

The Nodejitsu Handbook

A gentle introduction to the art of Nodejitsu

Table of Contents

- [Deploying Applications](#)
- [Using the Jitsu Client](#)
 - ◆ [Installation](#)
 - ◆ [Usage](#)
- [Nodejitsu's Open-source Projects](#)
 - ◆ [Why open-source](#)
 - ◆ [Where to find them](#)
 - ◆ [How to contribute](#)
- [Additional Information](#)
 - ◆ [Installing Node.js](#)
 - ◆ [Installing NPM](#)
 - ◆ [New to Node?](#)
 - ◆ [Creating a package.json](#)
 - ◆ [Using Databases](#)
- [Using the JSON API](#)
 - ◆ [Applications](#)
 - ◆ [Snapshots](#)
 - ◆ [Users](#)
 - ◆ [Databases](#)
 - ◆ [Logging](#)
 - ◆ [Marketplace](#)

NAME

introduction

Introduction

The Nodejitsu handbook will help you to familiarize yourself with deploying your Node.js applications to the cloud using Nodejitsu's services. It also provides detailed information about the Nodejitsu platform's advanced features and information on getting support when you need it.

This is a living document which you can submit patches to at <http://github.com/nodejitsu/handbook>.

Who Is Nodejitsu?

We are a collective of seasoned developers who have been devoted to the Node.js project since 2009. We are community leaders who have created and contributed to hundreds of open-source Node.js projects. If you have used Node.js, you've probably used code we've helped create. You can find our open source code at <http://github.com/nodejitsu>.

What Is Nodejitsu?

[Nodejitsu](#) is a Platform as a Service and a Marketplace for Node.js applications. Nodejitsu allows you to seamlessly deploy your Node.js applications into the cloud with a myriad of additional features. Our platform provides a robust suite of functionality to assist in the development, management, and deployment of Node.js applications. Our deployment tools are the most user-friendly in the industry and our customer support is unparalleled.

Getting Started

So you wish to learn the ways of Nodejitsu? Excellent! Reading this sentence is the first step! Below, you will find the Table of Contents which provides an overview of the systems which comprise Nodejitsu. We suggest starting at [Deploying Applications](#). You can also always visit our website at <http://nodejitsu.com>.

You only need to know 3 things to get started:

- We're Nodejitsu, and we can give you scalable, fault-tolerant cloud hosting for your Node.js apps - and we're the best you'll find.
- Our entire stack is [open source](#) and you can use our tools anywhere else you'd like to.
- Getting started with [your first app](#) is [simple](#) with our Jitsu command-line interface.

Hello World: A Tutorial

In this tutorial, you will write a simple "hello world" web application in Node.js, and then deploy it using `jitsu`, Nodejitsu's command line interface.

Before you get started, you should have both [Node.js](#) and the [Node Package Manager](#) (npm) installed.

Write A Server:

Let's start with a very basic node.js http server. Create a folder called `myapp/` and then create a file inside the folder called `server.js` inside of it with the following code:

```
// requires node's http module
var http = require('http');

// creates a new httpServer instance
http.createServer(function (req, res) {
  // this is the callback, or request handler for the httpServer

  // respond to the browser, write some headers so the
  // browser knows what type of content we are sending
  res.writeHead(200, {'Content-Type': 'text/html'});

  // write some content to the browser that your user will see
  res.write('<h1>hello, i know nodejitsu.</h1>')

  // close the response
  res.end();
}).listen(80); // the server will listen on port 80
```

That's all the code you'll need for starters.

Deploy with `jitsu`:

There are three basic ways to deploy applications to Nodejitsu:

!-Make sure that all these links point to the proper URLs-- * `jitsu`, The Nodejitsu command line tool * The Nodejitsu [Web Application](#), An easy to use web interface for managing your applications * The [JSON API](#)

`jitsu` is a [Command Line Interface \(CLI\)](#) for interacting with the Nodejitsu platform. It's open-source and easy to use. We've designed Jitsu to be suitable for command line beginners, but still be powerful and extensible enough for production usage. If you aren't a fan of the command line or don't have terminal access you can still do everything `jitsu` can do through the [Nodejitsu Web Application](#).

In this tutorial, we use `jitsu` to deploy our "hello world" application. If this is your first time deploying an application and you are eager to get started, we recommend using `jitsu`; it has a one line installer, it's self-documenting, and with it you'll be able to deploy your app in seconds.

