



# Emerging Technologies Blog

thoughts, inquiries, observations, hacking and test results

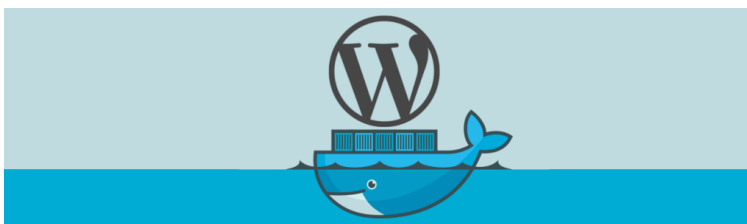


## WordPress on Bluemix Containers

May 22, 2015

Miguel Clement

3 comments



*A guide to Using app binding to connect a WordPress instance to a database.*

### Why?



#### Recent Posts

- [Learning Microservices Architecture with Bluemix and Docker \(Part 3\)](#)
- [Learning Microservices Architecture with Bluemix and Docker \(Part 2\)](#)
- [Learning Microservices Architecture with](#)

*Docker Containers are ideal for hosting WordPress sites for three reasons: official images, Volumes, and scalability.*

- The Official WordPress Image: WordPress maintains the [WordPress docker image](#) on [docker hub](#). This is an official image so you can be sure it is stable and set up properly.
- Docker Volumes: [Volumes](#) are used as persistent file storage. We use volumes as a file system for WordPress. If you stop, restart or even delete your wordpress container the volume will persist. This allows us to save media and install plugins/themes directly to the volume. Using volumes also makes migrating and scaling your WordPress site very easy!
- Scalability: With Bluemix docker containers you can increase the Memory of your containers, and Number of CPU's. You can also create a cluster of identical containers to host your site. This is great for load balancing! We also have the added benefit of being able to scale the database services independently. Your WordPress Database can come from any of the MySQL services in the [Bluemix catalog](#). (this tutorial will use clearDB)

---

## How?

This tutorial assumes you have already installed the cf "ic" plugin and docker CLI. For instructions on how to install the CLIs see the [bluemix docs](#). The first three steps are done with docker and the cf ic clis.



## Bluemix and Docker (Part 1)


- [Market Data Analytics: Advanced Research Techniques](#)
- [Drone assisted Selfie](#)

---

## Twitter Widget

**Tweets** Follow

**IBM jStart Team** 14h  
@IBMjStart  
"So You Want to Do Machine Learning?" on @LinkedIn [linkedin.com/puls](#)  
[pic.twitter.com/3z](#)  
  
Expand

**IBM jStart Team** 16h  
@IBMjStart  
Miguel forked the [#docker](#)  
- 12 14 15 16 17 18 19 20 21 22 23 24

Tweet to @IBMjStart

---

## Other Emerging Tech Blogs

- [Randy Wilcox](#)

## Step 1: Pull the official image locally

```
1 $ docker pull ibmjstart/bluemix-wordpress
```

## Step 2: Tag the wordpress image for pushing up to your bluemix registry

you'll need to replace [namespace] with your namespace.

```
1 $ docker tag ibmjstart/bluemix-wordpress registry
```

## Step 3: Push [namespace]/wordpress up to your registry

```
1 $ docker push registry.ng.bluemix.net/[namespace]
```

note: you may need to use "cf ic login" before pushing up to your bluemix registry.

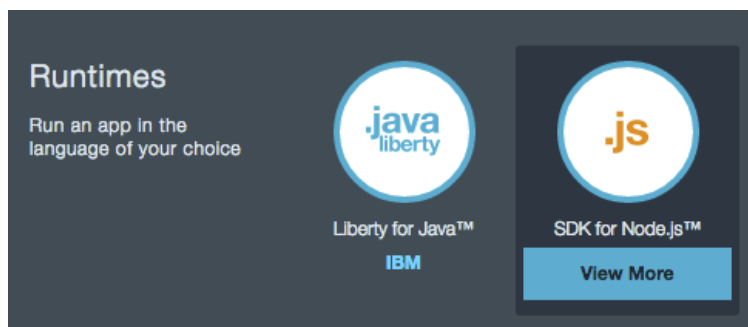
## Step 4: Create the volume that will serve as the WordPress file system

```
1 $ cf ic volume create [Volume Name]
```

You can name the volume whatever you want but remember the name for when you run the container in step 6.

