Programa :
Curso :
Duración :

1. OBJETIVO

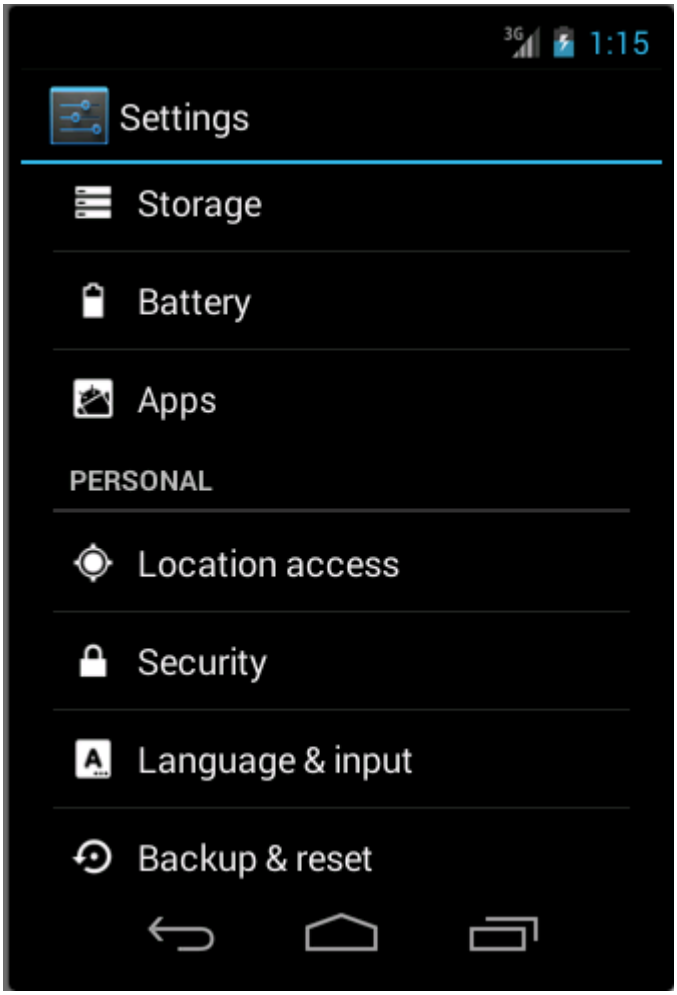
Consumir un ContentProvider y Creación de uno nuevo.

