201730145033宗玥 #9

实现跨程序数据共享

1. 作业要求

在上一章中DatabaseTest项目的基础上继续开发，通过内容提供器来给它加入外部访问接口。

1. 打开已有的DatabaseTest项目
2. 创建一个内容提供器，New->Other->Content Provider
3. 修改上一步创建的DatabaseProvider中的代码

定义常量用于表示访问表中的各种模式数据

实现抽象方法

1. 在AndroidManifest.xml文件中注册内容提供器
2. 创建新项目ProviderTest来访问DatabaseTest中的数据
3. 作业展示
4. 创建DatabaseProvider定义常量用于表示访问表中的各种模式数据

public static final int *BOOK\_DIR*=**0;**public static final int *BOOK\_ITEM*=**1;**public static final int *CATEGORY\_DIR*=**2;**public static final int *CATEGORY\_ITEM*=**3;**

1. 实现抽象方法

public int delete(Uri uri**,** String selection**,** String[] selectionArgs) {  
 // Implement this to handle requests to delete one or more rows.  
 // 删除数据  
 SQLiteDatabase db = dbHelper.getWritableDatabase()**;** int deletedRows = **0;** switch (*uriMatcher*.match(uri)) {  
 case *BOOK\_DIR*:  
 deletedRows = db.delete("Book"**,** selection**,** selectionArgs)**;** break**;** case *BOOK\_ITEM*:  
 String bookId = uri.getPathSegments().get(**1**)**;** deletedRows = db.delete("Book"**,** "id = ?"**,** new String[] { bookId })**;** break**;** case *CATEGORY\_DIR*:  
 deletedRows = db.delete("Category"**,** selection**,** selectionArgs)**;** break**;** case *CATEGORY\_ITEM*:  
 String categoryId = uri.getPathSegments().get(**1**)**;** deletedRows = db.delete("Category"**,** "id = ?"**,** new String[] { categoryId })**;** break**;** default:  
 break**;** }  
 return deletedRows**;**}  
  
@Override  
public String getType(Uri uri) {  
 // *TODO: Implement this to handle requests for the MIME type of the data* // at the given URI.  
 switch (*uriMatcher*.match(uri)) {  
 case *BOOK\_DIR*:  
 return "vnd.android.cursor.dir/vnd.com.example.databasetest. provider.book"**;** case *BOOK\_ITEM*:  
 return "vnd.android.cursor.item/vnd.com.example.databasetest. provider.book"**;** case *CATEGORY\_DIR*:  
 return "vnd.android.cursor.dir/vnd.com.example.databasetest. provider.category"**;** case *CATEGORY\_ITEM*:  
 return "vnd.android.cursor.item/vnd.com.example.databasetest. provider.category"**;** }  
 return null**;**}  
  
@Override  
public Uri insert(Uri uri**,** ContentValues values) {  
 // *TODO: Implement this to handle requests to insert a new row.* // 添加数据  
 SQLiteDatabase db = dbHelper.getWritableDatabase()**;** Uri uriReturn = null**;** switch (*uriMatcher*.match(uri)) {  
 case *BOOK\_DIR*:  
 case *BOOK\_ITEM*:  
 long newBookId = db.insert("Book"**,** null**,** values)**;** uriReturn = Uri.*parse*("content://" + *AUTHORITY* + "/book/" + newBookId)**;** break**;** case *CATEGORY\_DIR*:  
 case *CATEGORY\_ITEM*:  
 long newCategoryId = db.insert("Category"**,** null**,** values)**;** uriReturn = Uri.*parse*("content://" + *AUTHORITY* + "/category/" + newCategoryId)**;** break**;** default:  
 break**;** }  
 return uriReturn**;**}  
  
@Override  
public boolean onCreate() {  
 // *TODO: Implement this to initialize your content provider on startup.* dbHelper=new MyDatabaseHelper(getContext()**,**"BookStore.db"**,**null**,2**)**;** return true**;**}  
  
