User side

- **Users will be able to add comments to articles.**
  The form, located at the bottom of article pages, will contain a field for the user's name and a textarea for the comment. Comments will be written in Markdown using an implementation functionally identical to NodeBB's implementation[1] so that the forum works the same way as the front page.
- **Users will be able to log in if they want a persistent identity.**
  Login will be available via NodeBB[2] after migration.
- **Anonymous users will be able to add comments after solving a captcha.**
  We'll use reCaptcha[3]. Good news: for most people, just tick a box. Bad news: no more "captcha: asdosungèg" comments.
- **A maximum number of comments will be displayed per page.**
  20 comments per page. Page navigation will be at the top and bottom of the comments list with direct links to the first and last pages as well as up to three pages before or after the current page.
- **Logged-in users will be able to add information to the end of their comments, but not change the already-posted content.**
  (Yes, we're bringing back addendums!) The editing screen will contain the current comment and a box for the addendum, and the addendum will be timestamped. Adding an addendum appends to the original comment content. There is no special database structure for addendums.
- **Comments can be replies to other comments.**
  Each comment will either be a reply directly to the article or a reply to another comment on the same article. This is simply a nullable comment ID in the database. We're not doing anything crazy like Discourse's ability for a post to reply to itself recursively.

Admin side

- **Admins will be able to edit comments.**
  This will be done through the admin interface.
- **Admins will be able to mass-delete comments.**
  This will be done through the admin interface using checkboxes with support for shift-clicking to select a group of consecutive comments.
- **Admins will be able to see all the comments made by a specific user.**
  Either logged-in or per IP address. Edit/delete actions will be available for admins on user pages as well as on article pages.

---

[1] https://www.npmjs.com/package/nodebb-plugin-markdown
[2] https://www.npmjs.com/package/nodebb-plugin-ns-login
[3] https://www.google.com/recaptcha/intro/index.html

- **Admins will be able to feature comments.**
  Just like the old system[4]! On article comments pages in the admin area, there will be a “feature” link that marks the comment as featured. Comments can be un-featured by using the “un-feature” link that replaces the “feature” link on featured comments.
- **We will migrate the Articles category from Discourse.**
  Because the current system barely works. Because Discourse barely works. Since both Discourse and the new system use Markdown, any problem will be on Discourse’s side.

[4] http://thedailywtf.com/Articles/What-Should-He-Do.-Indeed.aspx

Supplemental information

Important notes

NodeBB will need access to Community Server's database, preferably on a LAN. Because of this, it will probably be easier to do the import on a local VM or a spare PC running Linux and then copy it to a server when it's done. The uploads folder does not need to be on the machine doing the import if it is not the production server. The import takes 10-15 hours on my 2000s-era PC, and with Community Server added it will probably take longer.

An incremental import can be done by re-copying Discourse's database and repeating the import starting from the part where nodebb-plugin-import is installed. Make sure to choose to NOT flush the database. It is advisable to make a backup between incremental imports so that if something goes wrong you only have to repeat the current import and not the entire forum's import.

Users and categories will only be collected from Discourse. If a user or category was deleted after the original CS import, or a user changed their email address on Discourse between the original CS import and now, the posts related to that user or category will not be imported. Topics and posts imported during the original CS import will be re-obtained from the CS database. Discourse-era posts in imported topics will remain as they are.

It is very important that only the person doing the import can access the forum until the category permissions are applied.

NodeBB setup instructions (Docker)

Memory (-m) is for a machine with 3GB of RAM and other servers running in Docker. Adjust to taste. If it gives a warning about not limiting swap, that's okay.

First things first: get Postgres running with Discourse's database. We need a copy of the postgres_data folder from Discourse. Run this command with postgres_data in the current directory or edit it with the path to postgres_data. Do this on a copy of postgres_data just in case something goes wrong.

```
docker run -d --name wtdwtf-postgres-temp -v
"`pwd`"/postgres_data:/var/lib/postgresql/data postgres:9.3

docker run -d --name wtdwtf-postgres -m 250M -e
POSTGRES_PASSWORD=discourse postgres:9.3

docker exec --user postgres wtdwtf-postgres-temp pg_dump -bCxo
discourse | docker run -i --link wtdwtf-postgres:postgres --rm
```

```
postgres:9.3 sh -c 'PGPASSWORD=discourse exec psql -h
"$POSTGRES_PORT_5432_TCP_ADDR" -p "$POSTGRES_PORT_5432_TCP_PORT" -U
postgres'
```
Note: If this fails with `psql: could not connect to server: Connection refused`, wait a few seconds and try again. The second PostgreSQL server simply hasn't started yet.

