Introducción a la programación en ANDROID

Instalación

Laboratorio de Programación Avanzada
Facultad de Informática
Universidad Nacional del Comahue

Sistemas Operativos de los Dispositivos Móviles

- Android, iOS, Windows Phone …

Desarrollo de aplicaciones

- Java, Scala, Kotlin, Groovy ...
- El SDK (Software Development Kit) de Android, incluye un conjunto de herramientas de desarrollo (librerías, ejemplo, etc)
- La plataforma integral de desarrollo (IDE, Integrated Development Environment) soportada oficialmente es Android Studio junto con el complemento ADT (Android Development Tools plugin)
- Otros entornos:
  - Eclipse
  - HyperNext Android Creator (no requiere saber Java ni SDK de android)
  - App Inventor for Android (entorno de desarrollo visual)
  - Basic4android (similar al desarrollo en Visual Basic)
  - WinDev Mobile (para el lenguaje W-Language)

Entorno de Desarrollo Android

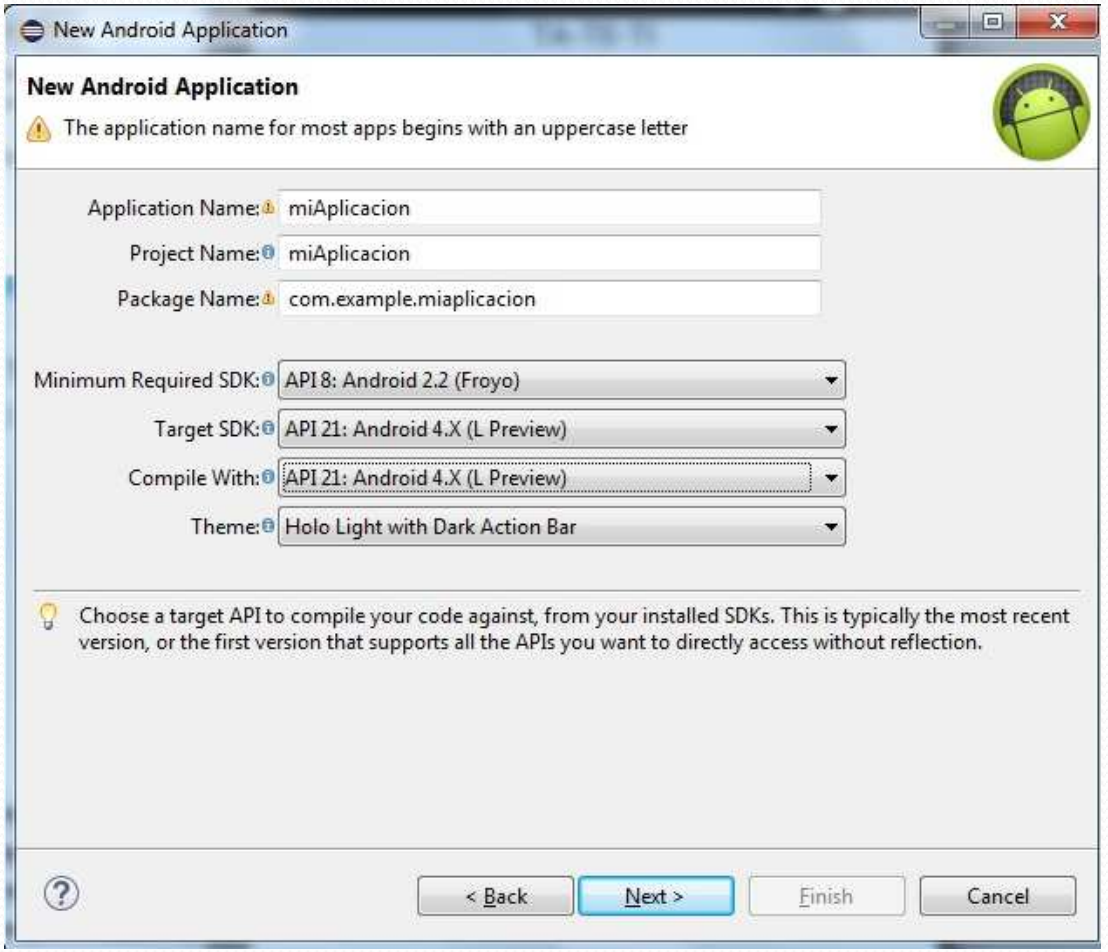
- Eclipse + Java Development Kid (JDK) + Android SDK
- Pasos
  1. descargar e instalar el JDK
  2. descargar y descomprimir el Android SDK
  3. Instalar Eclipse IDE
  4. Instalar el ADT (abrir eclipse e ir a Help -> Install New Software)
  5. Abrir Eclipse y usar el Android SDK Manager para descargar e instalar plataformas Android
  6. Configurar Android Devices (AVD) para usar emuladores o instalar los drivers necesarios para usar dispositivos reales.
- http://www.desarrolloweb.com/articulos/entorno-desarrollo-android.html