@Override  
public Cursor query(Uri uri**,** String[] projection**,** String selection**,** String[] selectionArgs**,** String sortOrder) {  
 // *TODO: Implement this to handle query requests from clients.* //查询数据  
 SQLiteDatabase db=dbHelper.getReadableDatabase()**;** Cursor cursor=null**;** switch(*uriMatcher*.match(uri)){  
 case *BOOK\_DIR*:  
 cursor=db.query("Book"**,**projection**,**selection**,**selectionArgs**,**null**,**null**,**sortOrder)**;** break**;** case *BOOK\_ITEM*:  
 String bookId=uri.getPathSegments().get(**1**)**;** cursor=db.query("Book"**,**projection**,**"id=?"**,**new String[]{bookId}**,**null**,**null**,**sortOrder)**;** break**;** case *CATEGORY\_DIR*:  
 cursor=db.query("Category"**,**projection**,**selection**,**selectionArgs**,**null**,**null**,**sortOrder)**;** break**;** case *CATEGORY\_ITEM*:  
 String categoryId=uri.getPathSegments().get(**1**)**;** cursor=db.query("Category"**,**projection**,**"id=?"**,**new String[]{categoryId}**,**null**,**null**,**sortOrder)**;** break**;** default:  
 break**;** }  
 return cursor**;**}  
  
@Override  
public int update(Uri uri**,** ContentValues values**,** String selection**,** String[] selectionArgs) {  
 // *TODO: Implement this to handle requests to update one or more rows.* // 更新数据  
 SQLiteDatabase db = dbHelper.getWritableDatabase()**;** int updatedRows = **0;** switch (*uriMatcher*.match(uri)) {  
 case *BOOK\_DIR*:  
 updatedRows = db.update("Book"**,** values**,** selection**,** selectionArgs)**;** break**;** case *BOOK\_ITEM*:  
 String bookId = uri.getPathSegments().get(**1**)**;** updatedRows = db.update("Book"**,** values**,** "id = ?"**,** new String[] { bookId })**;** break**;** case *CATEGORY\_DIR*:  
 updatedRows = db.update("Category"**,** values**,** selection**,** selectionArgs)**;** break**;** case *CATEGORY\_ITEM*:  
 String categoryId = uri.getPathSegments().get(**1**)**;** updatedRows = db.update("Category"**,** values**,** "id = ?"**,** new String[] { categoryId })**;** break**;** default:  
 break**;** }  
 return updatedRows**;**}

1. 在AndroidManifest.xml文件中注册内容提供器（创建时已自动注册）

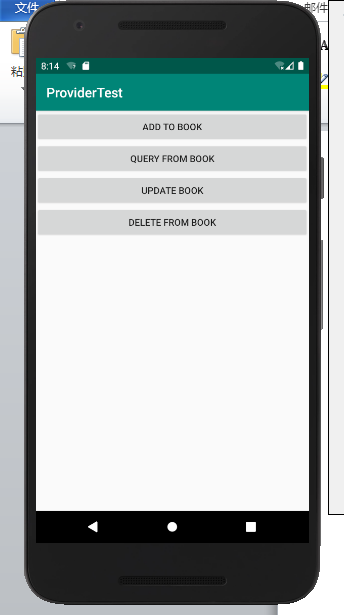
<provider  
 android:name=".DatabaseProvider"  
 android:authorities="com.example.databasetest"  
 android:enabled="true"  
 android:exported="true">  
</provider>

1. 创建新项目ProviderTest来访问DatabaseTest中的数据

activity\_main.xml文件👇

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" >  
  
 <Button  
 android:id="@+id/add\_data"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Add To Book" />  
  
 <Button  
 android:id="@+id/query\_data"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Query From Book" />  
  
 <Button  
 android:id="@+id/update\_data"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Update Book" />  
  
