id="@+id/FrgListado"
      android:name="com.germangascon.fragments.FragmentListado"
      android:layout_width="match_parent"
      android:layout_height="0px"
      android:layout_weight="40"/>
   <fragment
      android:id="@+id/FrgDetalle"
      android:name="com.germangascon.fragments.FragmentDetalle"
      android:layout_width="match_parent"
      android:layout_height="0px"
      android:layout_weight="60"/>
</LinearLayout>
```

Exemple – Visualització (I)

- Arribats a aquest punt, ja podríem provar si es visualitza bé en mòbil i en tablets.
- Per a això cal tenir **dos emuladors**. Un amb **pantalla normal** i un altre amb **pantalla gran**.
- Podem **canviar l'orientació del dispositiu** prement **Ctrl+F12**.
- **Faltaria per implementar la lògica** de l'aplicació, és a dir, el que ha d'ocórrer en prémer un element de la llista de correus.
- Assignarem l'esdeveniment onItemClick() a la llista dins del mètode onActivityCreated() de la classe **FragmentListado**. Depenent del que s'està mostrant en pantalla:
  - Si existeix el Fragment de detall caldria **obtenir una referència** a ell i cridar al seu mètode mostrarDetalle() amb el text del correu seleccionat.
  - En cas contrari, hauríem de **navegar a l'Activity secundària DetalleActivity** per a mostrar el detall.

Exemple – Visualització (II)

- No obstant això existeix un problema, **un Fragment no té per què conèixer l'existència de cap altre**, és més, haurien de dissenyar-se de manera que siguen el més independents possible, d'aquesta forma conseguirem que puguen reutilitzar-se en diferents situacions sense problemes de dependències amb altres elements de la interfície.
- Per aquest motiu, el patró utilitzat normalment en aquestes circumstàncies no serà tractar l'esdeveniment en el propi Fragment, sinó definir i llançar un esdeveniment personalitzat en prémer l'item de la llista i **delegar a l'Activity contenidora** la lògica de l'esdeveniment, ja que ella sí que ha de conèixer quins Fragments componen la seua interfície.
- Per a això **definim una interfície** amb el mètode associat a l'esdeveniment, en aquest cas anomenada **ICorreosListener** amb un únic mètode anomenat onCorreoSeleccionado(), **declarem un atribut** amb aquesta interfície i definim un **mètode** setXXXListener() per a poder assignar l'esdeveniment des de fora de la classe.

Exemple – Interface ICorreosListener

- En primer lloc definim la interfície amb un només mètode: onCorreoSeleccionado()

```
public interface ICorreosListener {

    void onCorreoSeleccionado(Correo c);

}
```

Exemple – FragmentListado amb listener

```
public class FragmentListado extends Fragment {
    //...
    private ICorreosListener listener;
    //...
    @Override
    public void onActivityCreated(@Nullable Bundle savedInstanceState) {
        super.onActivityCreated(savedInstanceState);
        lstListado = getView().findViewById(R.id.LstListado);
        lstListado.setAdapter(new AdaptadorCorreos(this, datos));
        lstListado.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int pos, long id) {
                if(listener!=null) {
                    listener.onCorreoSeleccionado((Correo)lstListado.getAdapter().getItem(pos));
                }
            }
        });
    }
    public void setCorreosListener(ICorreosListener listener) {
        this.listener = listener;
    }
}
```

Exemple – MainActivity (I)

```
public class MainActivity extends AppCompatActivity implements  ICorreosListener {
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);

