public abstract class

SQLiteOpenHelper

extends Object

Summary: Ctors | Methods | Inherited Methods | [Expand All]
Added in API level 1

java.lang.Object
 ↳android.database.sqlite.SQLiteOpenHelper

Class Overview

A helper class to manage database creation and version management.

You create a subclass implementing `onCreate(SQLiteDatabase)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onCreate(android.database.sqlite.SQLiteDatabase))`, `onUpgrade(SQLiteDatabase, int, int)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onUpgrade(android.database.sqlite.SQLiteDatabase, int, int))` and optionally `onOpen(SQLiteDatabase)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onOpen(android.database.sqlite.SQLiteDatabase))`, and this class takes care of opening the database if it exists, creating it if it does not, and upgrading it as necessary. Transactions are used to make sure the database is always in a sensible state.

This class makes it easy for `ContentProvider (/reference/android/content/ContentProvider.html)` implementations to defer opening and upgrading the database until first use, to avoid blocking application startup with long-running database upgrades.

For an example, see the NotePadProvider class in the NotePad sample application, in the *samples/* directory of the SDK.

> **Note:** this class assumes monotonically increasing version numbers for upgrades.

Summary

Public Constructors

SQLiteOpenHelper (Context context, String name, SQLiteDatabase.CursorFactory factory, int version) Create a helper object to create, open, and/or manage a database.	
SQLiteOpenHelper (Context context, String name, SQLiteDatabase.CursorFactory factory, int version, DatabaseErrorHandler errorHandler) Create a helper object to create, open, and/or manage a database.	

Public Methods

synchronized void	close () Close any open database object.
String	getDatabaseName () Return the name of the SQLite database being opened, as given to the constructor.
SQLiteDatabase	getReadableDatabase () Create and/or open a database.
SQLiteDatabase	getWritableDatabase () Create and/or open a database that will be used for reading and writing.
void	onConfigure (SQLiteDatabase db) Called when the database connection is being configured, to enable features such as write-ahead logging or foreign key support.
abstract void	onCreate (SQLiteDatabase db) Called when the database is created for the first time.
void	onDowngrade (SQLiteDatabase db, int oldVersion, int newVersion) Called when the database needs to be downgraded.
void	onOpen (SQLiteDatabase db) Called when the database has been opened.
abstract void	onUpgrade (SQLiteDatabase db, int oldVersion, int newVersion) Called when the database needs to be upgraded.
void	setWriteAheadLoggingEnabled (boolean enabled) Enables or disables the use of write-ahead logging for the database.

Inherited Methods [Expand]

▸ From class java.lang.Object

Public Constructors

public **SQLiteOpenHelper** (Context context, String name, SQLiteDatabase.CursorFactory factory, int version)

Added in API level 1

Create a helper object to create, open, and/or manage a database. This method always returns very quickly. The database is not actually created or opened until one of `getWritableDatabase()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#getWritableDatabase())` or `getReadableDatabase()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#getReadableDatabase())` is called.

Parameters

context	to use to open or create the database
name	of the database file, or null for an in-memory database
factory	to use for creating cursor objects, or null for the default
version	number of the database (starting at 1); if the database is older, `onUpgrade(SQLiteDatabase, int, int)` will be used to upgrade the database; if the database is newer, `onDowngrade(SQLiteDatabase, int, int)` will be used to downgrade the database

public **SQLiteOpenHelper** (Context context, String name, SQLiteDatabase.CursorFactory factory, int version, DatabaseErrorHandler errorHandler)

Added in API level 11

Create a helper object to create, open, and/or manage a database. The database is not actually created or opened until one of `getWritableDatabase()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#getWritableDatabase())` or `getReadableDatabase()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#getReadableDatabase())` is called.

Accepts input param: a concrete instance of `DatabaseErrorHandler` `(/reference/android/database/DatabaseErrorHandler.html)` to be used to handle corruption when sqlite reports database corruption.

Parameters

context	to use to open or create the database
name	of the database file, or null for an in-memory database
factory	to use for creating cursor objects, or null for the default
version	number of the database (starting at 1); if the database is older, `onUpgrade(SQLiteDatabase, int, int)` will be used to upgrade the database; if the database is newer, `onDowngrade(SQLiteDatabase, int, int)` will be used to downgrade the database
errorHandler	the `DatabaseErrorHandler` to be used when sqlite reports database corruption, or null to use the default error handler.

Public Methods

public synchronized void **close** ()

Added in API level 1

Close any open database object.

public String **getDatabaseName** ()

Added in API level 14

Return the name of the SQLite database being opened, as given to the constructor.

public SQLiteDatabase **getReadableDatabase** ()

Added in API level 1

Create and/or open a database. This will be the same object returned by `getWritableDatabase()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#getWritableDatabase())` unless some problem, such as a full disk, requires the database to be opened read-only. In that case, a read-only database object will be returned. If the problem is fixed, a future call to `getWritableDatabase()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#getWritableDatabase())` may succeed, in which case the read-only database object will be closed and the read/write object will be returned in the future.

> Like `getWritableDatabase()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#getWritableDatabase())`, this method may take a long time to return, so you should not call it from the application main thread, including from `ContentProvider.onCreate()` `(/reference/android/content/ContentProvider.html#onCreate())`.