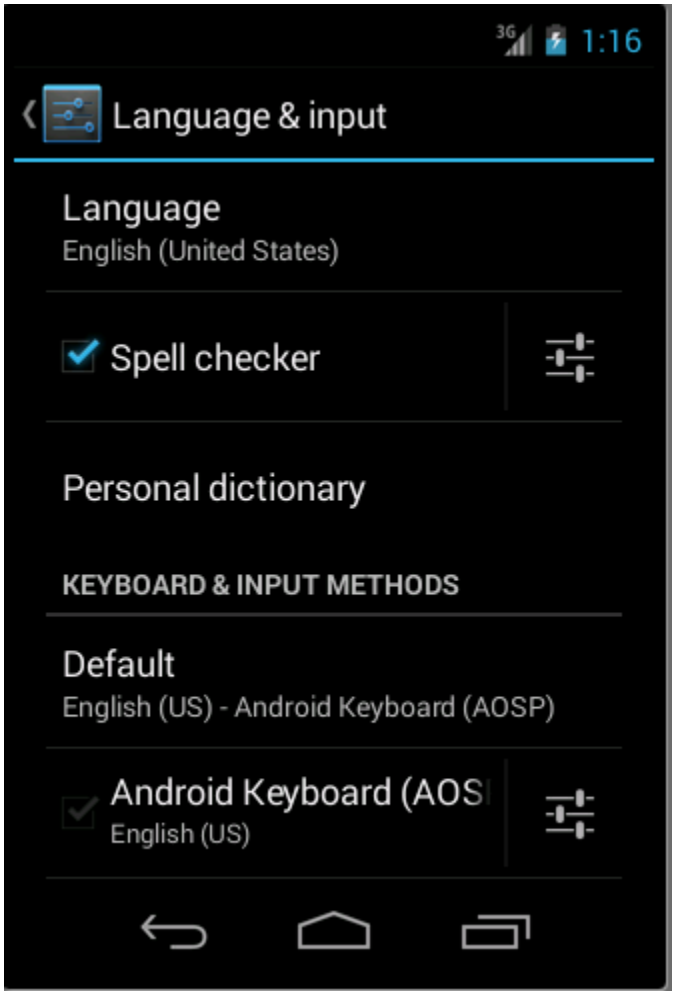
2. DESCRIPCION

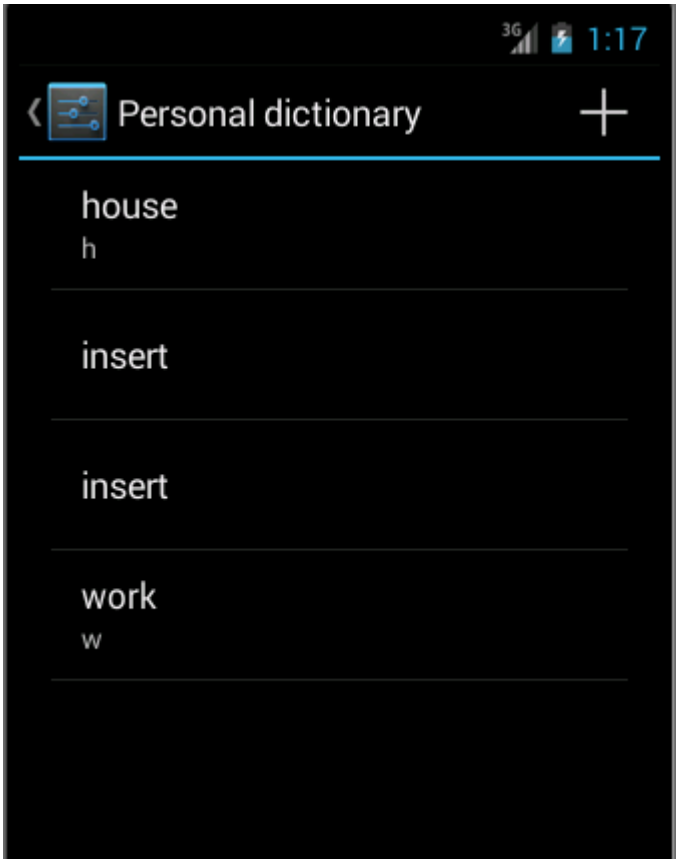
Se elaborará un activity que permite consumir un content provider de android, en la segunda parte se construirá un content provider para luego consumirlo.

3. PASOS

1. Crear un nuevo proyecto llamado ContentProviderDemo.
2. Crear la clase QueryContentProvider que extienda de ListActivity, mostrara las palabras que han sido almacenadas en el diccionario del android utilizando el content provider: “user_dictionary”


1:15
Settings
Storage
Battery
Apps
PERSONAL
Location access
Security
Language & input
Backup & reset


1:16
Language & input
Language
English (United States)
Spell checker
Personal dictionary
KEYBOARD & INPUT METHODS
Default
English (US) - Android Keyboard (AOSP)
Android Keyboard (AOS
English (US)



3. Crear layout: rowcursor.xml que contendrá los elementos que se mostrara por cada palabra que contenga el diccionario.

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtid"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtword"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtlocale"/>

</LinearLayout>
```

4. Editar QueryContentProvider:

    1. Agregar atributos :

```
String[] mProyeccion = {UserDictionary.Words._ID, UserDictionary.Words.WORD, UserDictionary.Words.LOCALE};
String mSeleccion = null;
String[] mSeleccionArgumentos = {""};

int[] mWordListItems = {R.id.txtid, R.id.txtword, R.id.txtlocale};

SimpleCursorAdapter simpleCursorAdapter;
```

    2. Crear el método obtenerPalabrasdeProvider:

```
void obtenerPalabrasdeProvider(){
    Cursor cursor= getContentResolver().query(UserDictionary.Words.CONTENT_URI,
            mProyeccion,null,null,null);
    if(null==cursor){
        Log.d("obtenerPalabrasdeProvider","Error en consultar content provider");
    }else if(cursor.getCount()<1){
        Log.d("obtenerPalabrasdeProvider","No se obtiene palabras del provider");
    }else{
        simpleCursorAdapter= new SimpleCursorAdapter(getApplicationContext(),
                R.layout.rowcursor,cursor,mProyeccion,mWordListItems,0);
        getListView().setAdapter(simpleCursorAdapter);
    }
    String palabras=null;
    int index=cursor.getColumnIndex(UserDictionary.Words.WORD);
    if(cursor!=null){
        while(cursor.moveToNext()){
            palabras+=palabras+cursor.getString(index);
        }
    }
    Log.d("palabras::",palabras);
```

    3. Sobrescribir el método onCreate:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    obtenerPalabrasdeProvider();
}
```

5. Modificar el AndroidManifest:

    1. Agregar permisos de lectura :

```
<uses-permission android:name="android.permission.READ_USER_DICTIONARY"></uses-permission>
```

    2. Declarar el activity como principal:

```
<activity
    android:name=".mycontentprovider.QueryMyContentProvider"
    android:label="Query Content Provider">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

6. Ejecutar la aplicación:

7. Ahora crearemos un content provider que simulara el comportamiento del diccionario.

8. Crearemos una clase MySqlOpenHelper que extienda SQLiteOpenHelper, que nos permitirá crear la base de datos de nuestra aplicación.

