ATOMIC OBJECT

March 27, 2019 by: **Joe Bustamante** **16 Comments**

A Comprehensive Guide to Fixing Node-Gyp Issues on Windows

`node-gyp` <https://github.com/nodejs/node-gyp> is a tool that enables the compilation of native add-on modules for **Node** <https://nodejs.org/en> in multiple platforms. It has widespread use and is included as a dependency in many NPM packages.

On most systems, this isn't an issue, and installing `node-gyp` with the rest of your packages works as expected. Unfortunately, this is not the case with Windows, as is evidenced by **this thread from 2015** <https://github.com/nodejs/node-gyp/issues/629#issuecomment-153196245>. The Windows environment makes getting `node-gyp` to work a less-than-stellar developer experience, full of multiple pitfalls and many ways for things to go wrong.

This guide is meant to help solve the issues that can arise when installing a package that requires `node-gyp`.

1. Try running `npm install` with the `--no-optional` flag.

If you're lucky, the dependency that requires `node-gyp` will be optional, and you can skip the entire process required to get it working. Try running `npm install --no-optional` to install only the required dependencies.

If you run this and still get the error, I have bad news: You're in for a bit of a ride. As we begin our journey into getting `node-gyp` up and running, here's an important note for all of the steps that follow. Make sure that you're always working in an elevated terminal (with administrator privileges) and that you restart your console whenever a download is complete.

2. Try downloading the `windows-build-tools` package.

According to the `node-gyp` documentation, this step should be the end-all-be-all solution to fixing `node-gyp` problems. For most people, that's true. NPM has a package called `windows-build-tools` that should automatically install everything you need to get `node-gyp` working,

including the Microsoft build tools, compilers, Python, and everything else required to build native Node modules on Windows.

The good news is that installing this package should take care of all of the wiring up of these components. The bad news is that there are a lot of things included in it.

Depending on the version you download, it can hover between three and eight gigabytes (all to get some dependencies installed!). The install can take upwards of 30 minutes depending on your connection, so don't despair if it seems like the install is hanging for a while.

You can download them by running this command: `npm install --global --production windows-build-tools --vs2015`

Important note

If you run this command without any additional flags, you'll install the files associated with the latest version of **Visual Studio** <https://visualstudio.microsoft.com>, which is VS2017 at the time of writing. However, `node-gyp` requires the `v140` distributable, not the `v150` (which comes with VS2017). This is why the `--vs2015` flag is added to the end of the command, since that's the last version of Visual Studio that came with the `v140` package. You can see more notes about that near the bottom of **the package's website** <https://www.npmjs.com/package/windows-build-tools>.

Hopefully, that's all it will take for you to get everything installed. If you've still got issues, then you're going to have to grab all of the required files manually.

3. Download the Visual Studio 2015 build tools manually.

Rather than installing the build tools through NPM, download them manually. You can find them **on the Microsoft download page** <https://www.microsoft.com/en-us/download/details.aspx?id=48159>. Once they're downloaded, just run the installer.

4. Tell Node to use the 2015 build tools.

Now that we have the build tools, we need to tell Node to use them. You'll have to run this command: `npm config set msvs_version 2015 -global`

5. Make sure you have Python 2.7 installed.

Next up is to download Python 2.7. This is important—by default, you'll install the newest version (3.x.x), but only 2.7 is compatible with `node-gyp`. If you try to use a newer version of Python,

you'll get a syntax error due to `print` being made into an actual function in Python 3.

If you have another version of Python already installed, that's okay. They can coexist with each other. You can grab Python 2.7 **at this link** <https://www.python.org/download/releases/2.7/>.

6. Set your Node config to use Python 2.7.

Now that you have Python 2.7, you're going to have to set Node to use it by default. To do that, run this command: `npm config set python python2.7`

If you followed the last few steps, you should now have everything necessary to get `node-gyp` working. Make sure you've restarted your terminal and are running it as an administrator, and try doing your install again. Hopefully, at this point, you can successfully install the dependencies you need. If not, we're going to have to try one last thing.

7. Repeat Step 2 with the Visual Studio 2017 build tools.

I've personally had issues when I've tried to download the 2017 version of the build tools, even when trying to use newer versions of `node-gyp`. If you look online, you'll see lots of other people with the same problem, including some of the commenters on **this StackOverflow question** <https://stackoverflow.com/questions/21658832/npm-install-error-msb3428-could-not-load-the-visual-c-component-vcbuild-ex>.

However, most of the documentation around `node-gyp` and compiling native Node modules on Windows doesn't specify to use the `--vs2015` flag, and some even mention downloading the 2017 version. If you're still having issues with getting `node-gyp` to run, try repeating Step 2 while omitting the `--vs2015` flag.

8. Try installing an older version of Node.

Still getting an installation error? Try installing an older version of Node. If you're on an experimental version, try going back to the last stable release (and then make sure that you're actually using that version when you try and do `npm install`).

If that still doesn't work, try going back to Node 8. You'll find some issues online of people having issues with Node 10 that were resolved by downgrading to Node 8, and sometimes newer versions of Node don't play nice with `node-gyp`.

9. File an issue.

If you're still having trouble getting `node-gyp` to work, then your issue probably doesn't fall into one of the more common problems. It's worth doing some research online to see if that solves your problem. Otherwise, your best option is to file an issue on **the GitHub page** <https://github.com/nodejs/node-gyp> for `node-gyp` and see what advice you can get there.

Other Resources

Much of this information can now be found on **the GitHub readme for node-gyp** <https://github.com/nodejs/node-gyp> , but it wasn't always there. It only made an appearance after a few thousand posts from disgruntled Windows users who just wanted to install some dependencies. Even then, the information is missing some of the issues that arise with installing the wrong version of the build tools.

Another great resource is Microsoft's guide on **working with Node on Windows.** <https://github.com/Microsoft/nodejs-guidelines/blob/master/windows-environment.md#command-line-console-and-other-useful-tools> It even has sections dedicated to compiling native Node modules and resolving basic `node-gyp` problems.

Additionally, this **question on StackOverflow** <https://stackoverflow.com/questions/21658832/npm-install-error-msb3428-could-not-load-the-visual-c-component-vcbuild-ex> contains useful answers and comments regarding this issue. Leave a note in the comments if you've found anything else helpful in solving `node-gyp` issues!