## Step 5: Create a MySQL Service

Lets head over to [Bluemix](#) for the rest of the tutorial. In bluemix, you will need to create a new app to bind a MySQL DB to. I recommend using the **SDK for node.js runtime** from the catalog.



Go ahead and create it:

- [Nick Heidloff](#)
- [Ryan Baxter](#)

---

### Archives

- [July 2015](#)
- [June 2015](#)
- [May 2015](#)

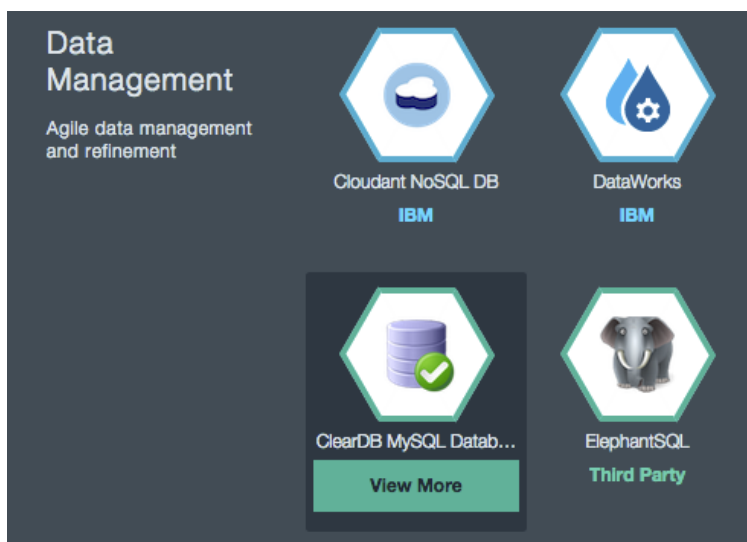
---

### Meta

- [Log in](#)
  - [Entries RSS](#)
  - [Comments RSS](#)
  - [WordPress.org](#)
-

| Plan      | Features  | Price              |
|-----------|---|--------------------|
| ✓ Default | Run one or more apps free for 30 days (\$75 GB-hours free). | \$0.07 USD/GB-Hour |

Make sure to choose a unique name for your app. Once your app is running go ahead and create a MySQL Service instance in the catalog:

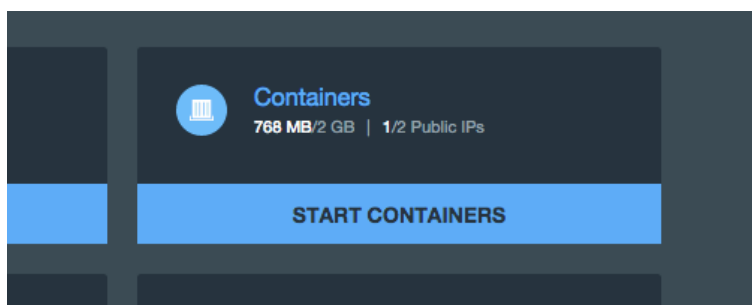


Finally make sure to bind it to the node.js app we just created. This is done in the App section.

| Plan       | Features  | Price |
|------------|---|-------|
| ✓ Spark DB | Price: FREE!<br>DB size: up to 5 MB<br>Connections: 4<br>I/O Performance: Low<br>Daily Backups<br>Perfect for proof-of-concept and initial development. | Free  |

## Step 6: Run the Container with the WordPress Image

Go on to your Dashboard and select  
Start Containers:



Next you will see this (your wordpress image  
should be listed):



Click on the wordpress image. That will take you  
to the following page. there a quite a few details  
on this page that we need to get right.

1. Make sure you are under the scalable  
group tab (Important!)
2. Give your container group a unique name
3. choose mybluemix.net as the route  
domain.

4. choose as many instances as you would like (1 should be fine)
5. choose a unique hostname
6. open HTTP port 80 (very important)
7. Click advanced options
8. Select the volume you created in step 4 from the dropdown. Set the mount path to `"/var/www/html"`
9. Bind the bridge app we made in step 5. by selecting the bridge app from the dropdown.