  FragmentListado frgListado = (FragmentListado)getFragmentManager().findFragmentById(R.id.FrgListado);
  //Si hemos utilizado la librería de soporte deberemos hacerlo de la siguiente forma:
  //FragmentListado frgListado = (FragmentListado)getSupportFragmentManager().findFragmentById(R.id.FrgListado);
  frgListado.setCorreosListener(this);
}
@Override
public void onCorreoSeleccionado(Correo c) {
  boolean hayDetalle = (getFragmentManager().findFragmentById(R.id.FrgDetalle) != null);
  if(hayDetalle) {
     ((FragmentDetalle)getFragmentManager().findFragmentById(R.id.FrgDetalle)).mostrarDetalle(c.getTexto());
  } else {
     Intent i = new Intent(this, DetalleActivity.class);
     i.putExtra(DetalleActivity.EXTRA_TEXTO, c.getTexto());
      startActivity(i);
  }
}
}
```

Exemple – MainActivity (II)

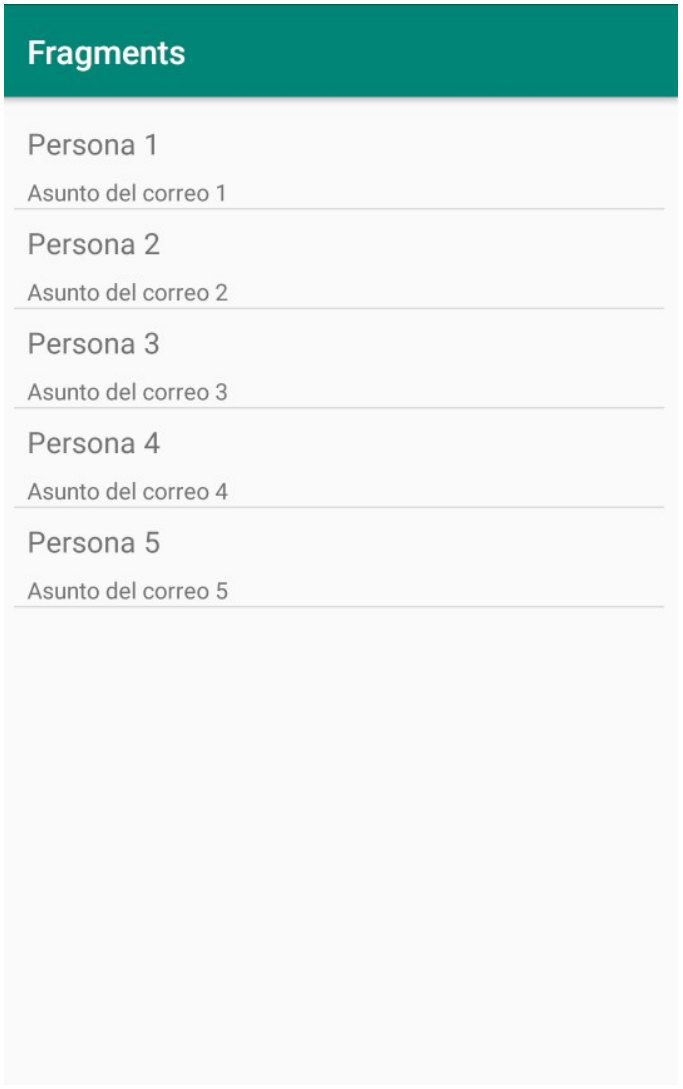
- Com podem observar, hem fet que **MainActivity** herete de la nostra interfície **ICorreosListener**, per tant ens basta passar this al mètode setCorreosListener().
- El mètode onCorreoSeleccionado() s'executarà quan se seleccione un determinat item de la llista.