Entorno de Desarrollo Android

- Para testear la aplicación se puede abrir una aplicación existente:
  1. Ir al Android SDK Manager
  2. Elegir Samples for SDK (ver el API elegido) e instalar el paquete (seguir los pasos)
  3. Abrir el proyecto (File/New/other/Android -> Android sample project)
  4. Elegir en Build Target según el API del ejemplo elegido y seleccionar el ejemplo.
  5. Crear un AVD (es el emulador)
  6. Crear una configuración para la ejecución (Launch)
  7. Ejecutar la aplicación: Debug As, elegir la configuración creada.
  8. La aplicación se debería ejecutar dentro del emulador.

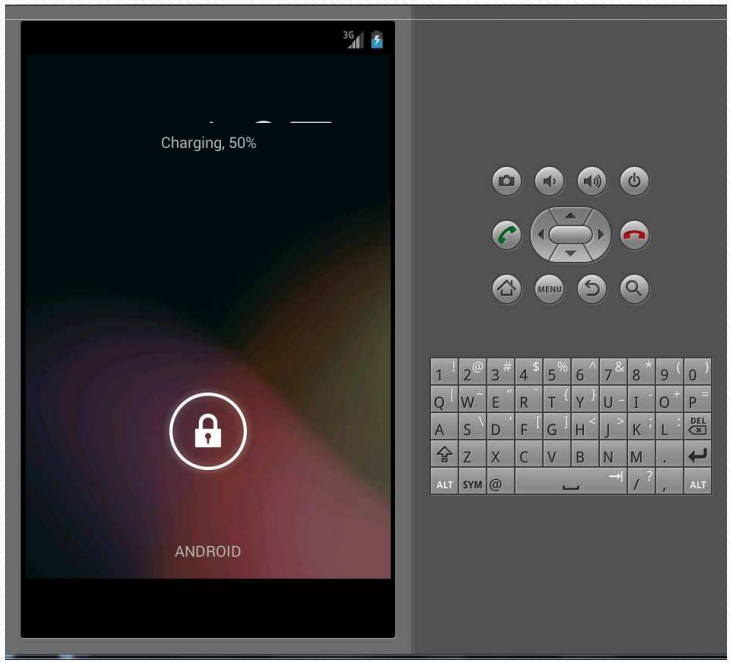
Primera Aplicación

- File /new / project
- Android Application Project
- Name
- Create Launch icon
- Create Activity → empty



Ejecución de la aplicación

- Crear un AVD (Android Virtual Device)
- Crear una configuración de Debug
- Ejecutar dicha configuración en el AVD



