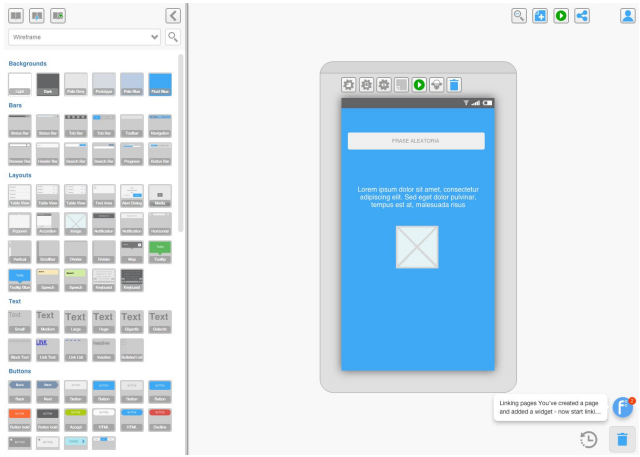
CREACIÓN DE UNA ANDROID APP

PROYECTO ANDROID

Desarrollo de una aplicación que muestre frases de inspiración cada vez que hacemos “tap” en un botón.

DISEÑO DEL CONCEPTO

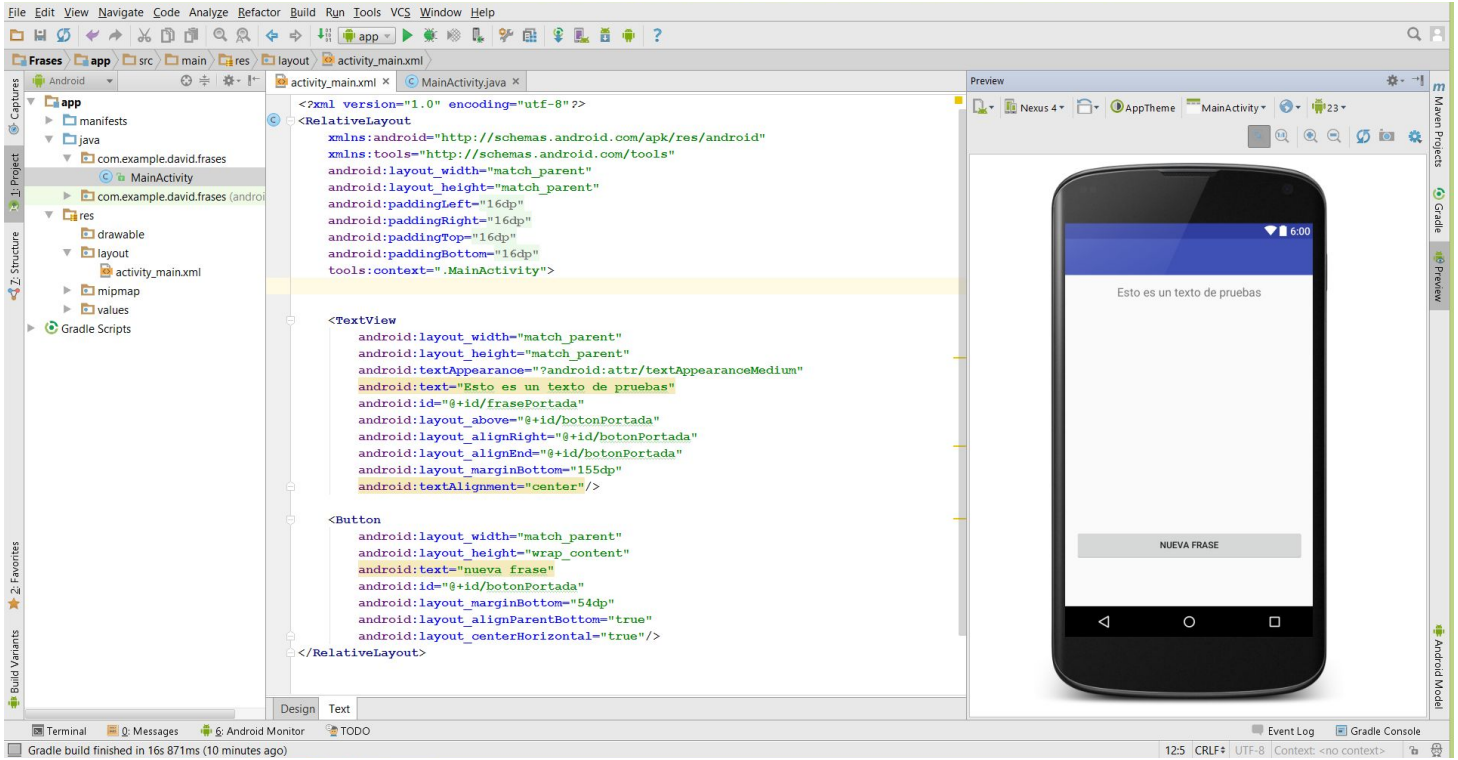
Antes de comenzar el desarrollo, es bueno dedicar un tiempo al diseño del mockup de nuestra app. Una gran herramienta de prototipdo, es: https://www.fluidui.com



