How to use sqlite3 module with electron?

Asked 4 years, 1 month ago Active 4 months ago Viewed 65k times

72

31

I want to develop desktop app using electron that uses sqlite3 package installed via npm with the command

```
npm install --save sqlite3
```

but it gives the following error in electron browser console

```
Uncaught Error: Cannot find module
'E:\allcode\eapp\node_modules\sqlite3\lib\binding\node-v45-win32-x64\node_sqlite3.node'
```

My development environment is windows 8.1 x64 node version 12.7

my **package.json** file looks like this:

```
{
  "name": "eapp",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "electron ."
  },
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "electron-prebuilt": "^0.32.1"
  },
  "dependencies": {
    "angular": "^1.3.5",
    "sqlite3": "^3.1.0"
  }
}
```

index.js file

```
var app = require('app');
var BrowserWindow = require('browser-window');
require('crash-reporter').start();
var mainWindow = null;


app.on('window-all-closed', function() {
    if (process.platform != 'darwin') {
        app.quit();
    }
});

app.on('ready', function() {
    // Create the browser window.
    mainWindow = new BrowserWindow({width: 800, height: 600});
    mainWindow.loadUrl('file://' + __dirname + '/index.html');
    mainWindow.openDevTools();
```

```
    mainWindow.on('closed', function() {
        mainWindow = null;
    });
});
```

my.js file

```
var sqlite3 = require('sqlite3').verbose();
var db = new sqlite3.Database('mydb.db');

db.serialize(function() {
    db.run("CREATE TABLE if not exists lorem (info TEXT)");

    var stmt = db.prepare("INSERT INTO lorem VALUES (?)");
    for (var i = 0; i < 10; i++) {
        stmt.run("Ipsum " + i);
    }
    stmt.finalize();

    db.each("SELECT rowid AS id, info FROM lorem", function(err, row) {
        console.log(row.id + ": " + row.info);
    });
});

db.close();
```

index.html file

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title></title>
</head>
<body>
<div >
    <div>
        <h2>Hello</h2>
    </div>

</div>
<!--<script src="js/jquery-1.11.3.min.js"></script>-->
<script src="js/my.js"></script>
</body>
</html>
```

node.js sqlite npm electron

edited Aug 29 '17 at 9:00
statox
1,348 ● 1 ● 13 ● 32

asked Sep 10 '15 at 14:13
manas
2,479 ● 6 ● 32 ● 45

8 Answers

104

By far the easiest way to use SQLite with electron is with `electron-builder`.

First, add a postinstall step in your package.json:

```
"scripts": {
   "postinstall": "install-app-deps"
   ...
}
```

and then install the necessary dependencies and build:

```
npm install --save-dev electron-builder
npm install --save sqlite3
npm run postinstall
```

electron-builder will build the native module for your platform, with the correct name for the Electron binding; and you can then `require` it in code as normal.

See my github repo and blog post - it took me quite a while to figure this out too.

edited Mar 5 '17 at 5:25

Alex Beals
676 ● 3 ● 8 ● 22

answered Dec 19 '16 at 20:48

Steve Melia
1,061 ● 1 ● 5 ● 4

1 Using your electron-boilerplate-sqlite is indeed the easier method, but with `npm install` – Bernardo Ramos Feb 24 '17 at 8:22

1 when run 'npm run postinstall', i got this error "Building the projects in this solution one at a time. To enable parallel build, please add the "/m" switch." – chirag Sep 25 '18 at 8:55

This has saved my hide not once, but twice so far! – John Nesbitt Feb 4 at 15:43

Does anyone know whether the `--save sqlite3` option can be reliably tagged onto the first line (with `--save-dev electron-builder`), where `npm install` would only be run once? – Phillip Aug 16 at 17:21

This does not work for me, I get errors like ` ../../nan/nan_object_wrap.h:67:18: error: ‘class Nan::Persistent<v8::Object>’ has no member named ‘MarkIndependent’` – Michael Sep 9 at 1:09

16

I would not recommend the native node sqlite3 module. It requires being rebuild to work with electron. This is a massive pain to do - At least I can never get it to work and their a no instructions to for rebuilding modules on windows.

Instead have a look at kripken's 'sql.js' module which is sqlite3 that has been compiled 100% in JavaScript. https://github.com/kripken/sql.js/

edited Dec 18 '16 at 10:16

answered Dec 18 '16 at 9:27

Joue Bien
177 ● 1 ● 5

3 Electron docs are pretty clear that in order to use bundled native modules, you are required to rebuild them in order to work with electron. Either use the electron-rebuild utility which works most of the time or manually set the gyp flags: electron.atom.io/docs/tutorial/using-native-node-modules – Bret Feb 6 '17 at 18:57

1 Sorry for the slow response. github.com/electron/electron-rebuild is a handy tool for rebuilding in development. github.com/electron-userland/electron-builder/wiki/… describes a multi-platform build strategy for production builds. Jut remember, once you introduce native deps, you lose the ability to

cross compile for different OSs. github.com/nodejs/node-gyp#installation has decent docs on necessary windows compiler tools. – Bret Mar 16 '17 at 21:17

1 none of these comments are relevant to this answer! – user3791372 May 21 '17 at 0:14

9 It should be mentioned that sql.js can not operate on the filesystem. Everytime you need to change the database you have to write the whole thing to disk. Making it pretty much useless for most scenarios. – mode777 Jun 14 '17 at 17:32

2 sql.js its more like a toy, made for fun. It cannot be better than NeDB and other nosql databases, because its store all database in memory. So there is no good reason to use it. For small database use nosql like NeDB, for larger you have to compile sqlite – Daimos Oct 22 '17 at 20:21

|

8

Two aspects are to be considered here:

1. Setting `NODE_PATH` : this lets electron know where to find your modules (see this answer for a thorough explanation)
2. Compiling native modules against electron headers: see official docs

And checkout the following questions, that ask the same thing:

- Electron App with Database
- Using NodeJS plugins in Elelectron

My tip would be to give lovefield (by Google) a try.

edited May 23 '17 at 12:34
Community ♦
1 ● 1

answered Sep 12 '15 at 10:19
Yan Foto
7,194 ● 3 ● 36 ● 67

In Nodewebkit, I must compile the sqlite. Same rule apply for the electron? – Vedran Maricevic. Dec 17 '15 at 9:10

@Wexoni AFAIK every node module with native bindings must be compiled against electron headers. – Yan Foto Dec 17 '15 at 9:16

Forgive my ignorance, but the way I understand it... same rule apply for Electron and for NW? – Vedran Maricevic. Dec 17 '15 at 9:20

1 @Wexoni I haven't worked with NW, but I know that `sqlite3` must be compiled before being able to use it in electron. Am I understanding your question correctly? – Yan Foto Dec 17 '15 at 10:17

1 @YannBertrand they still use own headers. So i suppose you'd still need to compile it against their headers. – Yan Foto May 25 '16 at 20:37

|

7

I was having same problem. Tried everything and atlast this worked for me :-