The rest can be left as default.

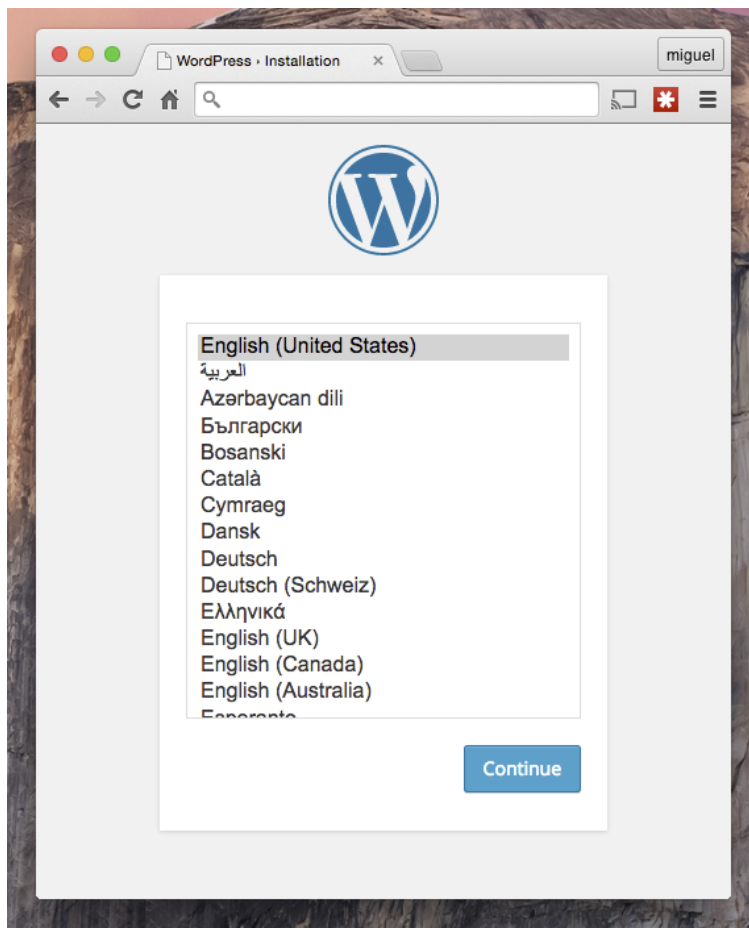
The screenshot shows the Bluemix console interface for creating a new container group. On the left sidebar, the 'wordpress' app is selected under 'My Org'. The main panel is titled 'Scalable Group' and contains the following configuration fields:

- Space:** MFCdev
- Container group name:** myWordPressName
- Instances:** 1
- Size:** Tiny(512 MB Memory, 32 GB Storage)
- Host:** myWordPressName
- Domain:** mybluemix.net
- HTTP port:** 80
- ☒ Enable automatic recovery
- Advanced Options:**
  - Volumes:** wp (dropdown) /var/www/html (text) ☐ Read-only (+)
  - Environment Variables:** Enter key / Enter value (+)
  - Service binding:** tokenAPI (dropdown)

hit **create**.

## Step 7: Hello WordPress

visit your page using the `[Hostname].mybluemix.net` url you selected in step 5 (note: this may take a few minutes to deploy, so give it a while). Your page should look like this:



You should now have a fully functional WordPress install on containers! run through the setup and you will be good to go. **If you would like to learn more about how our WordPress image works, you can read about it [HERE](#)**

---

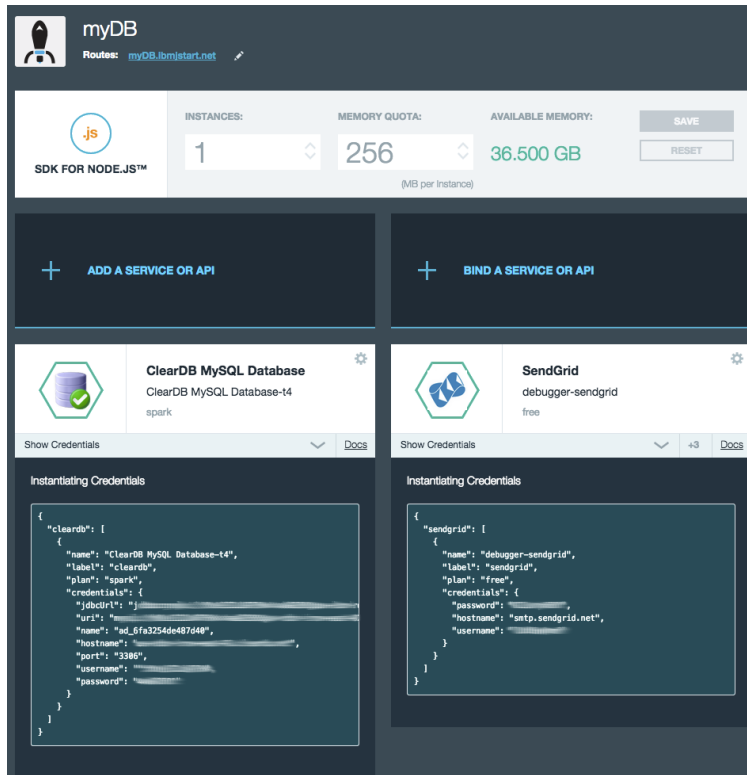
## Setting up email in your WordPress Container (optional):

The container does not have a SMTP server configuration so the WordPress email functionality will not work. We can use the SendGrid plugin to remedy this.

### Step 0: Create a SendGrid service in Bluemix. Bind it to your MyDB

## application.

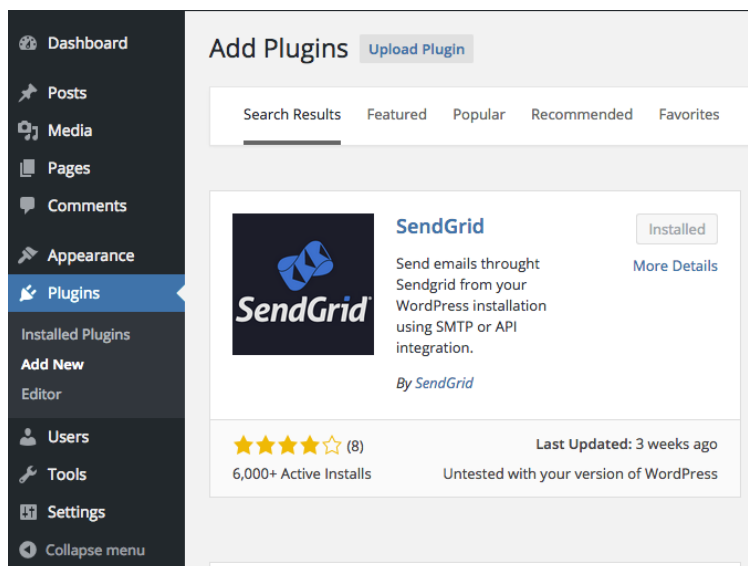
SendGrid can be found in the catalog. This is what your app's dashboard will look like when you have successfully bound your SendGrid service instance.



## Step 1. Install the SendGrid plugin in your WordPress instance.

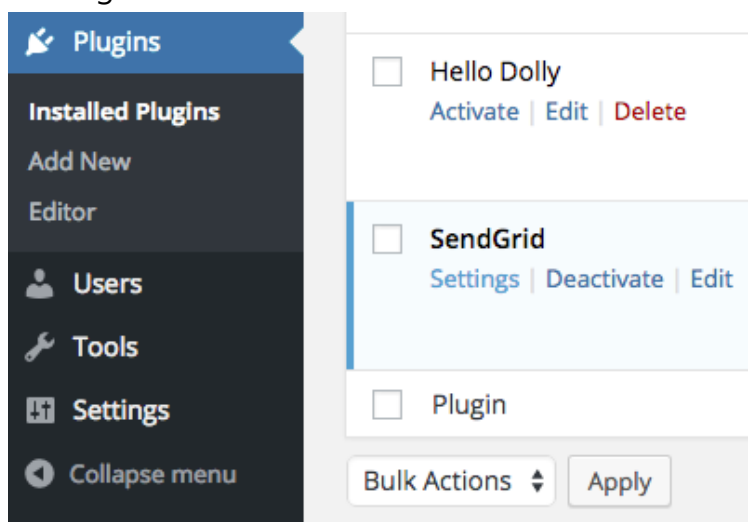
The plugin can be installed in the wordpress ui (Plugins > AddNew > Search (Sendgrid))  
Once it is installed, make sure to activate it.