Introducción a la programación en ANDROID

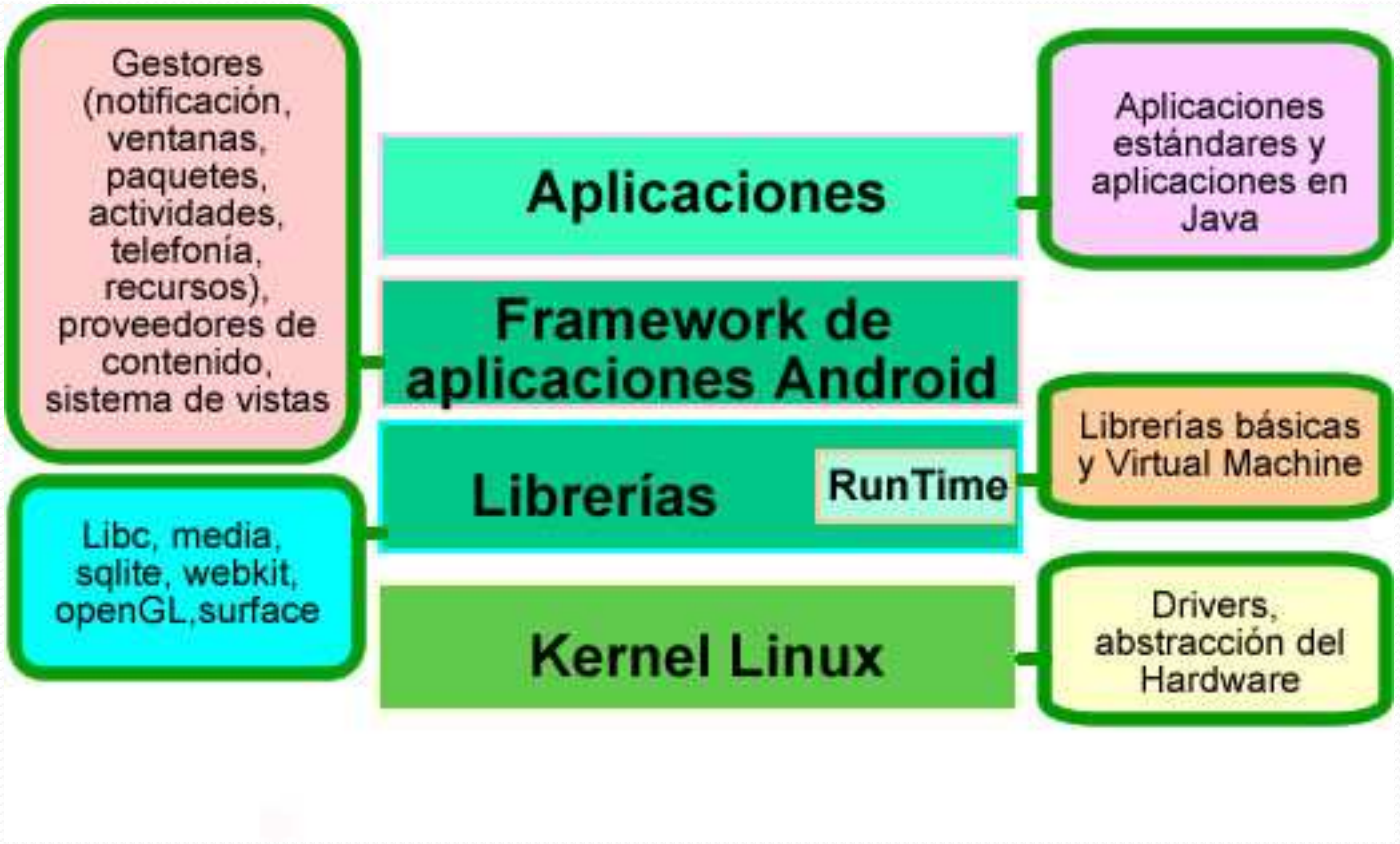
Primera parte

Laboratorio de Programación Avanzada
Facultad de Informática
Universidad Nacional del Comahue

Introducción

- Plataforma subyacente: Android es una pila de software para dispositivos móviles
- Licencia Apache Open Source

Arquitectura



Versiones de plataformas Android

Version	Codename	API	Distribution
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	4.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	3.7%
4.1.x	Jelly Bean	16	12.1%
4.2.x		17	15.2%
4.3		18	4.5%
4.4	KitKat	19	39.2%
5.0	Lollipop	21	15.9%
5.1		22	5.1%

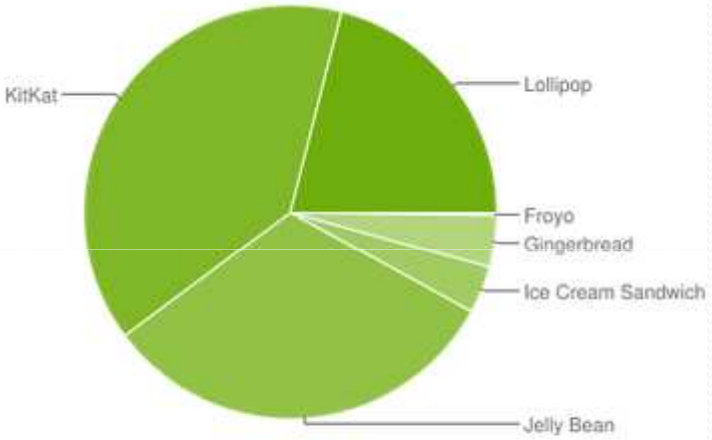
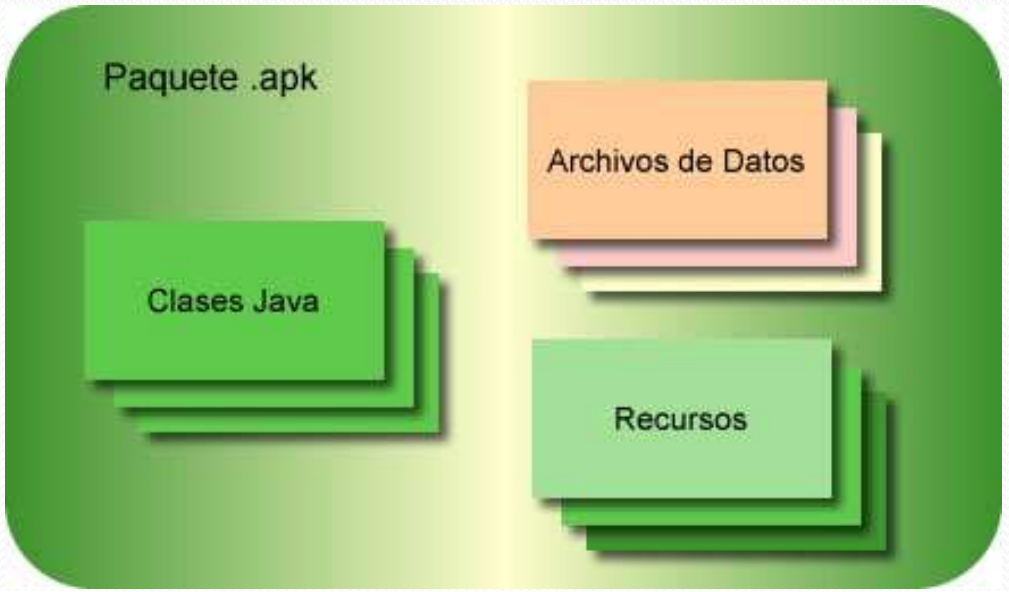


