Ciclo 4
Fundamentos de programación
Aplicaciones móviles Android con Java
Sesión 4: Metodologia de diseño

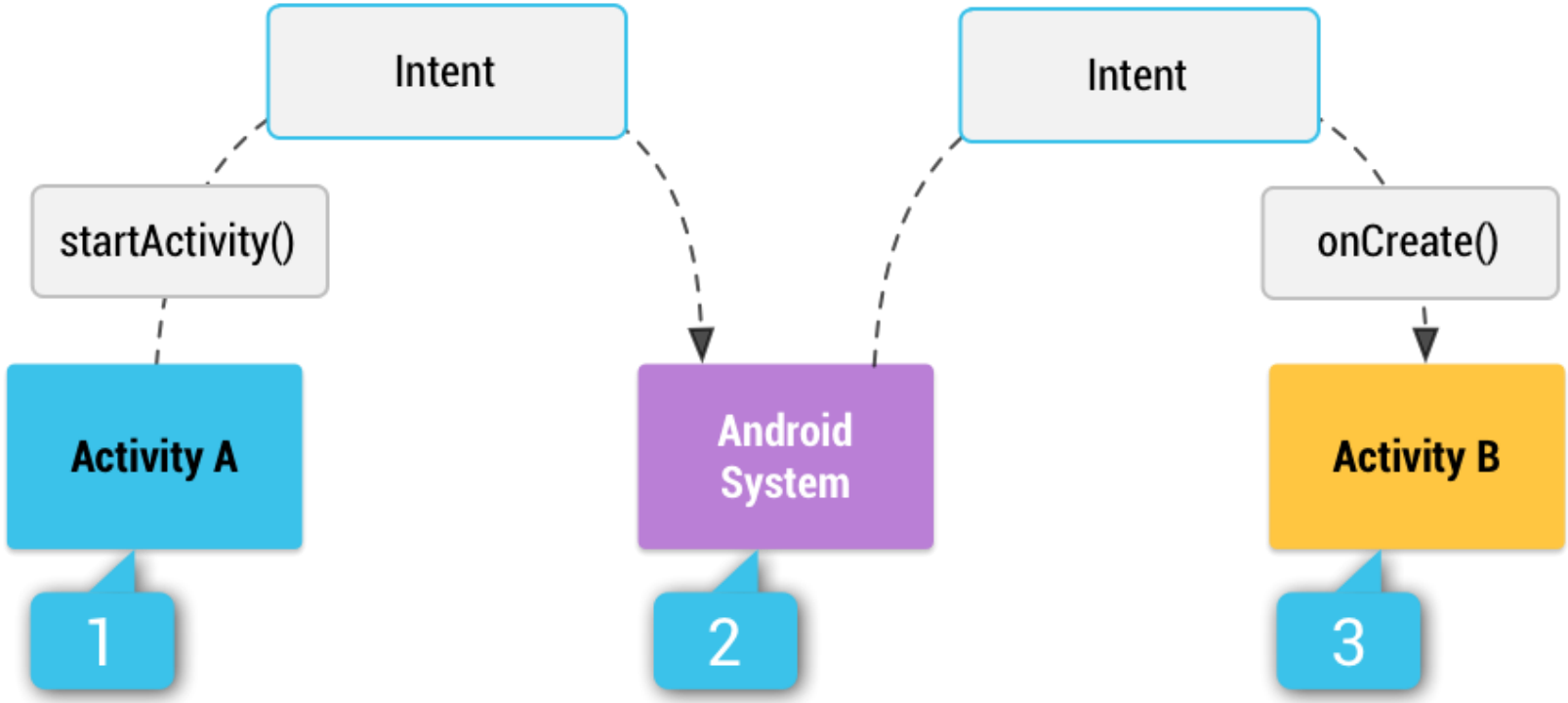
Programa Ciencias de la Computación e Inteligencia Artificial
Escuela de Ciencias Exactas e Ingeniería
Universidad Sergio Arboleda
Bogotá

Agenda

1. Intención (Intent)
2. usando código, XML, diseño visual de vistas, edición visual de las vistas, material design, paleta de colores
3. Creación y uso de iconos e imágenes.
4. Wireframe
5. Mockup
6. Prototipo

1. Intención (Intent)

El Intent es un objeto de mensajería que puedes usar para solicitar una acción de otro componente de una app. Si bien las intents facilitan la comunicación entre componentes de varias formas, existen tres casos de uso principales:

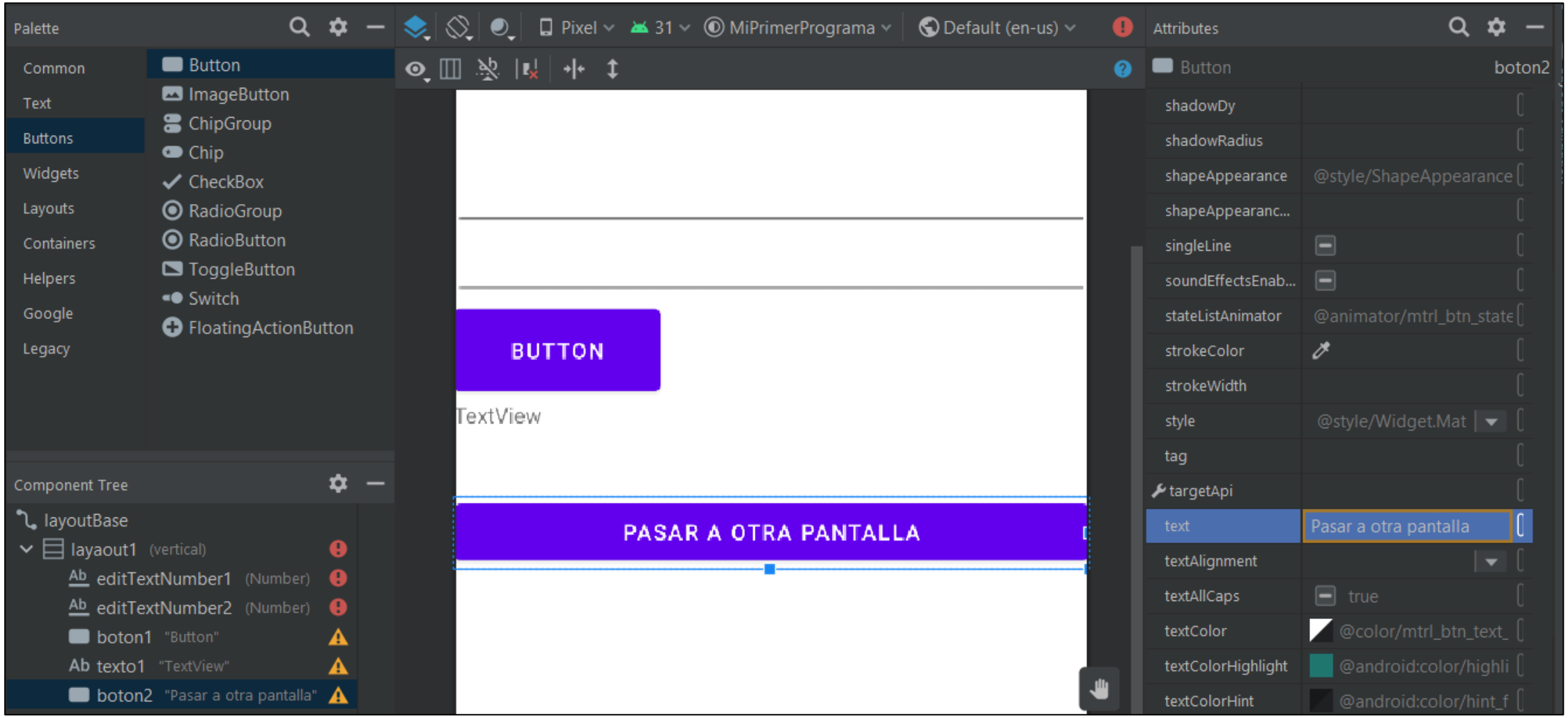


1. **Iniciar una actividad (por ahora solo usaremos este caso)**
2. **Iniciar un servicio**
3. **Transmitir una emisión**

Tomado de: https://developer.android.com/guide/components/intents-filters?hl=es-419

