LIN OXIDE
Linux Freedom Thoughts

How Install NodeJS in Ubuntu 16.04 LTS Xenial

17, 2016 | By Arun Pyasi More

JS is a free and open source javascript runtime developed on Chrome's V8 JavaScript engine which is designed to build scalable network cations. It allows the use of Javascript in the server side programming which has the ability to interact with the operating system and orking. It uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time cations that run across distributed devices. Here in this article, we'll learn how to install the latest and stable NodeJS in our machine running ıtu 16.04 LTS Xenial. To install NodeJS, we have many methods we can use. The following are the ways we'll feature in this article.

- Installing using the Official Repository
- Installing using the Github Source Code Clone
- Installing using Node Version Manager (NVM)

Installing using the Official Repository

First of all, as NodeJS is available in the official repository of Ubuntu 16.04 LTS Xenial, we can easily install it using the repository. In order to do so, we'll first need to update the local repository index of our apt package manager.

```
$ sudo apt-get update
```

Once the update is completed, we'll move ahead and run the following command to upgrade our system which will upgrade the packages to the latest available versions.

```
$ sudo apt-get upgrade
```

Then, we'll install nodejs using the apt-get command. Doing so will automatically install the node package manager which comes by default with nodejs. NPM allows us to install node packages from the Node Package Manager Repository.

```
$ sudo apt-get install nodejs nodejs-legacy
```

```
arun@localhost:~$ sudo apt-get install nodejs nodejs-legacy
[sudo] password for arun:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libuv1
The following NEW packages will be installed:
  libuv1 nodejs nodejs-legacy
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 3247 kB of archives.
After this operation, 13.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Once done, we'll be able to install and run our node applications successfully.

Installing using the Github Source Code Clone

If we wanna install the nodejs from the latest clone of Github Source Code then we'll need to follow this method of installation.

of all, we'll need to make sure that the dependencies required for the compilation of NodeJS is installed in our Ubuntu 16.04 machine. So, in
· to install it, we'll need to update the local repository index of the apt package manager.

$ sudo apt-get update

› done, we'll now install the required dependencies from the repository.

$ sudo apt-get install make gcc g++ python

```
localhost:~$  sudo apt-get install make gcc  g++ python
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils cpp cpp-5 g++-5 gcc-5 libasan2 libatomic1 libc-dev-bin libc6-dev libcc1-0
  libcilkrts5 libgcc-5-dev libgomp1 libisl15 libitm1 liblsan0 libmpc3 libmpx0
  libpython-stdlib libpython2.7-minimal libpython2.7-stdlib libquadmath0
  libstdc++-5-dev libtsan0 libubsan0 linux-libc-dev manpages-dev python-minimal
  python2.7 python2.7-minimal
Suggested packages:
  binutils-doc cpp-doc gcc-5-locales g++-multilib g++-5-multilib gcc-5-doc
  libstdc++6-5-dbg gcc-multilib autoconf automake libtool flex bison gdb gcc-doc
  gcc-5-multilib libgcc1-dbg libgomp1-dbg libitm1-dbg libatomic1-dbg libasan2-dbg
  liblsan0-dbg libtsan0-dbg libubsan0-dbg libcilkrts5-dbg libmpx0-dbg libquadmath0-dbg
  glibc-doc libstdc++-5-doc make-doc python-doc python-tk python2.7-doc binfmt-support
The following NEW packages will be installed:
  binutils cpp cpp-5 g++ g++-5 gcc gcc-5 libasan2 libatomic1 libc-dev-bin libc6-dev
  libcc1-0 libcilkrts5 libgcc-5-dev libgomp1 libisl15 libitm1 liblsan0 libmpc3 libmpx0
  libpython-stdlib libpython2.7-minimal libpython2.7-stdlib libquadmath0
  libstdc++-5-dev libtsan0 libubsan0 linux-libc-dev make manpages-dev python
  python-minimal python2.7 python2.7-minimal
0 upgraded, 34 newly installed, 0 to remove and 0 not upgraded.
Need to get 41.4 MB of archives.
After this operation, 155 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

We'll now need to download the official release of nodejs from its Offical Github Repo . To do so, we'll run the following wget command to the respective release of nodejs.

$ wget https://github.com/nodejs/node/archive/v6.2.1.tar.gz

Once the tarballs are downloaded, we'll want to extract it using the following command.

$ tar zxvf v6.2.1.tar.gz

Then, we'll move ahead towards the compilation of it by running the following commands.

$ cd node-6.2.1
$ ./configure

```
arun@localhost:~/node-6.2.1$ ./configure
creating  ./icu_config.gypi
* Using ICU in deps/icu-small
creating  ./icu_config.gypi
{ 'target_defaults': { 'cflags': [],
                       'default_configuration': 'Release',
                       'defines': [],
                       'include_dirs': [],
                       'libraries': []},
  'variables': { 'asan': 0,
                 'gas_version': '2.26',
                 'host_arch': 'x64',
                 'icu_data_file': 'icudt57l.dat',
                 'icu_data_in': '../../deps/icu-small/source/data/in/icudt57l.dat',
                 'icu_endianness': 'l',
                 'icu_gyp_path': 'tools/icu/icu-generic.gyp',
                 'icu_locales': 'en,root',
                 'icu_path': 'deps/icu-small',
                 'icu_small': 'true',
                 'icu_ver_major': '57',
                 'node_byteorder': 'little',
                 'node_enable_v8_vtunejit': 'false',
                 'node_install_npm': 'true',
                 'node_no_browser_globals': 'false',
                 'node_prefix': '/usr/local',
```

its compiled successfully, we'll gonna install it in our machine running 16.04 LTS.