Installation

In order to install `jitsu`, open a terminal and type:

```
[sudo] npm install -g jitsu
```

```

Marak-Squiere--MacBook-Pro:hellonode maraksquires$ jitsu
info: Welcome to Nodejitsu
info: It worked if it ends with Nodejitsu ok
info: Executing command help
help:
help:
help:  _/ / / / /
help:  _/ / / / /
help:
help: Flawless deployment of Node.js apps to the cloud
help: open-source and fully customizable.
help: https://github.com/nodejitsu/jitsu
help:
help: Usage:
help:
help:   jitsu <resource> <action> <param1> <param2> ...
help:
help: Common Commands:
help:
help:   Deploys current path to Nodejitsu
help:
help:     jitsu deploy
help:
help:   Creates a new application on Nodejitsu
help:
help:     jitsu create
help:
help:   Lists all applications for the current user
help:
help:     jitsu list
help:
help: Additional Commands
help:
help:     jitsu apps
help:     jitsu snapshots
help:     jitsu users
help:     jitsu conf
help:     jitsu logout
help:
help:
info: Nodejitsu ok
Marak-Squiere--MacBook-Pro:hellonode maraksquires$

```

!-- Update this image!--

This command will install `jitsu` on your system; the `-g` makes npm install it globally.

After installation, run the `jitsu` command from your command line. Since it's your first time using `jitsu`, you will be prompted to login with an existing account or to create a new account.

```

Marak-Squiere--MacBook-Pro:hellonode maraksquires$ jitsu
info: Welcome to Nodejitsu
info: It worked if it ends with Nodejitsu ok
info: No user has been setup on this machine
prompt: username: marak
prompt: password:
info: Authenticated as marak
info: Successfully configured user marak
info: Nodejitsu ok
Marak-Squiere--MacBook-Pro:hellonode maraksquires$

```

!-- Update this image?--

Once you've logged in, you can deploy your app immediately.

One Line Deployment

Open a terminal:

```
cd /path/to/myapp  
jitsu deploy
```

!-- Expand! Run through the details of below, assuming it's the first time-- This will create a new application snapshot, generate and or update project metadata, and deploy the project in the current path to [Nodejitsu](#). If it's your first deployment, you'll be prompted for some information such as *subdomain* and *start script* but it's really easy and we promise it will only take a few seconds.

Now just open up your favorite browser, and go to `yoursubdomain.nodejitsu.com`. If everything has been set up correctly, then you, too, are on the path of nodejitsu!

Open Source Projects

Why Open Source

A lot of Nodejitsu's technology stack is released as open source software. We choose to do this for many reasons. Aside from being able to give back to the Node.js community, releasing pieces of our stack as open-source allows other developers to review and improve our software. We've already received invaluable contributions to our platform from developers who would have never seen our code if we had not open-sourced it.

Where To Find Our Projects

Nodejitsu hosts its open-source projects on [Github](https://github.com) at <http://github.com/nodejitsu>. Github is a web site for sharing and collaborating on source code using [git](https://git-scm.com/), a popular version control system. You can get our source code without creating an account at github, and if you want to create an account it's free. You will need a [git client](https://git-scm.com/) if you wish to clone any of our code repositories.

How To Contribute

Anyone can contribute to any of our Nodejitsu open-source projects at any time. Our [github site](https://github.com/nodejitsu) has the facilities for managing patches, issues, code comments, version control, and just about anything else an open source developer could need. If you have any questions about a project, simply sign up and create a Github issue. We'll make sure one of our ninjas gets back to you with ninja speed!

Features of the Nodejitsu Platform

Deployment

!--stub out!--

Snapshots

!--stub out!--

Databases

Applications on Nodejitsu are ready to be connected to any database. If you have already have a database running, Nodejitsu can connect to your pre-existing database. If you require a new database, Nodejitsu can provide you FREE instances of several different types of databases. These free instances are great for development purposes or hobby sites. If you require a high traffic or production database we provide an easy upgrade path to industrial strength database hosting.

Creating new Databases

If you require database hosting you can create a new database instance of any of our supported databases using [Samurai](#), [Jitsu](#), or our [API](#).

Existing Databases

If you already have an externally hosted Database, Nodejitsu is capable of connecting to it. We've got Database hosting if you need it, but we fully support externally hosted Databases. Feel free to drop us an email if you have any questions.