Puedes hacer de forma sencilla y gratuita, un prototipo de cómo va a ser el resultado final sin demasiado esfuerzo.

FASE 1

En primer lugar, vamos a desarrollar la vista de la app. Para este ejemplo, debemos añadir 2 elementos, un TextViews y un Button que es el aporta la interactividad. El resultado debe ser algo asi:



FASE 1

Recuerda que es una buena práctica cambiar los id de los elementos. Una vez que tenemos la estructura, vamos a personalizar su estética, para ello cambiamos los colores, puedes usar los que quieras, pero los recomendados, aparecen en esta lista:

https://www.google.com/design/spec/style/color.html

Una cosa interesante, es quitar el "Action Bar" de la app, para tener más espacio y que todo quede más limpio. Para ello, abre el archivo: styles.xml (dentro de la carpeta values) y modifica esta linea:

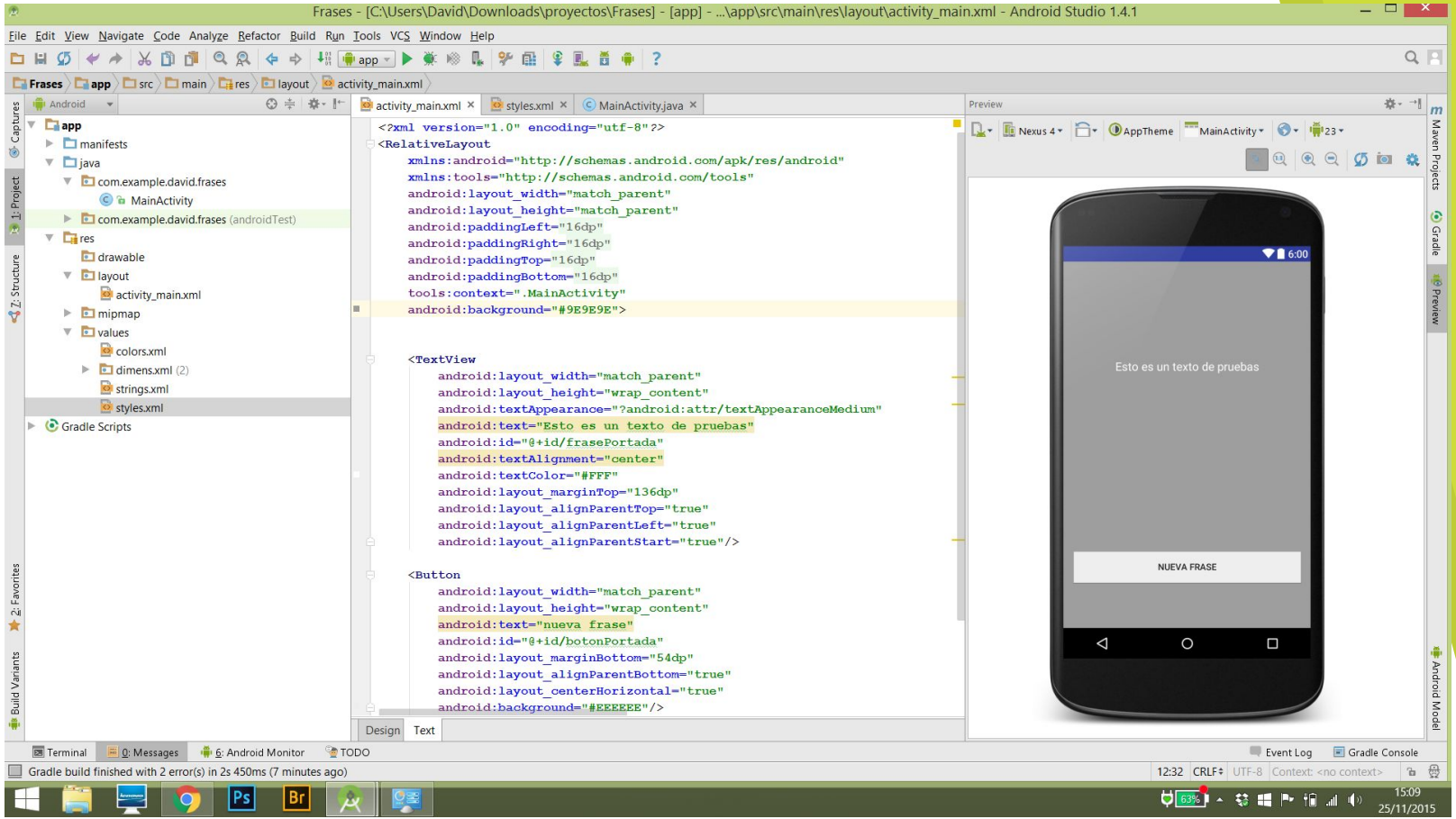
```
<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
```

