Node.jsをバージョン管理ツールを使ってインストール

Macで Node.jsのバージョン管理

Mac OSでNode.jsのバージョン管理をする場合は、nodebrew 03 の使用をおすすめする。nodebrewは日本のフロントエンドエンジニア、外村和仁氏が開発したNode.jsのバージョン管理ツールで、導入と管理方法の手軽さが特徴だ。また、Node.jsからフォークされたio.jsの管理も行うことができる。

Node.jsのアンインストール

nodebrewをインストールする前にまず、Node.jsがインストールされている場合はアンインストールしなければならない。アンインストールするには、Githubで公開されているNode.jsをアンインストールするためのシェルスクリプトを利用しよう。ターミナルを起動して 04 のコマンドを順に実行していくと、アンインストールは完了だ。試しにnode -vとコマンドを入力してみて、バージョンが表示されないか確認しておこう。

nodebrewのインストール

Node.jsがアンインストールできたらnodebrewの インストールを行う。

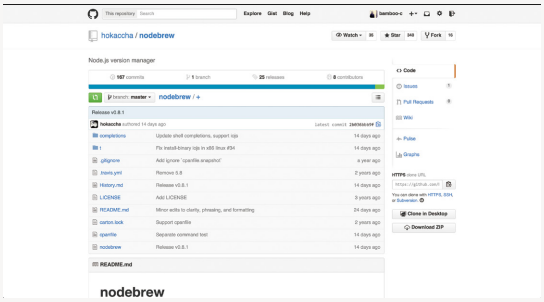
「`curl -L git.io/nodebrew | perl - setup`」というコマンドをコンソールに入力してインストールしよう。続いて、インストールしたデータにパスを通すために「`echo 'export PATH=$HOME/.nodebrew/current/bin:$PATH' >> ~/.bashrc`」と入力する。そして、設定ファイル（シェルがデフォルトのbashの場合はbashrc）を読み込み直すコマンド「`source ~/.bashrc`」を入力すれば、インストールは完了だ。ただし、.bashrcファイルが存在しない場合にはエラーが出力される。その場合には新しく生成する必要があるので、「`vi ~/.bashrc`」のコマンドを入力しよう。開かれた編集画面に、「`export PATH=$HOME/.nodebrew/current/bin:$PATH`」を入力して保存する。ここでも同じように「`source ~/.bashrc`」というコマンドで設定ファイルを読み直そう。ここまでできたら試しに「`nodebrew help`」というコマンドでヘルプ情報が表示されるかを確認してみよう。表示されていれば成功だ。

nodebrewを使ったNode.jsのインストール

続いてnodebrewを使って任意のバージョンのNode.jsをインストールしてみよう。nodebrew install-binaryのコマンドに、バージョンを追加すればOKだ。仮に0.10.29というバージョンのNode.jsをインストールしたい場合には、「`nodebrew install-binary v0.10.29`」というコマンドでインストールできる。Node.jsのインストールが終わったら、「`nodebrew ls`」というインストール済みのバージョンを一覧で確認できるコマンドで確認してみよう。v0.10.29と表示されていれば成功だ。

nodebrewで利用するバージョンを指定する

nodebrew lsでバージョンを確認した際に、一緒にcurrent: none と表示されたはずだ。これは現在利用中のバージョンを表示してくれるもので、noneはまだ指定が行われていない状況を指す。メインで利用するバージョンを指定する際はuseコマンドを利用する。先ほどインストールした0.10.29のバージョンをメインで利用したい場合には、「`nodebrew use v0.10.29`」とコマンドを入力すればいい。切り替える際にも、別のバージョンをuseコマンドで指定することで簡単に切り替えること



03 nodebrew

https://github.com/hokaccha/nodebrew

```
$ curl -o uninstall-node.sh https://gist.
githubusercontent.com/nicerobot/2697848/raw/uninstall-
node.sh
$ chmod u+x uninstall-node.sh
$ ./uninstall-node.sh
$ rm uninstall-node.sh
$ sudo rm -rf /usr/local/include/node
$ sudo rm -rf /usr/local/lib/dtrace
$ rm -rf ~/.node-gyp
$ rm -rf ~/.npm
$ rm -rf ~/.sourcemint
```

04 Node.jsをアンインストール

sudoのコマンドの初回実行時はマシンのパスワードの入力が求められる。

ができる。このバージョンの切り替えがバージョン管理ツールの一番の強みだ。

Windows で Node.js のバージョン管理

Windows で Node.js のバージョン管理を行うには、nodist 05 を利用するのが一般的だ。nodist をインストールする前に、すでに PC に Node.js がインストールされている場合はアンインストールする必要がある。通常のアプリケーションと同様にコントロールパネルから削除できる。

また、インストールには Git が必要になるので、必要に応じてインストールしておこう（P.17 参照）。この際、Git もコマンドプロンプトから呼び出せる必要があるので、Git をインストーラーでインストールする際に「Adjusting your PATH environment」の設定を「Run Git from the Windows Command Prompt」に変更しておくか、下記の nodist と同様の要領で手動でパスを通しておこう。

nodist のインストールはコマンドプロンプトを利用して以下の手順で行う。

①`git clone git://github.com/marcelklehr/nodist.git` コマンドでファイルのダウンロード
②コントロールパネルから環境変数でパスを通す
③`nodist update` コマンドで nodist のアップデートを行う

