Servicios Basados en Localización (LBS)

Tema 6
Desarrollo de aplicaciones en Android

Interfaz de Usuario

Miguel Ángel Manso
Ramón Alcarria

Recursos

- Carpeta "res" dentro del proyecto
  - drawable-xxx
    - Iconos y recursos gráficos de la aplicación
  - layout
    - layouts de los activities; se cargan desde el activity
  - menu
    - Menús de los activities
  - values/strings.xml
    - Strings usados por la aplicación y sus layouts
    - Es importante que no estén "harcoded", es decir, definidos dentro del propio layout, para favorecer la internacionalización (traducción a diversos idiomas)

Referencias a recursos

- Referencia a un recurso desde el layout
  - android:text="@string/more"
  - android:src="@drawable/separator"
- Referencia a componentes de un layout desde el código
  - En el layout
    - @+id/boton1
    - @+ significa que se crea el id para acceder desde el código
    - @ significa que el recurso se ha definido previamente, o que está en los recursos de la app
- Más información: http://www.intertech.com/Blog/Post/Android-Layout-and-ID-Attribute.aspx

Acceder desde el código

- Para acceder a un componente definido en un layout, primero debemos identificarlo en el layout con un nombre

```
<org.osmdroid.views.MapView
android:id="@+id/openmapview"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
/>
```

- Después, desde el código, usamos el método findViewById(int resourceID) para obtener una referencia al objeto usando el android:id definido en el XML

```
myOpenMapView = (MapView)findViewById(R.id.openmapview);
```

Acceder desde el código

- Como cambiar el texto de un recurso con una cadena definida en el fichero de strings
  - Obtener el objeto con findViewById

```
TextView texto = (TextView)findViewById(R.id.text1);
```

  - Acceder al string por el ID

```
String mensaje = getResources().getString(R.string.bye_world)
```

  - Cambiar el texto

```
texto.setText(mensaje);
```


Asignar manejadores a botones

- Desde código

```
final Button button = (Button) findViewById(R.id.button_id);
button.setOnClickListener(new View.OnClickListener() {
   public void onClick(View v) {
      // Perform action on click
    }
});
```

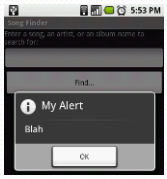
- Desde el propio layout

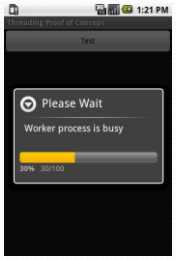
```
<Button
android:id="@+id/button1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="right"
android:text="@string/more"
android:onClick="doMore" />
```

```
public void doMore(View view) {
  Toast.makeText(this, "Hola, manejador",
Toast.LENGTH_LONG).show();
}
```

Diálogos

- Ventana emergente que capta el foco hasta que el usuario lo cierra o el proceso termina.
  - Alert Dialog: Gestiona hasta 3 botones
  - Progress Dialog: Muestra una barra de progreso
  - DatePickerDialog: Permite al usuario seleccionar una fecha
  - TimePickerDialog: Permite al usuario seleccionar una hora





Diálogos

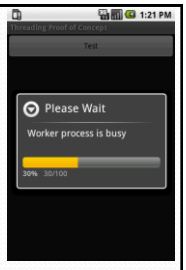
- Alert Dialog

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setMessage("¿Quieres terminar la actividad?");
builder.setCancelable(true);
builder.setPositiveButton("Si", new OkOnClickListener());
builder.setNegativeButton("No", new CancelOnClickListener());
AlertDialog dialog = builder.create();
dialog.show();

private final class CancelOnClickListener implements
DialogInterface.OnClickListener {
public void onClick(DialogInterface dialog, int which) {
Toast.makeText(getApplicationContext(), "La Actividad continúa",
Toast.LENGTH_LONG).show();}}

private final class OkOnClickListener implements
DialogInterface.OnClickListener {
public void onClick(DialogInterface dialog, int which) {
AndroidDialogsActivity.this.finish();}}
```

Diálogos



- Progress Dialog

```
ProgressDialog progressDialog;
progressDialog = new ProgressDialog(this);
progressDialog.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
progressDialog.setMessage("Loading...");
progressDialog.setCancelable(false);
progressDialog.show();
```

```
//Para avanzar el progreso de la barra
progressDialog.setProgress(int progress);