$ sudo make install

```
lling /usr/local/include/node/openssl/rc4.h
lling /usr/local/include/node/openssl/blowfish.h
lling /usr/local/include/node/openssl/e_os2.h
lling /usr/local/include/node/openssl/archs/VC-WIN64A/opensslconf.h
lling /usr/local/include/node/openssl/archs/linux32-s390x/opensslconf.h
lling /usr/local/include/node/openssl/archs/BSD-x86/opensslconf.h
installing /usr/local/include/node/openssl/archs/solaris-x86-gcc/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux-ppc/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux-armv4/opensslconf.h
installing /usr/local/include/node/openssl/archs/aix64-gcc/opensslconf.h
installing /usr/local/include/node/openssl/archs/darwin64-x86_64-cc/opensslconf.h
installing /usr/local/include/node/openssl/archs/BSD-x86_64/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux-x86_64/opensslconf.h
installing /usr/local/include/node/openssl/archs/aix-gcc/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux-aarch64/opensslconf.h
installing /usr/local/include/node/openssl/archs/darwin-i386-cc/opensslconf.h
installing /usr/local/include/node/openssl/archs/solaris64-x86_64-gcc/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux-ppc64/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux64-s390x/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux-x32/opensslconf.h
installing /usr/local/include/node/openssl/archs/VC-WIN32/opensslconf.h
installing /usr/local/include/node/openssl/archs/linux-elf/opensslconf.h
installing /usr/local/include/node/openssl/opensslconf.h
installing /usr/local/include/node/zconf.h
installing /usr/local/include/node/zlib.h
arun@localhost:~/node-6.2.1$
```

The installation process may take much more time depending upon the performance of the machine.

Installing using Node Version Manager (NVM)

The Node Version Manager also known as NVM is a version managing script for nodejs that allows us to manage multiple versions of Node.js to use in the same machine. In order to install NVM, we'll require that curl, libssl-dev and build-essential are installed. In order to do so, we'll need to run the following command.

$ sudo apt-get update
$ sudo apt-get install build-essential libssl-dev curl

```
arun@localhost:~$ sudo apt-get install build-essential libssl-dev curl
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.47.0-1ubuntu2).
The following additional packages will be installed:
  binutils cpp cpp-5 dpkg-dev fakeroot g++ g++-5 gcc gcc-5 libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan2 libatomic1 libc-dev-bin
  libc6-dev libcc1-0 libcilkrts5 libdpkg-perl libfakeroot libfile-fcntllock-perl
  libgcc-5-dev libgomp1 libisl15 libitm1 liblsan0 libmpc3 libmpx0 libquadmath0
  libssl-doc libstdc++-5-dev libtsan0 libubsan0 linux-libc-dev make manpages-dev
  zlib1g-dev
Suggested packages:
  binutils-doc cpp-doc gcc-5-locales debian-keyring g++-multilib g++-5-multilib
  gcc-5-doc libstdc++6-5-dbg gcc-multilib autoconf automake libtool flex bison gdb
  gcc-doc gcc-5-multilib libgcc1-dbg libgomp1-dbg libitm1-dbg libatomic1-dbg
  libasan2-dbg liblsan0-dbg libtsan0-dbg libubsan0-dbg libcilkrts5-dbg libmpx0-dbg
  libquadmath0-dbg glibc-doc libstdc++-5-doc make-doc
The following NEW packages will be installed:
  binutils build-essential cpp cpp-5 dpkg-dev fakeroot g++ g++-5 gcc gcc-5
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan2
  libatomic1 libc-dev-bin libc6-dev libcc1-0 libcilkrts5 libdpkg-perl libfakeroot
  libfile-fcntllock-perl libgcc-5-dev libgomp1 libisl15 libitm1 liblsan0 libmpc3
  libmpx0 libquadmath0 libssl-dev libssl-doc libstdc++-5-dev libtsan0 libubsan0
  linux-libc-dev make manpages-dev zlib1g-dev
raded, 39 newly installed, 0 to remove and 0 not upgraded.
to get 41.2 MB of archives.
this operation, 154 MB of additional disk space will be used.
u want to continue? [Y/n] y
```

the dependencies are installed, we'll now install the latest release of NVM from its Github repository using curl.

$ curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.1/install.sh | bash

```
localhost:~$ curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.1/insta
 | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  7766  100  7766    0     0   5329      0  0:00:01  0:00:01 --:--:--  5333
=> Downloading nvm from git to '/home/arun/.nvm'
=> Cloning into '/home/arun/.nvm'...
remote: Counting objects: 4724, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4724 (delta 0), reused 0 (delta 0), pack-reused 4721
Receiving objects: 100% (4724/4724), 1.27 MiB | 559.00 KiB/s, done.
Resolving deltas: 100% (2807/2807), done.
Checking connectivity... done.
* (HEAD detached at v0.31.1)
  master

=> Appending source string to /home/arun/.bashrc
=> Close and reopen your terminal to start using nvm
```

Now, in order to gain access to the NVM functionalities and binaries, we'll need to make sure to source ~/.profile file as the installer has appended the required settings in it.

$ source ~/.profile

In order to apply the changes, we'll need to make sure to logout and login to the session.

Once done, we'll move ahead towards the installation of the latest nodejs in our machine using NVM. Here, we can see all the available versions of nodejs that we want to install. To do so, we'll need to execute the following command.

$ nvm ls-remote

```
        v5.6.0
        v5.7.0
        v5.7.1
        v5.8.0
        v5.9.0
        v5.9.1
       v5.10.0
       v5.10.1
       v5.11.0
       v5.11.1
        v6.0.0
        v6.1.0
        v6.2.0
        v6.2.1
arun@localhost:~$
```

Then, we'll need to run the following command to install it.

$ nvm install v6.2.1

```
arun@localhost:~$ nvm install v6.2.1
Downloading https://nodejs.org/dist/v6.2.1/node-v6.2.1-linux-x64.tar.xz...
######################################################################## 100.0%
manpath: can't set the locale; make sure $LC_* and $LANG are correct
Now using node v6.2.1 (npm v3.9.3)
Creating default alias: default -> v6.2.1
arun@localhost:~$
```

Once its installed, we can switch the version of nodejs by simply running the following command.

$ nvm use v6.2.1

Testing NodeJS Installation

As we have completed the installation of NodeJS using the above steps, we should be now able to even check the version of nodejs by running the following command.

$ node -v

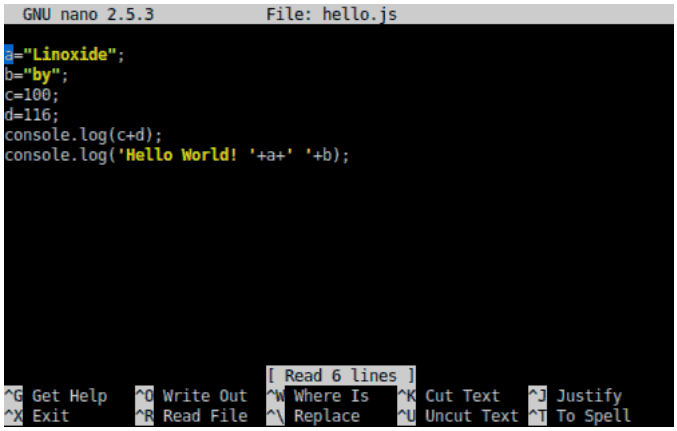
v6.2.1

we'll gonna create a simple nodejs app printing our all time favorite "Hello World" statement. We'll create a file named hello.js using a text r.

$ nano hello.js

, we'll write the following javascript code to the hello.js file.

```
a="Linoxide";
b="by";
c=100;
d=116;
console.log(c+d);
console.log('Hello World! '+a+' '+b);
```



Once done, we'll gonna save and execute the javascript file using nodejs.

$ node hello.js

```
arun@localhost:~$ nano hello.js
arun@localhost:~$ node hello.js
216
Hello World! Linoxide by
arun@localhost:~$
```

Conclusion

Finally we have successfully installed the latest and stable available NodeJS in our machine running Ubuntu 16.04 LTS Xenial. This tutorial should work fine in almost all the derivatives of Ubuntu and Debian. NodeJs has completely changed the way we used to run Javascripts. Nodejs has made Javascript available to run out of the web browsers ranging from servers to home desktop applications. After the above installation of

NodeJs is completed, we can now run our various nodejs applications. If you have any questions, comments, feedback please do write on the comment box below and let us know what stuffs needs to be added or improved.

might also like

- 3 Options To Install Node.js on Ubuntu 14.04
- How to Setup Node.JS on Ubuntu 15.04 with Different Methods
- How to Setup Node JS v4.0.0 on Ubuntu 14.04 / 15.04
- How to Install Ghost on Ubuntu Server 14.04 LTS (Trusty)
- How to Install Grunt on Ubuntu 16.04
- How to Setup Email Server with Mailinabox on Ubuntu 14.04
- How To Install The Latest Google Chrome On Ubuntu 12.04 LTS
- Install Ubuntu 14.04 LTS (Trusty Tahr) – Welcome New Features
- How to Configure Apache Solr on Ubuntu 14 / 15
- How to Install SQLite 3.9.1 with JSON Support on Ubuntu 15.04

Filed Under : UBUNTU HOWTO

PREVIOUS ARTICLE
How to Run Single-Node Cassandra Cluster in Ubuntu 16.04

NEXT ARTICLE
How To Install and Setup RabbitMQ on Ubuntu 16.04

Comments (1)

Trackback URL | Comments RSS Feed