Imagen extraída de http://developer.android.com/intl/es/about/dashboards/index.html

Aplicaciones

- Son escritas en JAVA
- El código compilado, junto con los archivos de datos y recursos requeridos, son reunidos en un paquete Android (.apk)



Estructura del proyecto

Recursos

- Los recursos son archivos adicionales cuyo contenido es independiente del código fuente
- Recursos del sistema (android.R)
- Recursos de la aplicación
  - Son guardados en la carpeta Res
  - Cada tipo de recurso se almacena en una carpeta diferente:
    - layout, images, menus, values (strings, dimentions, colors, styles).
- Ventajas:
  - Personalizar la aplicación para diferentes dispositivos, lugares, idiomas
  - El código usa los recursos apropiados según la configuración del dispositivo

Recursos Alternativos

- Las aplicaciones deben proporcionar recursos alternativos para adaptarse a diferentes dispositivos.
- Por ejemplo pueden suministrarse cadenas en castellano y en ingles, creando un archivo con el mismo nombre pero guardado dentro de una carpeta con igual nombre que el original pero seguido de un guión y el cualificador: /res/values/strings.xml y /res/values/strings-es.xml
- Por ejemplo para suministrar un icono igual para pantallas de alta densidad deberían existir: drawable/icon.png y drawable-hdpi/icon.png

Otros archivos

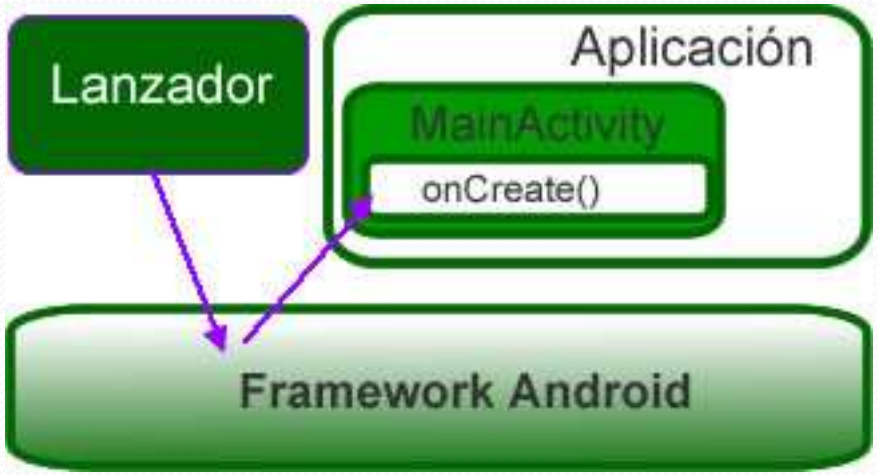
- En la carpeta gen se encuentra el archivo R que es generado en forma automática.
- R es una clase de Java generada por Android con clases en su interior llamadas drawable, layout y string, Contienen constantes enteras que apuntan a los recursos dentro de la carpeta res.
- Todas las aplicaciones tienen un archivo llamado AndroidManifest.xml en la raíz del proyecto.

