
NodeBB Documentation

Release 0.4.1

The NodeBB Team (<https://github.com/NodeBB/NodeBB/graphs/contributors>)

thg 9 27, 2017

Contents

1 Cài đặt NodeBB	3
1.1 Cài đặt NodeBB theo HĐH	3
1.2 Cài đặt NodeBB trên dịch vụ điện toán đám mây (Cloud)	10
2 Cấu hình NodeBB	17
2.1 Configuring Databases	17
2.2 Configuring Web Server / Proxies	20
3 Chạy NodeBB	25
3.1 Running NodeBB	25
4 Cập nhật NodeBB	27
4.1 Upgrading NodeBB	27
5 Quản trị NodeBB	31
5.1 Administrative Functions	31
5.2 Social Network SSOs	33
5.3 Image Hosting APIs	34
6 Đóng góp cho NodeBB	35
6.1 NodeBB Style Guide	35
6.2 Core Modules	36
7 Hệ thống plugin	39
7.1 Writing Plugins for NodeBB	39
7.2 Available Hooks	42
7.3 Settings Framework	47
8 Hệ thống widgets	55
8.1 Writing Widgets for NodeBB	55
9 Nhân tạo giao diện	57
9.1 Creating a new NodeBB Theme	57
9.2 Rendering Engine	58
10 Tài nguyên cho nhà phát triển	63
10.1 Developer's Resources	63

11 Tr giúp cho d án NodeBB	65
11.1 Helping out the NodeBB Project	65
12 Khắc phục / Tr giúp	67
12.1 Need Help?	67
13 Ch mục và bng	71

NodeBB là mt nn tng din đần th h mi s dng công ngh web sockets đ có th tng tác ngay lp tc và thông báo thì gian thc. NodeBB có nhieu tính năng hin đi vì t duy sáng to nh tng tác mng xã hi và dòng chy tho lun.

NodeBB là mt d án mã ngun m mà có th to nhánh trên [GitHub](#). Đng b qua các cách mà bn có th giúp đ đóng góp cho d án này t dch thut, yêu cu “pull”, và chia s vì bn bè ca bn.

Cài đặt NodeBB theo HĐH

Đây là hướng dẫn tổng quát để giúp bạn cài đặt và chạy.

Ghi chú: Nếu bạn điều hành các máy không có trong danh sách, hãy đưa ra yêu cầu trên [cộng đồng](#) của chúng tôi hoặc còn thì, hãy đề trình lên.

Arch Linux

Đầu tiên, chúng ta cài đặt gói phần mềm cần bạn thực. Hãy thực hiện `pacman -S` trước tiên để chắc chắn rằng bạn đã đăng ký vị trí và tất cả các gói cần thiết.

```
$ sudo pacman -S git nodejs redis imagemagick
```

Nếu bạn muốn sử dụng MongoDB, LevelDB, hoặc một nền tảng cơ sở dữ liệu khác ngoài Redis, hãy đọc tài liệu [Cấu hình cơ sở dữ liệu](#).

Tiếp theo, tạo bản sao của kho sau:

```
$ git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git nodebb
```

Cài đặt các trình phụ thuộc theo yêu cầu của NodeBB:

```
$ cd nodebb
$ npm install
```

Bắt đầu đơn giản cài đặt bằng cách chạy ứng dụng ví dụ như `setup`:

```
$ ./nodebb setup
```

Mục đích cài đặt là cho máy chủ của bạn và chạy trên cùng một máy/công.

Cui cùng, ta chy din đần.

```
$ ./nodebb start
```

NodeBB cũng có th khi đng vi các trình tr giúp, nh `supervisor` và `forever`. *Đc thêm v các tùy chn đây.*

Ubuntu

Đu tiên, chúng ta cài đt các trình nn:

```
$ sudo apt-get install git nodejs redis-server imagemagick npm
```

Nu bn mun s dng MongoDB, LevelDB, hoc mt c s d liu khác thay cho Redis xin đc *Cu hình c s d liu*.

Nu trình qun lý gói ca bn ch cài đt phiên bn Node.js nh hn 0.8 (vd: Ubuntu 12.10, 13.04), dùng lnh “node –version” đ xác đnh phiên bn Node.js ca bn

```
$ sudo add-apt-repository ppa:chris-lea/node.js
$ sudo apt-get update && sudo apt-get dist-upgrade
```

Nu bn mun cài đt Node.js v0.11, s dng url repo sau `ppa:chris-lea/node.js-devel`.

Sau đó, to bn sao repo:

```
$ git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git nodebb
```

Cài đt các trình ph thuc bi NodeBB:

```
$ cd nodebb
$ npm install
```

The hin cài đt bng cách chy ng dng vi th `setup`:

```
$ ./nodebb setup
```

Mc đnh cài đt là cho máy ch ni đa và chy trên cng mc đnh, vi redis lu tr trên cùng mt máy/cng.

Cui cùng, ta chy din đần.

```
$ ./nodebb start
```

NodeBB cũng có th khi đng vi các trình tr giúp, nh `supervisor` và `forever`. *Đc thêm v các tùy chn đây.*

Debian

Hng dn trên Ubuntu hin ti không hoàn toàn tng thích vi Debian và có mt vài nét đc trng và đc bit là cách cài đt Node.js, và cách ly đc bn mi nht ca Redis.

Yêu cu

NodeBB cn có các phn mm sau đc cài đt:

- Node.js ít nht là bn 0.10 hoc cao hn
- Redis, bn 2.6 hoc cao hn
- cURL, ch cn the hin lnh `sudo apt-get install curl` đ cài đt

Cài đặt Node.js

Debian 7 và 6 học cũ hơn mặc định không có gói cài đặt *nodejs*, nhưng có một vài giải pháp để cài đặt Node.js trên Debian cả hai.

Wheezy Backport :

Giải pháp này **CHỈ cho Debian 7**, đơn giản là chỉ đơn giản đi đây vì **quyền root** :

```
$ echo "deb http://ftp.us.debian.org/debian wheezy-backports main" >> /etc/apt/
↪sources.list
$ apt-get update
```

Để cài đặt Node.js + NPM, chỉ :

```
$ apt-get install nodejs-legacy
$ curl --insecure https://www.npmjs.org/install.sh | bash
```

Nó sẽ cài đặt một phiên bản Node.js hiện hành 0.8 (vào 29 tháng 3 năm 2014 : 0.10.21)

Tập hợp tự nguyện :

Giải pháp này cho Debian 6 (Squeeze) và hơn, để cài đặt NodeJS, chỉ đơn giản sau vì **quyền root** :

```
$ sudo apt-get install python g++ make checkinstall
$ src=$(mktemp -d) && cd $src
$ wget -N http://nodejs.org/dist/node-latest.tar.gz
$ tar xzvf node-latest.tar.gz && cd node-v*
$ ./configure
$ fakeroot checkinstall -y --install=no --pkgversion $(echo $(pwd) | sed -n -re's/./
↪+node-v(.+)\$/\1/p') make -j$(($(nproc)+1)) install
$ sudo dpkg -i node_*
```

Liệt kê phiên bản miễn phí qua DotDeb

Dotdeb là một kho chứa các gói để bin các box Debian cả hai thành minh m, n định và LAMP để copy nh.

- Nginx,
- PHP 5.4 và 5.3 (Nhưng trình m rợng PHP hữu ích : APC, imagick, Pinba, xcache, Xdebug, XHpro..)
- MySQL 5.5,
- Percona toolkit,
- Redis,
- Zabbix,
- Passenger...

Dotdeb hỗ trợ :

- Debian 6.0 “Squeeze“ và 7 “Wheezy“
- c 2 kiến trúc amd64 và i386

Debian 7 (Wheezy) :

Đã có mã kho DotDeb đây đây :

```
$ sudo echo 'deb http://packages.dotdeb.org wheezy all' >> /etc/apt/sources.list
$ sudo echo 'deb-src http://packages.dotdeb.org wheezy all' >> /etc/apt/sources.list
```

Sau đó, thêm khóa GPG sau :

```
$ wget http://www.dotdeb.org/dotdeb.gpg
$ sudo apt-key add dotdeb.gpg
```

Và cập nhật nguồn:

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

Debian 6 (Squeeze)

Đã có mã kho DotDeb đây đây :

```
$ sudo echo 'deb http://packages.dotdeb.org squeeze all' >> /etc/apt/sources.list
$ sudo echo 'deb-src http://packages.dotdeb.org squeeze all' >> /etc/apt/sources.list
```

Sau đó, thêm khóa GPG sau :

```
$ wget http://www.dotdeb.org/dotdeb.gpg
$ sudo apt-key add dotdeb.gpg
```

Và cập nhật nguồn:

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

Cài đặt NodeBB

Bây giờ, ta đã có NodeJS và Redis để cài đặt, chỉ cần sau đó cài đặt gói phần mềm này:

```
$ apt-get install redis-server imagemagick git
```

Tip theo dõi bạn sao cả kho sau :

```
$ cd /path/to/nodebb/install/location
$ git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git nodebb
```

Chúng ta sẽ tiến hành cài đặt các trình phụ thực cho NodeBB qua NPM :

```
$ cd /path/to/nodebb/install/location/nodebb (hoặc nếu bạn đang đăng nhập cài đặt run : cd_
→nodebb)
$ npm install
```

Cài đặt NodeBB bằng cách chỉ cần dùng ví dụ `–setup` :

```
$ ./nodebb setup
```

1. *URL of this installation* là địa chỉ IP hoặc tên miền trong IP đó.

Ví dụ: `http://0.0.0.0` hoc `http://example.org`

2. **Port number of your NodeBB là cổng để có thể truy cập vào trang:** **Chú ý:** Nếu bạn không tạo proxy cho cổng của bạn vì một giải pháp nào đó như Nginx thì nên sử dụng cổng 80.
3. Nếu bạn đã thực hiện các bước trên để cài đặt redis-server thì sử dụng giá trị cài đặt mặc định là redis.

Và sau cùng.. hãy bắt đầu chạy NodeBB

```
$ ./nodebb start
```

Chú ý: Nếu NodeBB hoặc máy chủ của bạn gặp sự cố, NodeBB sẽ không khởi động lại, đây là lý do tại sao bạn nên đi qua các cách khác để khởi động NodeBB vì các trình trợ giúp như `supervisor` và `forever`, chúng *đều* nên giữ gìn như 1 cú nhấp chuột!

Phục vụ, thiết lập và lời khuyên

Bạn nên bỏ một cài đặt NodeBB của bạn, [đọc thêm đây](#).

Bạn nên sử dụng Nginx (hoặc tương tự) để tạo proxy cho NodeBB và cổng 80, [đọc thêm đây](#)

SmartOS

Yêu cầu

NodeBB cần có những chương trình này để cài đặt:

- Node.js (phiên bản 0.10 hoặc hơn, hướng dẫn đi).
- Redis (phiên bản 2.6 hoặc hơn, hướng dẫn đi) hoặc MongoDB (phiên bản 2.6 hoặc hơn).
- nginx (phiên bản 1.3.13 hoặc hơn, **chỉ khi** bạn có ý định sử dụng nginx tạo proxy cho NodeBB).

Truy cập máy chủ

1. Đăng nhập vào tài khoản Joyent của bạn: Joyent.com
2. Chọn: Create Instance
3. To image smartos nodejs miễn phí.