Connecting Applications to Databases

If you want to connect a Database to your Node.js application, Nodejitsu provides you with sample code for each Database type as well as the ability to specify database connection strings in your application's package.json

Marketplace

!--stub out!--

The Jitsu Client

`jitsu` is a [Command Line Interface \(CLI\)](#) for interacting with the Nodejitsu platform. It's open-source and easy to use. We've designed Jitsu to be suitable for command line beginners, but still be powerful and extensible enough for production usage.

Installation

`jitsu` is distributed using the Node Package Manager (npm). Installing `jitsu` with npm is a snap:

```
[sudo] npm install -g jitsu
```

This command installs `jitsu` on the system globally.

Usage

Commands for `jitsu` follow this pattern:

```
jitsu <resource> <action> <param1> <param2> ...
```

For example, in `jitsu apps deploy`, "apps" is the resource and "deploy" is the action.

!- alphabetize these? Also, is ### appropriate? --

jitsu deploy(jitsu apps deploy)

`jitsu deploy` will attempt to deploy the application in the current directory to [Nodejitsu](#). It deploys your application using the following steps:

1. Creates the application (if necessary)
2. Creates or validates the package.json
3. Packages and creates a new snapshot
4. Stops the application (if necessary)
5. Starts the application

jitsu create(jitsu apps create)

`jitsu create` will create a new application. This entails generating a package.json for your app, for the purposes of deployment.

jitsu list(jitsu apps list)

`jitsu list` lists your applications, as well as their respective states, subdomains, entry points and latest snapshots.

!- Screenshot --

jitsu apps <action>

In addition to the commands aliased to `jitsu create`, `jitsu deploy` and `jitsu list`, the `apps` resource contains a number of other actions.

jitsu apps view <name>:

Lists the information for the application in the current directory. If `<name>` is supplied, then that application's information is listed instead.

jitsu apps update <name>:

Updates the application in the current directory with the information in the `package.json` file. If `<name>` is supplied, the application with `<name>` is updated instead.

jitsu apps destroy <name>

Destroys the application in the current directory. If <name> is supplied then that application is destroyed instead.

jitsu apps start <name>

Starts the application in the current directory. If <name> is supplied then that application is started instead.

jitsu apps restart <name>

Restarts the application in the current directory. If <name> is supplied then that application is restarted instead.

jitsu apps stop <name>

Stops the application in the current directory. If <name> is supplied then that application is stopped instead.

jitsu config <action>

jitsu config commands allow you to edit your local jitsu configuration file.

jitsu config list

Lists all configuration values currently set for the user.

jitsu config set <key> <value>

Sets the specified <key>/<value> pair in the jitsu configuration.

jitsu config get <key> <value>

Gets the specified <key>/<value> pair in the jitsu configuration.

jitsu config delete <key>

Deletes the specified <key> in the jitsu configuration.

jitsu help <resource> <action>

All commands will yield friendly messages to you if you specify incorrect parameters. Additionally, jitsu help will return useful help messages about any given resource or resource/action pair. for instance:

```
josh@pidgey:~$ jitsu help apps deploy
info: Welcome to Nodejitsu
info: It worked if it ends with Nodejitsu ok
info: Executing command help apps deploy
help:
help:
help: Deploys an application using the following steps:
help:
help: 1. Creates the application (if necessary)
help: 2. Creates or validates the package.json
help: 3. Packages and creates a new snapshot
help: 4. Stops the application (if neccessary)
help: 5. Starts the application
help:
help: jitsu deploy
help: jitsu apps deploy
help:
info: Nodejitsu ok
josh@pidgey:~$
```

If no resource and/or action are specified, then jitsu help alone will describe what resources are available.

jitsu snapshots

`jitsu snapshots` * commands allow you to work with snapshots for your Applications on Nodejitsu. Snapshots are images of your Application's code that are deployed to the Nodejitsu Platform.

For commands that take a `<name>` parameter, if no parameter is supplied, `jitsu` will attempt to read the `package.json` from the current directory.

!--stub out!--

`jitsu snapshots create`

`jitsu snapshots list`

`jitsu snapshots list <app-name>`

`jitsu snapshots activate`

`jitsu snapshots activate <app-name>`

`jitsu snapshots destroy`

`jitsu snapshots destroy <app-name>`

jitsu users

`jitsu users` * commands allow you to work with new or existing Nodejitsu user accounts. You will be prompted for additional user information as required.

