**Using your own SQLite database in Android applications**

Most all of the Android examples and tutorials out there assume you want to create and populate your database at runtime and not to use and access an independent, preloaded database with your Android application.

The method I'm going to show you takes your own SQLite database file from the "assets" folder and copies into the system database path of your application so the SQLiteDatabase API can open and access it normally.

**1. Preparing the SQLite database file.**

Assuming you already have your sqlite database created, we need to do some modifications to it.  
If you don't have a sqlite manager I recommend you to download the opensource [SQLite Database Browser](http://sourceforge.net/projects/sqlitebrowser/) available for Win/Linux/Mac.

Open your database and add a new table called "android\_metadata", you can execute the following SQL statement to do it:

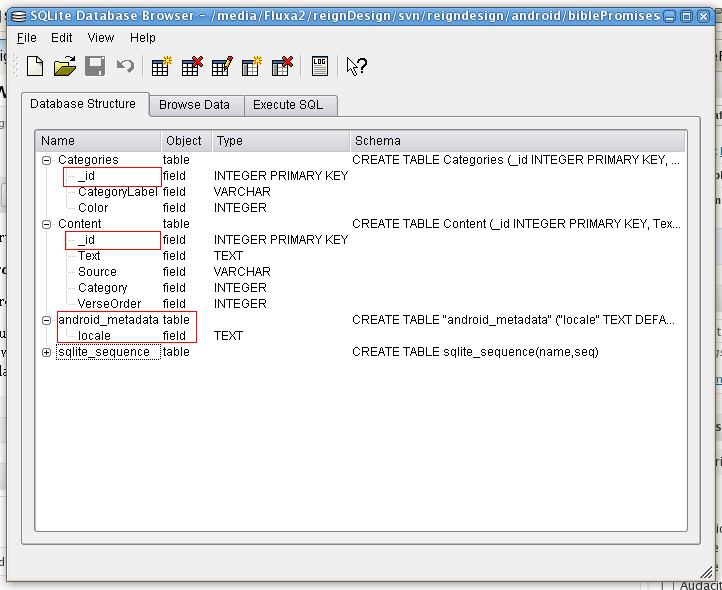
1. **CREATE** **TABLE** "android\_metadata" ("locale" TEXT **DEFAULT** 'en\_US')

Now insert a single row with the text 'en\_US' in the "android\_metadata" table:

1. **INSERT** **INTO** "android\_metadata" **VALUES** ('en\_US')

Then, it is necessary to rename the primary id field of your tables to "\_id" so Android will know where to bind the id field of your tables.  
You can easily do this with [SQLite Database Browser](http://sourceforge.net/projects/sqlitebrowser/) by pressing the edit table button Edit Table, then selecting the table you want to edit and finally selecting the field you want to rename.

After renaming the id field of all your data tables to "\_id" and adding the "android\_metadata" table, your database it's ready to be used in your Android application.

Modified database

Note: in this image we see the tables "Categories" and "Content" with the id field renamed to "\_id" and the just added table "android\_metadata".

**2. Copying, opening and accessing your database in your Android application.**

Now just put your database file in the "assets" folder of your project and create a Database Helper class by extending the [SQLiteOpenHelper](http://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html) class from the "android.database.sqlite" package.

Make your DataBaseHelper class look like this:

1. **public** **class** DataBaseHelper **extends** SQLiteOpenHelper{
3. *//The Android's default system path of your application database.*
4. **private** **static** [**String**](http://www.google.com/search?hl=en&q=allinurl%3AString+java.sun.com&btnI=I%27m%20Feeling%20Lucky) DB\_PATH = "/data/data/YOUR\_PACKAGE/databases/";
6. **private** **static** [**String**](http://www.google.com/search?hl=en&q=allinurl%3AString+java.sun.com&btnI=I%27m%20Feeling%20Lucky) DB\_NAME = "myDBName";
8. **private** SQLiteDatabase myDataBase;
10. **private** **final** [**Context**](http://www.google.com/search?hl=en&q=allinurl%3AContext+java.sun.com&btnI=I%27m%20Feeling%20Lucky) myContext;
12. */\*\**
13. *\* Constructor*
14. *\* Takes and keeps a reference of the passed context in order to access to the application assets and resources.*
15. *\* @param context*
16. *\*/*
17. **public** DataBaseHelper(**[Context](http://www.google.com/search?hl=en&q=allinurl%3AContext+java.sun.com&btnI=I%27m%20Feeling%20Lucky)** context) {
19. **super**(context, DB\_NAME, **null**, 1);
20. **this**.myContext = context;
21. }
23. */\*\**
24. *\* Creates a empty database on the system and rewrites it with your own database.*
25. *\* \*/*
26. **public** void createDataBase() **throws** [**IOException**](http://www.google.com/search?hl=en&q=allinurl%3AIOException+java.sun.com&btnI=I%27m%20Feeling%20Lucky){
28. boolean dbExist = checkDataBase();
30. if(dbExist){
31. *//do nothing - database already exist*
32. }else{
34. *//By calling this method and empty database will be created into the default system path*
35. *//of your application so we are gonna be able to overwrite that database with our database.*
36. **this**.getReadableDatabase();
38. **try** {
40. copyDataBase();
42. } **catch** (**[IOException](http://www.google.com/search?hl=en&q=allinurl%3AIOException+java.sun.com&btnI=I%27m%20Feeling%20Lucky)** e) {
44. **throw** **new** [**Error**](http://www.google.com/search?hl=en&q=allinurl%3AError+java.sun.com&btnI=I%27m%20Feeling%20Lucky)("Error copying database");
46. }
47. }
49. }
51. */\*\**
52. *\* Check if the database already exist to avoid re-copying the file each time you open the application.*
53. *\* @return true if it exists, false if it doesn't*
54. *\*/*
55. **private** boolean checkDataBase(){
57. SQLiteDatabase checkDB = **null**;
59. **try**{
60. [**String**](http://www.google.com/search?hl=en&q=allinurl%3AString+java.sun.com&btnI=I%27m%20Feeling%20Lucky) myPath = DB\_PATH + DB\_NAME;
61. checkDB = SQLiteDatabase.openDatabase(myPath, **null**, SQLiteDatabase.OPEN\_READONLY);
63. }**catch**(SQLiteException e){
65. *//database does't exist yet.*
67. }
69. if(checkDB != **null**){
71. checkDB.close();
73. }
75. **return** checkDB != **null** ? **true** : **false**;
76. }
78. */\*\**
79. *\* Copies your database from your local assets-folder to the just created empty database in the*
80. *\* system folder, from where it can be accessed and handled.*
81. *\* This is done by transfering bytestream.*
82. *\* \*/*
83. **private** void copyDataBase() **throws** [**IOException**](http://www.google.com/search?hl=en&q=allinurl%3AIOException+java.sun.com&btnI=I%27m%20Feeling%20Lucky){
85. *//Open your local db as the input stream*
86. [**InputStream**](http://www.google.com/search?hl=en&q=allinurl%3AInputStream+java.sun.com&btnI=I%27m%20Feeling%20Lucky) myInput = myContext.getAssets().open(DB\_NAME);
88. *// Path to the just created empty db*
89. [**String**](http://www.google.com/search?hl=en&q=allinurl%3AString+java.sun.com&btnI=I%27m%20Feeling%20Lucky) outFileName = DB\_PATH + DB\_NAME;
91. *//Open the empty db as the output stream*
92. [**OutputStream**](http://www.google.com/search?hl=en&q=allinurl%3AOutputStream+java.sun.com&btnI=I%27m%20Feeling%20Lucky) myOutput = **new** [**FileOutputStream**](http://www.google.com/search?hl=en&q=allinurl%3AFileOutputStream+java.sun.com&btnI=I%27m%20Feeling%20Lucky)(outFileName);
94. *//transfer bytes from the inputfile to the outputfile*
95. byte[] buffer = **new** byte[1024];
96. int length;
97. while ((length = myInput.read(buffer))>0){
98. myOutput.write(buffer, 0, length);
99. }
101. *//Close the streams*
102. myOutput.flush();
103. myOutput.close();
104. myInput.close();
106. }
108. **public** void openDataBase() **throws** [**SQLException**](http://www.google.com/search?hl=en&q=allinurl%3ASQLException+java.sun.com&btnI=I%27m%20Feeling%20Lucky){
110. *//Open the database*
111. [**String**](http://www.google.com/search?hl=en&q=allinurl%3AString+java.sun.com&btnI=I%27m%20Feeling%20Lucky) myPath = DB\_PATH + DB\_NAME;
112. myDataBase = SQLiteDatabase.openDatabase(myPath, **null**, SQLiteDatabase.OPEN\_READONLY);
114. }
116. @Override
117. **public** **synchronized** void close() {
119. if(myDataBase != **null**)
120. myDataBase.close();
122. **super**.close();
124. }
126. @Override
127. **public** void onCreate(SQLiteDatabase db) {
129. }
131. @Override
132. **public** void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
134. }
136. *// Add your public helper methods to access and get content from the database.*
137. *// You could return cursors by doing "return myDataBase.query(....)" so it'd be easy*
138. *// to you to create adapters for your views.*
140. }

That's it.  
Now you can create a new instance of this DataBaseHelper class and call the createDataBase() and openDataBase() methods. Remember to change the "YOUR\_PACKAGE" to your application package namespace (i.e: com.examplename.myapp) in the DB\_PATH string.

1. ...
3. DataBaseHelper myDbHelper = **new** DataBaseHelper();
4. myDbHelper = **new** DataBaseHelper(**this**);
6. **try** {
8. myDbHelper.createDataBase();
10. } **catch** (**[IOException](http://www.google.com/search?hl=en&q=allinurl%3AIOException+java.sun.com&btnI=I%27m%20Feeling%20Lucky)** ioe) {
12. **throw** **new** [**Error**](http://www.google.com/search?hl=en&q=allinurl%3AError+java.sun.com&btnI=I%27m%20Feeling%20Lucky)("Unable to create database");
14. }
16. **try** {
18. myDbHelper.openDataBase();
20. }**catch**(**[SQLException](http://www.google.com/search?hl=en&q=allinurl%3ASQLException+java.sun.com&btnI=I%27m%20Feeling%20Lucky)** sqle){
22. **throw** sqle;
24. }
26. ...

**This article has also been translated into different languages**

* [Serbo-Croatian](http://science.webhostinggeeks.com/koristite-svoju-sqlite-bazu) by **Anja Skrba** from [Webhostinggeeks.com](http://webhostinggeeks.com/)
* [Armenian](http://students.studybay.com/?p=134) by **Gajk Melikyan** from [StudyBay](http://studybay.com/" \t "_blank)