


Open Opened 3 weeks ago by  **Rubén Montero**

Un primer fragmento

Resumen

- Hablaremos de *Fragments* en Android
- Añadiremos un `FrameLayout` a `activity_monsters.xml` que servirá de contenedor para distintos *fragments*
- Crearemos un primer *Fragment*
- Se verá en el contenedor cuando el usuario pulse el primer *item* de la barra inferior

Descripción

Los [Fragments](#) son algo así como *sub-actividades* en Android. Representan una parte reutilizable de la interfaz de usuario y definen y administran su propio diseño. También tienen su propio [ciclo de vida](#).

¡Trabajemos con *Fragments*!

La tarea

Como primer paso, **elimina** el `ImageView` de `activity_monsters.xml`:

```
- <ImageView
-     android:layout_width="0dp"
-     android:layout_height="0dp"
-     android:src="@drawable/togemon"
-     android:contentDescription="Un Digimon muy poderoso"
-     app:layout_constraintTop_toBottomOf="@id/monstersTitle"
-     app:layout_constraintStart_toStartOf="parent"
-     app:layout_constraintEnd_toEndOf="parent"
-     app:layout_constraintBottom_toTopOf="@id/bottomNavigation"/>
```

En su lugar, **añade** un `<FrameLayout>` [^1]. Así:

```
<FrameLayout
    android:id="@+id/fragmentContainer"
    android:layout_width="0dp"
    android:layout_height="0dp"
    app:layout_constraintTop_toBottomOf="@id/monstersTitle"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintBottom_toTopOf="@id/bottomNavigation"
/>
```

Ahora que hemos sustituido nuestro `<ImageView>` por un `<FrameLayout>`, lo usaremos como *contenedor* de los *Fragments*.

Fragments

Tienen dos elementos: Interfaz XML y clase Java.

Interfaz XML

Haz *click* derecho en *res > layout* y **selecciona** *New > Layout Resource File*.

El **Root Element** será `androidx.constraintlayout.widget.ConstraintLayout`, como viene por defecto.

Dale el nombre `fragment_1.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

</androidx.constraintlayout.widget.ConstraintLayout>
```

Ahora, dentro de este **ConstraintLayout** **añade** la imagen que borramos anteriormente. **Así:**

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:src="@drawable/togemon"
        android:contentDescription="Un Digimon muy poderoso" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

(Como ves, se usa **match_parent** para **Layout_width** y **Layout_height**. Esto hace que la imagen encaje (rellene) el padre).

Clase Java

Nuestro *Fragment* debe tener una clase asociada.

Haz click derecho en *src > com.afundacion.fp.library* y **selecciona** *New > Java Class*.

Dale el nombre **Fragment1**:

```
public class Fragment1 {

}
```

A continuación, **declara** que la clase *herede* de la superclase **Fragment**. **Así:**

```
import androidx.fragment.app.Fragment;

public class Fragment1 extends Fragment {

}
```

Además, estamos *obligados* a proveer un *constructor vacío*. Esta es una cuestión propia de cómo Android gestiona los *Fragments*. **Añádelo** así:

```
import androidx.fragment.app.Fragment;

public class Fragment1 extends Fragment {

    public Fragment1() {
    }

}
```

¡Ya casi hemos terminado con nuestro *Fragment*!

Sólo falta implementar el método **onCreateView**. Este método es análogo al **onCreate** de las actividades.

Teclea **onCreateView** debajo del constructor y verás que Android Studio nos permite autocompletar:

```
public class Fragment1 extends Fragment {

    public Fragment1() {
    }

    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        return super.onCreateView(inflater, container, savedInstanceState);
    }

}
```

```

    }
}

```

Para terminar este método, **reemplaza** la línea `return super.onCreateView` por el siguiente contenido:

```

@Override
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
    // 'Inflamos' el XML fragment_1 y lo ponemos en el 'container'
    return inflater.inflate(R.layout.fragment_1, container, false);
}

```

(Usamos el método `.inflate` del objeto provisto `inflater`. Esto sirve para leer el archivo `fragment_1.xml` y convertirlo un objeto tipo `View` que Android podrá usar).

Uniendo barra inferior y *Fragments*

Ahora, en un par de líneas de código, vamos a [unir los dos conceptos](#) con los que hemos trabajado recientemente.

Para empezar, en `MonstersActivity.java`, **instanciamos** un nuevo `Fragment1` dentro del `if` del ejercicio anterior, correspondiente al código que se ejecuta cuando el usuario hace *click* en el primer elemento de la barra inferior:

```

import androidx.fragment.app.Fragment;

public class MonstersActivity extends AppCompatActivity {
    // ...

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        // ...
        bar.setOnItemClickListener(new NavigationBarView.OnItemClickListener() {
            @Override
            public boolean onNavigationItemSelected(@NonNull MenuItem item) {
                if (item.getItemId() == R.id.item1) {
                    // Aquí va el Toast de la tarea anterior. No lo borres
                    // ...

                    // Instanciamos un Fragment1
                    Fragment myFragment = new Fragment1();
                }
                // Aquí van los if de item2 e item3
                // ...
            }
        });
    }
}

```

A continuación, con una línea **indica** que quieres *reemplazar* (visualmente) el *contenedor* con el *Fragment*.

Empieza así:

```

Fragment myFragment = new Fragment1();
getSupportFragmentManager()

```

...para obtener una *instancia* de un objeto administrador de *Fragments*, sobre el que indicarás que quieres empezar una nueva *transacción* así:

```

Fragment myFragment = new Fragment1();
getSupportFragmentManager().beginTransaction()

```

...y ahora sólo falta indicar que queremos reemplazar una vista con un `android:id` específico por el *Fragment* de la línea anterior. **Termina** así:

```

Fragment myFragment = new Fragment1();
// Recuerda que R.id.fragmentContainer es el id de nuestro FrameLayout añadido al principio
getSupportFragmentManager().beginTransaction().replace(R.id.fragmentContainer, myFragment)

```

¡Ah! Un último paso. **Haz** efectivo el cambio con `.commit();`:

```

Fragment myFragment = new Fragment1();

```

```
getSupportFragmentManager().beginTransaction().replace(R.id.fragmentContainer, myFragment).commit();
```

¡Ya has conseguido que se vea un *Fragment* cuando el usuario hace *click* en **Digimon 1** de la barra inferior!

Pero inicialmente **MonstersActivity** está vacía...

Cierto. Hemos eliminado el `<ImageView>` al principio así que era algo que cabía esperar.

Añade la siguiente línea al principio de `onCreate` en `MonstersActivity.java` para que, inicialmente, el *Fragment* sea visible:

```
public class MonstersActivity extends AppCompatActivity {  
    // ...  
  
    @Override  
    protected void onCreate(@Nullable Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_monsters);  
  
+        // Aquí, inicialmente también añadimos un Fragment1 a la interfaz  
+        getSupportFragmentManager().beginTransaction().replace(R.id.fragmentContainer, new Fragment1()).commit();  
  
        // Otro código, como .setOnItemSelectedListener  
        // ...  
    }  
}
```

¡Listo! Puedes verificar cómo funciona.

Por último

Sube tus cambios al repositorio en un nuevo *commit*.

[^1]. Es un tipo de *layout* especial que se usa normalmente para albergar *un único hijo*.



[Rubén Montero @ruben.montero](#) changed milestone to [%Sprint 1](#) 3 weeks ago



[Ania Blanco @ania.blanco](#) mentioned in commit [4167e77d](#) 1 week ago