!--stub out!--

`jitsu users create <username>`

`jitsu users available <username>`

`jitsu users confirm <username>`

`jitsu users logout (jitsu logout)`

.jitsuconf file

All configuration data for your local `jitsu` install is located in the `.jitsuconf` file located in your home directory. Directly modifying this file is not advised. You should be able to make all configuration changes using `jitsu config`.

Web Admin

!--STUB OUT!--

JSON API

Nodejitsu provides a web API for developers who want to interact with the Nodejitsu platform programatically. This API is built to be [RESTful](#) and communicates via [JSON](#). The API is the most low-level way of interacting with the Nodejitsu platform. For most deployment scenarios you should use our command line tool, [jitsu](#), or login directly at <http://nodejitsu.com>

- [Applications](#)
- [Snapshots](#)
- [Users](#)
- [Databases](#)
- [Logging](#)
- [Marketplace](#)

The documentation here should be an accurate representation of our current API, but you can always look directly at our [API wrappers](#) in `jitsu` to see a working example of an application built against Nodejitsu's REST API.

Authentication

Most of the calls to the API will require that you authenticate using your Nodejitsu account. If you do not have an account it is possible to create one using the User API, the Jitsu CLI, or just by visiting <http://nodejitsu.com>. Currently, we support [Basic Authentication](#). If you haven't used Basic Auth before don't fret, it's easy!

Here is an example using the command line utility, [Curl](#).

```
// get all applications for User "Marak"
curl --user Marak:password http://nodejitsu.com/apps/marak
```

If you are trying to use our API directly and are having issues with Basic Auth, please feel free to email support@nodejitsu.com

Applications

Applications are the core of the Nodejitsu API. Each application represents a set of Node.js code plus a package.json which contains meta-data about the application such as it's dependencies, database connections, configuration settings, authors, etc. For more information about the package.json format see: [package.json](#)

Get all Applications for a User

```
GET /apps/:user-id
```

Create a new Application

```
POST /apps/:user-id
{ package.json }
```

Start an Application

```
POST /apps/:user-id/:app-id/start
```

Stop an Application

```
POST /apps/:user-id/:app-id/stop
```

Restart an Application

```
POST /apps/:user-id/:app-id/restart
```

Update an Application

```
PUT /apps/:user-id
{ package.json }
```

Delete an Application

```
DELETE /apps/:user-id/:app-id/remove
```

Snapshots

Snapshots are an easy way to capture the current state of your application. Once a Snapshot of your application is created you can roll back and activate that Snapshot at any time.

Make an existing snapshot the active app

```
PUT /apps/:user-id/:app-id/snapshots/:id/active
```

Activate / Deploy a snapshot

```
POST /apps/:user-id/:snapshots/:id
```

Show a catalog of all Snapshot for an Application

```
GET /apps/:user-id/:app-id/snapshots
```

Show the contents of a Snapshot

```
GET /apps/:user-id/:app-id/snapshots/:id
```

Users

Create a new User / Sign-up for a free Nodejitsu account

Email address is the only required field.

```
POST /users/:user-id
{
  email: "youremail@theinternet.com"
}
```

Confirm a User account

All User accounts must be confirmed. When a new User is created, a confirmation email will be sent to the email address associated with that user. In this email there will be an invite code. This code must be sent to the API to confirm the account.

```
POST /users/:user-id
{
  inviteCode: "SecretCode"
}
```

Update User

```
PUT /users/:user-id
{
  password: "new_password"
}
```

Databases

Databases are an integral part of most applications. The Nodejitsu API allows you to dynamically create new hosted database instances for your applications. Cloud database hosting is provided by: CouchOne, Redis2Go and MongoHQ.

Create a new Database

```
POST /databases/:user-id/:database-id

{
  type: "Couch || Redis || Mongo"
}
```

Get information about a Database

```
GET /databases/:user-id/:database-id
```

Delete a Database

```
DELETE /databases/:user-id/:database-id
```

Logging

Logging is a very important feature in any professional grade Node.js application. Nodejitsu provides integrated logging solutions for your applications. Your logs are always saved and ready to be retrieved. TODO: add better description on logging

Get all logs for a user

```
GET /logs/:user-id/
```

Get logs for a specific application

```
GET /logs/:user-id/:app-id
```

Marketplace

The Marketplace is an online store where you can browse ready to deploy Node.js Applications. The Marketplace is a great place to start if you are new to Node.js development or want to share your existing Node.js Application with the world.