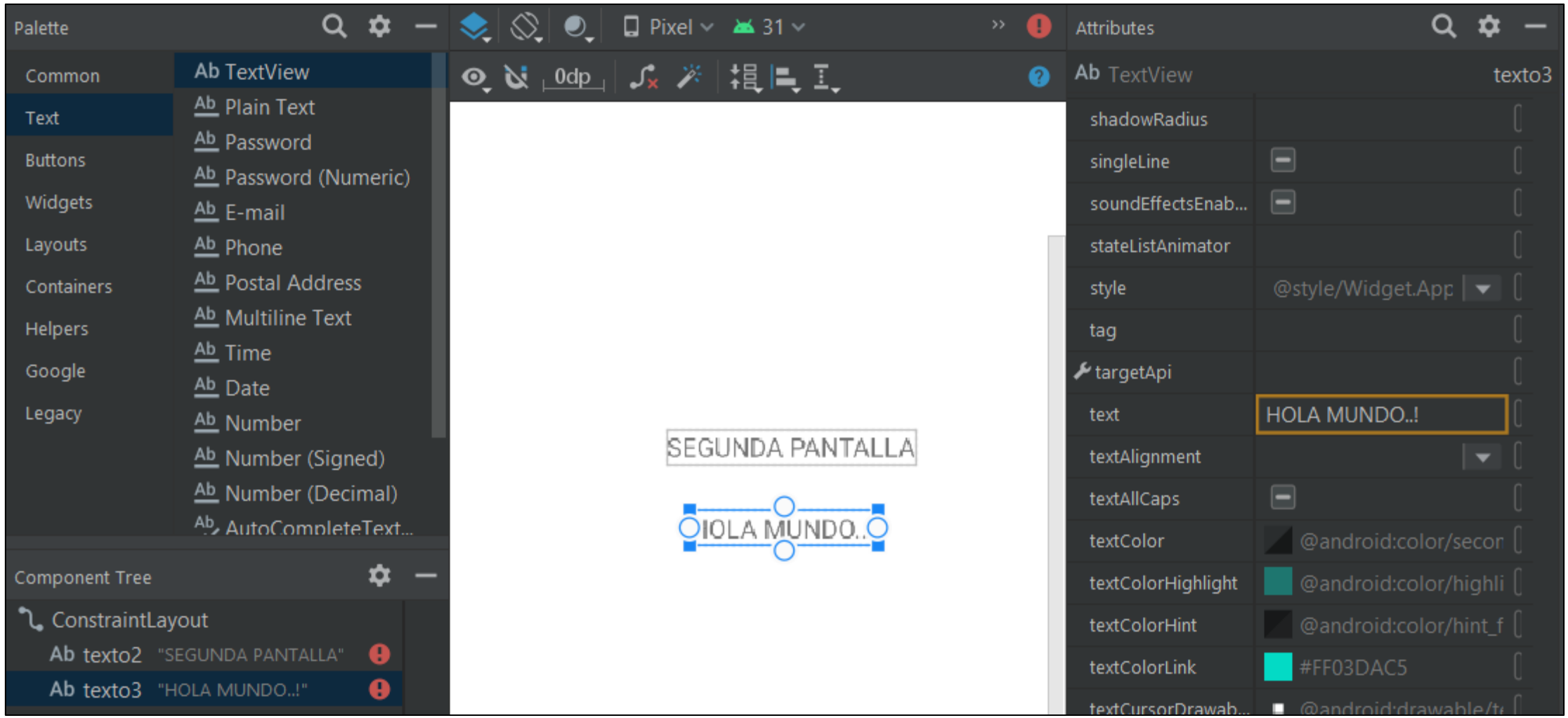
1. Intención (Intent)

Para pasar de una pantalla a otra o Actividad, usaremos el Intent para hacerlo:



PANTALLA 1

1. Intención (Intent)



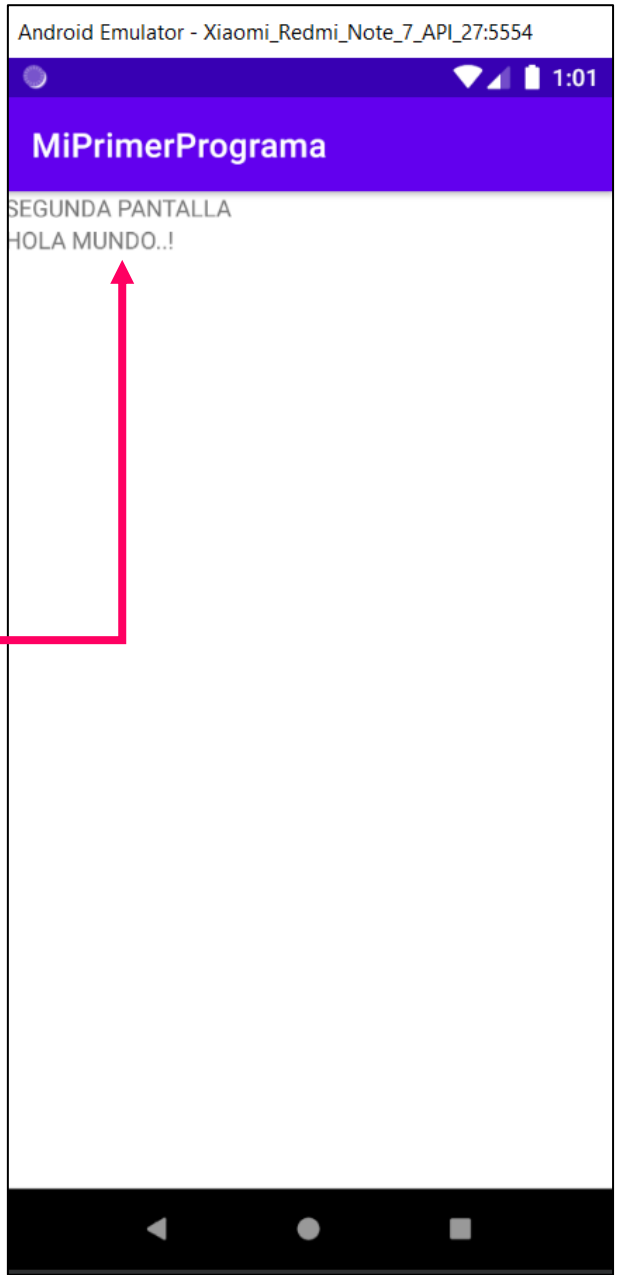
PANTALLA 2

2. Cómo iniciar otra actividad

Dentro de otro botón en el proyecto MiPrimerPrograma ponemos el código para cambiar a otra pantalla usando la clase Intent.

Intent nombreActividad = new Intent (contexto, nuevaActividad);

```
//--------------------------------------------------------------------
Button botonSiguiente =(Button) findViewById(R.id.boton2);
botonSiguiente.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        Intent pantalla2 = new Intent(getApplicationContext(), MainActivity2.class);
        startActivity(pantalla2);
    }
});
//--------------------------------------------------------------------
```

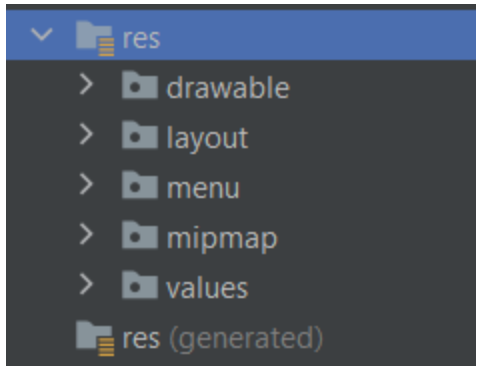


3. Programa completo

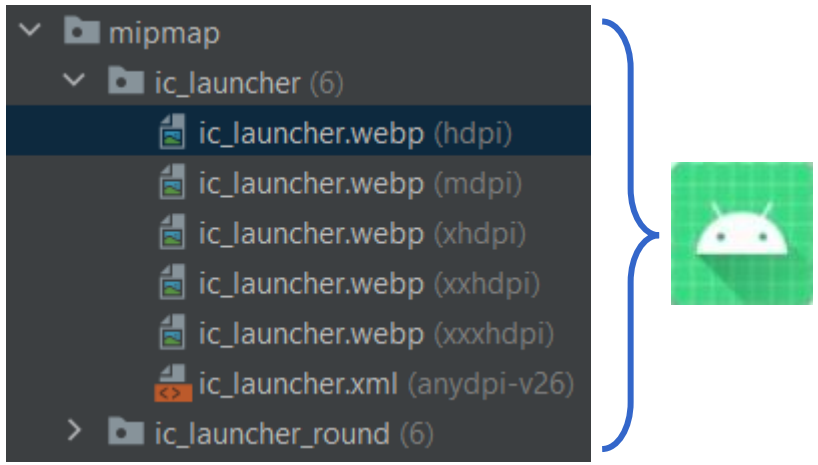

```
public class MainActivity extends AppCompatActivity {
    private EditText myEditText1;
    private EditText myEditText2;
    private TextView myTexto1;
    private Button myBoton1, botonSiguiente;
    //*****************************************************************
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        myEditText1 =(EditText)findViewById(R.id.editTextNumber1);
        myEditText2= (EditText) findViewById(R.id.editTextNumber2);
        myTexto1 =(TextView) findViewById(R.id.texto1);
        //-----------------------------------------------------------------
        Button myBoton1 =(Button) findViewById(R.id.boton1);
        // Se implementa el escuchador del evento clic usando una clase anonima
        myBoton1.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                String valor1=myEditText1.getText().toString();
                String valor2=myEditText2.getText().toString();
                int nro1=Integer.parseInt(valor1);
                int nro2=Integer.parseInt(valor2);
                int suma=nro1+nro2;
                String resu=String.valueOf(suma);
                myTexto1.setText(resu);
            }
        });
        //-----------------------------------------------------------------
        Button botonSiguiente =(Button) findViewById(R.id.boton2);
        botonSiguiente.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                Intent pantalla2 = new Intent(getApplicationContext(), MainActivity2.class);
                startActivity(pantalla2);
            }
        });
        //-----------------------------------------------------------------
    //************************************ MENU ****************************
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menuopciones, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id==R.id.opcion1) {
            Toast.makeText(this,"Se seleccionó la primera opción",Toast.LENGTH_LONG).show();
        }
        if (id==R.id.opcion2) {
            Toast.makeText(this,"Se seleccionó la segunda opción",Toast.LENGTH_LONG).show();
        }
        if (id==R.id.opcion3) {
            Toast.makeText(this,"Se seleccionó la tercera opción", Toast.LENGTH_LONG).show();
        }
        return super.onOptionsItemSelected(item);
    }
    //*********************************************************************
} //fin de la clase
```