Get rid of the temporary database server. We only needed it because Discourse's configuration for postgres was really weird. At this point, you no longer need the postgres_data directory, but keep a backup just in case.
```
docker stop wtdwtf-postgres-temp
docker rm -v wtdwtf-postgres-temp
```

Start MongoDB with its legacy storage engine that doesn't have a minimum 1GB disk cache in addition to the OS's disk cache.
```
docker run -d --name wtdwtf-mongo -m 2G mongo --storageEngine mmapv1
```

Start NodeBB without the setup command running first. This means that reboots will not need to be interactive. Also make the uploads directory a volume so we can upgrade more easily.
```
docker run -d --name wtdwtf-nodebb --link wtdwtf-postgres:postgres
--link wtdwtf-mongo:mongo -v
/usr/share/nginx/wtdwtf-nodebb.uploads:/usr/src/app/public/uploads
benlubar/nodebb:v1.0.0 npm start
```

Apply https://github.com/NodeBB/NodeBB/pull/4288 so this doesn't take forever
```
curl
https://gist.githubusercontent.com/BenLubar/a4a24cad319e7f2c664e/raw/
5bd58d0ca2440d9ae7a5b6106f7d8e76898f2049/batch-mongodb-fast-path.diff
| docker exec -i wtdwtf-nodebb git apply
```

Create the folders NodeBB expects inside the uploads folder.
```
sudo mkdir -p
/usr/share/nginx/wtdwtf-nodebb.uploads/{category,files,profile,sounds
,system}
```

Download the suggested plugins for WTDWTF.
```
docker exec -ti wtdwtf-nodebb npm install
'git://github.com/BenLubar/nodebb-plugin-import.git#1.0.0-support'
'git://github.com/BenLubar/nodebb-plugin-import-discourse.git#tdwtf'
'git://github.com/barisusakli/nodebb-plugin-emailer-amazon.git#patch-
1'
nodebb-plugin-{htmlcleaner,ns-login,google-analytics,gravatar,categor
y-notifications,sso-{facebook,github,google,twitter},write-api,questi
on-and-answer,shortcuts,emoji-{static,one}}
```

Run the setup command manually. There will be an error about kerberos from node-gyp, but it's not something we need to worry about.
```
docker exec -ti wtdwtf-nodebb ./nodebb setup
```
important settings:
database: **mongo**
database host: **mongo**
database port **(leave blank)**
database username **(leave blank)**
database password **(leave blank)**
which database **(leave blank)**

Set up a username and password for the initial admin (the one doing the import). Make sure that the username and email address are the same as their original Discourse username/email address, or at the very least, not used by any other users on the Discourse forum.

Restart the NodeBB image so that it runs with the configuration we just made.
```
docker restart wtdwtf-nodebb
```

Make a directory for the socket we're going to talk to the reverse proxy through.
```
sudo mkdir -p /usr/share/nginx/wtdwtf-nodebb.sock
sudo chown www-data:www-data /usr/share/nginx/wtdwtf-nodebb.sock
```

Make a socat image. socat will convert NodeBB's port (on a dynamic IP address) to a unix socket (on a shared filesystem path) so that the nginx configuration doesn't need to be updated every time the docker image is changed.
```
docker run -d --name wtdwtf-socat --link wtdwtf-nodebb:nodebb -v
/usr/share/nginx/wtdwtf-nodebb.sock:/sock --user www-data:www-data -m
25M verb/socat -d -d UNIX-LISTEN:/sock/sock,fork,reuseaddr,umask=0
TCP4:nodebb:4567
```

Copy the Discourse uploads folder from /var/discourse/shared/standalone/uploads on the old server to /usr/share/nginx/wtdwtf-nodebb.uploads on the new server. The new directory will already have been created by the previous commands.