Android Manifest

- Aquí se especifican
  - El nombre del paquete Java
  - Versión de la aplicación (usada por Google Play)
  - Actividades que integran la aplicación
  - Las librerías usadas (Google Maps)
  - Permisos (por ejemplo para acceder a Internet)
  - Nivel de API mínimo requerido

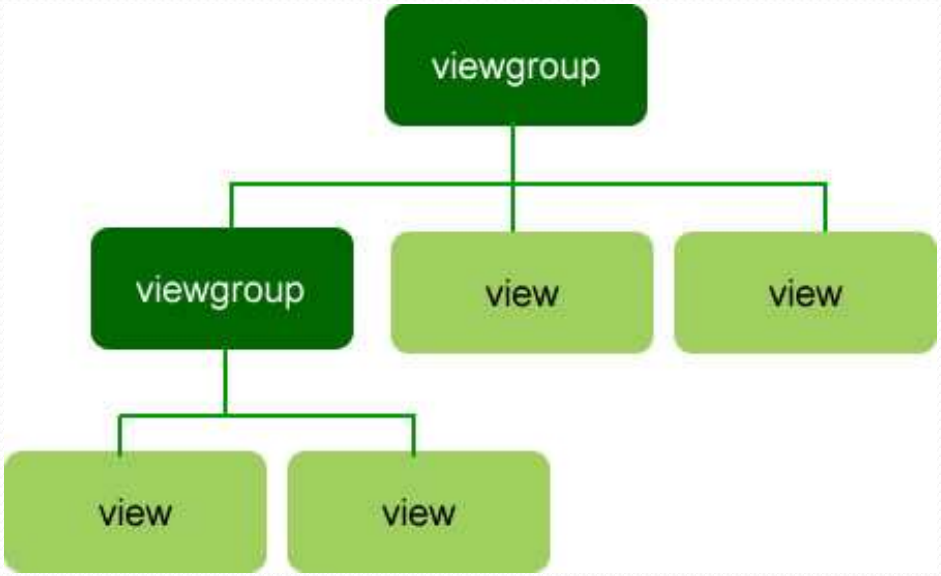
Actividades (Activity)

- Es una clase escrita en Java que extiende Activity
- Una actividad puede representar una interface de usuario
- Extiende la clase Activity
- Muestra los controles para interactuar con la aplicación (vistas – views)
- El usuario interactúa a través de ella



Actividades e Interfaz de Usuario

- Cada actividad tiene asociada una UI
- Esta UI es especificada en un archivo layout
- Este archivo es especificado como parámetro en la sentencia SetContentView dentro del método OnCreate de la actividad
- La UI se construye usando View y ViewGroup
- View: TextView, EditText, Button, etc.
- Viewgroup: Linear, Relative, Tabular, etc.



Acceso a recursos desde Java

- Son utilizados desde el código para implementar la aplicación
  - <Nombre_paquete>.R.<tipo_recurso>.<id_recurso>
  - R.layout.activity_main
  - R.id.mensaje

/res/values/strings.xml

```
<resources>
    <string name="app_name">Hello</string>
    <string name="hello_world">Hello world!</string>
<string name="mensaje">Buen día!</string>
</resources>
```

/src/MainActivity.java

```
public class MainActivity extends Activity {

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

      TextView tv = (TextView) findViewById(R.id.msj);

     String mensajito =  getResources().getString(R.string.mensaje);
      tv.setText(mensajito);
   }
}
```