//Para cerrar el diálogo
progressDialog.dismiss();
```


Preferencias

- Datos que guarda una aplicación para personalizar la experiencia de usuario: Información personal, opciones de presentación.
- Las preferencias se puede guardar en la base de datos SQLite de Android y también en documentos XML
- La actividad que define las preferencias tiene que extender a PreferenceActivity

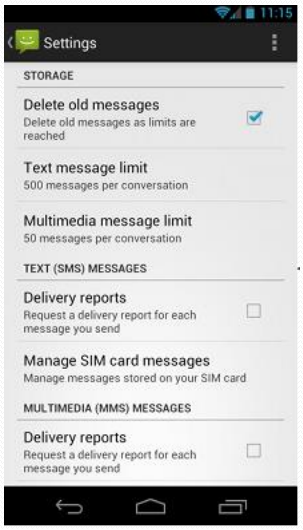
Preferencias

- PreferenceScreen: Describe la pantalla de las preferencias
- PreferenceCategory: Para dividir en categorías
- Tipos de Preference:

Tipo	Descripción
CheckBoxPreference	Marca seleccionable.
EditTextPreference	Cadena simple de texto.
ListPreference	Lista de valores seleccionables (exclusiva).
MultiSelectListPreference	Lista de valores seleccionables (múltiple).

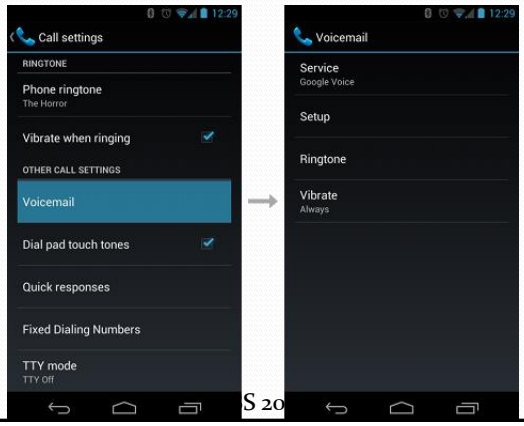
Preferencias

- Las *Preference* tienen:
  - android:key
  - android:title
  - android:summary
  - andorid:defaultValue
- Las *PreferenceCategory* tienen:
  - android:key
  - android:title



Preferencias

- Las *PreferenceScreen* tienen:
  - android:key
  - android:title



Utilizando intents

- Para llamar a otras actividades desde una preferencia

```
<Preference android:title="@string/prefs_web_page" >
    <intent android:action="android.intent.action.VIEW"
            android:data="http://www.example.com" />
</Preference>
```

Creando la PreferenceActivity

- Extensión de la clase Activity que permite almacenar preferencias.
- Para Android 3.0 o superior hay que utilizar la clase PreferenceFragment en su lugar.

```
public class SettingsActivity extends PreferenceActivity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        addPreferencesFromResource(R.xml.preferences);
    }
}
```


Preference Fragments

- Para Android 3.0 o superior se recomienda utilizar PreferenceFragments.
  - Primero se define el Fragmento y luego se añade a una actividad

```
public static class SettingsFragment extends PreferenceFragment {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Load the preferences from an XML resource
        addPreferencesFromResource(R.xml.preferences);
    }
    ...
}
```

```
public class SettingsActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Display the fragment as the main content.
        getFragmentManager().beginTransaction()
                .replace(android.R.id.content, new SettingsFragment())
                .commit();
    }
}
```

Guardar y leer preferencias

- Asociar valores por defecto
  - En el XML

```
<!-- default value is a string -->
<ListPreference
    android:defaultValue="@string/pref_syncConnectionTypes_default"
    ... />
```

  - En código
    - this: context
    - false: si el usuario ya ha modificado la preferencia, entonces no se modifica ahora

```
PreferenceManager.setDefaultValues(this, R.xml.advanced_preferences, false);
```


Guardar y leer preferencias

- Se utiliza un objeto SharedPreferences para guardar y leer preferencias

```
SharedPreferences prefs =
     getSharedPreferences("MisPreferencias",Context.MODE_PRIVATE);

SharedPreferences.Editor editor = prefs.edit();
editor.putString("email", "modificado@email.com");
editor.putString("nombre", "Prueba");
editor.commit();
```

```
SharedPreferences sharedPref = PreferenceManager.getDefaultSharedPreferences(this);
String syncConnPref = sharedPref.getString("email", "");
```

Guardar y leer preferencias

- Se utiliza un objeto SharedPreferences para guardar y leer preferencias
- Vamos a probar a escribir preferencias y luego obtenerlas con estos dos métodos.
- Vamos a buscar el archivo con el DDMS a ver cual es su aspecto.

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="nombre">prueba</string>
    <string name="email">modificado@email.com</string>
</map>
```


Ejercicio de preferencias

- Creamos un proyecto nuevo con una actividad en blanco y utilizamos el menú que ha sido creado automáticamente para que nos lleve a un menú de preferencias que vamos a elaborar.
  - Opción rápida: Utilizar el asistente para crear una nueva *SettingsActivity* (Proyecto Preference)
  - Crear todas las clases al estilo de estos tutoriales:
    - http://rominirani.com/android-preferences-tutorial/
    - http://www.101apps.co.za/articles/preference-settings.html
    - Proyecto Preference2