Git でダウンロードした nodist は、デフォルトでは「C:¥Users¥ユーザー名（自身のユーザー名のフォルダ）¥nodist」に配置されている。ここではこの状態を前提に説明を進める。

まず、コマンドプロンプトから nodist を利用できるように、パスを通す作業が必要だ。コントロールパネルの「システム＞システムの詳細設定＞環境変数」を開き、「システム環境変数」エリアの［新規］ボタンをクリックする。表示されたダイアログで［変数名：`NODIST_PREFIX`］、［変数値：`C:¥Users¥ユーザー名¥nodist`］に設定、適用して環境変数を追加する。次に「システム環境変数」の一覧にある「Path」をクリックして選択し、［編集］をクリック。変数値の先頭に「`C:¥Users¥ユーザー名¥nodist¥bin;`」を追記しよう。

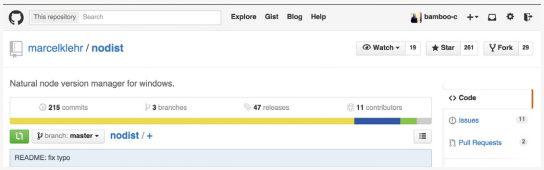
これらの設定は Node.js や OS のバージョンによって変わる場合があるため、動作しない場合は nodist の readme.md を確認してほしい。

ここまで完了したら、「`nodist update`」コマンドで最新の状態にアップデートしておく。

終わったら、「`nodist -v`」コマンドでバージョンが表示されるかを確認しよう。きちんと表示されていればインストールは完了だ。

nodist で Node.js のインストール

nodist でバージョンを指定してインストールを行うには、「`nodist 0.10.29`」のようにバージョン名をつけて実行する。これでインストールと使用バージョンの指定が同時に行える。すでにインストールされているバージョン名をつけた場合は、使用バージョンの指定のみになる。「`nodist`」コマンドでインストールしたバージョンを一覧表示できるので、確認してみよう。

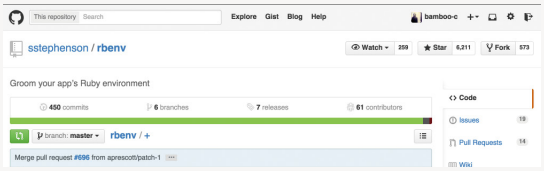


05 nodist

https://github.com/marcelklehr/nodist

```
$ echo 'export PATH="$HOME/.rbenv/bin:$PATH"' >>
~/.bash_profile
$ echo 'if which rbenv > /dev/null; then eval "$(rbenv
init -)"; fi' >> ~/.bash_profile
```

07 Mac で rbenv のインストール 1



06 rbenv

https://github.com/sstephenson/rbenv

```
$ brew install ruby-build
$ brew install rbenv-gemset
$ brew install rbenv-gem-rehash
```

08 Mac で rbenv のインストール 2

い）を回避するため、サーバに置きたい場合もある。

そんなときは簡易サーバを利用しよう。複雑な機能はないが、前述したようなローカルファイルで起こる問題は回避できる 24。

- grunt-contrib-connect
  https://www.npmjs.com/package/grunt-contrib-connect
- gulp-webserver
  https://www.npmjs.com/package/gulp-webserver

使い方

grunt connectないしgulp webserverで起動する。Webブラウザから"http://localhost:8000/"を開くと表示される。終了は［control］（［Ctrl］）＋［C］を押す。

起動時「Fatal error: Port 8000 is already in use by another process.」あるいは「Error: listen EADDRINUSE」というエラーが出た場合、設定ファイルのportの数字を他のものに変更する。8000番代を用いる場合が多い。

サーバのルートディレクトリは、Gruntはbaseオプションで、gulpはgulp.src()の引数で指定する。

オプションでkeepaliveを指定しないと一瞬で終了してしまうが、watchのように継続するタスクと一緒に実行する場合は不要だ 25。

ファイル削除

Gruntの場合、ファイル削除を行う次のプラグインをインストールする。

- grunt-contrib-clean
  https://www.npmjs.com/package/grunt-contrib-clean

オプションはなく、単純に対象ファイル、ディレクトリを指定する。削除対象は細かく分けておき、grunt clean:publicのように個別に実行すると小回りが利く。

gulpの場合は、プラグインではなく通常のnpmパッケージを使う。

- del
  https://www.npmjs.com/package/del

ファイル複製

出力ファイルを1つのディレクトリにまとめるのは、後々まとめて本番サーバへ転送する際の手間を考えるとよい手だ。コンパイルなど何の処理もいらない、編集対象のファイルをそのまま用いる場合は、Gruntのプラグインなどを利用して複製する。gulpはストリームがあるので、単純にgulpの基本機能のみを使用して入力と出力を行える。

- grunt-contrib-copy
  https://www.npmjs.com/package/grunt-contrib-copy

ファイル結合

ファイルを取得すると通信コストがかかる。ファイルを結合して通信の数を減らすことで、読み込みを速めることができる。通信パフォーマンスについては、P.152も参照してほしい。

- grunt-contrib-concat
  https://www.npmjs.com/package/grunt-contrib-concat
- gulp-concat
  https://www.npmjs.com/package/gulp-concat

オプション	説明
base	公開するディレクトリ（Gruntのみ）
keepalive	すぐ終了しない。watchなどと組み合わせるなら不要（Gruntのみ）
port	ポート

24 簡易サーバの主なオプション

```
// デフォルトタスクを設定
grunt.registerTask('default', ['connect', 'watch']);
```

25 watchと一緒に実行する