Chú ý: Những bước này đã được thử nghiệm vì image: `smartos nodejs 13.1.0 smartos nodejs 13.2.3`

4. Đi phiên bản của bạn hiện Running sau đó click chọn tên của nó.
5. Tìm Login và mật khẩu quản trị. Nếu khu vực Credentials bị thiếu, hãy tìm lại trang.

Ví dụ: `ssh root@0.0.0.0 A#Ca{c1@3`

6. Truy cập SSH theo quyền admin không root: `ssh admin@0.0.0.0`

Chú ý: Đi vì những người dùng Windows chưa có cài đặt SSH thì đây là 1 tùy chọn: Cygwin.com

Cài đặt

1. Cài đặt trình thực thi cho NodeBB:

```
$ sudo pkgin update
$ sudo pkgin install scmgit nodejs build-essential ImageMagick redis
```

Nu mt trong s dòng trên tht bi thì chy lệnh di:

```
$ pkgin search *failed-name*
$ sudo pkgin install *available-name*
```

2. Nu cn cài đt redis-server vi mc đnh là service (t đng khi đng và khi đng li):

Nu bn mun s dng MongoDB, LevelDB, hoc 1 c s d liu khác ngoài Redis thì hãy đc *Cu hình c s d liu*.

Chú ý: Nhng bc này đ cài đt nhanh 1 server redis nhng không ti u đ dùng chính thc.

Chú ý: Nu bn đã chy redis-server ri thì hãy thoát khi nó ngay.

```
$ svcadm enable redis
$ svcs svc:/pkgsrc/redis:default
```

Chú ý: Nu STATE là maintenance thì:

```
$ scvadm clear redis
```

- Đ tt redis-server và ngán nó khi đng li:

```
$ scvadm disable redis
```

- Đ khi đng redis-server và cho nó luôn chy:

```
$ scvadm enable redis
```

3. Chuyñ đn ni mà bn mun to folder cài đt nodebb:

```
$ cd /parent/directory/of/nodebb/
```

4. To bn sao NodeBB's repository (bn có h đi nodebb cui thành tên folder khác mà bn mun):

```
$ git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git nodebb
```

5. Cài đt trình ph thuc npm ca NodeBB:

```
$ cd nodebb
$ npm install
```

6. Chy mã cài đt NodeBB:

```
$ ./nodebb setup
```

(a) URL used to access this NodeBB là đđ ch ip bn dùng đ đđng nhp SSH hoc tên min tr đđ IP này.

Ví d: http://0.0.0.0 hoc http://example.org

(b) Port number of your NodeBB là cng cn đ truy cp đđn trang ca bn:

Chú ý: Nu bn không s dng nginx hoc 1 server web nào khác đ to proxy thì nên đ cng 80.

(c) Please enter a NodeBB secret - Không gi email và công khai mã này.

(d) IP or Hostname to bind to - S dng giá tr mc đnh tr khi máy ch bn có yêu cu khác.

(e) Nu bn s dng các bc cài đt redis-server trên thì hãy s dng giá tr cài đt mc đnh cho redis.

7. **Khi đng tin trình NodeBB th công:** **Chú ý:** Điều này không nên s dng cho bn chính thc nhng thay vì s dng daemon thì nên s dng Forever hoc Supervisor. *Đc các tùy chnh đây.*

```
$ node app
```

8. **Truy cp ng dng ca bn! Ví d:** Vi cng là 4567: `http://0.0.0.0:4567` or `http://example.org:4567`

Chú ý: Vi cng là 80 thì :80 không cn phi nhp.

Chú ý: Nu hng dn trên không rõ ràng vì bn hoc nu bn có gp rc ri gì, hãy cho chúng tôi bit bng cách báo cáo li ti đây.

Cp nht NodeBB

Chú ý: Chi tit v vic cp nht NodeBB xin đc *Cp nht NodeBB*.

Windows 8

Phn mm yêu cu

Đu tiên, cài đt các chng trình đi đây:

- <https://windows.github.com/>
- <http://nodejs.org/>
- <http://sourceforge.net/projects/redis/files/redis-2.6.10/>
- <http://imagemagick.org/script/binary-releases.php#windows/>

Bn có th phi khi đng li máy tính ca mình.

Chy NodeBB

Khi đng Redis Server

Ghi chú: The default location of Redis Server is

C:\Program Files (x86)\Redis\StartRedisServer.cmd

M Git Shell, và đánh dòng lnh sau. To bn sao NodeBB repo:

```
git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git
```

Truy cp đng dn:

```
cd NodeBB
```

Cài đt trình ph thuc:

```
npm install
```

Chy trình tng tác cài đt:

```
node app.js
```

Bn có th đ mi tùy chn là mc đnh.

Và bn đã xong! Sau khi cài đt, chy

```
node app.js
```

Bn có th truy cp din đản ti `http://127.0.0.1:4567/`

Phát trin trên Windows

S có đôi chút khó chu khi phi tt và khi đng li NodeBB mi khi bn thay đi. Hãy cài đt supervisor:

```
npm install -g supervisor
```

M bash:

```
bash
```

Và chy NodeBB trong ch đ “watch”:

```
./nodebb watch
```

Nó s bt NodeBB trong ch đ nhà phát trin, và s theo dõi các file đc thay đi và t đng khi đng li din đản.

- *Arch Linux*
- *Ubuntu*
- *Debian*
- *SmartOS*
- *Windows*
- *CentOS (external)*

Cài đt NodeBB trên dch v đin toán đám mây (Cloud)

Di đây là hng dn tng bc đ bn cài đt và chy trên các gii pháp PaaS.

Ghi chú: Nu gii pháp PaaS ca bn không có trong danh sách, hãy hãy đ ra yêu cu trên [cng đng](#) ca chúng tôi học thm chí tt hn, hãy đ trình lên.

Cloud 9 IDE

Hng dn cài đt di đây dành cho IDE nn web [Cloud 9](#).

Bc 1: To bn sao NodeBB vào workspace ca bn t GitHub. Bn có th s đng lnh sau t terminal:

```
git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git nodebb
```

lnh nodebb sau git url s to 1 file tên là nodebb vì vy bn phi CD vào trong file đó sau khi bn to xong bn sao ca NodeBB.

Bc 2: Cài đt redis vi Cloud9's package manager

```
nada-nix install redis
```

Bc 3: Chy redis server ti cng 16379 - cng 6379 đã đc s dng bi Cloud 9. “&” đ dòng lnh thc hin nn. Bn có th hy tin trình bt c lúc nào sau này. \$IP là bin h thng ca Cloud 9 cha đã ch IP instance ca bn.

```
redis-server --port 16379 --bind $IP &
```

Bc 4: Xác đnh đã ch IP instance ca bn đ NodeBB có th chy chính xác. Đây là 1 trong nhng yêu cu ca Cloud 9 và có v nh là cách duy nht mi làm nó chy. Bn không th s dng \$IP trong file config.json (có nghĩa là bn không th nhp \$IP trong node app --setup).

```
echo $IP
```

Bc 5: Cài NodeBB và trình ph thuc:

```
npm install
```

Bc 6: Chy trình cài đt nodebb:

```
node app --setup
```

URL ca phn cài đt này nên là ‘http://workspace_name-c9-username.c9.io’, thay th workspace_name bng tên workspace ca bn và username vi username ca bn. Chú ý rng NodeBB vn đang s dng kt ni không bo mt http đ ti jQuery bn s th y đ s dng <http://> hn nhiu thay vì <https://> cho url nn ca bn. Nu không jQuery s không ti và NodeBB s không chy.

S cng không thc s quan trng - Cloud9 s bt bn s dng cng 80. Hãy đt nó là 80. Nu nó là cng khác, nh 4567, cũng vn s bình thng.

Use a port number to access NodeBB? Li ln na, nó cũng không quá khác bit. Hãy đt là “no”. Hoc khác cũng vn đc.

Host IP or address of your Redis instance: localhost (giá tr xut ra ca lnh \$IP cũng đc chp nhn)

IP or Hostname to bind to: Nh p giá tr mà \$IP cha Bc 4. Nó s có th ging nh sau: 123.4.567.8

Host port of your Redis instance: 16379

Redis Password: Tr khi bn cài đt nó th công, không thì Redis s đc cu hình không có mt khu. Hãy đ trng và nhn Enter

Ln cài đt đt tiên yêu cu cn có Admin name, email address và password đc đt.

Vy là xong! Đng s dng nút Run trên cùng ca IDE, nó thng có li đi vi tôi. Bn nên s dng command line thì tt hn. Chy lnh sau:

```
node app
```

Sau đó truy cp http://workspace_name-c9-username.c9.io ti trình duy.

Khc phc

Vn đ thng gp là dch v c s d liu cha đc khi đng. Hãy chc chn bn đã cài đt Redis 1 cách chính xác và chy:

```
redis-server --port 16379 --bind $IP
```

Heroku

Chú ý: Cài đt lên Heroku yêu cu máy local phi chy 1 phiên bn unix, NodeBB không chy trên Windows.

1. Ti và cài đt [Heroku Toolbelt](#) cho h điu hành ca bn
2. Đăng nhp vào tài khon Heroku: `heroku login`
3. Xác nhn tài khon Heroku bng cách thêm th credit (ti <http://heroku.com/verify>)
4. To bn sao repository: `git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git /path/to/repo/clone`
5. `cd /path/to/repo/clone`
6. Cài đt dependencies `npm install`
7. To heroku app: `heroku create`
8. Bt WebSocket (beta): `heroku labs:enable websockets -a {APP_NAME}, {APP_NAME}` đc cung cp bi Heroku, và ging nh sau `adjective-noun-wxyz.herokuapp.com` (Chú ý: [Đc tài liu sau](#)): `b.herokuapp.com` khi nhp `{APP_NAME}` phía trên.
9. Bt [Redis To Go](#) cho tài khon heroku ca bn: `heroku addons:add redistogo:nano`
10. Chy mã cài đt NodeBB: `node app --setup` (thông tin v Heroku server và Redis to Go instance có th tìm thy trang tài khon ca bn)
 - Tên server ca bn nm trong trang cài đt Heroku app và ging nh sau `adjective-noun-wxyz.herokuapp.com`
 - S dng bt c cng nào, nó s b b qua.
 - Redis server có th tìm trong redis url. Ví d vi url là: `redistogo://redistogo:h28h3wgh37fns7@crestfish.redistogo.com:12345/`
 - Server là `fishyfish.redistogo.com`
 - Cng là `12345`
 - Mt khu là `h28h3wgh37fns7`
12. Thêm 3 gói sau vào phn dependencies trong file `package.json`:

```
"dependencies": {
  ...
  "nodebb-plugin-dbsearch": "0.0.10",
  "redis": "~0.10.1",
  "connect-redis": "~2.0.0"
},
"devDependencies": {
```

13. To 1 Procfile cho Heroku: `echo "web: node loader.js" > Procfile`
14. Commit Procfile:

```
git add -f Procfile config.json package.json && git commit -am "adding Procfile and
↪ configs for Heroku"
```

15. Push lên heroku: `git push heroku master` * Hãy chc chn rng khóa SSH đúng đã đc thêm vào tài khon ca bn, nu không thì push s không thành công!
16. Khi to 1 dyno: `heroku ps:scale web=1`
17. Truy cp vào ng dng!

Nu hng dn trên không rõ ràng hoc bn gp rc ri, hãy cho chúng tôi bit bng cách [to báo cáo](#).

Luôn cp nht

Nu bn mun pull nhng thay đi mi nht t git repository vào ng dng Heroku ca bn:

1. Chuy n hng repository ti /path/to/nodebb
2. `git pull`
3. `npm install`
4. `node app --upgrade`
5. `git commit -am "upgrading to latest nodebb"`
6. `git push heroku master`

Nitrous.IO

Hng dn cài đt đi đây dành cho *Nitrous.IO* <<http://nitrous.io>>.

Bc 1: To ng dng mi vi NodeJS :

<https://www.nitrous.io/app#/boxes/new>

Bc 2: M terminal / SSH vào ng dng / M IDE

Bc 3: Ly file ca NodeBB v, unzip, xóa master.zip và cd vào folder

```
wget https://github.com/NodeBB/NodeBB/archive/v0.5.x.zip && unzip NodeBB-v0.5.x.zip &&
↪ rm NodeBB-v0.5.x.zip && cd NodeBB-v0.5.x
```

Bc 4: NPM Install

```
npm install
```

Bc 5: Cài đt Redis

```
parts install redis
```

Bc 6: Cài đt NodeBB

```
./nodebb setup
```

Đ mi th là mc đnh, nhng bn vn có th t thay đi.

Tôi đ ngh nên dùng cng : 8080

Bc 14: Và cui cùng, khi đng NodeBB

```
./nodebb start
```

Và sao đó m “Preview URI” không có cng nu bn đã đt cng là : 8080.

Chú ý

Bn có th m rng tài nguyên ca ng dng : <http://www.nitrous.io/app#/n2o/bonus>.

Openshift Paas

Hng dn di đây dành cho *Openshift* <<http://openshift.com>> Paas.

Bc 1: To ng dng mi :

```
rhc app create nodebb nodejs-0.10
```

Bc 2: Thêm cartridge Redis

```
rhc add-cartridge http://cartreflect-claytondev.rhcloud.com/reflect?
↪github=smarterclayton/openshift-redis-cart -a nodebb
```

Bc 3: SSH đn ng dng ca bn

```
rhc app ssh -a nodebb
```

Bc 4: Xác đnh ip ca NodeJS và Redis đ NodeBB có th chy chính xác. Đây là mt yêu cu ca Openshift và là cách duy nht đ nó chy. Bn không th s dng \$IP trong config.json (Có nghĩa là bn không th nhp \$IP trong lnh node app --setup). Dòng đt tiên : NodeJS và dòng th hai : Redis Giá tr xut ra ca lnh echo \$REDIS_CLI s nh sau : -h ip_redis -p port_redis -a password

```
echo $OPENSHIFT_NODEJS_IP && echo $REDIS_CLI
```

Bc 5: Thoát SSH

Bc 6: Thêm mã ngun ca NodeBB vào repository ca ng dng

```
cd nodebb && git remote add upstream -m master https://github.com/NodeBB/NodeBB.git
```

Bc 7: Ly file v và push

```
git pull -s recursive -X theirs upstream v0.5.x && git push
```

Bc 8: Ngng chy ng dng

```
rhc app stop -a nodebb
```

Bc 9: SSH vào ng dng

```
rhc app ssh -a nodebb
```

Bc 10: Sa môi trng chy ca NodeJS trong terminal vi SSH

```
cd ~/nodejs/configuration && nano node.env
```

Bc 11: Thay server.js bng app.js và thoát khi editor

```
ctrl + x
```

Bc 12: Trong terminal khác, bt đt chy ng dng

```
rhc app start -a nodebb
```

Bc 13: Chy mã cài đt NodeBB trên terminal vi SSH

```
cd ~/app-root/repo && node app --setup
```

URL để cài đặt nên đặt là 'http://nodebb-username.rhcloud.com', thay username vì username của bạn.

Port number : 8080

IP or Hostname to bind to: Nhập giá trị xuất của \$OPENSIFT_NODEJS_IP trong Bước 4.

Host IP or address of your MongoDB instance: Nhập giá trị xuất của \$REDIS_CLI trong Bước 4.

Host port of your MongoDB instance: Nhập giá trị xuất của \$REDIS_CLI trong bước Bước 4.

Redis Password: Nhập giá trị xuất của \$REDIS_CLI trong Bước 4.

Bc 14: Và cuối cùng, trong terminal khác, khi đang khởi động

```
rhc app restart -a nodebb
```

Và sau đó truy cập <http://nodebb-username.rhcloud.com> trong trình duyệt.

Chú ý

Restart NodeBB trong trang Admin không hoạt động. Sử dụng :

```
rhc app restart -a nodebb
```

Koding

Chú ý: Việc cài đặt trên Koding cần có tài khoản miễn phí trên Koding.com.

1. Tạo tài khoản học đăng nhập vào *Koding.com* <<http://koding.com>>
2. Nhấn vào nút màu xanh lá trông như sau >_
3. Bạn sẽ nhìn thấy VM của bạn vì chế độ Off bên phải màu đỏ, hãy nhấn vào đó để bật VM
4. Nhấn tiếp vào nó khi có thông báo Ready.
5. Bây giờ bạn đang trong cửa sổ terminal. Việc cài đặt gần giống với Ubuntu, nhưng hơi khác một chút vì một số gói đã được cài đặt sẵn.
6. Đầu tiên, ta phải cập nhật hệ thống đã được cập nhật - `sudo apt-get update && sudo apt-get upgrade`
7. Nhập vào một khu vực mà bạn đã đăng ký, nếu bạn đăng ký bằng Github học qua 1 bên thì bạn nào khác, bạn phải cài đặt 1 cái trong trang cài đặt tài khoản. Sau đó quay trở lại.
8. Bây giờ chỉ cần sau `sudo apt-get install python-software-properties python g++ make`
9. Cài đặt gói http - `sudo apt-get install redis-server imagemagick`
10. Tiếp, tạo bản sao NodeBB vào trong NodeBB folder - `git clone -b v0.5.x https://github.com/NodeBB/NodeBB.git nodebb` (Tùy chọn: Thay thế nodebb cuối nếu bạn muốn có tên folder khác)
11. Truy cập vào NodeBB folder - `cd nodebb` (tr khi bạn đã đặt tên folder khác trước và nếu bạn đã quên tên folder, chạy `ls` để nhìn thấy tên của folder)
12. Bây giờ chúng ta cài đặt các trình phụ thuộc của NodeBB - `npm install` (có thể mất 1 đến 2 phút)
13. Cài đặt nodebb bằng dòng lệnh - `./nodebb setup`
14. Câu hỏi đầu tiên sẽ hỏi tên miền của bạn, đặt nó là localhost. Tên miền của bạn sẽ như sau `http://{yourkodingusername}.kd.io` - Username của bạn có thể nhìn thấy góc trên cùng bên phải.

15. Hoàn tất việc cài đặt (giá trị mặc định sau khi cài đặt tên miền đều có thể chấp nhận, hãy nhấn enter cho đến khi có yêu cầu “Create an Admin”)
16. Tạo username và mật khẩu admin, sau đó nó sẽ tạo các categories và một vài thứ khác mà làm NodeBB tuyệt vời.
17. Giờ chúng ta có thể chạy NodeBB - `./nodebb start`
18. Một tab khác trên trình duyệt và truy cập vào địa chỉ `http://{yourkodingusername}.kd.io:4567` (nếu bạn không thay đổi cổng mặc định trong phần cài đặt)
19. Bạn sẽ nhìn thấy 1 màn hình tiếp tục truy cập vào trang của bạn, nhấn vào link `na` bên dưới để tiếp tục truy cập trang.

Chúc mừng, bạn đã cài đặt thành công NodeBB trên Koding.com

Nếu bạn thấy trên không rõ ràng học bạn gặp rắc rối, hãy cho chúng tôi biết bằng cách [tạo báo cáo](#). (Hãy tag @a5mith trong issue của bạn, vì tôi đã viết hướng dẫn này)

Một vài vấn đề khi chạy trên Koding

Vì Koding là miễn phí, nên nó sẽ có 1 vài sự cố thái của host cloud:

1. VM sẽ ngừng hoạt động sau 15 phút không có hoạt động. Điều này không may là không ảnh hưởng đến trang web, mà là của Terminal (Bạn có thể khắc phục điều này bằng cách luôn mở terminal và chạy lệnh `ls` mỗi 10 phút để làm việc liên tục)
2. Thông báo bạn sẽ nhận được thông báo rằng “Your VM is unavailable, try again later”, bạn có thể thử đăng xuất và đăng nhập lại, tải lại trang, hoặc báo cáo lỗi vì điều gì xảy ra.
3. Koding.com sử dụng ubuntu để host VM, vì vậy hãy biết căn bản về Ubuntu sẽ có ích.
 - [Cloud9](#)
 - [Heroku](#)
 - [Nitrous.IO](#)
 - [Openshift](#)
 - [Digital Ocean](#) (external)
 - [Koding.com](#)
 - [Codio](#) (external)

Configuring Databases

NodeBB has a Database Abstraction Layer (DBAL) that allows one to write drivers for their database of choice. Currently we have the following options:

MongoDB

If you're afraid of running out of memory by using Redis, or want your forum to be more easily scalable, you can install NodeBB with MongoDB. This tutorial assumes you know how to SSH into your server and have root access.

These instructions are for Ubuntu. Adjust them accordingly for your distro.

Note: If you have to add `sudo` to any command, do so. No one is going to hold it against you ;)

Step 1: Install MongoDB

The latest and greatest MongoDB is required (or at least greater than the package manager). The instructions to install it can be found on the [MongoDB manual](#).

Step 2: Install node.js

Like MongoDB, the latest and greatest node.js is required (or at least greater than the package manager), so I'm leaving this to the official wiki. The instructions to install can be found on [Joyent](#).

Note: NPM is installed along with node.js, so there is no need to install it separately

Step 3: Install the Base Software Stack

Enter the following into the terminal to install the base software required to run NodeBB:

```
# apt-get install git build-essential imagemagick
```

Step 4: Clone the Repository

Enter the following into the terminal, replacing `/path/to/nodebb/install/location` to where you would like NodeBB to be installed.

```
$ cd /path/to/nodebb/install/location
$ git clone git://github.com/NodeBB/NodeBB.git nodebb
```

Step 5: Install The Required NodeBB Dependencies

Go into the newly created `nodebb` directory and install the required dependencies by entering the following.

```
$ cd nodebb
$ npm install
```

Step 6: Adding a New Database With Users

To go into the MongoDB command line, type:

```
$ mongo
```

To add a new database called `nodebb`, type:

```
> use nodebb
```

To add a user to access the `nodebb` database, type:

For MongoDB 2.6.x

```
> db.createUser( { user: "nodebb", pwd: "<Enter in a secure password>", roles: [
↪ "readWrite" ] } )
```

Note: NodeBB requires MongoDB 2.6.0 or higher. The role `readWrite` provides read or write any collection within a specific database to user.

Step 7: Configure MongoDB

Modify `/etc/mongodb.conf`.

```
# nano /etc/mongodb.conf
```

To enable authentication, uncomment `auth = true`. Restart MongoDB.

```
# service mongodb restart
```

Step 8: Configuring NodeBB

Make sure you are in your NodeBB root folder. If not, just type:

```
$ cd /path/to/nodebb
```

To setup the app, type:

```
$ node app --setup
```

- Change the hostname to your domain name.
- Accept the defaults by pressing enter until it asks you what database you want to use. Type `mongo` in that field.
- Accept the default port, unless you changed it in the previous steps.
- Change your username to `nodebb`, unless you set it to another username.
- Enter in the password you made in step 5.
- Change the database to `nodebb`, unless you named it something else.

Continue with the installation, following the instructions the installer provides you.

Step 9: Starting the App

To start the app, run:

```
$ ./nodebb start
```

Now visit `yourdomainorip.com:4567` and your NodeBB installation should be running.

NodeBB can also be started with helper programs, such as *supervisor* or *forever*. You can also use `nginx` as a *reverse proxy*).

LevelDB

Follow the *installation instructions* for your particular OS but feel free to omit the Redis installation.

After cloning NodeBB, ensure that you run:

```
npm install levelup leveledown
```

Finally, set up a directory to store your LevelDB database, for example:

```
mkdir /var/level/
```

Run the NodeBB install, select `level` when it prompts you for your database. If you created the folder as above, you can leave the rest of the questions as default.

- Redis (default, see *installation guides*)
- *Mongo*
- *Level*

Ghi chú: If you would like to write your own database driver for NodeBB, please visit our [community forum](#) and we can point you in the right direction.

Running a Secondary Database

Cnh báo: This option is experimental and should not be used on a production environment.

Both databases **must** be flushed before beginning - there isn't a mechanism yet that detects an existing installation on one database but not another. Until fail-safe's such as these are implemented this option is hidden under the `--advanced setup` flag.

```
node app --setup --advanced
```

Consult the other database guides for instructions on how to set up each specific database. Once you select a secondary database's modules, there's no turning back - until somebody writes an exporter/importer.

Currently this setup is being tested with Redis as the primary store (sets, lists, and sorted sets, because Redis is super fast with these), and Mongo as the hash store (post and user data, because ideally we wouldn't want this in RAM).

Configuring Web Server / Proxies

Here a few options that you can use to proxy your NodeBB forum.

Configuring nginx as a proxy

NodeBB by default runs on port 4567, meaning that builds are usually accessed using a port number in addition to their hostname:

```
http://example.org:4567
```

In order to allow NodeBB to be served without a port, nginx can be set up to proxy all requests to a particular hostname (or subdomain) to an upstream NodeBB build running on any port.

Requirements

- **NGINX version v1.3.13 or greater**
 - Package managers may not provide a new enough version. To get the latest version, [compile it yourself](#), or if on Ubuntu, use the [NGINX Stable](#) or [NGINX Development](#) PPA builds, if you are on Debian, use [DotDeb repository](#) to get the latest version of Nginx.
 - To determine your nginx version, execute `nginx -V` in a shell

Configuration

NGINX-served sites are contained in a `server` block. This block of options goes in a specific place based on how nginx was installed and configured:

- `/path/to/nginx/sites-available/*` – files here must be aliased to `../sites-enabled`
- `/path/to/nginx/conf.d/*.conf` – filenames must end in `.conf`
- `/path/to/nginx/httpd.conf` – if all else fails

Below is the basic nginx configuration for a NodeBB build running on port 4567:


```
server {
    listen 80;

    server_name forum.example.org;

    location / {
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $http_host;
        proxy_set_header X-NginX-Proxy true;

        proxy_pass http://127.0.0.1:4567/;
        proxy_redirect off;

        # Socket.IO Support
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }
}
```

Notes

- Remember to also edit *config.json* and change *use_port* from *true* to *false*
- nginx must be on version 1.4.x to properly support websockets. Debian/Ubuntu use 1.2, although it will work there will be a reduction in functionality.
- The `proxy_pass` IP should be `127.0.0.1` if your NodeBB is hosted on the same physical server as your nginx server. Update the port to match your NodeBB, if necessary.
- This config sets up your nginx server to listen to requests for `forum.example.org`. It doesn't magically route the internet to it, though, so you also have to update your DNS server to send requests for `forum.example.org` to the machine with nginx on it!

Configuring apache as a proxy

Prerequisites to making this work: Apache 2.4.x

What if I'm on 2.2.x (Debian/Ubuntu)?

you need to manually compile and add the module "mod_proxy_wstunnel" to the Apache 2.2 branch. If you're running Ubuntu or Debian, you're likely on the 2.2 branch of code.

The following guide will assist with that if you're on Debian or Ubuntu. This is what I used to backport the mod_proxy_wstunnel module to the 2.2 code base of Apache;

http://www.amoss.me.uk/2013/06/apache-2-2-websocket-proxying-ubuntu-mod_proxy_wstunnel/

NOTE: On ubuntu, if you're missing the ./configure file

You need to first run `./buildconf`. After this is complete, you will then be able to use `./configure`.

automake & libtool package was needed too.

```
apt-get install automake libtool
```

Enable the necessary modules

1. `sudo a2enmod proxy`
2. `sudo a2enmod proxy_http`
3. `sudo a2enmod proxy_wstunnel`

Add the config to Apache

The next step is adding the configuration to your `virtualhost.conf` file, typically located in `/etc/apache2/sites-available/`. The below configuration assumes you've used 4567 (default) port for NodeBB installation. It also assumes you have the bind address set to 127.0.0.1.

```
ProxyRequests off

<Proxy *>
    Order deny,allow
    Allow from all
</Proxy>
ProxyPass /socket.io/1/websocket ws://127.0.0.1:4567/socket.io/1/websocket
ProxyPassReverse /socket.io/1/websocket ws://127.0.0.1:4567/socket.io/1/websocket

ProxyPass /socket.io/ http://127.0.0.1:4567/socket.io/
ProxyPassReverse /socket.io/ http://127.0.0.1:4567/socket.io/

ProxyPass / http://127.0.0.1:4567/
ProxyPassReverse / http://127.0.0.1:4567/
```

The last thing you need to be sure of is that the `config.json` in the NodeBB folder has `use_port: false`. Otherwise some functionality will not work properly.

Example `nodebb/config.json`

```
{
  "base_url": "http://www.yoursite.com",
  "port": "4567",
  "use_port": false,
  "secret": "55sb254c-62e3-4e23-9407-8655147562763",
  "bind_address": "127.0.0.1",
  "database": "redis",
  "redis": {
    "host": "127.0.0.1",
    "port": "6379",
    "password": "",
    "database": "0"
  },
  "bcrypt_rounds": 12,
  "upload_path": "/public/uploads",
  "relative_path": ""
}
```

Change the domain and don't use the secret in the example above.

Configuring Varnish Cache

To be sure Varnish will work properly with NodeBB check that your configuration `/etc/varnish/default.vcl` is optimized for **websockets**.

```
backend nodebb {
    .host = "127.0.0.1"; # your nodebb host
    .port = "4567"; # your nodebb port
}

sub vcl_recv {

    # Pipe websocket connections directly to Node.js
    if (req.http.Upgrade ~ "(?i)websocket") {
        set req.backend = nodebb;
        return (pipe);
    }

    # NodeBB
    if (req.http.host == "forum.yourwebsite.com") { # change this to match your host
        if (req.url ~ "^/socket.io/") {
            set req.backend = nodebb;
            return (pipe); # return pass seems not working for websockets
        }
        return (pass); # don't cache
    }
}

sub vcl_pipe {
    # Need to copy the upgrade header
    if (req.http.upgrade) {
        set bereq.http.upgrade = req.http.upgrade;
    }
}
```

- *Nginx*
- *Apache*
- *Varnish Cache*

Running NodeBB

The preferred way to start and stop NodeBB is by invoking its executable:

- `./nodebb start` Starts the NodeBB server
- `./nodebb stop` Stops the NodeBB server
- Alternatively, you may use `npm start` and `npm stop` to do the same

The methods listed below are alternatives to starting NodeBB via the executable.

Simple Node.js Process

To start NodeBB, run it with `node` (some distributions use the executable `nodejs`, please adjust accordingly):

```
$ cd /path/to/nodebb/install
$ node app
```

However, bear in mind that crashes will cause the NodeBB process to halt, bringing down your forum. Consider some of the more reliable options, below:

Supervisor Process

Using the [supervisor package](#), you can have NodeBB restart itself if it crashes:

```
$ npm install -g supervisor
$ supervisor app
```

As `supervisor` by default continues to pipe output to `stdout`, it is best suited to development builds.

Forever Daemon

Another way to keep NodeBB up is to use the [forever package](#) via the command line interface, which can monitor NodeBB and re-launch it if necessary:

```
$ npm install -g forever  
$ forever start app.js
```

Upgrading NodeBB

NodeBB's periodic releases are located in the [Releases](#). These releases contain what is usually considered the most bug-free code, and is designed to be used on production-level instances of NodeBB.

You can utilise `git` to install a specific version of NodeBB, and upgrade periodically as new releases are made.

To obtain the latest fixes and features, you can also `git clone` the latest version directly from the repository (`master` branch), although its stability cannot be guaranteed. Core developers will attempt to ensure that every commit results in a working client, even if individual features may not be 100% complete.

As always, the NodeBB team is not responsible for any misadventures, loss of data, data corruption, or any other bad things that may arise due to a botched upgrade - so please **don't forget to back up** before beginning!

Upgrade Path

NodeBB's upgrade path is designed so that upgrading between versions is straightforward. NodeBB will provide upgrade compatibility (via the `--upgrade` flag) between the latest version of a lower branch and the latest version of the higher branch. For example, if `v0.2.2` is the latest version in the `v0.2.x` branch, you can switch to the `v0.3.x` branch and suffer no ill effects. Upgrading from `v0.2.0` to `v0.3.x` is not supported, and NodeBB will warn you when attempting to upgrade that you are not upgrading cleanly.

Upgrading between patch revisions

e.g. v0.1.0 to v0.1.1

Patch revisions contain bugfixes and other minor changes. Updating to the latest version of code for your specific version branch is all that is usually required.

Execute steps 1 through 3.

Upgrading between minor revisions

e.g. v0.1.3 to v0.2.0

Minor revisions contain new features or substantial changes that are still backwards compatible. They may also contain dependent packages that require upgrading, and other features may be deprecated (but would ideally still be supported).

Execute steps 1 through 4.

Upgrade Steps

Note: After upgrading between revisions (i.e. v0.0.4 to v0.0.5), it may be necessary to run the following upgrade steps to ensure that any data schema changes are properly upgraded as well:

1. Shut down your forum

While it is possible to upgrade NodeBB while it is running, it is definitely not recommended, particularly if it is an active forum:

```
$ cd /path/to/nodebb
$ ./nodebb stop
```

2. Back up your data

Ghi chú: This section is incomplete, please take care to back up your files properly!

Backing up Redis

As with all upgrades, the first step is to **back up your data!** Nobody likes database corruption/misplacement.

All of the textual data stored in NodeBB is found in a `.rdb` file. On typical installs of Redis, the main database is found at `/var/lib/redis/dump.rdb`.

Store this file somewhere safe.

Backing up MongoDB

To run a backup of your complete MongoDB you can simply run

```
mongodump
```

which will create a directory structure that can be restored with the `mongorestore` command.

It is recommended that you first shut down your database. On Debian / Ubuntu it's likely to be: `sudo service mongod stop`

Backing up LevelDB

As LevelDB is simply a collection of flat files, just copy the database over to a safe location, ex.

```
cp -r /path/to/db /path/to/backups
```

Store this file somewhere safe.

Avatars

Uploaded images (avatars) are stored in `/public/uploads`. Feel free to back up this folder too:

```
cd /path/to/nodebb/public
tar -czf ~/nodebb_assets.tar.gz ./uploads
```

3. Grab the latest and greatest code

Navigate to your NodeBB: `$ cd /path/to/nodebb`.

If you are upgrading from a lower branch to a higher branch, switch branches as necessary. ***Make sure you are completely up-to-date on your current branch!***

For example, if upgrading from `v0.3.2` to `v0.4.3`:

```
$ git fetch      # Grab the latest code from the NodeBB Repository
$ git checkout v0.4.x  # Type this as-is! Not v0.4.2 or v0.4.3, but "v0.4.x"!
$ git merge origin/v0.4.x
```

If not upgrading between branches, just run the following command:

```
$ git pull
```

This should retrieve the latest (and greatest) version of NodeBB from the repository.

Alternatively, download and extract the latest versioned copy of the code from [the Releases Page](#). Overwrite any files as necessary. This method is not supported.

4. Run the NodeBB upgrade script

This script will install any missing dependencies, upgrade any plugins or themes (if an upgrade is available), and migrate the database if necessary.

```
$ ./nodebb upgrade
```

Note: `./nodebb upgrade` is only available after `v0.3.0`. If you are running an earlier version, run these instead:

- `npm install`
- `ls -d node_modules/nodebb* | xargs -n1 basename | xargs npm update`
- `node app --upgrade`

6. Start up NodeBB & Test!

You should now be running the latest version of NodeBB.

Administrative Functions

Ghi chú: These docs are out of date. If you are interested in updating these (preferably with screenshots) let us know on the [community forum](#) because we are planning on a design overhaul of the ACP soon.

To view the admin panel (if you are an admin): http://your_nodebb_domain.com/admin

Top Menu (all can be easily reached by other means)

- NodeBB ACP (Administrator Control Panel: this view) * http://your_nodebb_domain.com/admin/index (see Home below)
- Forum * http://your_nodebb_domain.com (your main forum)
- Home * http://your_nodebb_domain.com/admin/index (see Home below)
- Settings * http://your_nodebb_domain.com/admin/settings (see Settings below)

Side Menu: NodeBB

- Home * http://your_nodebb_domain.com/admin/index * Links all go to nodebb.com homepage
 - NOTE: should all the links go to same place?
 - Message reminder of what version this is and to check for updates (See [Upgrading NodeBB](#)) * NOTE: would a link to where the latest stable version is help?
 - Active Users * lists number users per page-path (?) * NOTE: not clear exactly what the paths mean or how to visit that path
- Categories * http://your_nodebb_domain.com/admin/categories * Filters: Active, Disabled, Unit Tests * List of Categories:
 - Icon, Name, Desc, Action: Disable
 - Actions: Save, Add New

- Users * http://your_nodebb_domain.com/admin/users * Filters: Latest Users, Top Posters, Most Reputation, Action: Search * List of Users:
 - Icon, Link: Name, Reputation(star), Number Posts(pencil), Action: Ban
 - Action: Load More
- Groups * http://your_nodebb_domain.com/admin/groups
 - List of Groups
 - Name, Desc, Icon
 - Action: Delete Group * NOTE: What exactly can Groups be set up to do, besides Admin?
- Topics * http://your_nodebb_domain.com/admin/topics * List of Topics
 - Name [link to topic], Posted When and By, Number posts (Topic+Replies), Thread Actions: Pin(pushpin), Lock(lock), Delete(trashcan)
 - Action: Load More Topics
 - Topic [from List of Topics link]
 - Normal View of Topic+Reply Posts but with: * Link, Edit, Delete Actions all enabled for each Post * Thread Tools:
 - * Pin, Lock, Move, Delete
- Themes (See [Theming NodeBB](#)) * http://your_nodebb_domain.com/admin/themes * List of (Custom | Bootswatch) Themes
 - Actions: Use, Preview
 - Action: Revert (to base)
- Plugins (See [Writing Plugins for NodeBB](#)) * http://your_nodebb_domain.com/admin/plugins * List of Plugins
 - Action: De/activate
 - Info on making plugins
- Settings * http://your_nodebb_domain.com/admin/settings
 - General Settings * (textbox) Site Title * (textbox) Site Description * (textbox) Site Keywords * (textbox) Imgur Client ID
 - * NOTE: How does this function?
 - * (textbox) Maximum User Image Size
 - Privilege Thresholds (Use privilege thresholds to manage how much reputation a user must gain to receive moderator access.) * (textbox) Manage Thread * (textbox) Moderate Users * (textbox) Create Pinned Topics
 - Email Settings * (textbox) Email Address (The following email address refers to the email that the recipient will see in the “From” and “Reply To” fields.) * (textbox) SMTP Server Host (Default: 127.0.0.1) * (textbox) SMTP Server Port
 - User Settings * (textbox) Minimum Username Length * (textbox) Maximum Username Length * (textbox) Minimum Password Length
 - Post Settings * (textbox) Post Delay * (textbox) Minimum Title Length * (textbox) Minimum Post Length * (checkbox) Use Outgoing Links Warning Page
 - Action: Save

- Redis * http://your_nodebb_domain.com/admin/redis * Redis data storage stats
- Logger * http://your_nodebb_domain.com/admin/logger * (checkbox) Enable HTTP logging * (checkbox) Enable socket.io event logging * (textbox) Path to log file
- MOTD (Message of the Day) * http://your_nodebb_domain.com/admin/motd * (textarea) You can enter either full HTML or Markdown text. * (checkbox) Show the Message of the Day

Side Menu: Social Authentication (See [Enabling Social Network Logins](#))

- Twitter
- http://your_nodebb_domain.com/admin/twitter
- Facebook
- http://your_nodebb_domain.com/admin/facebook
- Google+
- http://your_nodebb_domain.com/admin/gplus

Side Menu: Plugins (Shows installed plugins)

Side Menu: Unit Tests (Will run qunit tests)

Social Network SSOs

NodeBB supports integration for Facebook, Twitter, and Google through third party plugins:

- `npm install nodebb-plugin-sso-facebook`
- `npm install nodebb-plugin-sso-twitter`
- `npm install nodebb-plugin-sso-google`

Other SSO vendors are available, such as GitHub. Please check the [plugin directory](#) for a list of all SSO vendors.

After installing and activating them, they require an API key in order to function:

Facebook

Register an application via the [Facebook Developers](#) page. A credit card or mobile phone number may be required in order to create a Developer account.

Create a new application, and obtain an Application Key and Application Secret:

Ensure that “Website with Facebook Login” is checked, and that the URL to your NodeBB instance is specified in the “Site URL” box. Add that site’s domain to the “App Domains” field.

Paste this key and secret into the appropriate boxes in the NodeBB Administration Panel (accessible via `/admin` on your NodeBB install)

Twitter

Register an application at the [Twitter Developers](#) page. Create a new Application, and obtain the Access Token and Secret:

Important: While setting up your application, be sure to specify a Callback URL. It does not have to correspond to your installation, it just cannot be blank.

Paste this token and secret into the appropriate boxes in the NodeBB Administration Panel (accessible via /admin on your NodeBB install)

Google

Register an application at the [Google API Console](#), and obtain a Client ID and Secret.

Paste this ID and secret into the appropriate boxes in the NodeBB Administration Panel (accessible via /admin on your NodeBB install)

Image Hosting APIs

Enabling Imgur Image Uploads

To enable post image attachments, first create an imgur app from :

<https://api.imgur.com/oauth2/addclient>

You can use : “Anonymous usage without user authorization”

After that you will get a “Client ID”.

Then install nodebb-plugin-imgur:

```
npm install nodebb-plugin-imgur
```

Activate the plugin from the control panel and restart NodeBB.

You should see a Imgur menu item in the control panel. Paste the Client ID to the “Imgur Client ID” in the plugin page. Save and you should be able to upload images by dragging them into the composer window.

Uploading to Amazon S3

Ghi chú: No documentation for this yet! See [the plugin thread](#) for more information.

NodeBB Style Guide

For the most part, NodeBB follows the [Google Javascript Style Guide](#).

Code Formatting

Ghi chú: The existing codebase as of July 2013 does not adhere to this style guide 100%. If you see instances where the style guide is not adhered to, feel free to restyle and send off a pull request.

Indentation & Bracing

NodeBB uses tabbed indentation. Bracing should follow the [One True Brace Style](#):

```
if (condition) {
  // code here ...
} else {
  // otherwise ...
}
```

Put conditionals and statements on separate lines and wrap with curly braces even if it's just one line:

```
if (leTired) {
  haveANap();
}
```

Errors

Most callbacks return an error as the first parameter. Handle this error first before processing further.

```
someFunction(parameters, function(err, data) {
  if(err) {
    return callback(err); // or handle error
  }
  // proceed as usual
});
```

Variables

Variables should always be prefaced with the *var* keyword:

```
var foo = 'bar';
```

Multiple declarations are to be included in the same *var* statement:

```
var foo = 'bar',
    bar = 'baz';
```

Semicolons

Use semicolons if at all possible

Nomenclature

CamelCase if at all possible:

```
functionNamesLikeThis, variableNamesLikeThis, ClassNamesLikeThis, EnumNamesLikeThis,
↪methodNamesLikeThis, CONSTANT_VALUES_LIKE_THIS, foo.namespaceNamesLikeThis.bar, and
↪filenameslikethis.js.
```

Core Modules

Ghi chú: This section is under construction. Have a look at the modules folder for more information, located at:

```
public/src/modules
```

Alerts

The alert module is a toaster notification that can be called via the following syntax:


```
app.alert({
  title: 'Success!',
  message: 'Here\'s an example of an alert!',
  location: 'left-bottom',
  timeout: 2500,
  type: 'success',
  image: 'https://i.imgur.com/dJBzcGT.jpg'
});
```

The above code will result in this notification (default styling):

To style this, have a look at the vanilla theme's `modules/alert.less` and `templates/alert.tpl`.

Parameters:

1. `title` - string, which can be a language string as well. Some core language strings that you can use here include: `[[global:alert.success]]` and `[[global:alert.error]]`
2. `message` - string, which can be a language string as well.
3. `location` (optional) - `right-top` (default), `left-top`, `right-bottom`, `left-bottom`
4. `timeout` (optional) - integer in milliseconds, default is permanent until closed.
5. `type` - `error`, `success`, `info`, `warning/notify`
6. `image` (optional) - string, URL to image.
7. `closefn` (optional) - function. This is called when the user closes the alert via the (X) button.
8. `clickfn` (optional) - function. This is called when the user clicks on the alert.

Writing Plugins for NodeBB

So you want to write a plugin for NodeBB, that's fantastic! There are a couple of things you need to know before starting that will help you out.

Like WordPress, NodeBB's plugins are built on top of a hook system in NodeBB. This system exposes parts of NodeBB to plugin creators in a controlled way, and allows them to alter content while it passes through, or execute certain behaviours when triggered.

See the full *list of hooks* for more information.

Filters and Actions

There are two types of hooks: **filters** and **actions**.

Filters act on content, and can be useful if you want to alter certain pieces of content as it passes through NodeBB. For example, a filter may be used to alter posts so that any occurrences of "apple" gets changed to "orange". Likewise, filters may be used to beautify content (i.e. code filters), or remove offensive words (profanity filters).

Actions are executed at certain points of NodeBB, and are useful if you'd like to *do* something after a certain trigger. For example, an action hook can be used to notify an admin if a certain user has posted. Other uses include analytics recording, or automatic welcome posts on new user registration.