Get all Marketplace Applications

```
GET /marketplace
```

Get a specific Marketplace Application

```
GET /databases/:user-id/:id
```

Haibu

!-basically a copy-paste job from the haibu docs- haibu is the open-source node.js project for spawning and managing several node.js applications on a single server. It's an integral part of Nodejitsu's production stack and is fully supported by a dedicated team of core node.js developers.

By installing haibu, a user creates a development environment for themselves that mirrors the functionality of Nodejitsu's cloud platform.

Troubleshooting

Frequently Asked Questions

!- What are these FAQs? --

Common Problems and Mistakes

!- What are these problems, anyway? --

Appendix

Installing Node.js

Building and Installing Node.js

Step 1 - Pick Your Platform

Node should install out of the box on Linux, Macintosh, and Solaris.

With some effort you should be able to get it running on other Unix platforms and Windows (either via Cygwin or MinGW).

Step 2 - Prerequisites

Node has several dependencies, but fortunately most of them are distributed along with it. If you are building from source you should only need 2 things.

- **python** - version 2.4 or higher. The build tools distributed with Node run on python.
- **libssl-dev** - If you plan to use SSL/TLS encryption in your networking, you'll need this. Libssl is the library used in the [openssl](#) tool. On Linux and Unix systems it can usually be installed with your favorite package manager. The lib comes pre-installed on OS X.

Step 3a - Installing on Unix (including BSD and Mac)

Building from source

Use make to build and install Node (execute the following on the command line)

```
git clone https://github.com/joyent/node.git
cd node
export JOBS=2 # optional, sets number of parallel commands.
mkdir ~/local
./configure --prefix=$HOME/local/node
make
make install
export PATH=$HOME/local/node/bin:$PATH
```

If you have any installation problems, look at [Troubleshooting Installation](#), try an [alternate installation method](#), or stop into [#node.js](#) and ask questions.

Pre-built binaries

You can also install node from packages: [\[\[Installing Node.js via package manager\]\]](#)

Step 3b - Building on Windows

Pre-built binaries

Self-contained binaries are available at node-js.prcn.co.cc

Building from source

There are two ways of building Node on Windows. One is over the Cygwin emulation layer the other is using MinGW (GNU toolchain for windows). See the [Cygwin](#) and [MinGW](#) pages.

Neither builds are satisfactorily stable but it is possible to get something running.

Step 4 - Install NPM

NPM is a package manager that has become the de-facto standard for installing additional node libraries and programs. Here's the quick and easy one-liner for installing on Unix.

```
$ curl http://npmjs.org/install.sh | sh
```

To install a library e.g. [jitsu](#) (The Nodejitsu deployment CLI tool)

```
$ npm install jitsu
```

The number of npm modules available is vast - try a [search!](#)

And visit <https://github.com/isaacs/npm> for details.

New to Node.js?

Don't be scared! There are plenty of resources out there for beginners. Here are just a few:

- [The nodejs.org Official Docs](#)
- The #Node.js, #nodejitsu and #nodesupport rooms on [irc.freenode.net](#)
- [@NodeKohai](#) on Twitter

Understanding the package.json format

A package.json file describes your application, its dependencies, and other various application metadata. For a detailed spec on creating a package.json you can check out Isaac's fine documentation [here](#).

Preparing a package.json for your application

Nodejitsu requires that you create a valid [package.json](#) for your application. The package.json will determine certain important pieces of information about your application which are required for deployment. Since sometimes it can get confusing when constructing your package.json file, we provide wizards in our CLI tool and on our website for creating one.

Here is an example of what your package.json might look like:

```
{
  "name": "hellonode",
  "subdomain": "hellonode",
  "scripts": {
    "start": "server.js"
  },
  "version": "0.0.0"
}
```

Notice the "scripts" property? This is where you'll store information about specific scripts in your application. The "start" property indicates the script that will get called when your application is started.

Specifying dependencies in your package.json

If your application requires additional dependencies or third-party libraries, Nodejitsu fully supports npm module dependency resolution. All you have to do is list your dependencies the exact same way you would if you were packaging a module for npm. Here is an example of the same package.json with a few dependencies.

```
{
  "name": "hellonode",
  "subdomain": "hellonode",
  "scripts": {
    "start": "server.js"
  },
  "dependencies": {
    "async": ">= 0.1.8",
    "colors": ">= 0.5.0",
    "request": ">= 1.9.0",
  }
}
```

```
    "vows": ">= 0.5.8",  
  },  
  "version": "0.0.0"  
}
```

Your dependencies will be resolved when your application deploys to Nodejitsu.