por esta otra:

```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
```

Ahora el resultado debe ser algo asi:

FASE 1


Frases - [C:\Users\David\Downloads\proyectos\Frases] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 1.4.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Frases app src main res layout activity_main.xml
Android
app
manifests
java
com.example.david.frases
MainActivity
com.example.david.frases (androidTest)
res
drawable
layout
activity_main.xml
mipmap
values
colors.xml
dimens.xml (2)
strings.xml
styles.xml
Gradle Scripts
activity_main.xml
styles.xml
MainActivity.java
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingLeft="16dp"
android:paddingRight="16dp"
android:paddingTop="16dp"
android:paddingBottom="16dp"
tools:context=".MainActivity"
android:background="#9E9E9E">
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceMedium"
android:text="Esto es un texto de pruebas"
android:id="@+id/frasePortada"
android:textAlignment="center"
android:textColor="#FFF"
android:layout_marginTop="136dp"
android:layout_alignParentTop="true"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true"/>
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="nueva frase"
android:id="@+id/botonPortada"
android:layout_marginBottom="54dp"
android:layout_alignParentBottom="true"
android:layout_centerHorizontal="true"
android:background="#EEEEEE"/>
Design
Text
Preview
Nexus 4
AppTheme
MainActivity
23
6:00
Esto es un texto de pruebas
NUEVA FRASE
Terminal
0: Messages
6: Android Monitor
TODO
Event Log
Gradle Console
Gradle build finished with 2 error(s) in 2s 450ms (7 minutes ago)
12:32 CRLF UTF-8 Context: <no context>
15:09
25/11/2015

FASE 1

Vamos a jugar ahora con los paddings de la app. Vi vas al activity_main.xml e intenta smodificar esta linea:

```
android:paddingRight="16dp"
```

Notarás que automáticamente el la cambia a esto otro:

```
android:paddingRight="@dimen/activity_horizontal_margin"
```

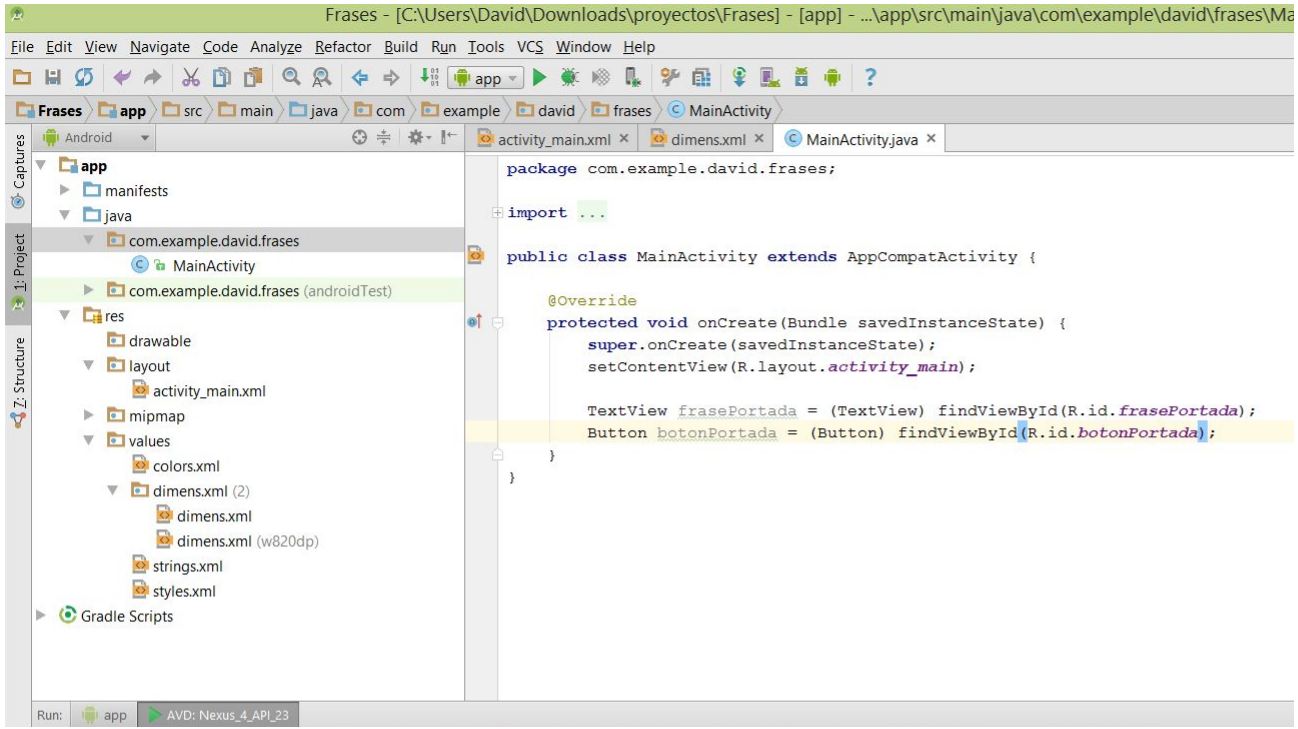
Esto es correcto, ahora debemos actualizar los valores en: dimens.xml (dentro de la carpeta values)