When you are writing your plugin, make sure a hook exists where you'd like something to happen. If a hook isn't present, [file an issue](#) and we'll include it in the next version of NodeBB.

Configuration

Each plugin package contains a configuration file called `plugin.json`. Here is a sample:

```
{
  "id": "my-plugin",
  "name": "My Awesome Plugin",
  "description": "Your plugin's description",
  "url": "Absolute URL to your plugin or a Github repository",
  "library": "./my-plugin.js",
  "staticDirs": {
    "images": "public/images"
  },
  "less": [
    "assets/style.less"
  ],
  "hooks": [
    { "hook": "filter:post.save", "method": "filter" },
    { "hook": "action:post.save", "method": "emailme" }
  ],
  "languages": "path/to/languages"
}
```

The `id` property is a unique name that identifies the plugin.

The `library` property is a relative path to the library in your package. It is automatically loaded by NodeBB (if the plugin is activated).

The `staticDirs` property is an object hash that maps out paths (relative to your plugin's root) to a directory that NodeBB will expose to the public at the route `/plugins/{YOUR-PLUGIN-ID}`.

- e.g. The `staticDirs` hash in the sample configuration maps `/path/to/your/plugin/public/images` to `/plugins/my-plugin/images`

The `less` property contains an array of paths (relative to your plugin's directory), that will be precompiled into the CSS served by NodeBB.

The `hooks` property is an array containing objects that tell NodeBB which hooks are used by your plugin, and what method in your library to invoke when that hook is called. Each object contains the following properties (those with * are required):

- `hook`, the name of the NodeBB hook
- `method`, the method called in your plugin
- `priority`, the relative priority of the method when it is eventually called (default: 10)

The `languages` property is optional, which allows you to set up your own internationalization for your plugin (or theme). Set up a similar directory structure as core, for example: `language/en_GB/myplugin.json`.

Writing the plugin library

The core of your plugin is your library file, which gets automatically included by NodeBB if your plugin is activated.

Each method you write into your library takes a certain number of arguments, depending on how it is called:

- Filters send a single argument through to your method, while asynchronous methods can also accept a callback.
- Actions send a number of arguments (the exact number depends how the hook is implemented). These arguments are listed in the *list of hooks*.

Example library method

If we were to write method that listened for the `action:post.save` hook, we'd add the following line to the `hooks` portion of our `plugin.json` file:

```
{ "hook": "action:post.save", "method": "myMethod" }
```

Our library would be written like so:

```
var MyPlugin = {
  myMethod: function(postData) {
    // do something with postData here
  }
};
```

Using NodeBB libraries to enhance your plugin

Occasionally, you may need to use NodeBB's libraries. For example, to verify that a user exists, you would need to call the `exists` method in the `User` class. To allow your plugin to access these NodeBB classes, use `module.parent.require`:

```
var User = module.parent.require('./user');
User.exists('foobar', function(err, exists) {
  // ...
});
```

Installing the plugin

In almost all cases, your plugin should be published in [npm](#), and your package's name should be prefixed "nodebb-plugin-". This will allow users to install plugins directly into their instances by running `npm install`.

When installed via `npm`, your plugin **must** be prefixed with "nodebb-plugin-", or else it will not be found by NodeBB.

As of v0.0.5, "installing" a plugin by placing it in the `/plugins` folder is still supported, but keep in mind that the package `id` and its folder name must match exactly, or else NodeBB will not be able to load the plugin. *This feature may be deprecated in later versions of NodeBB.*

Testing

Run NodeBB in development mode:

```
./nodebb dev
```

This will expose the plugin debug logs, allowing you to see if your plugin is loaded, and its hooks registered. Activate your plugin from the administration panel, and test it out.

Disabling Plugins

You can disable plugins from the ACP, but if your forum is crashing due to a broken plugin you can reset all plugins by executing

```
./nodebb reset plugins
```

Alternatively, you can disable one plugin by running

```
./nodebb reset plugin="nodebb-plugin-im-broken"
```

Available Hooks

The following is a list of all hooks present in NodeBB. This list is intended to guide developers who are looking to write plugins for NodeBB. For more information, please consult *Writing Plugins for NodeBB*.

There are two types of hooks, **filters**, and **actions**. Filters take an input (provided as a single argument), parse it in some way, and return the changed value. Actions take multiple inputs, and execute actions based on the inputs received. Actions do not return anything.

Important: This list is by no means exhaustive. Hooks are added on an as-needed basis (or if we can see a potential use case ahead of time), and all requests to add new hooks to NodeBB should be sent to us via the [issue tracker](#).

Filters

`filter:admin.header_build`

Allows plugins to create new navigation links in the ACP

`filter:post.save`

Argument(s): A post's content (markdown text)

Executed whenever a post is created or edited, but before it is saved into the database.

`filter:post.get`

Argument(s): A post object (javascript Object)

Executed whenever a post is retrieved, but before being sent to the client.

`filter:header.build`

Allows plugins to add new navigation links to NodeBB

`filter:register.build`

Argument(s):

- *req* the express request object (javascript Object)
- *res* the express response object (javascript Object)
- *data* the data passed to the template (javascript Object)

Allows plugins to add new elements to the registration form. At the moment, the only one supported is 'data.captcha'

filter:post.parse

Argument(s): A post or signature's raw text (String)

Executed when a post or signature needs to be parsed from raw text to HTML (for output to client). This is useful if you'd like to use a parser to prettify posts, such as [Markdown](#), or [BBCode](#).

filter:posts.custom_profile_info

Allows plugins to add custom profile information in the topic view's author post block

filter:register.check

Argument(s):

- *req* the express request object (javascript Object)
- *res* the express response object (javascript Object)
- *userData* the user data parsed from the form

Allows plugins to run checks on information and deny registration if necessary.

filter:scripts.get

Allows to add client-side JS to the header and queue up for minification on production

filter:uploadImage**filter:uploadFile****filter:widgets.getAreas****filter:widgets.getWidgets****filter:search.query****filter:post.parse****filter:messaging.save****filter:messaging.parse****filter:sounds.get****filter:post.getPosts****filter:post.getFields****filter:auth.init****filter:composer.help**

`filter:topic.thread_tools`

`filter:topic.get`

Passes in the final parsed topic data.

`filter:user.create`

`filter:user.delete`

`filter:user.profileLinks`

`filter:user.verify.code`

Parameters: confirm_code

Ability to modify the generated verification code (ex. for using a shorter verification code instead for SMS verification)

`filter:user.custom_fields`

Parameters: userData

Allows you to append custom fields to the newly created user, ex. mobileNumber

`filter:register.complete`

Parameters: uid, destination

Set the post-registration destination, or do post-register tasks here.

`filter:widget.render`

`filter:templates.get_virtual`

Allows you to modify the `api/get_templates_listing` API call, allowing ajaxification to custom templates that are not served physically via the `templates`` parameter in ``plugin.json`

`filter:templates.get_config`

Allows you to add custom ajaxification rules in the `api/get_templates_listing` API call. See <https://github.com/NodeBB/nodebb-theme-vanilla/blob/master/templates/config.json> for more details

Actions

`action:app.load`

Argument(s): None

Executed when NodeBB is loaded, used to kickstart scripts in plugins (i.e. cron jobs, etc)

action:page.load

Argument(s): An object containing the following properties:

- `template` - The template loaded
- `url` - Path to the page (relative to the site's base url)

action:plugin.activate

Argument(s): A String containing the plugin's `id` (e.g. `nodebb-plugin-markdown`)

Executed whenever a plugin is activated via the admin panel.

Important: Be sure to check the `id` that is sent in with this hook, otherwise your plugin will fire its registered hook method, even if your plugin was not the one that was activated.

action:plugin.deactivate

Argument(s): A String containing the plugin's `id` (e.g. `nodebb-plugin-markdown`)

Executed whenever a plugin is deactivated via the admin panel.

Important: Be sure to check the `id` that is sent in with this hook, otherwise your plugin will fire its registered hook method, even if your plugin was not the one that was deactivated.

action:post.save

Argument(s): A post object (javascript Object)

Executed whenever a post is created or edited, after it is saved into the database.

action:post.upvote

Argument(s): `pid`, `uid`

Executed whenever a post is upvoted. `uid` is the user that has triggered the upvote.

action:post.downvote

Argument(s): `pid`, `uid`

Executed whenever a post is downvoted. `uid` is the user that has triggered the downvote.

action:email.send**action:post.setField****action:topic.edit****action:topic.pin**

Called when toggling pinned state Object: `tid`, `isPinned`, `uid`

action:topic.lock

Called when toggling locked state Object: tid, isLocked, uid

action:topic.move

Called when moving a topic from one category to another Object: tid, fromCid, toCid, uid

action:post.edit

action:post.delete

action:post.restore

action:notification.pushed

Argument(s): A notification object (javascript Object)

Executed whenever a notification is pushed to a user.

action:config.set

action:topic.save

action:topic.delete

action:user.create

action:user.verify

Parameters: uid; a hash of confirmation data (ex. confirm_link, confirm_code) Useful for overriding the verification system. Currently if this hook is set, the email verification system is disabled outright.

action:user.follow

Parameters: fromUid, toUid

action:user.unfollow

Parameters: fromUid, toUid

action:user.set

Parameters: field (str), value, type ('set', 'increment', or 'decrement') Useful for things like awarding badges or achievements after a user has reached some value (ex. 100 posts)

action:settings.set

Parameters: hash (str), object (obj) Useful if your plugins want to cache settings instead of pulling from DB everytime a method is called. Listen to this and refresh accordingly.

Client Side Hooks

`filter:categories.new_topic`
`action:popstate`
`action:ajaxify.start`
`action:ajaxify.loadingTemplates`
`action:ajaxify.loadingData`
`action:ajaxify.contentLoaded`
`action:ajaxify.end`
`action:reconnected`
`action:connected`
`action:disconnected`
`action:categories.loading`
`action:categories.loaded`
`action:categories.new_topic.loaded`
`action:topic.loading`
`action:topic.loaded`
`action:composer.loaded`
`action:composer.topics.post`
`action:composer.posts.reply`
`action:composer.posts.edit`
`action:widgets.loaded`

Settings Framework

If you want to make your plugin customizable you may use the Settings Framework NodeBB offers.

Server-Side Access

First you need some default settings, just create a new object for this:

```

var defaultSettings = {
  booleans: {
    someBool: true,
    moreBools: [false, false, true]
  },
  strings: {
    someString: 'hello world',
    multiLineString: 'some\nlong\ntext',
    arrayOfStrings: ['some\nlong\ntexts', 'and another one']
  },
  numbers: {
    multiArrayDimensions: [[42,42],[21,21]],
    multiArrayDimensions2: [[42,42],[ ]],
    justSomeNumbers: [],
    oneNumber: 3,
    anotherNumber: 2
  },
  someKeys: ['C+S+#13'] // Ctrl+Shift+Enter
};

```

Now you can use the server-side settings-module to access the saved settings like this:

```

var Settings = module.parent.require('./settings');
var mySettings = new Settings('myPlugin', '0.1', defaultSettings, function() {
  // the settings are ready and can accessed.
  console.log(mySettings === this); // true
  console.log(this.get('strings.someString') === mySettings.get().strings.
  ↪someString); // true
});

```

The second parameter should change at least every time the structure of default settings changes. Because of this it's recommended to use your plugins version.