Returns

a database object valid until `getWritableDatabase()` or `close()` is called.

Throws

SQLiteException if the database cannot be opened

public SQLiteDatabase **getWritableDatabase** () Added in API level 1

Create and/or open a database that will be used for reading and writing. The first time this is called, the database will be opened and `onCreate(SQLiteDatabase)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onCreate(android.database.sqlite.SQLiteDatabase))`, `onUpgrade(SQLiteDatabase, int, int)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onUpgrade(android.database.sqlite.SQLiteDatabase, int, int))` and/or `onOpen(SQLiteDatabase)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onOpen(android.database.sqlite.SQLiteDatabase))` will be called.

Once opened successfully, the database is cached, so you can call this method every time you need to write to the database. (Make sure to call `close()` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#close())` when you no longer need the database.) Errors such as bad permissions or a full disk may cause this method to fail, but future attempts may succeed if the problem is fixed.

> Database upgrade may take a long time, you should not call this method from the application main thread, including from `ContentProvider.onCreate()` `(/reference/android/content/ContentProvider.html#onCreate())`.

Returns

a read/write database object valid until `close()` is called

Throws

SQLiteException if the database cannot be opened for writing

public void **onConfigure** (SQLiteDatabase db) Added in API level 16

Called when the database connection is being configured, to enable features such as write-ahead logging or foreign key support.

This method is called before `onCreate(SQLiteDatabase)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onCreate(android.database.sqlite.SQLiteDatabase))`, `onUpgrade(SQLiteDatabase, int, int)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onUpgrade(android.database.sqlite.SQLiteDatabase, int, int))`, `onDowngrade(SQLiteDatabase, int, int)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onDowngrade(android.database.sqlite.SQLiteDatabase, int, int))`, or `onOpen(SQLiteDatabase)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onOpen(android.database.sqlite.SQLiteDatabase))` are called. It should not modify the database except to configure the database connection as required.

This method should only call methods that configure the parameters of the database connection, such as `enableWriteAheadLogging()` `(/reference/android/database/sqlite/SQLiteDatabase.html#enableWriteAheadLogging())` `setForeignKeyConstraintsEnabled(boolean)` `(/reference/android/database/sqlite/SQLiteDatabase.html#setForeignKeyConstraintsEnabled(boolean))`, `setLocale(Locale)` `(/reference/android/database/sqlite/SQLiteDatabase.html#setLocale(java.util.Locale))`, `setMaximumSize(long)` `(/reference/android/database/sqlite/SQLiteDatabase.html#setMaximumSize(long))`, or executing PRAGMA statements.

Parameters

db The database.

public abstract void **onCreate** (SQLiteDatabase db) Added in API level 1

Called when the database is created for the first time. This is where the creation of tables and the initial population of the tables should happen.

Parameters

db The database.

public void **onDowngrade** (SQLiteDatabase db, int oldVersion, int newVersion) Added in API level 11

Called when the database needs to be downgraded. This is strictly similar to `onUpgrade(SQLiteDatabase, int, int)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onUpgrade(android.database.sqlite.`

`SQLiteDatabase, int, int))` method, but is called whenever current version is newer than requested one. However, this method is not abstract, so it is not mandatory for a customer to implement it. If not overridden, default implementation will reject downgrade and throws SQLiteException

This method executes within a transaction. If an exception is thrown, all changes will automatically be rolled back.

Parameters

db	The database.
oldVersion	The old database version.
newVersion	The new database version.

public void **onOpen** (SQLiteDatabase db)

Added in API level 1

Called when the database has been opened. The implementation should check `isReadOnly()` `(/reference/android/database/sqlite/SQLiteDatabase.html#isReadOnly())` before updating the database.

This method is called after the database connection has been configured and after the database schema has been created, upgraded or downgraded as necessary. If the database connection must be configured in some way before the schema is created, upgraded, or downgraded, do it in `onConfigure(SQLiteDatabase)` `(/reference/android/database/sqlite/SQLiteOpenHelper.html#onConfigure(android.database.sqlite.SQLiteDatabase))` instead.

Parameters

db	The database.

public abstract void **onUpgrade** (SQLiteDatabase db, int oldVersion, int newVersion)

Added in API level 1

Called when the database needs to be upgraded. The implementation should use this method to drop tables, add tables, or do anything else it needs to upgrade to the new schema version.

The SQLite ALTER TABLE documentation can be found here (http://sqlite.org/lang_altertable.html). If you add new columns you can use ALTER TABLE to insert them into a live table. If you rename or remove columns you can use ALTER TABLE to rename the old table, then create the new table and then populate the new table with the contents of the old table.

This method executes within a transaction. If an exception is thrown, all changes will automatically be rolled back.

Parameters

db	The database.
oldVersion	The old database version.
newVersion	The new database version.

public void **setWriteAheadLoggingEnabled** (boolean enabled)

Added in API level 16

Enables or disables the use of write-ahead logging for the database. Write-ahead logging cannot be used with read-only databases so the value of this flag is ignored if the database is opened read-only.

Parameters

enabled	True if write-ahead logging should be enabled, false if it should be disabled.

See Also

`enableWriteAheadLogging()`