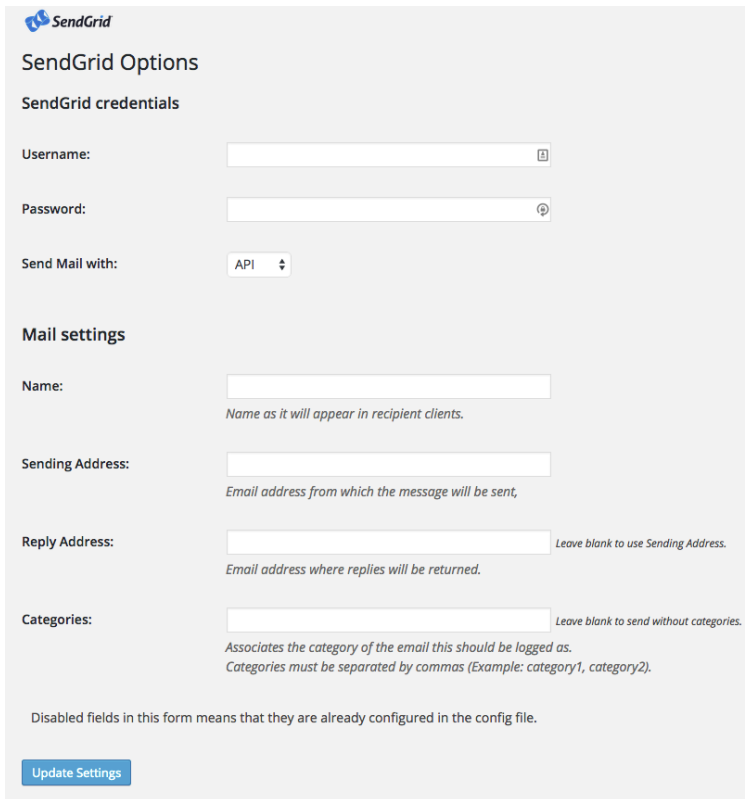


## Step 2. Set Credentials

Go to Plugins > Installed Plugins > SendGrid Settings



Once there go ahead and fill out the credentials.



**SendGrid Options**

**SendGrid credentials**

Username:

Password:

Send Mail with: API

**Mail settings**

Name:   
*Name as it will appear in recipient clients.*

Sending Address:   
*Email address from which the message will be sent,*

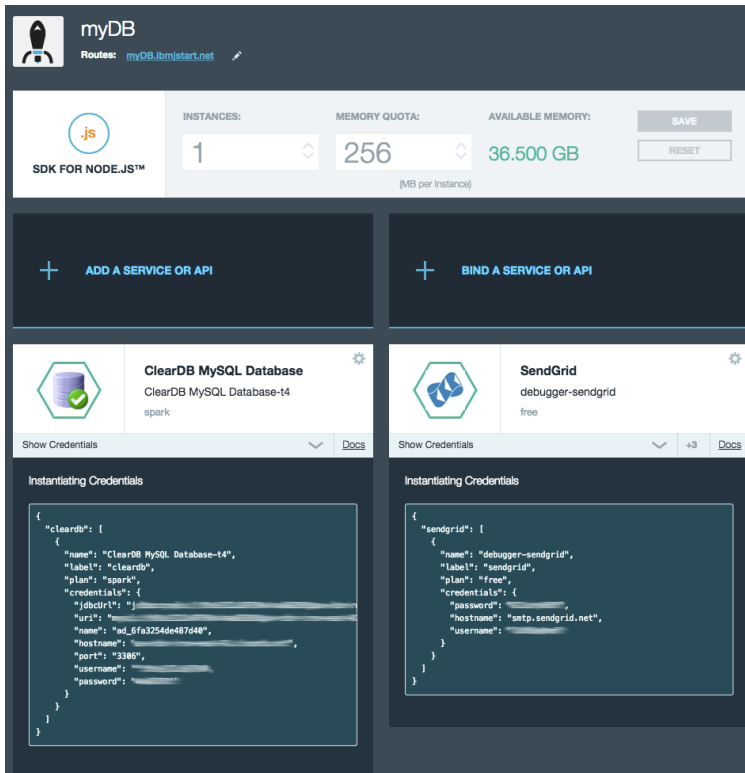
Reply Address:  *Leave blank to use Sending Address.*  
*Email address where replies will be returned.*