Notarás que ahora sí, inmediatamente cambia el padding.

Tomate tu tiempo para crear una vista que te guste y pasamos al siguiente apartado, donde comenzamos a desarrollar la app.

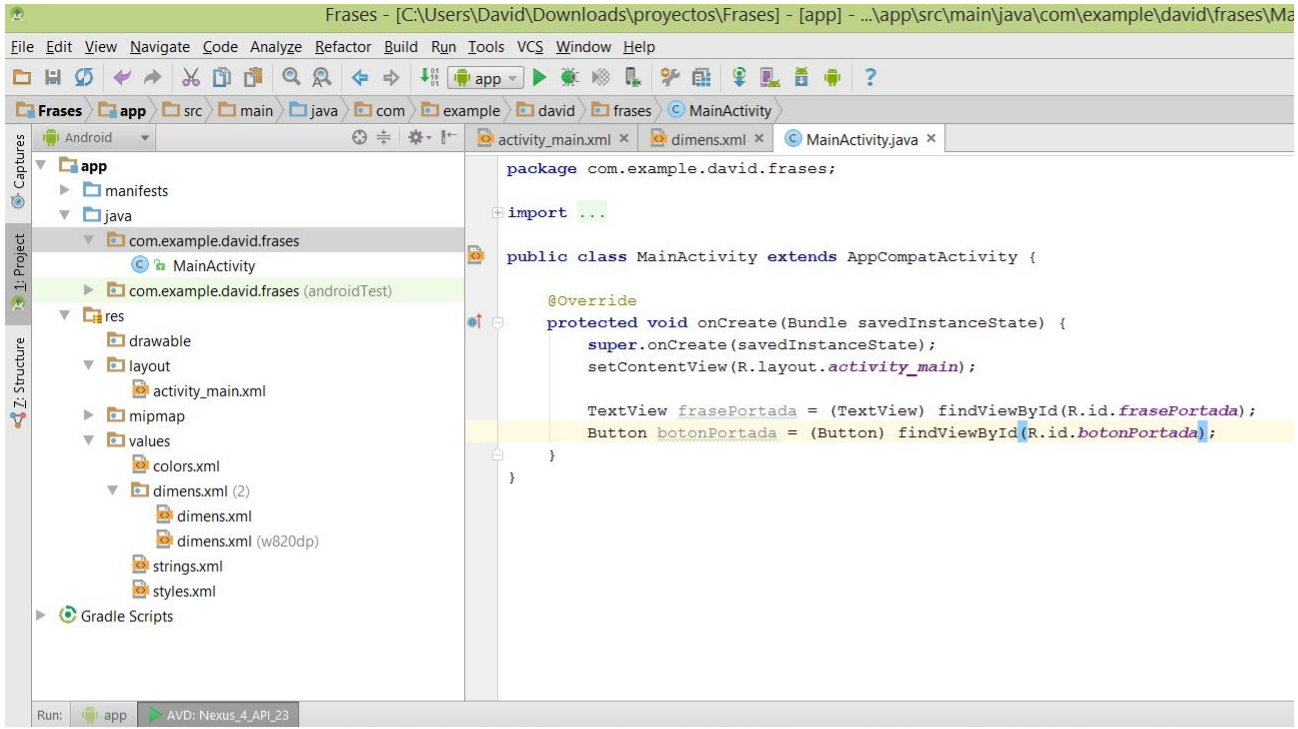
FASE 2

En esta fase, lo primero que debemos hacer es referenciar los elementos de la vista, declarando un par de variables. Dentro del método onCreate() del archivo MainActivity vamos a hacer esto:



FASE 2

En esta fase, lo primero que debemos hacer es referenciar los elementos de la vista, declarando un par de variables. Dentro del método onCreate() del archivo MainActivity vamos a hacer esto:



FASE 2

Siguiendo con el mismo planteamiento, añadimos un método onclicklistener para determinar la acción del botón.

```
botonPortada.setOnClickListener( new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        frasePortada.setText("Nuestra recompensa se encuentra en el esfuerzo y no en el resultado. Un esfuerzo total es una victoria completa.\n Mahatma Gandhi");
    }
});
```

Una vez hecho esto, haz click en emular y presiona el botón, si lo hemos hecho todo bien, la frase se mostrará en pantalla.

FASE 2

¡Genial! Ahora vamos a darle un poco de dinamismo, ya que en este momento, mostramos un string estático y lo que queremos es implementar un sencillo algoritmo que muestre frases de forma aleatoria.

Lo primero, es crear una variable vacia, a la que luego le añadiremos las frases, dentro del método onClick del botón:

```
public void onClick(View v) {
     String frase = "";
     frasePortada.setText(frase);
}
```

Ahora, vamos a usar el método Random(); de Java, que hace exactamente eso, seleccionar al azar. Lo más práctico, es generar un número aleatorio, y luego asociarlo a una de las frases. Añadimos el método, y le pasamos un rango para evitar números desorbitados, debajo de :

```
String frase = "";
```

incluye esto:

```
Random alAzar = new Random();
int numeroAlAzar = alAzar.nextInt( 4);
```

FASE 2

Vamos a comprobar que nuestro planteamiento, de momento funciona, vamos a convertir el número int, que recibimos del método random en un texto, asi podemos pasarlo y verlo en el emulador. En lugar de una frase, tendremos un número, pero comprobaremos que todo va en el camino correcto:

```
public void onClick(View v) {
        String frase =  "";


        Random alAzar =  new Random();
        int numeroAlAzar = alAzar.nextInt( 5);


        frase = numeroAlAzar  +  "";


        frasePortada.setText(frase);
    }
});
```

numeroAlAzar, es un valor numérico, pero añadiendole + “” lo convertimos en el equivalente en texto. Emula la app y comprueba el resultado.

FASE 3

En esta fase, vamos a resolver el almacenamiento de frases mediante un array. Se podria usar una base de datos del tipo SQLite pero sería demasiado tedioso para un caso asi.

```
String[] frases = {
        "Nuestra recompensa se encuentra en el esfuerzo y no en el resultado. Un esfuerzo total es una victoria completa.\n Mahatma Gandhi",
        "Si ya sabes lo que tienes que hacer y no lo haces entonces estás peor que antes.\n Confucio",
        "Buscando el bien de nuestros semejantes, encontramos el nuestro.\n Platon",
        "Es mejor ser rey de tu silencio que esclavo de tus palabras.\n William Shakespeare"
};
```

Y posteriormente, la variable frases debe cargar la frase asociada, para evitar tener que estar contando, vamos a usar el “length”

```
int numeroAlAzar = alAzar.nextInt(frases. length);
```

FASE 3

Finalmente, antes de cargar la frase, asociamos la variable frase, a una de las frases de forma aleatoria y el resultado quedaría asi:

```
public void onClick(View v) {
   String[] frases = {
           "Nuestra recompensa se encuentra en el esfuerzo y no en el resultado. Un esfuerzo
total es una victoria completa.\n Mahatma Gandhi",
           "Si ya sabes lo que tienes que hacer y no lo haces entonces estás peor que antes.\n
Confucio",
           "Buscando el bien de nuestros semejantes, encontramos el nuestro.\n Platon",
           "Es mejor ser rey de tu silencio que esclavo de tus palabras.\n William Shakespeare"
   };

   String frase = "";
   Random alAzar = new Random();
   int numeroAlAzar = alAzar.nextInt(frases.length);

   frase = frases[numeroAlAzar];

   frasePortada.setText(frase);
}
```