Acceso a recursos desde XML

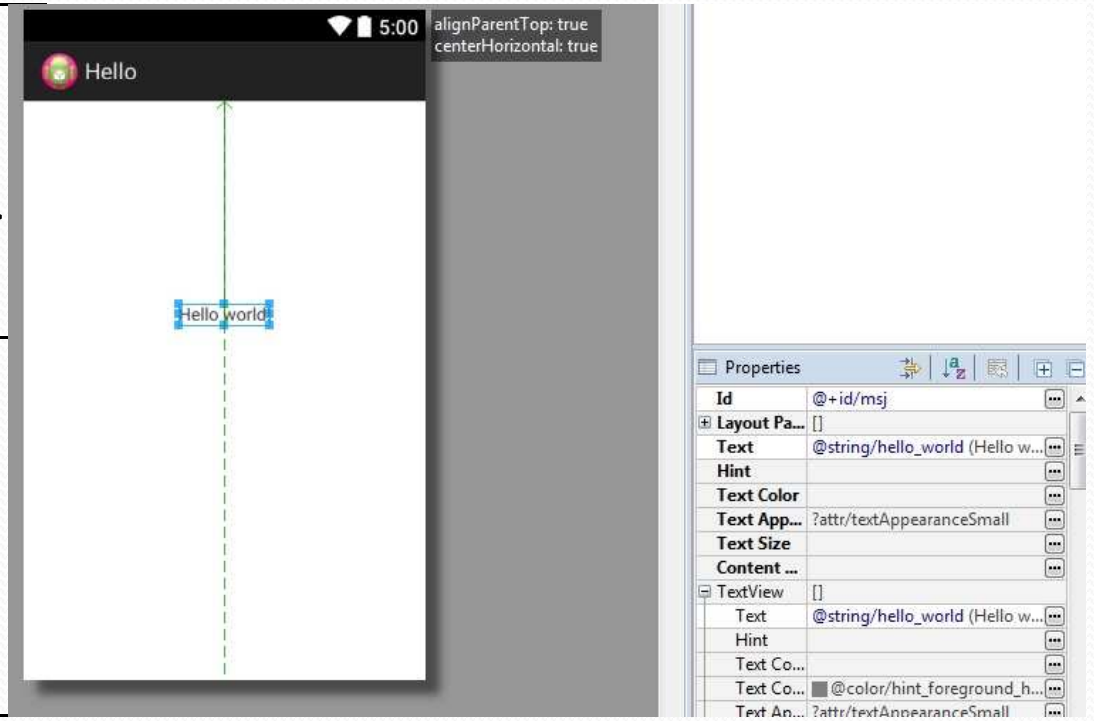
- @[<Nombre_paquete>].<tipo_recurso>/<id_recurso>
- @string/mensaje
- @android:color/blue

/res/values/strings.xml

```
<resources>
   <string name="app_name">Hello</string>
   <string name="hello_world">Hello world!</string>
<string name=“mensaje">Buen día!</string>
</resources>
```

/res/layout/activity_main.xml

```
<RelativeLayout   …   >
       <TextView
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="@string/hello_world" />
</RelativeLayout>
```



Otros Recursos: Arreglos

/res/values/array.xml

```
<resources>
      <integer-array name="numeros">
         <item >1</item>
         <item >2</item>
         <item >3</item>
      </integer-array>
 </resources>
```

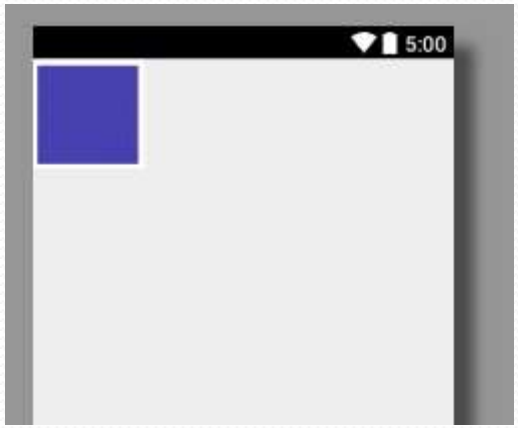
Acceso al arreglo desde Java

```
int[] numeros= getResources().getIntArray(R.array.numeros);
```

Otros Recursos: Imágenes

/res/layout/activity_main.xml

```
<LinearLayout…>
      <ImageView
          android:id="@+id/circulo"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:contentDescription="@string/circulo"
          android:onClick="circulo"
          android:scaleType="fitCenter"
          android:src="@drawable/circulo" />
….
```

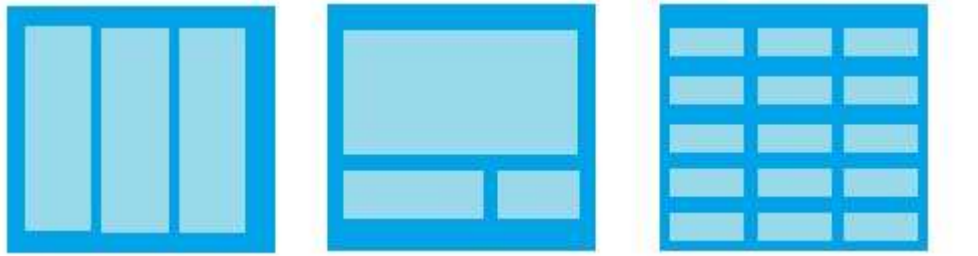



```
public class MainActivity extends Activity {
    ImageView primera;
     ImageView segunda;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        primera = (ImageView) findViewById(R.id.circulo);
        segunda = (ImageView) findViewById(R.id.cuadrado);
    }

    public void circulo (View view){
       segunda.setVisibility(View.VISIBLE);
       primera.setVisibility(View.GONE);
    }

    public void cuadrado (View view){
        primera.setVisibility(View.VISIBLE);
        segunda.setVisibility(View.GONE);
    }
}
```

Interfaces de Usuario

- La UI es definida en un archivo de diseño ubicada en la carpeta /res/layout
- Es una especificación XML
- Cada elemento en la interface tiene un ID que permite acceder a él desde el código JAVA
- Ejemplo: R.id.mi_boton

Los Layouts



- LinearLayout: apila uno tras otro todos sus elementos hijos de forma horizontal o vertical según se establezca su propiedad android:orientation="vertical" y android:orientation="Horizontal" .
- RelativeLayout: permite especificar la posición de cada elemento de forma relativa a su elemento padre o a cualquier otro elemento incluido en el propio layout. De esta forma, al incluir un nuevo elemento A podremos indicar por ejemplo que debe colocarse debajo del elemento B y alineado a la derecha del layout padre.
- FrameLayout: coloca todos sus controles hijos alineados con su esquina superior izquierda, de forma que cada control quedará oculto por el control siguiente
- Table/GridLayout : permite distribuir sus elementos hijos de forma tabular, definiendo las filas y columnas necesarias, y la posición de cada componente dentro de la tabla.

http://www.sgoliver.net/blog/interfaz-de-usuario-en-android-layouts/

Atributos de las vistas

- ***Android:id***= “@id.boton”
- ***Android:layout_width:*** “40dp”
  - dp: Densidad de píxeles independientes, se basa en la densidad física de la pantalla.
  - px. Píxeles, corresponde a píxeles reales en la pantalla.
  - en. Cm - basado en el tamaño físico de la pantalla.
  - mm. Milímetros - en función del tamaño físico de la pantalla.
  - pt. Puntos - 1/72 de una pulgada en función del tamaño físico de la pantalla.
  - sp. Escala de píxeles independientes
  - FILL_PARENT que indica que la vista intentará ser tan grande como su padre
  - WRAP_CONTENT indica que la vista intentará ser lo suficientemente grande para mostrar su contenido
- ***android:layout_weigth***: permite repartir en forma proporcional el espacio.
- ***android:layout_gravity***="center". Esta propiedad es la que se usa para centrar, center_Horizontal|center_Vertical centra en forma horizontal y lo ajusta en vertical arriba.

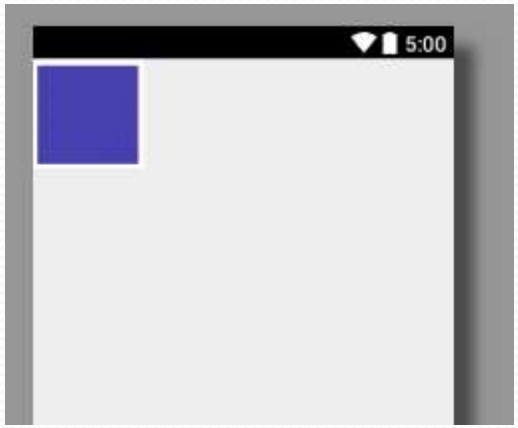
Eventos

- Un evento es una acción disparada por un usuario que interactúa con la UI (presionar o soltar un botón o una tecla, tocar un área de la pantalla, ...)
- Para reaccionar a los eventos que ocurren en la UI es necesario implementar los métodos adecuados en la Actividad
- Se definen escuchadores (listener) de eventos
  - Atributo onClick, el valor que se le da a este atributo es el nombre del método que se ejecutará
  - Implementar el escuchador de eventos y registrarlo en el botón
  - La actividad implementa la interface escuchador de eventos.