To use the settings client-side you need to create a WebSocket that delivers the result of `mySettings.get()`.

The `mySettings`-object will cache the settings, so be sure to use methods like `mySettings.sync(callback)` when the settings got changed from somewhere else and `mySettings.persist(callback)` when you finished `mySettings.set(key, value)` calls.

You need to create a socket-listener like following to allow the admin to initiate a synchronization with the settings stored within database:

```

var SocketAdmin = module.parent.require('./socket.io/admin');
SocketAdmin.settings.syncMyPlugin = function() {
  mySettings.sync();
};

```

If you want to add a reset-functionality you need to create another socket-listener:

```

SocketAdmin.settings.getMyPluginDefaults = function (socket, data, callback) {
  callback(null, mySettings.createDefaultWrapper());
};

```

The methods of the `mySettings` object you probably want to use:

- `constructor()`
- `sync([callback])` Reloads the settings from database, overrides local changes.
- `persist([callback])` Saves the local changes within database.

- **get ([key])** Returns the setting(s) identified by given key. If no key is provided the whole settings-object gets returned. If no such setting is saved the default value gets returned.
- **set ([key,]value)** Sets the setting of given key to given value. Remember that it's just a local change, you need to call `persist` in order to save the changes.
- **reset ([callback])** Persists the default settings.
- **getWrapper ()** Returns the local object as it would get saved within database.
- **createWrapper(version, settings)** Creates an object like it would get saved within database containing given information and settings.
- **createDefaultWrapper ()** Creates an object like it would get saved within database containing the default settings.

Client-Side Access

The next step is making the settings available to the admin.

You need to use the `hooks` filter: `admin.header.build` (to display a link to your page within ACP) and `action:app.load` (to create the needed route).

Within your page you can access the client-side Settings API via

```
require(['settings'], function (settings) {
  var wrapper = $('#my_form_id');
  // [1]
  settings.sync('myPlugin', wrapper);
  // [2]
});
```

To make a button with the id `save` actually save the settings you can add the following at [2]:

```
$('#save').click(function(event) {
  event.preventDefault();
  settings.persist('myPlugin', wrapper, function(){
    socket.emit('admin.settings.syncMyPlugin');
  });
});
```

As said before the server-side settings-object caches the settings, so we emit a `WebSocket` to notify the server to synchronize the settings after they got persisted.

To use a reset-button you can add the following at [2]:

```
$('#reset').click(function(event) {
  event.preventDefault();
  socket.emit('admin.settings.getMyPluginDefaults', null, function (err, data) {
    settings.set('myPlugin', data, wrapper, function(){
      socket.emit('admin.settings.syncMyPlugin');
    });
  });
});
```

There you go, the basic structure is done. Now you need to add the form-fields.

Each field needs an attribute `data-key` to reference its position within the settings. The Framework does support any fields whose jQuery-object provides the value via the `val ()` method.

The plugin to use for a field gets determined by its `data-type`, `type` or `tag-name` in this order.

Additionally the following plugins are registered by default:

- **array (types: `div`, `array`)** An Array of any other fields. Uses the object within `data-attributes` to define the array-elements. Uses `data-new` to define the value of new created elements.
- **key (types: `key`)** A field to input keyboard-combinations.
- **checkbox, number, select, textarea** Handle appropriate fields.

A full list of all attributes that may influence the behavior of the default Framework:

- `data-key`: the key to save/load the value within configuration-object
- `data-type`: highest priority type-definition to determine what kind of element it is or which plugin to associate
- `type`: normal priority type-definition
- `data-empty`: if `false` or `0` then values that are assumed as empty turn into null. `data-empty` of arrays affect their child-elements
- `data-trim`: if not `false` or `0` then values will get trimmed as defined by the elements type
- `data-split`: if set and the element doesn't belong to any plugin, it's value will get split and joined by its value into the field
- **array-elements:**
 - `data-split`: separator (HTML allowed) between the elements, defaults to `' , '`
 - `data-new`: value to insert into new created elements
 - `data-attributes`: an object to set the attributes of the child HTML-elements. `tagName` as special key will set the tag-name of the child HTML-elements
- **key-fields:**
 - `data-trim`: if `false` or `0` then the value will get saved as string else as object providing following properties: `ctrl`, `alt`, `shift`, `meta`, `code`, `char`
 - `data-split`: separator between different modifiers and the key-code of the value that gets saved (only takes effect if trimming)
 - `data-short`: if not `false` or `0` then modifier-keys get saved as first uppercase character (only takes effect if trimming)
- **select:**
 - `data-options`: an array of objects containing `text` and `value` attributes.

The methods of the `settings` module:

- **registerPlugin(plugin[, types])** Registers the given plugin and associates it to the given types if any, otherwise the plugins default types will get used.
- **get ()** Returns the saved object.
- **set(hash, settings[, wrapper[, callback[, notify]])** Refills the fields with given settings and persists them. `hash` Identifies your plugins settings. `settings` The object to save in database (settings-wrapper if you use server-side Settings Framework). `wrapper` (default: `'form'`) The DOM-Element that contains all fields to fill. `callback` (default: null) Gets called when done. `notify` (default: true) Whether to display saved- and fail-notifications.
- **sync(hash[, wrapper[, callback]])** Resets the settings to saved ones and refills the fields.

- `persist(hash[, wrapper[, callback[, notify]])` Reads the settings from given wrapper (default: 'form') and saves them within database.

For Settings 2.0 support the methods `load` and `save` are still available but not recommended.

Client-Side Example Template

An example template-file to use the same settings we already used server-side:

```

<h1>My Plugin</h1>
<hr />

<form id="my_form_id">
  <div class="row">
    <p>
      <h2>Settings</h2>
      A boolean: <input type="checkbox" data-key="booleans.someBool"></input>
      <br>
      An array of checkboxes that are selected by default:
      <div data-key="booleans.moreBools" data-attributes="{"data-type":"checkbox
      <br>
      A simple input-field of any common type: <input type="password" data-key=
      <br>
      A simple textarea: <textarea data-key="strings.multiLineString"></
      <br>
      Array of textareas:
      <div data-key="strings.arrayOfStrings" data-attributes="{"data-type":
      <br>
      2D-Array of numbers that persist even when empty (but not empty rows):
      <div data-key="numbers.multiArrayDimensions" data-split="<br>"
      data-attributes="{"data-type":"array","data-attributes":{"type":
      <br>
      Same with persisting empty rows, but not empty numbers, if no row is
      <br>
      <div data-key="numbers.multiArrayDimensions2" data-split="<br>" data-
      <br>
      data-attributes="{"data-type":"array","data-empty":true,"data-
      <br>
      Array of numbers (new: 42, step: 21):
      <div data-key="numbers.justSomeNumbers" data-attributes="{"data-type":
      <br>
      Select with dynamic options:
      <select data-key="numbers.oneNumber" data-options='[{"value":"2","text":"2
      <br>
      Select that loads faster:
      <select data-key="numbers.anotherNumber"><br>
      <option value="2">2</option>
      <option value="3">3</option>
      </select>

      Array of Key-shortcuts (new: Ctrl+Shift+7):
      <div data-key="someKeys" data-attributes="{"data-type":"key"}" data-new=
      <br>
    </p>
  </div>
  <button class="btn btn-lg btn-warning" id="reset">Reset</button>

```

```
<button class="btn btn-lg btn-primary" id="save">Save</button>
</form>

<script>
  require(['settings'], function (settings) {
    var wrapper = $('#my_form_id');
    // [1]
    settings.sync('myPlugin', wrapper);
    $('#save').click(function(event) {
      event.preventDefault();
      settings.persist('myPlugin', wrapper, function(){
        socket.emit('admin.settings.syncMyPlugin');
      });
    });
    $('#reset').click(function(event) {
      event.preventDefault();
      socket.emit('admin.settings.getMyPluginDefaults', null, function (err,
↪data) {
        settings.set('myPlugin', data, wrapper, function(){
          socket.emit('admin.settings.syncMyPlugin');
        });
      });
    });
  });
</script>
```

Custom Settings-Elements

If you want to define your own element-structure you can create a **plugin** for the Settings Framework.

This allows you to use a whole object like a single field which - besides comfort in using multiple similar objects - allows you to use them within arrays.

A plugin is basically an object that contains at least an attribute `types` that contains an array of strings that associate DOM-elements with your plugin.

You can add a plugin at [1] using the method `settings.registerPlugin`.

To customize the way the associated fields get interpreted you may add the following methods to your plugin-object:

All given elements are instances of JQuery.

All methods get called within Settings-scope.

- **use ()** Gets called when the plugin gets registered.
- **[HTML-Element|JQuery] create (type, tagName, data)** Gets called when a new element should get created (eg. by expansion of an array).
- **destruct (element)** Gets called when the given element got removed from DOM (eg. by array-splice).
- **init (element)** Gets called when an element should get initialized (eg. after creation).
- **[value] get (element, trim, empty)** Gets called whenever the value of the given element is requested. `trim` Whether the result should get trimmed. `empty` Whether considered as empty values should get saved too.
- **set (element, value, trim)** Gets called whenever the value of the given element should be set to given one. `trim` Whether the value is assumed as trimmed.

For further impression take a look at the [default plugins](#).

You should also take a look at the helper-functions within [Settings](#) in order to create your own plugins. There are a few methods that take response to call the methods of other plugins when fittingly.

Writing Widgets for NodeBB

See the original [blog post](#) for a high level overview and screenshots of the widget system.

Embedding HTML and JavaScript

You don't need to be a developer to figure this out. Head over to the Themes control panel and click on the Widgets tab. Create a new HTML widget by dragging and dropping the widget onto whatever template you want.

Copy and paste HTML or JavaScript into the widget and hit save - you're done!

You can optionally give your widget a container by dragging and dropping from the containers section onto your selected widget.

If you're looking for some sample scripts, head over to our [plugins section](#) and look for any topic labelled `nodebb-script-xyz`. Don't forget to submit your scripts and ideas as well!

Creating Widgets

You can define widgets in both plugins and themes. If you're building a plugin which simply delivers a widget (or collection of widgets), we strongly suggest you follow the `nodebb-widget-xyz` nomenclature instead when publishing.

Registering your widget

Listen to this hook to register your widget:

```
"hook": "filter:widgets.getWidgets", "method": "defineWidgets", "callbacked": true
```

Pass this back in the array:

Content defines the form that is displayed to customize your widget in the admin panel.

Listening to your widget

NodeBB core will call your widget on the appropriate page load by way of the hooks system. The hook will be named after your widget's namespace (see previous example) - like so: `filter:widget.render:widget_namespace`

This will pass in an object with the following useful properties:

- `obj.area` - will have `location`, `template`, `url`
- `obj.data` - will have your admin-defined data; in the example from the previous section you will be exposed an `obj.data.myKey`

Defining Widget Areas in Themes

A Widget Area is characterized by a template and a location. Themes can share widgets if they define the same Widget Areas. If an admin switches themes, widgets that were previously defined in a Widget Area incompatible with the new theme are saved.

Listen to this hook to register your Widget Area:

```
"hook": "filter:widgets.getAreas", "method": "defineWidgetAreas", "callbacked": true
```

Pass this back in the array:

```
{
  name: "Category Sidebar",
  template: "category.tpl",
  location: "sidebar"
}
```

And that's all. You can define as many Widget Areas in your theme as you wish. If you're still stuck, have a look at [this commit](#) which upgraded the Cerulean theme to use the widget system.