```
npm install --save sqlite3
npm install --save electron-rebuild
npm install --save electron-prebuilt
.\node_modules\.bin\electron-rebuild.cmd
```

This will create "electron-v1.3-win32-x64" folder in .\node_modules\sqlite3\lib\binding\ location which is used by electron to use sqlite3.

Just start application and you will be able to use sqlite3 now.

answered Aug 21 '16 at 10:38

Rj-s

278 ● 1 ● 5 ● 15

It just created an empty folder, but still no `node_sqlite3.node` file in it – Mehdi Dehghani Mar 2 '18 at 12:10

electron-prebuilt has been renamed to electron. For more details, see electron.atom.io/blog/2016/08/16/npm-install-electron – Jacob Nelson Mar 1 at 8:37

6

A simpler solution:

1. Install electron-rebuild `npm i electron-rebuild --save-dev`
2. Launch electron-rebuild `./node_modules/.bin/electron-rebuild` (or `.\node_modules\.bin\electron-rebuild.cmd` on windows)
3. Go to "**node_modules/sqlite3/lib/binding/**" and rename the folder "**electron-v0.36-darwin-x64**" to "**node-*v47*-darwin-x64**"

PS: *v47* is my version, be careful to choose the good one (in your case *v45*)

answered Dec 21 '15 at 0:32

Fabien Sa

6,217 ● 2 ● 32 ● 38

2 Can you explain the third step? Why rename? – m4heshd Apr 3 '18 at 22:00

3

I encounter this error too. Here is how i solve it:
`npm install --save-dev electron-rebuild`
then:
`./node_modules/.bin/electron-rebuild`

from: https://electronjs.org/docs/tutorial/using-native-node-modules

ps: While it's on rebuilding, don't use `npm start` to lanch the electron app. Otherwise the rebuild process would fail.

answered Jun 19 '18 at 11:43

valleygtc

71 ● 3

2

Have a look at a similar answer here

TL;DR

```
cd .\node_modules\sqlite3
npm install nan --save
npm run prepublish
node-gyp configure --module_name=node_sqlite3 --module_path=../lib/binding/electron-
v1.3-win32-x64
node-gyp rebuild --target=1.3.2 --arch=x64 --target_platform=win32 --dist-
url=http://electron.atom.io/ --module_name=node_sqlite3 --
module_path=../lib/binding/electron-v1.3-win32-x64
```

edited May 23 '17 at 12:34
Community ♦
1 ● 1

answered Aug 12 '16 at 6:00

smkndblvr
83 ● 6

3 what the script `prepublish` looks like? – Victor Ivens Sep 26 '17 at 19:31

0

It works for me in version 3 and 4, unfortunately NOT version 5. See the sqlite3 documentation for details: https://www.npmjs.com/package/sqlite3#custom-builds-and-electron or otherwise run the following line: `npm install sqlite3 --runtime=electron --target=4.0.0 --dist-url=https://atom.io/download/electron`

answered Jul 6 at 22:35

Sailab Rahi
412 ● 1 ● 6 ● 10