- Si en la pantalla existeix el fragment de detall, simplement ho actualitzarem mitjançant mostrarDetalle().
- En cas contrari iniciarem l'activitat **DetalleActivity**. Per a això, hem de crear un nou **Intent** amb **la referència a aquesta classe**, i li afegirem com a **paràmetre extra** un camp de text **amb el contingut del correu seleccionat**. Finalment cridem a startActivity() per a iniciar la nova activitat.

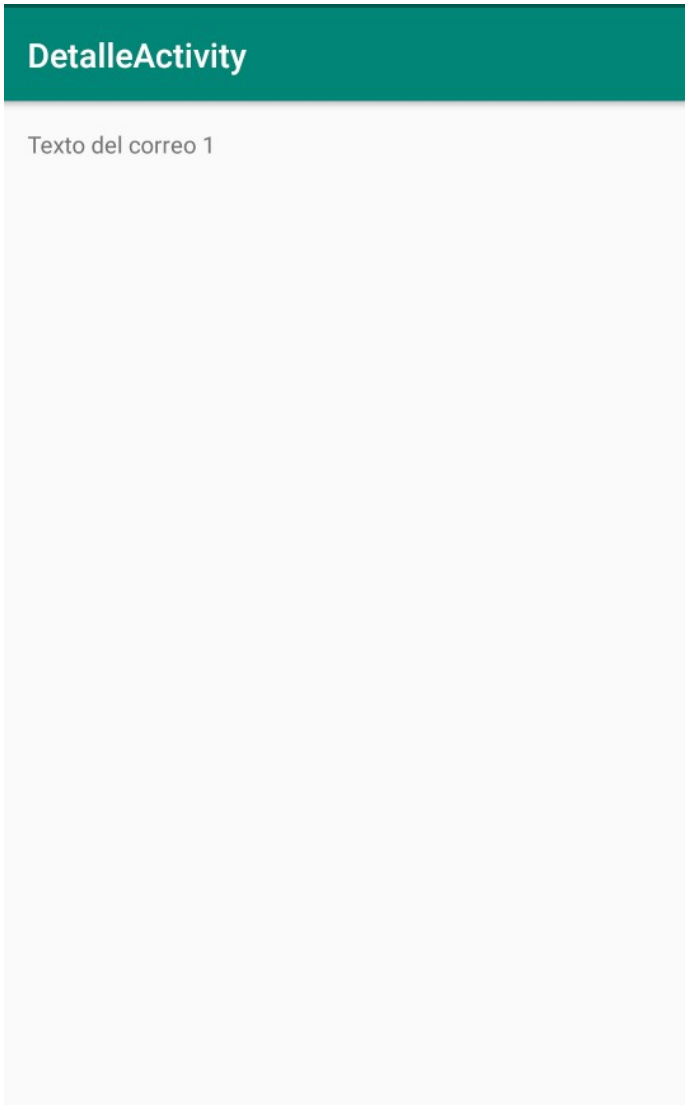
Exemple – DetalleActivity

- Finalment, només quedaria crear la classe **DetalleActivity** i afegir-la a l'AndroidManifest.xml.

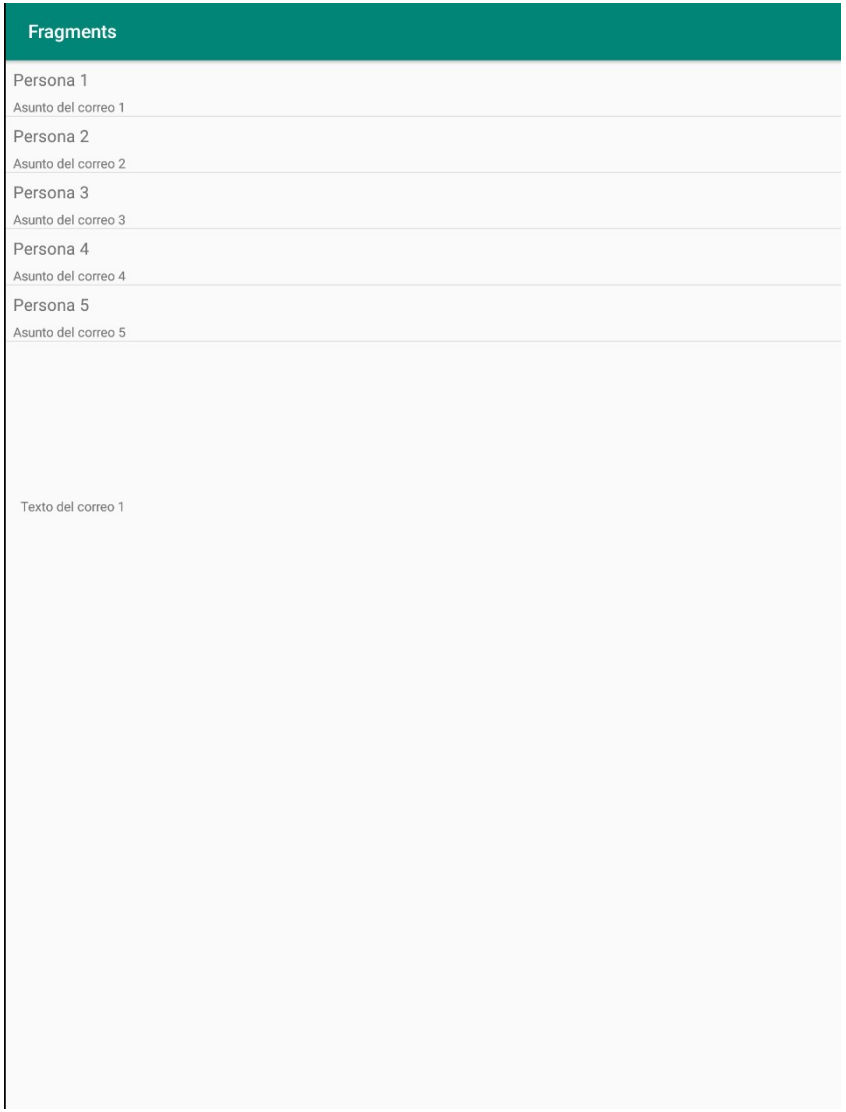
```
public class DetalleActivity extends AppCompatActivity {

    public static final String EXTRA_TEXTO = "com.germangascon.fragments.EXTRA_TEXTO";

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_detalle);

        FragmentDetalle detalle =
(FragmentDetalle)getFragmentManager().findFragmentById(R.id.FrgDetalle);

        detalle.mostrarDetalle(getIntent().getStringExtra(EXTRA_TEXTO));

    }

}
```

Exemple - Vista dispositius mòbils





Exemple - Vista tablets vertical



Exemple - Vista tablets horitzontal