Creating a new NodeBB Theme

NodeBB is built on [Twitter Bootstrap](#), which makes theming incredibly simple.

Packaging for NodeBB

NodeBB expects any installed themes to be installed via `npm`. Each individual theme is an `npm` package, and users can install themes through the command line, ex.:

```
npm install nodebb-theme-modern-ui
```

The theme's folder must contain at least two files for it to be a valid theme:

1. `theme.json`
2. `theme.less`

`theme.less` is where your theme's styles will reside. NodeBB expects LESS to be present in this file, and will precompile it down to CSS on-demand. For more information regarding LESS, take a look at [the project homepage](#).

Note: A *suggested* organization for `theme.less` is to `@import` multiple smaller files instead of placing all of the styles in the main `theme.less` file.

Configuration

The theme configuration file is a simple JSON string containing all appropriate meta data regarding the theme. Please take note of the following properties:

- `id`: A unique id for a theme (e.g. "my-theme")
- `name`: A user-friendly name for the theme (e.g. "My Theme")

- `description`: A one/two line description about the theme (e.g. “This is the theme I made for my personal NodeBB”)
- `screenshot`: A filename (in the same folder) that is a preview image (ideally, 370x250, or an aspect ratio of 1.48:1)
- `url`: A fully qualified URL linking back to the theme’s homepage/project
- `templates`: (Optional) A system path (relative to your plugin’s root directory) to the folder containing template files. If not specified, NodeBB will search for the “templates” directory, and then simply fall back to using vanilla’s template files.

Child Themes

If your theme is based off of another theme, simply modify your LESS files to point to the other theme as a base:

topic.less

As `topic.less` from the theme `nodebb-theme-vanilla` was imported, those styles are automatically incorporated into your theme.

Important: If you depend on another theme, make sure that your theme specifically states this in its `package.json`. For example, for the above theme, as we depend on `nodebb-theme-vanilla`, we would explicitly state this by adding a new section into the `package.json` file:

```
"peerDependencies": {
  "nodebb-theme-vanilla": "~0.0.1"
}
```

Rendering Engine

How it works

Every page has an associated API call, Template file, and Language File.

For example, if you navigate to `/topic/351/nodebb-wiki`, the application will load three resources. The API return `/api/topic/351/nodebb-wiki` and the `template`, in this example, “`topic.tpl`”, and the appropriate language file “`topic.json`”^{*}.

Just prepend `api/` to the URL’s path name to discover the JSON return. Any value in that return can be utilized in your template.

^{*}A page’s name corresponds to the template and language’s filename (ex. `http://domain.com/topic/xyz` correlates to `topic.tpl`). Sometimes this is not the case - ex. `/user/xyz` loads `account.tpl`. Have a look at the `custom_mapping` section in `public/templates/config.json` for more details.

Templating Basics

Using the API return as your guide, you can utilize any of those values in your template/logic. Using the above API call as an example, for anything in the root level of the return you can do something like:

```
{topic_name}
```

To access values in objects:

```
{privileges.read}
```

And finally you can loop through arrays and create blocks like so:

```
<!-- BEGIN posts -->
{posts.content}
<!-- END posts -->
```

The above will create X copies of the above block, for each item in the posts array.

Templating Logic

NodeBB's templating system implements some basic logic. Using the same API call as above for our example. You can write IF conditionals like so:

```
<!-- IF unreplied -->
This thread is unreplied!
<!-- ENDIF unreplied -->
```

Another example:

```
<!-- IF !disableSocialButtons -->
<button>Share on Facebook</button>
<!-- ELSE -->
Sharing has been disabled.
<!-- ENDIF !disableSocialButtons -->
```

We can check for the length of an array like so:

```
<!-- IF posts.length -->
There be some posts
<!-- ENDIF posts.length -->
```

While looping through an array, we can check if our current index is the @first or @last like so:

```
<!-- BEGIN posts -->
  <!-- IF @first -->
    <h1>Main Author: {posts.username}</h1>
  <!-- ENDIF @first -->
  {posts.content}
  <!-- IF @last -->
    End of posts. Click here to scroll to the top.
  <!-- ENDIF @last -->
<!-- END posts -->
```

For more advanced documentation, have a look at the [templates.js](#) repository

Exposing template variables to client-side JavaScript

There are two ways of letting our JS know about data from the server-side, apart from WebSockets (TODO: will be covered in a different article).

Via jQuery.get

If we require data from a different page we can make a \$.get call to any other API call. For example, if we wanted to know more about a specific user we could make a call like so:

```
$.get(RELATIVE_PATH + '/api/user/psychobunny', {}, function(user) {
  console.log(user)
});
```

See this API call in action: <http://community.nodebb.org/api/user/psychobunny>

Via Template Variables

In topic.tpl for example, we can add a hidden input like so:

```
<input type="hidden" template-variable="pageCount" value="{pageCount}" />
```

The template system will immediately parse all of these and expose them via the following method:

```
ajaxify.variables.get('pageCount');
```

This is the ideal method of letting JS know about important variables within the template.

Internationalization

The template engine interfaces with the internationalization system as well. We can embed variables into language strings. Let's use [this API call](#) as well as [this language file](#) as an example. We can now do something like the following:

```
[[register:help.username_restrictions, {minimumUsernameLength},
↪{maximumUsernameLength}]]
```

Which will translate this string:

```
A unique username between %1 and %2 characters
```

to

```
A unique username between 2 and 16 characters
```

Advanced Topics

Dynamically requiring and rendering a template file from client-side JavaScript

The template engine lazy loads templates on an as-needed basis and caches them. If your code requires a template or partial on-demand then you can :

```
ajaxify.loadTemplate('myTemplate', function(myTemplate) {
  var html = templates.parse(myTemplate, myData);
});
```

You can also access the individual blocks inside each template, which is handy for doing things like (for example) rendering a new post's and dynamically sticking it in an already loaded


```
Some stuff here...
<!-- BEGIN posts -->
We just want to pull this block only.
<!-- END posts -->
... some stuff here
```

```
ajaxify.loadTemplate('myTemplate', function(myTemplate) {
  var block = templates.getBlock(myTemplate, 'posts');
  var html = templates.parse(block, myData);
});
```

Rendering templates on server-side Node.js

The templating system hooks into Express just like most other templating frameworks. Just use either `app.render` or `res.render` to parse the appropriate template.

```
res.render('myTemplate', myData);
```

```
app.render('myTemplate', myData, function(err, parsedTemplate) {
  console.log(parsedTemplate);
});
```


Developer's Resources

Ghi chú: Khu vc này đang đc xây dng.

Lỗi

- Xây dng trang qun tr mi (Đã li thi)

Plugins

- Phát trin plugins vi Grunt và Coffeescript
- Vit NodeBB plugin đư tiên ca bn

Giao din

Widgets

Helping out the NodeBB Project

NodeBB is an open source project, and will forever remain free. Here's a number of ways you can help us, even if you aren't a programmer.

- [Like and share our content on Facebook](#)
- [Follow us on Twitter](#) and perhaps tweet **#NodeBB is most awesome forum software @NodeBB**
- Update our wiki! ;) We need everything from development/design tutorials to user friendly how-to guides.
- Tell everybody about NodeBB, including your grandma and her cats.
- [Submit a pull request, or two, or three..](#)
- Build a new theme
- Write a plugin
- Keep the link back to us on the footer of your own NodeBB :)
- Blog about us! Give the gift of SEO juice this Christmas
- [Help Translate NodeBB](#) - It's a really simple translation tool and you don't need to know how to code.
- Join our [community](#) and give us a hard time about bugs and missing features

Translating NodeBB to another language

NodeBB uses Transifex, which is a user friendly visual tool which allows any individual to translate text into a language of their choice. You don't need to be a programmer to do this, so what are you waiting for? [Join the translation team now](#) :)

Writing Documentation

These docs were written using [Sphinx](#) and published using [rtfd.org](#).

You can edit these docs [directly on GitHub](#), or by clicking on “View page source” on the top right of any page.

If you wish, you can clone the repository and compile the documentation yourself. Check out the [Getting Started](#) section for more info on how to accomplish the latter.

Documentation are auto-compiled and pushed to [rtfd.org](#) after every commit.

Need Help?

Frequently Asked Questions

If you experience difficulties setting up a NodeBB instance, perhaps one of the following may help.

How do I start/stop/restart NodeBB?

You can call the `./nodebb` executable to start and stop NodeBB:

```
$ ./nodebb
Welcome to NodeBB
Usage: ./nodebb {start|stop|reload|restart|log|setup|reset|upgrade|dev|watch}

start    Start the NodeBB server
stop     Stops the NodeBB server
reload   Restarts NodeBB
restart  Restarts NodeBB
log      Opens the logging interface (useful for debugging)
setup    Runs the NodeBB setup script
reset    Disables all plugins, restores the default theme.
upgrade  Run NodeBB upgrade scripts, ensure packages are up-to-date
dev      Start NodeBB in interactive development mode
watch    Start NodeBB in development mode and watch for changes
```

How do I upgrade my NodeBB?

Please consult *Upgrading NodeBB*

I upgraded NodeBB and now X isn't working properly!

Please consult *Upgrading NodeBB*

I installed an incompatible plugin, and now my forum won't start!

If you know which plugin caused problems, disable it by running: `./nodebb reset plugin="nodebb-plugin-pluginName"`

Otherwise, disable all plugins by running: `./nodebb reset plugins`

Is it possible to install NodeBB via FTP?

It is possible to transfer the files to your remote server using FTP, but you do require shell access to the server in order to actually “start” NodeBB. Here is a [handy guide for installing NodeBB on DigitalOcean](#)

I'm getting an “npm ERR!” error

For the most part, errors involving npm are due to Node.js being outdated. If you see an error similar to this one while running `npm install`:

```
npm ERR! Unsupported
npm ERR! Not compatible with your version of node/npm: connect@2.7.11
```

You'll need to update your Node.js version to 0.8 or higher.

To do this on Ubuntu:

```
# add-apt-repository ppa:chris-lea/node.js
# apt-get update && apt-get dist-upgrade -y
```

If successful, running the following command should show a version higher than 0.8

```
# apt-cache policy nodejs
```

URLs on my NodeBB (or emails) still have the port number in them!

If you are using *nginx* or *Apache* as a reverse proxy, you don't need the port to be shown. Simply run `.nodebb setup` and specify the base URL without a port number.

Alternatively, edit the `config.json` file using your favourite text editor and change `use_port` to `false`.

The “Recently Logged In IPs” section only shows 127.0.0.1

NodeBBs running behind a proxy may have difficulties determining the original IP address that requests come from. It is important that the proxy server provides the referral IP header.

In *nginx*, ensure that the following line is present in your `server` block:

```
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

In addition, ensure that the `use_port` option is set to `false` in your NodeBB's `config.json`

Submit Bugs on our Issue Tracker

Before reporting bugs, please ensure that the issue has not already been filed on our [tracker](#), or has already been resolved on our [support forum](#). If it has not been filed, feel free to create an account on GitHub and [create a new issue](#).

Ask the NodeBB Community

Having trouble installing NodeBB? Or did something break? Don't hesitate to join our forum and ask for help. Hopefully one day you'll be able to help others too :)

CHAPTER 13

Ch muc và bng

- genindex
- modindex
- search