Evento asociado a onClick

/res/layout/activity_main.xml

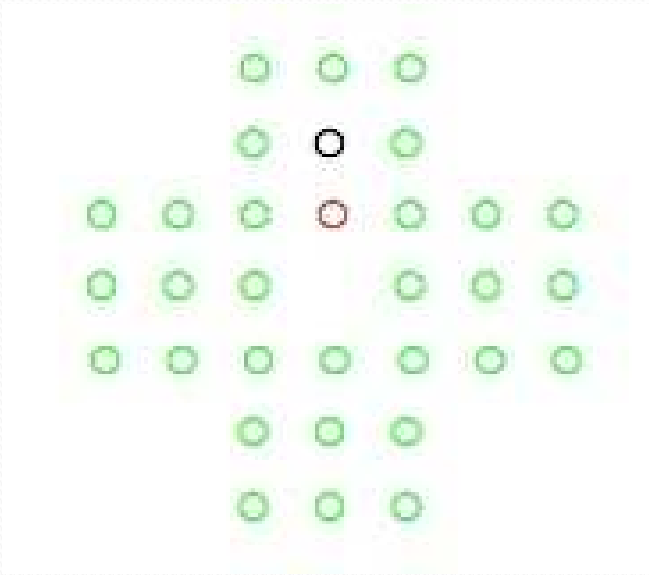
```
<LinearLayout…>
      <ImageView
          android:id="@+id/circulo"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:contentDescription="@string/circulo"
          android:onClick="circulo"
          android:scaleType="fitCenter"
          android:src="@drawable/circulo" />
….
```




```
public class MainActivity extends Activity {
    ImageView primera;
     ImageView segunda;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        primera = (ImageView) findViewById(R.id.circulo);
        segunda = (ImageView) findViewById(R.id.cuadrado);
    }

    public void circulo (View view){
    //pone visible al cuadrado
       segunda.setVisibility(View.VISIBLE);
     //pone invisible al circulo y es como si no estuviera
       primera.setVisibility(View.GONE);
    }
    public void cuadrado (View view){
        primera.setVisibility(View.VISIBLE);
        segunda.setVisibility(View.GONE);
    }   }
```

Botón como escuchador

El escuchador listener es una interfaz de tipo View.OnClickListener y por lo tanto a la vez que se instancia es necesario implementar el método callback onClick()



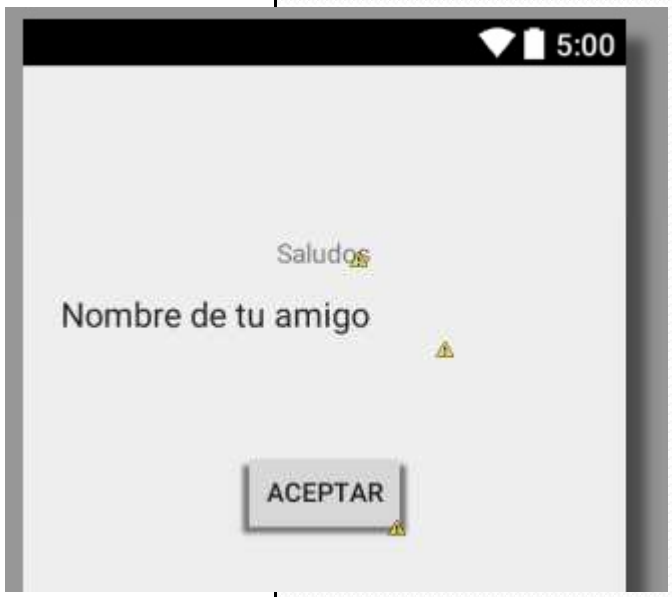
```
public class MainActivity extends Activity {
    private RadioButton button5;
    private RadioButton button10;
    private RadioButton button17;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     button5 = (RadioButton) findViewById(R.id.f5);
     button10 = (RadioButton) findViewById(R.id.f10);
     button17 = (RadioButton) findViewById(R.id.f17);
     button5.setOnClickListener(listener);
}
    private OnClickListener listener = new OnClickListener() {
      public void onClick(View v) {
        RadioButton button = (RadioButton) v;
         button.setChecked(false);
         button10.setChecked(false);
         button17.setChecked(true);
      }
   };
}
```

Eventos asociados a la clase

- La clase View define la interface Java que contiene los métodos callback
- La actividad implementa la interface del método callback
- El método es llamado por el Android Framework cuando el usuario interactúa con la vista sobre la cual se ha registrado el escuchador
- Métodos callback: onClick(), onLongClick(), onFocusChange(), onKey(), onTouch(), conCreateContextMenu()

Ejemplo la clase como escuchador

```
public class MainActivity extends Activity implements OnClickListener{

@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   Button btn = (Button) findViewById(R.id.btnAceptar);
   btn.setOnClickListener(this);
}

 @Override
   public void onClick(View v) {
      TextView tm = (TextView) findViewById(R.id.texto_mensaje);
      EditText edtNombre = (EditText) findViewById(R.id.nombre);
      String nombre = edtNombre.getText().toString();
      switch (v.getId()) {
         case R.id.btnAceptar:
         tm.setText("Buenos dìas "+nombre+"! Espero que estes muy bien "+h);
            break;
         case R.id.nombre:
         edtNombre.setText("");
            break;
      }
   }
}
```



Ver otro ejemplo Figuras Geometricas – onTouch eve

Manos a la Obra!