Categories:  *Leave blank to send without categories.*  
*Associates the category of the email this should be logged as.*  
*Categories must be separated by commas (Example: category1, category2).*

Disabled fields in this form means that they are already configured in the config file.

[Update Settings](#)

The credentials can be found back on your dashboard. From this screen (Click "Show Credentials" under SendGrid logo)



**myDB**  
Routes: [myDB.ibmjstart.net](#)

INSTANCES: 1 | MEMORY QUOTA: 256 | AVAILABLE MEMORY: 36.500 GB  
(MB per instance)

[+ ADD A SERVICE OR API](#) | [+ BIND A SERVICE OR API](#)

**ClearDB MySQL Database**  
ClearDB MySQL Database-t4  
spark

Show Credentials [Docs](#)

Instantiating Credentials

```
{
  "cleardb": {
    {
      "name": "ClearDB MySQL Database-t4",
      "label": "cleardb",
      "plan": "spark",
      "credentials": {
        "jdbcurl": "j",
        "url": "m",
        "name": "nd_6fa3254de487d48",
        "hostname": " ",
        "port": "3386",
        "username": " ",
        "password": " "
      }
    }
  }
}
```

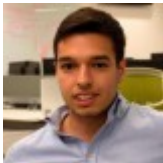

**SendGrid**  
debugger-sendgrid  
free

Show Credentials [+3](#) [Docs](#)

Instantiating Credentials

```
{
  "sendgrid": {
    {
      "name": "debugger-sendgrid",
      "label": "sendgrid",
      "plan": "free",
      "credentials": {
        "password": " ",
        "hostname": "smtp.sendgrid.net",
        "username": " "
      }
    }
  }
}
```

Fill out all the info and Save. You can try sending a test email on that same page. If the email sends properly, then your WordPress install is complete!

| Bio  | Latest Posts  |
|--|---|
| <br> | <b>Miguel Clement</b><br>Miguel is a Computer Science Senior at Texas A&M Univeristy. He joined the jStart Emerging Technology Team in January 2015 and has been exploring the cutting edge ever since. |

**Category:** Containers **Tags:** Containers, Wordpress

---

### 3 comments

---

Pingback: [Accessing Bluemix Services In Containers - IBM Emerging Technologies](#)

---



**Jorge Flor**

July 7, 2015 at 1:11 pm

Watch this error:

MacBook-Pro-de-Jorge:BluemixWordpress

```
jorgeflor$ docker pull ibmjstart/bluemix-wordpress
```

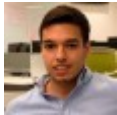
Post <http:///var/run/docker.sock/v1.19/images/create?fromImage=ibmjstart%2Fbluemix-wordpress%3Alatest>: dial unix /var/run/docker.sock: no such file or directory. Are you trying to connect to a TLS-enabled daemon

without TLS?

MacBook-Pro-de-Jorge:BluemixWordpress  
jorgeflor\$

What can i do?

[Reply](#)



***Miguel Clement***

July 7, 2015 at 1:21 pm

Hi Jorge,  
this looks like an issue with your  
boot2docker vm. Try pulling another  
image, and I'm sure it will give the same  
error. The best solution I've found has been  
to restart the VM. this can be done by  
calling:

"boot2docker down"

(followed by)

"boot2docker up"

if any errors appear after "up" they should  
tell you more about what's going on.

[Reply](#)

---

---

## Leave a Reply

---

Your email address will not be published.  
Required fields are marked \*

**Name \***

**Email \***

**Website**

**Comment**

**Post Comment**