 <Button  
 android:id="@+id/delete\_data"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Delete From Book" />  
  
</LinearLayout>

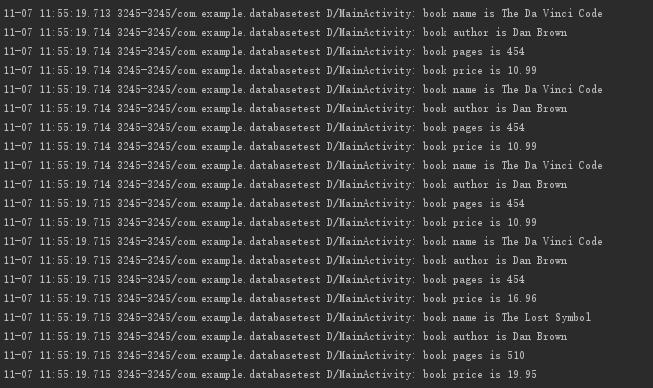
布局展示



MainActivity.java

package com.example.providertest**;**import androidx.appcompat.app.AppCompatActivity**;**import android.content.ContentValues**;**import android.database.Cursor**;**import android.net.Uri**;**import android.os.Bundle**;**import android.util.Log**;**import android.view.View**;**import android.widget.Button**;**public class MainActivity extends AppCompatActivity {  
  
 private String newId**;** @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState)**;** setContentView(R.layout.*activity\_main*)**;** Button addData = (Button) findViewById(R.id.*add\_data*)**;** addData.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 // 添加数据  
 Uri uri = Uri.*parse*("content://com.example.databasetest.provider/book")**;** ContentValues values = new ContentValues()**;** values.put("name"**,** "A Clash of Kings")**;** values.put("author"**,** "George Martin")**;** values.put("pages"**, 1040**)**;** values.put("price"**, 55.55**)**;** Uri newUri = getContentResolver().insert(uri**,** values)**;** newId = newUri.getPathSegments().get(**1**)**;** }  
 })**;** Button queryData = (Button) findViewById(R.id.*query\_data*)**;** queryData.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 // 查询数据  
 Uri uri = Uri.*parse*("content://com.example.databasetest.provider/book")**;** Cursor cursor = getContentResolver().query(uri**,** null**,** null**,** null**,** null)**;** if (cursor != null) {  
 while (cursor.moveToNext()) {  
 String name = cursor.getString(cursor. getColumnIndex("name"))**;** String author = cursor.getString(cursor. getColumnIndex("author"))**;** int pages = cursor.getInt(cursor.getColumnIndex ("pages"))**;** double price = cursor.getDouble(cursor. getColumnIndex("price"))**;** Log.*d*("MainActivity"**,** "book name is " + name)**;** Log.*d*("MainActivity"**,** "book author is " + author)**;** Log.*d*("MainActivity"**,** "book pages is " + pages)**;** Log.*d*("MainActivity"**,** "book price is " + price)**;** }  
 cursor.close()**;** }  
 }  
 })**;** Button updateData = (Button) findViewById(R.id.*update\_data*)**;** updateData.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 // 更新数据  
 Uri uri = Uri.*parse*("content://com.example.databasetest.provider/book/" + newId)**;** ContentValues values = new ContentValues()**;** values.put("name"**,** "A Storm of Swords")**;** values.put("pages"**, 1216**)**;** values.put("price"**, 24.05**)**;** getContentResolver().update(uri**,** values**,** null**,** null)**;** }  
 })**;** Button deleteData = (Button) findViewById(R.id.*delete\_data*)**;** deleteData.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 // 删除数据  
 Uri uri = Uri.*parse*("content://com.example.databasetest.provider/book/" + newId)**;** getContentResolver().delete(uri**,** null**,** null)**;** }  
 })**;** }  
}

DatabaseTest表格结果为：



将DatabaseTest表格数据进行增删查改后ProviderTest查询结果为：