4. Icono de la aplicación

En la carpeta res, hay cinco carpetas llamadas:



Los íconos e imágenes se almacenan en la carpeta mipmap, con cinco carpetas llamadas:



Las resoluciones de los dispositivos Android pueden ser muy diferentes (un celular, una tablet, un televisor etc.), por eso es importante proporcionar múltiples copias de cada imagen de recursos a diferentes resoluciones y almacenarlos en las carpetas nombradas respetando las siguientes reglas:

```
 res/mipmap-mdpi/
     El ícono debe ser de 48*48 píxeles.
res/mipmap-hdpi/
     150% del tamaño de las imágenes almacenadas en la carpeta mipmap-mdpi
     El ícono debe ser de 72*72 píxeles.
res/mipmap-xhdpi/
     200% del tamaño de las imágenes almacenadas en la carpeta mipmap-mdpi
     El ícono debe ser de 96*96 píxeles.
res/mipmap-xxhdpi/
     300% del tamaño de las imágenes almacenadas en la carpeta mipmap-mdpi
     El ícono debe ser de 144*144 píxeles.
res/mipmap-xxxhdpi/
     400% del tamaño de las imágenes almacenadas en la carpeta mipmap-mdpi
     El ícono debe ser de 192*192 píxeles.
res/mipmap-anydpi-v26/
     Archivo XML adaptable.
```

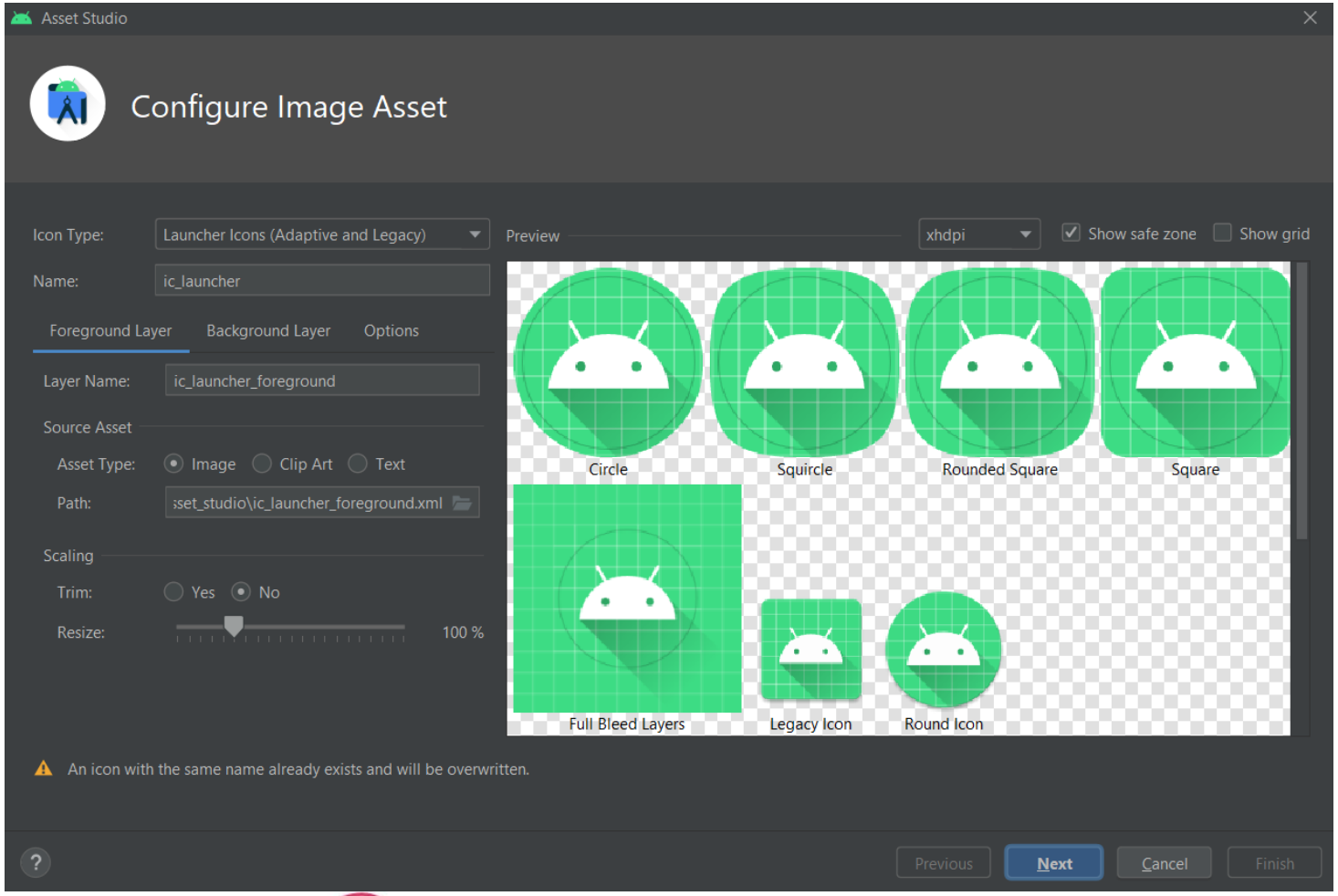

https://www.tutorialesprogramacionya.com/kotlinparaandroidya/detalleconcepto.php?punto=22&codigo=70&inicio=15

4. Icono de la aplicación

Para iniciar **Image Asset Studio**, sigue estos pasos:

1.En la ventana **Project**, selecciona la vista de Android.

2.Haz clic con el botón derecho en la carpeta **res** y selecciona **New** > **Image Asset**.



Al usar la herramienta se puede buscar una imagen corporativa y la herramienta lo copia en los diferentes tamaños de forma automática.

https://developer.android.com/studio/write/image-asset-studio?hl=es-419

Herramienta de generación de iconos: https://romannurik.github.io/AndroidAssetStudio/

4. Icono de la aplicación


```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="usa.sesion1.miprimerprograma">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="MiPrimerPrograma"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MiPrimerPrograma">
        <activity
            android:name=".MainActivity2"
            android:exported="true" />
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

En el archivo **manifest** y resaltado dentro de los recuadros amarillos, se encuentra el nombre de los iconos usados en la App.

Si se mantiene el mismo nombre al cambiar las imágenes los cambios se hacen automáticos. Pero si se cambian los nombres al momento de crearlos, hay que procurar venir al **manifest** y cambiarlos.

5. Recurso de imagen

Hacer referencia a un recurso de imagen en el código:

Se puede hacer referencia a un recurso de imagen de manera genérica desde el código y, cuando se ejecute tu app, se mostrará automáticamente la imagen correspondiente según el dispositivo:

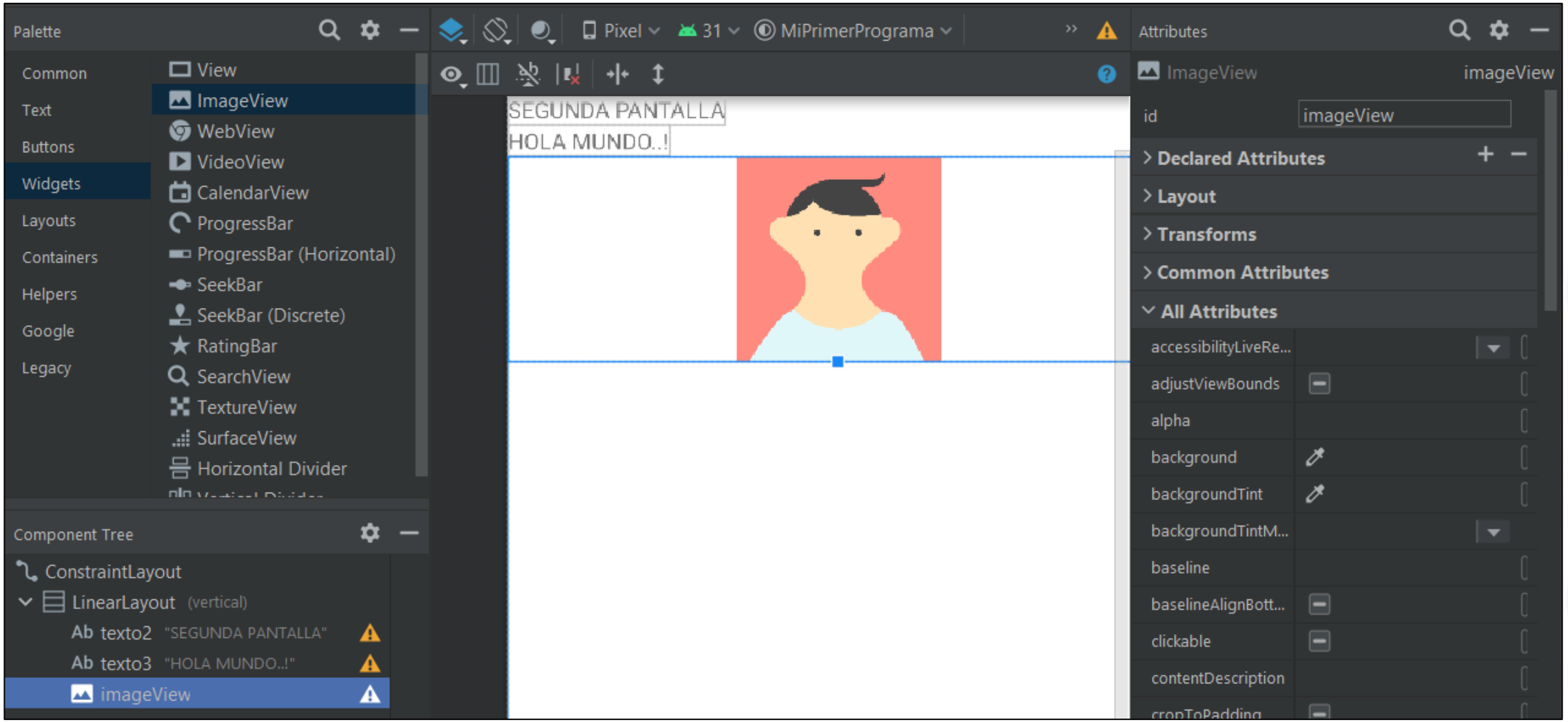
En la mayoría de los casos, puedes hacer referencia a recursos de imagen como **@drawable** en código XML o Drawable en código Java.

Por ejemplo, en el siguiente código XML de diseño, se muestra el elemento de diseño en una **ImageView**:

```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:src = "@drawable/abc_vector_test"
    tools:srcCompat="@tools:sample/avatars" />
```


https://developer.android.com/studio/write/image-asset-studio?hl=es-419

5. Recurso de imagen



5. Recurso de imagen


```java
public class MainActivity2 extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        Resources res = getResources();
        Drawable drawable = res.getDrawable(R.drawable.usa1, getTheme());

        ImageView imagen1 = (ImageView)findViewById(R.id.imageView);
        imagen1.setImageDrawable(drawable);

    }
}
```

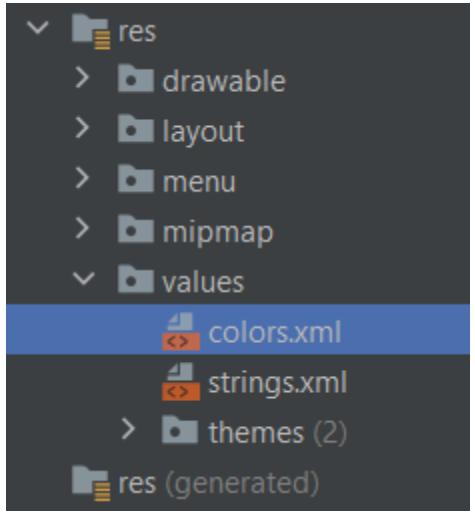
5. Recurso de imagen


```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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=".MainActivity2">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <TextView
            android:id="@+id/texto2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="SEGUNDA PANTALLA" />
        <TextView
            android:id="@+id/texto3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="HOLA MUNDO..!" />
        <ImageView
            android:id="@+id/imageView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:src = "@drawable/abc_vector_test"
            tools:srcCompat="@tools:sample/avatars" />
    </LinearLayout>

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

Archivo Manifest

6. Colores




```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

https://developer.android.com/guide/topics/resources/color-list-resource?hl=es

https://desarrollador-android.com/material-design/diseno-material-design/estilo/color/

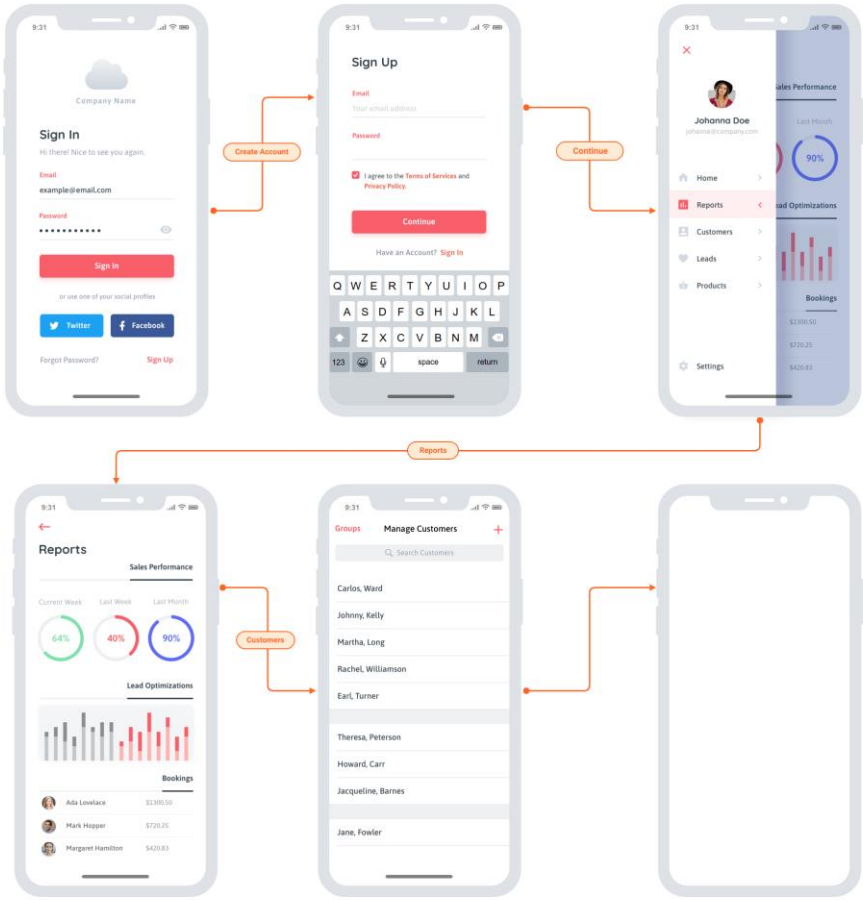
7. Diseño con Mockup

https://programmerclick.com/article/1165192066/

Una vez conocemos algunos objetos visuales y su funcionamiento, y como pasar entre pantallas, existen varias técnicas y herramientas hechas para definir un estilo de navegación entre pantallas, usando menús, botones y cualquier otro objeto visual que permita una navegación intuitiva a través de la aplicación.

Sin embargo, un trazado a mano alzada también es valido, antes de usar alguna herramienta tecnológica.

Lo importante es no perder la idea y diseñar con los objetos aprendidos o que nos son familiares de otras aplicaciones y que se pueden implementar mas adelante.



https://moqups.com/es/templates/wireframes-mockups/mobile-app/

8. Wireframe

Se trata de una especie de boceto de la aplicación móvil en el que definimos, sin nada de estilo o diseño, qué cosas deben aparecer y dónde van a estar en nuestra app. También añadiremos una explicación visual de cómo va a ser la interacción. Es importante trabajarlo desde la base ya que estos wireframes son el esqueleto de la app móvil. La idea es que se hagan de forma muy sencilla: incluso un boceto a mano puede servir como punto de partida.

https://www.yeeply.com/blog/como-definir-tu-aplicacion-movil-hacer-prototipo-de-app/

9. Mockup

Los mockups tienen como objetivo mostrar la parte más visual del proyecto. Presentan la estructura de la información, los contenidos y las funcionalidades de forma estática. Representan muy bien cómo va a ser la app, pero con una inversión mucho menor de lo que supondría un prototipo propiamente dicho.

https://www.yeeply.com/blog/como-definir-tu-aplicacion-movil-hacer-prototipo-de-app/

10. Prototipo

Avanzamos un poco más hacia el desarrollo de aplicaciones móviles en sí mismo cuando nos ponemos a hacer un prototipo. Llegados a este punto, utilizarás alguna herramienta para darle 'vida' a tu wireframe. La idea es que un prototipo muestre cómo va a ser la interacción de la app y que te permita hacerte una idea de cómo será finalmente. Son extremadamente útiles para testear la usabilidad de un proyecto.

https://www.yeeply.com/blog/como-definir-tu-aplicacion-movil-hacer-prototipo-de-app/

11. Buenas prácticas de Diseño UX.

Desde desplazamientos frustrantes hasta ubicaciones inexplicables de botones, el diseño de la UX determina cómo nos sentimos sobre los productos con los que interactuamos. Una mala UX puede tener efectos negativos para la marca. Por otra parte, un diseño de UX brillante que empatice con el cliente y sea intuitivo y elegante hace que el uso del producto sea disfrutable y puede impactar en la facturación.

¿Por qué no seguir tus instintos y diseñar como te plazca? Principalmente porque el buen diseño no tiene nada que ver contigo, sino con tu audiencia. Tu función como diseñador de UX es eliminar la fricción de la experiencia del usuario, y esa fricción es un blanco en constante movimiento.

1. **Ponte en el lugar de los usuarios**
2. **Hazlo accesible**
3. **Mantén la uniformidad**
4. **Crea un mapa del sitio**
5. **Usa herramientas para una navegación clara**
6. **Haz que los textos sean claros**
7. **Prueba y vuelve a probar**
8. **Diseña según el contexto**
9. **Mantenlo simple**
10. **Comprende la función de la tipografía**

Tomado de: https://lucidspark.com/es/blog/buenas-practicas-para-el-diseno-de-ux

12. Ejercicio

Realice un Wireframe del Reto 1 con los objetos visuales aprendidos, sobre las siguientes pantallas:

https://www.creativosonline.org/10-herramientas-imprescindibles-para-crear-mockups.html

Preguntas