Example nginx configuration (Docker)

```
server {
    # don't forget your SSL configuration from your SSL certificate
authority.

    listen 443 ssl;
    listen 80;
```

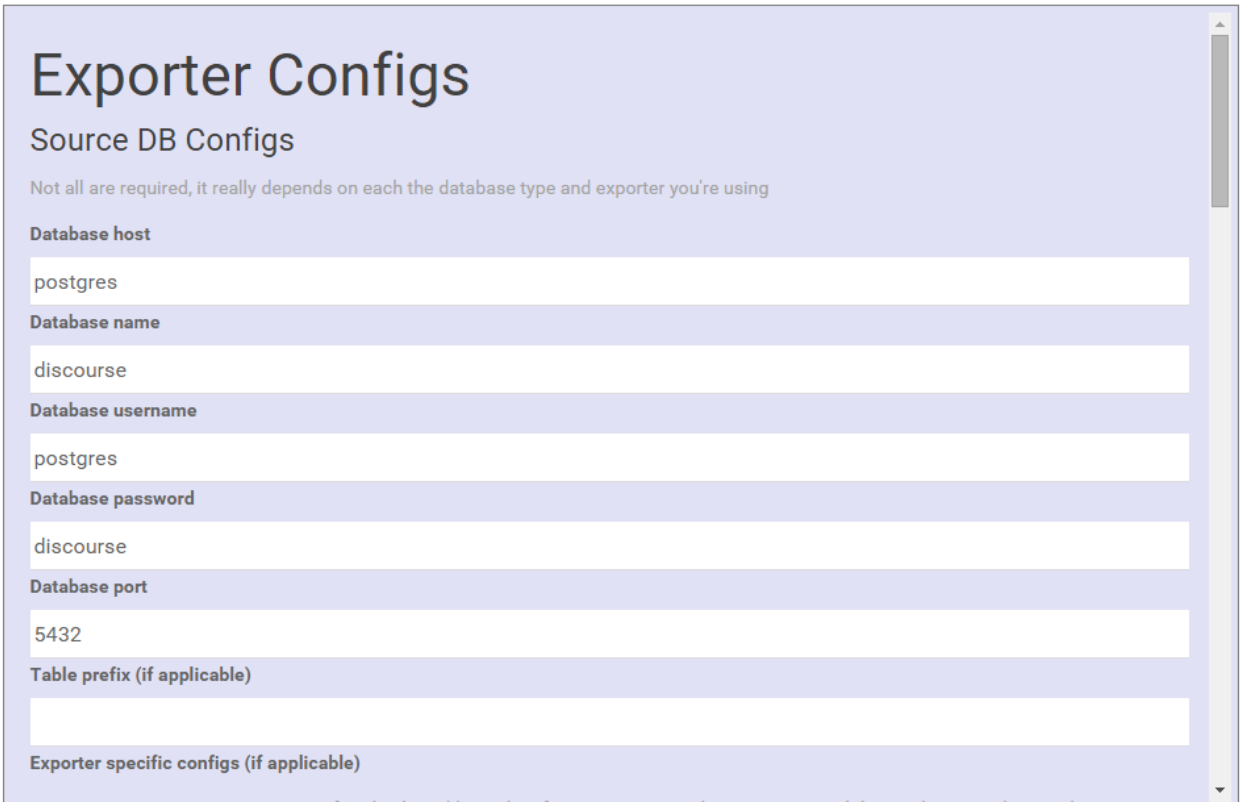
```
    server_name what.thedailywtf.com;

    location /uploads {
        root /usr/share/nginx/wtdwtf-nodebb.uploads;
        try_files $uri =404;
    }

    location / {
        allow 127.0.0.1; # change this to the IP address of the
person doing the import
        deny all; # delete this line and the one before it when the
import is done

        proxy_pass
http://unix:/usr/share/nginx/wtdwtf-nodebb.sock/sock:/;
        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_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_http_version 1.1;
    }
}
```

Discourse-to-NodeBB import (Docker)

Log in and go to **Extend > Plugins**. Deactivate all of the plugins. **Activate** nodebb-plugin-import and click the notification to **reload** the forum. If you miss the notification, there's a menu in the upper right of the page. Do not activate any other plugins.

Once you have reloaded, **refresh** your browser tab. Under the plugins menu should be an option labeled "**Import**". Click on it. Expand the **pre-import settings**.

Here are the important settings:



In the **exporter specific configs** section, put the connection string for Community Server's database wrapped in JSON, like `{"cs":"connection string goes here"}`. If you are doing an incremental import, you can use `{"skip_cs":true}` to skip the Community Server section of the import. There are other keys that will let you skip even more of the import when doing an incremental import. A list of them is at the top of index.js.

Under **Select an Exporter**, check the "don't install" box and paste the following into the box labeled "Or just enter the module's name or url you want to install":
`nodebb-plugin-import-tdwtf`

If the person doing the import had a Discourse or Community Server account:
Check the box labeled "**I want to take ownership of a specific user's posts**"
Enter the old username in the box labeled "**Old Username**".

Click "**Save Config**".
Re-open the pre-import settings and click the button labeled "**Flush NodeBB DB, then import**".

You can close the tab and the import will continue. The import takes **several hours**. (double digits of hours)

The second phase of the import is fixing the wide variety of quote styles. Open the **post-import tools** menu and check "**Pre-parse all content with my custom JavaScript**" and paste this code into the box: (ignore the unselectable placeholder text in the box)

```
var newContent;
while ((newContent =
content.replace(/\[quote\]\s*([\s\S]*?)\s*\[\/quote\]/g,
'<blockquote>$1</blockquote>')) !== content) content = newContent;
while ((newContent = content.replace(/\[quote="([^"\],]+), *post:
*([0-9]+), *topic: *([0-9]+)(?:, *full:
*true)?"\]\s*([\s\S]*?)\s*\[\/quote\]/g, '@$1 <a
href="/t/via-quote/$3/$2">said</a>:<blockquote>$4</blockquote>')) !==
content) content = newContent;
while ((newContent =
content.replace(/\[quote="([^"]+)"\]\s*([\s\S]*?)\s*\[\/quote\]/g,
'@$1 said:<blockquote>$2</blockquote>')) !== content) content =
newContent;
while ((newContent = content.replace(/\[quote
user="([^"]+)"\]\s*([\s\S]*?)\s*\[\/quote\]/g, '@$1
said:<blockquote>$2</blockquote>')) !== content) content =
newContent;
```

Leave everything else the same as it starts with and click **START CONVERT**.

After the import is finished, deactivate **nodebb-plugin-import** and activate **all other plugins**. Reload and refresh.

Now it's time to set up all the plugins and settings. **Remember to save** before leaving any of the settings pages.

You will need to install and configure a **mailer** plugin. Because Mandrill recently became a pay-only service, copying the settings from Discourse will not work.

You can use the following query to get site settings that I won't copy here for security reasons. The command to connect to postgres is in the "useful commands" section below.

```
SELECT name, value FROM site_settings WHERE name IN
('facebook_app_id', 'facebook_app_secret', 'github_client_id',
'github_client_secret', 'twitter_consumer_key',
'twitter_consumer_secret', 'google_oauth2_client_id',
'google_oauth2_client_secret', 'ga_universal_tracking_code');
```

Go to **/admin/plugins/spam-be-gone** and enter the keys for Akismet, ProjectHoneyPot, and reCaptcha. All three require free accounts on their respective services.

Go to **/admin/plugins/markdown** and check **Allow HTML**, then **Confirm**.

Go to **/admin/settings/general** and set:

- Site Title: **What the Daily WTF?**
- Title Layout: **{pageTitle} - {browserTitle}**

Download these images and **upload** them in the corresponding sections:

- Site Logo - Image: https://what.thedailywtf.com/uploads/default/original/3X/a/5/a55f6ae82ce16280bb7fb10636aa692045382979.png
- Favicon: https://what.thedailywtf.com/uploads/default/3/b22d0889376957fa.ico
- Homescreen / Touch Icon: https://what.thedailywtf.com/uploads/default/original/3X/a/5/a55f6ae82ce16280bb7fb10636aa692045382979.png

Go to **/admin/settings/email** and set the email address to **use-the-contact-form@thedailywtf.com** and the from name to **What the Daily WTF?**

Go to **/admin/settings/guest** and enable **Allow guests to search without logging in** and **Allow guests to search users without logging in**.

Go to **/admin/settings/uploads** and enable **Allow users to upload regular files** and **Convert profile image uploads to PNG**.

Go to **/admin/settings/post** and set **Minimum Post Length** to 1 and **Number of seconds users are allowed to edit posts after posting** to 604800 (7 days)

Go to **/admin/settings/user**:

- Check **disable username changes**
- Set **minimum username length** to 3
- Set **maximum username length** to 20

Go to **/admin/settings/tags** and set **Maximum Tags per Topic** to **2**, **Maximum Tag Length** to 20, and **Minimum Tag Length** to 3.

Go to **/admin/appearance/customise**, enable custom CSS, and enter the following in the box:

```
/* /users */
.users-container .users-box img, .users-container .users-box .user-icon {
    border-radius: 0; /* Get rid of women friendly shapes */
}
```

```
/* /topic/#/title.. */
.topic .posts .icon img, .topic .posts .icon .user-icon {
    margin-right: 30px; /* Push the dot over a bit more */
    border-radius: 0; /* Get rid of women friendly shapes */
}

.categories>li .content img, .category>ul>li .content img,
.categories>li .content .user-icon, .category>ul>li .content .user-icon,
.categories>li .card img, .category>ul>li .card img,
.categories>li .card .user-icon, .category>ul>li .card .user-icon,
.header #user_dropdown img, .header #user_dropdown .user-icon,
.expanded-chat .chat-content li.chat-message .chat-user-image,
.chat-modal .chat-content li.chat-message .chat-user-image,
.chats-list li .user-img, #menu .chat-list li a img, .header .chat-list li a img,
#menu .chat-list li a .user-icon, .header .chat-list li a .user-icon {
    border-radius: 0; /* Get rid of Jeff Atwood-friendly shapes */
}

.header .forum-logo {
       margin: 10px;
       height: 30px;
       width: 30px;
}

.navbar>.container .navbar-brand, .navbar>.container-fluid .navbar-brand {
    margin-left: -15px;
}

/* <kbd> */
kbd, kbd kbd {
    background-color: #fff;
    border: 1px solid #e9e9e9;
    border-radius: 3px;
    box-shadow: 0 1px 0 rgba(0,0,0,0.8);
    color: #222;
    display: inline-block;
    font-size: 0.857em;
    line-height: 1.4;
    margin: 0 .1em;
    padding: .1em .6em;
}

/* Stick a border around images */
.topic .posts .content iframe, .topic .posts .content .img-responsive {
    padding:1px;
    border:1px solid #D8E5FB;
    max-width:300px;
    max-height:300px;
}
```

```
/* Differentiate strike/del */
del{
    background-color: #FDD;
    border-color: #F1C0C0;
}
ins{
    background-color: #DBFFDB;
    border-color: #C1E9C1;
}
ins > del{
    background-color: #00CCFF;
    text-decoration: overline;
}
del > ins{
    background-color: #FFFF66;
    text-decoration: overline;
}

/* Show abbr and a title contents on mobile */
@media screen and (max-width: 991px) {
    abbr[title]:after{
        content: " [" attr(title) "]";
    }
    html,body {
        height:100%;
    }
}

/* Stop big+big...+big abuse http://what.thedailywtf.com/t/abusing-big-tags/5566 */
div.content big { font-size: 14px; }
div.content big big { font-size: 18px; }
div.content big big big { font-size: 22px; }
div.content big big big big { font-size: 26px; }
div.content big big big big big { font-size: 30px; }

/* And small-small */
div.content small { font-size: 9px; }
div.content small small { font-size: 8px; }
div.content small small small { font-size: 7px; }
div.content small small small small { font-size: 6px; }
div.content small small small small small { font-size: 5px; }

a[component="post/reply"], a[component="post/quote"] {
       background-color: #337ab7;
       color: white;
       padding: 10px;
       margin-right: 3px;
}
```

```
a[component="post/upvote"], a[component="post/downvote"]{
	background-color: #ccc;
	padding: 4px;
	padding-right: 2px;
	margin-right: 2px;
}

a[component="post/upvote"].upvoted > i.fa.fa-chevron-up {
	color: #0b0;
}

a[component="post/downvote"].downvoted > i.fa.fa-chevron-down {
	color: #c00;
}

span[component="post/vote-count"] {
	padding: 3px;
}
```

Go to **/admin/manage/groups**. **Hide or delete** trust_level_0, trust_level_1, and optionally trust_level_2. **Uncheck private** on the area_* groups to allow joining without admin approval. **Disable join requests** and set **hidden** on any group that should not be seen by users. **Uncheck show badge** on groups that should not show a badge.

Go to **/admin/general/languages** and change the default language to **en_US**.

Go to **/admin/general/homepage** and change the homepage route to **Recent**.

Go to **/admin/manage/categories** and set the category permissions. I suggest opening the edit link in a new tab because you won't have to scroll back to where you were. For admin-only categories, remove all permissions.

Here's some code you can paste into the MongoDB shell to do that for you:

```
var metaCid = db.objects.findOne({_key: '_imported:_categories',
value: '3'}).score;
db.objects.find({_key: 'categories:cid'}, {_id: 0, value:
1}).forEach(function(cid) {
	var cat = db.objects.findOne({_key: 'category:' + cid.value});

	function icon(name) {
		db.objects.update({_key: cat._key}, {$set: {icon: name}});
```

```
	}

	function disable() {
		icon('fa-trash');
		db.objects.update({_key: cat._key}, {$set: {disabled:
1}});
		onlyGroups([]);
	}

	function moveToMeta() {
		db.objects.remove({_key: 'cid:' + cat.parentCid +
':children', value: cid.value});
		db.objects.update({_key: cat._key}, {$set: {parentCid:
metaCid}});
		db.objects.insert({_key: 'cid:' + metaCid + ':children',
value: cid.value, score: +cid.value});
	}

	function onlyGroups(names) {
		db.objects.remove({_key: 'group:cid:' + cid.value +
':privileges:groups:find:members', value: {$ne: 'administrators'}});
		db.objects.remove({_key: 'group:cid:' + cid.value +
':privileges:groups:read:members', value: {$ne: 'administrators'}});
		db.objects.remove({_key: 'group:cid:' + cid.value +
':privileges:groups:topics:create:members', value: {$ne:
'administrators'}});
		db.objects.remove({_key: 'group:cid:' + cid.value +
':privileges:groups:topics:reply:members', value: {$ne:
'administrators'}});

		names.forEach(function(name) {
			db.objects.insert({_key: 'group:cid:' + cid.value +
':privileges:groups:find:members', value: name, score: +new Date()});
			db.objects.insert({_key: 'group:cid:' + cid.value +
':privileges:groups:read:members', value: name, score: +new Date()});
			db.objects.insert({_key: 'group:cid:' + cid.value +
':privileges:groups:topics:create:members', value: name, score: +new
Date()});
			db.objects.insert({_key: 'group:cid:' + cid.value +
':privileges:groups:topics:reply:members', value: name, score: +new
Date()});
		});
	}
```

```
     function noReply() {
          db.objects.remove({_key: 'group:cid:' + cid.value +
':privileges:groups:topics:reply:members', value: {$ne:
'administrators'}});
     }

     switch (cat._imported_path) {
     case '/c/article':
          disable();
          break;
     case '/c/article/authors-discussion':
          moveToMeta();
          onlyGroups(['authors', 'Global Moderators']);
          break;
     case '/c/coder-challenge':
          break;
     case '/c/funny-stuff':
          icon('fa-smile-o');
          break;
     case '/c/games':
          icon('fa-gamepad');
          break;
     case '/c/games/mafia':
          icon('fa-user-secret');
          break;
     case '/c/general':
          icon('fa-coffee');
          break;
     case '/c/general/look-at-me':
          break;
     case '/c/general-discussion':
          disable();
          break;
     case '/c/general-help':
          break;
     case '/c/general-help/coding-help':
          icon('fa-medkit');
          break;
     case '/c/meta':
          icon('fa-cogs');
          break;
     case '/c/meta/bug':
```

```
            icon('fa-bug');
            break;
      case '/c/meta/faqs':
            icon('fa-question-circle');
            break;
      case '/c/meta/flags-badges':
            icon('fa-flag-checkered');
            break;
      case '/c/meta/migration':
            break;
      case '/c/meta/one-post':
            noReply();
            break;
      case '/c/meta/the-lounge':
            icon('fa-paper-plane');
            onlyGroups(['trust_level_3', 'trust_level_4', 'Global
Moderators']);
            break;
      case '/c/meta/turn-left':
            onlyGroups(['trust_level_4', 'Global Moderators']);
            break;
      case '/c/meta/staff':
            onlyGroups(['Global Moderators']);
            break;
      case '/c/programmers-testing':
            icon('fa-code');
            onlyGroups(['programmers_testers', 'Global Moderators']);
            break;
      case '/c/programmers-testing/bot-testing':
            icon('fa-android');
            onlyGroups(['programmers_testers', 'bots', 'Global
Moderators']);
            break;
      case '/c/programmers-testing/tbd':
            icon('fa-code-fork');
            onlyGroups(['programmers_testers', 'Global Moderators']);
            break;
      case '/c/rubbish':
            disable();
            break;
      case '/c/side-bar-wtf':
            icon('fa-exclamation-triangle');
            break;
```

```
	case '/c/side-bar-wtf/codesod':
		icon('fa-code');
		break;
	case '/c/side-bar-wtf/errord':
		icon('fa-desktop');
		break;
	case '/c/the-i-hate-oracle-club':
		icon('fa-database');
		break;
	case '/c/uncategorized':
		disable();
		break;
	}
});
```

Go to **/admin/plugins/dbsearch**. Click the red "**clear index**" button. Click the orange "**re-index**" button. It's okay to leave the page. The forum will run slowly until this finishes, but it is required to do this once after the import for the search to work. Don't believe the progress bar. It takes much longer than it appears it is going to. (Probably around half an hour.) Do not refresh the forum or restart the container until this is complete or you will have to start it over.

Alternatively, you can run this script in the MongoDB shell to build the search index:

```
db.searchpost.dropIndexes();
db.searchtopic.dropIndexes();
db.searchpost.drop();
db.searchtopic.drop();
db.searchtopic.createIndex({id: 1});
db.searchpost.createIndex({id: 1});
db.objects.find({_key: /^topic:/, deleted: 0}, {_id: 0, tid: 1,
title: 1, cid: 1, uid: 1, mainPid: 1}).forEach(function(topic) {
	db.searchtopic.update({id: topic.tid}, {$set: {id: topic.tid,
content: topic.title, cid: topic.cid, uid: topic.uid}}, {upsert:
true, w: 1});
	function handlePost(pid) {
		var post = db.objects.findOne({_key: 'post:' + pid.value,
deleted: 0}, {_id: 0, content: 1, uid: 1});
		if (!post) {
			return;
		}
```

```
            db.searchpost.update({id: +pid.value}, {$set: {id:
+pid.value, content: post.content, cid: topic.cid, uid: post.uid}},
{upsert: true, w: 1});
      }
      handlePost({value: topic.mainPid});
      db.objects.find({_key: 'tid:' + topic.tid + ':posts'}, {_id: 0,
value: 1}).forEach(handlePost);
});
db.searchtopic.createIndex({content: 'text', uid: 1, cid: 1});
db.searchpost.createIndex({content: 'text', uid: 1, cid: 1});
```

After everything is done (you will have to wait for the search index), you can optionally compress the database and defragment its indices. This will make the forum inaccessible until it completes. If it is interrupted, you simply need to re-run the command. To do this, connect to mongodb (see the “useful commands” section below) and enter the following command:

```
["objects", "searchpost", "searchtopic"].forEach(function(c) {
db.runCommand({"compact": c}); });
```

It may be helpful to watch the mongo log from another terminal (again, see the “useful commands” section) because there will be no output for the command in the database shell, but it will write to the log. The database will use an additional 2GB of disk space while the command is running, but it will reuse that space for new records as they come in after the command finishes.

If you would like to remove the postgresql server, you will need to delete and re-create the nodebb and socat containers. However, because the uploads directory is outside the container and the database is in a separate container, the only data that gets lost is the database connection information. This is also how updating the NodeBB installation will work.

```
docker stop wtdwtf-socat wtdwtf-nodebb

docker rm -v wtdwtf-socat wtdwtf-nodebb

docker run -d --name wtdwtf-nodebb --link wtdwtf-mongo:mongo -v
/usr/share/nginx/wtdwtf-nodebb.uploads:/usr/src/app/public/uploads
benlubar/nodebb:v1.0.0 npm start

curl
https://gist.githubusercontent.com/BenLubar/a4a24cad319e7f2c664e/raw/
5bd58d0ca2440d9ae7a5b6106f7d8e76898f2049/batch-mongodb-fast-path.diff
| docker exec -i wtdwtf-nodebb git apply
```

```
docker exec -ti wtdwtf-nodebb npm install
'git://github.com/BenLubar/nodebb-plugin-import-discourse.git#tdwtf'
'git://github.com/barisusakli/nodebb-plugin-emailer-amazon.git#patch-
1'
nodebb-plugin-{htmlcleaner,ns-login,google-analytics,gravatar,categor
y-notifications,sso-{facebook,github,google,twitter},write-api,questi
on-and-answer,shortcuts,emoji-{static,one}}

docker exec -ti wtdwtf-nodebb ./nodebb setup
```
important settings:
database: **mongo**
database host: **mongo**
database port **(leave blank)**
database username **(leave blank)**
database password **(leave blank)**
which database **(leave blank)**

```
docker exec -ti wtdwtf-nodebb ./nodebb upgrade

docker restart wtdwtf-nodebb

docker run -d --name wtdwtf-socat --link wtdwtf-nodebb:nodebb -v
/usr/share/nginx/wtdwtf-nodebb.sock:/sock --user www-data:www-data -m
25M verb/socat -d -d UNIX-LISTEN:/sock/sock,fork,reuseaddr,umask=0
TCP4:nodebb:4567
```

Suggested NodeBB plugins:

- **nodebb-plugin-import-tdwtf** (keep this installed after the import for URL redirects)
- **nodebb-plugin-ns-login** (used by WtfWebApp)
- **nodebb-plugin-spam-be-gone** (included by default)
- **nodebb-plugin-dbsearch** (included by default, need to clear and rebuild index after import)
- **nodebb-plugin-markdown** (included by default)
- **nodebb-plugin-emoji-extended** (included by default)
- **nodebb-plugin-mentions** (included by default)
- **nodebb-plugin-sanitizehtml** (not required, but enabling HTML in the Markdown plugin is a huge vulnerability otherwise)
- **nodebb-plugin-emailer-amazon** (we're on EC2, so we may as well)
- **nodebb-plugin-google-analytics** (there's analytics on the main site, so why not on the forum too?)
- **nodebb-plugin-gravatar**

- **nodebb-plugin-category-notifications** (people apparently liked this feature of Discourse)
- ~~**nodebb-plugin-rainbows**~~ (pls no)
- **nodebb-plugin-sso-facebook**, **nodebb-plugin-sso-github**, **nodebb-plugin-sso-google**, **nodebb-plugin-sso-twitter** (I'm pretty sure people liked these from Discourse as well)
- **nodebb-plugin-write-api** (it might be helpful if we like SockBot enough to make it easy for them)
- **nodebb-plugin-question-and-answer** (for the Help categories)
- **nodebb-plugin-shortcuts** (people like their keyboard shortcuts)

Useful commands (Docker)

Connect to MongoDB (Ctrl+D to exit)
```
docker run --rm -it --link wtdwtf-mongo:mongo mongo sh -c 'exec mongo
"$MONGO_PORT_27017_TCP_ADDR:$MONGO_PORT_27017_TCP_PORT/0"'
```

Connect to PostgreSQL (Ctrl+D to exit)
```
docker run --rm -ti --link wtdwtf-postgres:postgres postgres:9.3 sh
-c 'PGPASSWORD=discourse exec psql -h "$POSTGRES_PORT_5432_TCP_ADDR"
-p "$POSTGRES_PORT_5432_TCP_PORT" -U postgres discourse'
```

Watch container log (Ctrl+C to exit) (replace `nodebb` with `mongo`, `postgres`, or `socat`)
```
docker logs -f --tail 100 wtdwtf-nodebb
```

Create MongoDB backup (creates a folder named mongo-backup-YYYY-MM-DD in the current directory)
```
docker run --rm -it --link wtdwtf-mongo:mongo -v "`pwd`":/data/backup
mongo sh -c 'exec mongodump --out "/data/backup/$(date --utc
'+mongo-backup-%Y-%m-%d')" --host "$MONGO_PORT_27017_TCP_ADDR"'
```

Restore MongoDB backup (backup (mongo-backup-YYYY-MM-DD) must be current directory)
```
docker run --rm -it --link wtdwtf-mongo:mongo -v "`pwd`":/data/backup
mongo sh -c 'exec mongorestore --host "$MONGO_PORT_27017_TCP_ADDR"
--drop /data/backup'
```