```
public class MySqlOpenHelper extends SQLiteOpenHelper {

    private static final String SQL_CREATE_MAIN = "create table main" +
            " (_id INTEGER PRIMARY KEY," +
            "word TEXT," +
            "frequency TEXT," +
            "locale TEXT)";
    private static final  String DBNAME ="PROVIDER" ;

    MySqlOpenHelper(Context context) { super(context,DBNAME,null,1); }

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) { sqLiteDatabase.execSQL(SQL_CREATE_MAIN); }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i2) {

    }
}
```

9. Crear la clase ContractProvider, que será la interface que los clientes utilizaran para saber los nombres de las columnas, la uri para consumir el provider que estamos creando.

```
public final class ContractProvider implements BaseColumns {

    public static final String TABLE="main";
    public static final String COLUMNA_ID="_id";
    public static final String COLUMNA_WORD="word";
    public static final String COLUMNA_FREQUENCY="frequency";
    public static final String COLUMNA_LOCALE="locale";
    public static final String AUTHORITY = "mycontentproviders";

    private  static final String uri="content://com.example.reapro.contentproviderdemo."+AUTHORITY+"/main";
    public static final Uri CONTENT_URI = Uri.parse(uri);
}
```

10. Crear la clase MyContentProvider que extienda de ContentProvider:

    1. Agregar atributos:

```
private static final UriMatcher uriMatcher;
private MySqlOpenHelper mySqlOpenHelper;
private SQLiteDatabase db;
```

    2. Declarar los atributos estaticos de Uris, para determinar que tipo de solicitud se está realizando desde el cliente:

```
static {
    uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
    uriMatcher.addURI("com.example.reapro.contentproviderdemo.mycontentproviders", "main", 1);
    uriMatcher.addURI("com.example.reapro.contentproviderdemo.mycontentproviders", "main/#", 2);
}
```

    3. Sobrescribir el método onCreate:

```
@Override
public boolean onCreate() {
    mySqlOpenHelper= new MySqlOpenHelper(getContext());

    return true;
}
```

4. Implementar el método query:

```
@Override
public Cursor query(Uri uri, String[] strings, String s, String[] strings2, String s2) {
    db=mySqlOpenHelper.getReadableDatabase();
    Cursor cursor=null;
    switch (uriMatcher.match(uri)){
        case 1:
            cursor= db.query("main",strings,s,strings2,null,null,s2);

        case 2:
            s=s+"_ID="+uri.getLastPathSegment();
            break;
        default:
    }
    return cursor;
}
```

5. Implementar el método insert:

```
@Override
public Uri insert(Uri uri, ContentValues contentValues) {
    db=mySqlOpenHelper.getWritableDatabase();
    long id=db.insert("main",null,contentValues);
    Log.d("id",""+id);
   return  uri.withAppendedPath(uri,""+id);

}
```

11. Crear la clase MyContentProvider que extienda de ContentProvider:

```java
public class QueryMyContentProvider extends ListActivity {

    String[] mProyeccion = {ContractProvider.COLUMNA_ID, ContractProvider.COLUMNA_WORD,ContractProvider.COLUMNA_LOCALE};
    String mSeleccion = null;
    String[] mSeleccionArgumentos = {""};

    int[] mWordListItems = {R.id.txtid, R.id.txtword, R.id.txtlocale};

    SimpleCursorAdapter simpleCursorAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        registrarPalabra();
        obtenerPalabrasdeProvider();
    }

    private void registrarPalabra() {

        Uri nuevoUri;
        ContentValues contentValues= new ContentValues();

        contentValues.put(ContractProvider.COLUMNA_LOCALE,"en_US");
        contentValues.put(ContractProvider.COLUMNA_WORD,"insert");
        contentValues.put(ContractProvider.COLUMNA_FREQUENCY,"100");

        nuevoUri=getContentResolver().insert(ContractProvider.CONTENT_URI,contentValues);
        Log.d("nuevoUri::", ""+ContentUris.parseId(nuevoUri));

    }

    void obtenerPalabrasdeProvider(){
        Cursor cursor= getContentResolver().query(ContractProvider.CONTENT_URI,mProyeccion,null,null,null);
        if(null==cursor){
            Log.d("obtenerPalabrasdeProvider",""+cursor);
        }else if(cursor.getCount()<1){
            Log.d("obtenerPalabrasdeProvider",""+cursor.getCount());
        }else{
            simpleCursorAdapter= new SimpleCursorAdapter(getApplicationContext(),R.layout.rowcursor,cursor,mProyeccion,mWordListItems,0);
            getListView().setAdapter(simpleCursorAdapter);
        }
        String palabras=null;
        int index=cursor.getColumnIndex(ContractProvider.COLUMNA_WORD);
        if(cursor!=null){
            while(cursor.moveToNext()){
                palabras+=palabras+cursor.getString(index);
            }
        }

        Log.d("palabras::",palabras);
    }

}
```

12. Modificar AndroidManifest:

    1. Declarar provider:

```xml
<provider
    android:name=".mycontentprovider.MyContentProvider"
    android:authorities="com.example.reapro.contentproviderdemo.mycontentproviders" />
```

    2. Cambiar el activity principal:

```
<activity
    android:name=".mycontentprovider.QueryMyContentProvider"
    android:label="Query Content Provider">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

13. Ejecutar aplicación: