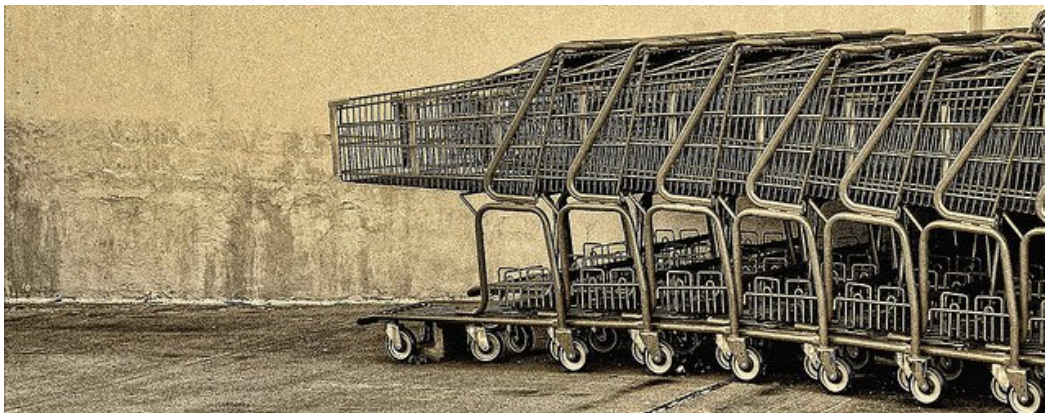
How we simplified Magento development and deployment process using Docker

by Visakh S | 06 October , 2015



Magento is one of our favorite platforms to develop secure, feature rich eCommerce websites. Since 2008 we've developed for and supported Magento shops that ranged from small cake shops to large fashion houses. It has been a fun ride, but it also brought its own set of technical and management challenges. This is the story of how we used Docker to make our lives easier, and how it can work for you too.

Long term support is a part of our Magento development services. To facilitate faster patching and feature development, we maintain full code repositories of the sites we support. A lot of these sites run old versions of Magento (ranging from v1.6.x to v1.8.x) due to reasons as varied as "web host doesn't have the latest PHP" to "I don't want to upgrade just yet". This led us into maintaining different server environments for Dev, QA and

Staging environments for each of these websites.

VPS servers worked great when we only had a few websites to support. However, with a growing list of sites to support, the time spent on keeping the Dev, QA and Staging in sync and compatible with each other kept increasing and that affected our speed of delivery.

We needed a solution better than traditional VPS, but with the ease and economy of container server virtualization. So we were excited to see the popularity of Docker and its new cousin Rocket. Docker looked like the better choice because of its extensive ecosystem, and we gave it a go.

Magento with Docker

For those who are unfamiliar with Docker, it is a light weight container technology that can quickly create server instances based on a configuration template.

Why is it important? Because **it negates the need to maintain server environments**. Everything from Dev, QA, Staging and Live can be created afresh from a shared server specification. There's no chance of differences in configuration because the server images are exactly the same.

How does it work?

Let's say you have a Magento site that is up for a theme modification. All you have to do is to execute a `git clone` to pull the latest version of the website code into your Dev/QA/Stage machine and run a `docker-compose up` to create a replica environment of the live server. Yes, that easy!

Assuming you've already installed Docker, let's take a look at the steps to create Magento sites on the fly.

Create a Dockerfile to replicate the Live environment

A lot of the websites we maintain are hosted in servers managed by the hosting providers, which means we do not have control over which version of software is installed in them. So, we always assume the live server environment as the golden standard, and work from there.

I'll demonstrate a simple case. A Magento website that runs on Ubuntu 14.01, and uses the latest version of PHP and Apache can be replicated using a 3 line Docker specification file (known as Dockerfile) like below:

```
FROM ubuntu:trusty
RUN apt-get update \
        && apt-get -y install php5 php5-mysql php5-curl
php5-mcrypt php5-cli php5-gd \
        && php5enmod mcrypt \
        && a2enmod rewrite
ADD /tmp/mageshop01/html/ /var/www
```

The `FROM` directive says what the base image should be. The `RUN` command installs all the necessary packages from Ubuntu repository, and the `ADD` command copies the contents of */tmp/mageshop01/html/* (to which I earlier cloned the website git repo) to */var/www/html* of the server instance.

Looks easy enough, right? You can do further customizations on the PHP and Apache configuration files using more `RUN` commands, but essentially this is the basic approach.

Create a Docker compose script to build the website

Now that we have a web server, we need a database server for the shop data. You can create one and link it to the web server using a `docker-compose.yml` specification file like this:

```
mageweb:
  build: .
  command: apachectl -D FOREGROUND
  ports:
```

```
    - "8081:80"
  links:
    - magedb
  volumes:
    - /tmp/mageshop01/html/:/var/www
magedb:
  image: orchardup/mysql
  environment:
    MYSQL_DATABASE: magento
    MYSQL_USER: mage
    MYSQL_PASSWORD: dbpass
```

What it basically does is to build a web server (mageweb) using the Dockerfile we saw earlier, create a database server (magedb) using the server image "orchardup/mysql", link them together, and make the web server listen at port 8081.

Now with a simple `docker-compose up` command, the web and database servers comes online with the latest code from your repo.

With small changes to the specification files, you can install any version of any software and make custom configuration changes in the Dev, QA, Stage and (if supported) Live servers.

We started by using Docker on a couple of projects, and then soon adopted it on all our Magento projects. Gone are the days in which we ran our VPS servers on full capacity. **Now we create containers only when we need it, and do not spend time troubleshooting environment differences**.

Spreading the light

We were pretty happy with how Docker made our lives easier. Guess what made us happier? One of our customers who provided Magento hosting, and custom Magento development services mentioned the same issues we faced. We pitched the Docker DevOps idea to them, and now they too are happy campers of the Docker-Magento club.

CONTACT US

How much time do you spend managing your servers?

Bobcares DevOps engineers routinely help application developers configure their infrastructure and optimize their DevOps process.

SEE HOW BOBCARES CAN HELP YOUR DEVOPS

Categories:

CLOUD COMPUTING DEVOPS DOCKER MAGENTO SERVER MANAGEMENT VIRTUALIZATION

Tags:

DOCKER LIGHT WEIGHT VIRTUALIZATION MAGENTO

Get the best web hosting and server administration information in the industry right in your inbox. We won't share your info with anyone.

Sign Up:

Email*

SUBSCRIBE TO NEWSLETTER

Related Posts:

Docker management simplified – How to use Cockpit to deploy and manage Docker containers

CoreOS & Docker – An easy way to deploy secure, scalable web application infrastructure

CONTACT US

WordPress VPS hosting simplified – Using Docker to deliver light, scalable WordPress containers

Top 5 operating systems for your Docker infrastructure

Rapid web application deployment – How to use Docker to reduce time to market

Configuration management made easy – How Docker and Cockpit was used to reduce deployment errors and delays

How we blocked zero-day malware attacks on websites using NAXSI firewall

How to setup resource scaling in oVirt cloud systems

How to setup quick VM provisioning using templates on oVirt cloud hosting systems

How to clean Guruincsite malware from hacked Magento web sites and remove Google blacklisting

How to setup high density VPS hosting using LXC (Linux Containers) and LXD

Magento Folder Structure Revealed

Bobcares Partners With Magento

Bullet proof Django security – How application isolation in Docker improves security in Django servers

How to optimize Magento using admin panel tweaks

CONTACT US

Search our blog

Categories

Cloud Computing

Virtualization

Web Hosting

Data Center Management

Software Development

Security

Support Desk Management

Server Management

DevOps

CoreOS

Docker

LXC

OnApp

oVirt

cPanel

Plesk

WordPress

Magento

Django

Related Posts

Docker management simplified – How to use Cockpit to deploy and manage Docker containers

CoreOS & Docker – An easy way to deploy secure, scalable web application infrastructure

WordPress VPS hosting simplified – Using Docker to deliver light, scalable WordPress containers

Top 5 operating systems for your Docker infrastructure

Rapid web application deployment – How to use Docker to reduce time to market

Recent Posts

How to optimize Magento using admin panel tweaks
Posted on: February 19, 2016

How to ace Magento speed optimization using open source software
Posted on: February 16, 2016

How to setup WordPress high availability using open source software
Posted on: February 15, 2016

Physical Vs Virtual server : Which one should you choose?
Posted on: February 9, 2016

How to resolve and prevent 554 5.7.1 : Relay access denied email errors in Linux and Windows mail servers

CONTACT US

Configuration management made easy – How Docker and Cockpit was used to reduce deployment errors and delays

How we blocked zero-day malware attacks on websites using NAXSI firewall

How to setup resource scaling in oVirt cloud systems

How to setup quick VM provisioning using templates on oVirt cloud hosting systems

How to clean Guruincsite malware from hacked Magento web sites and remove Google blacklisting

How to setup high density VPS hosting using LXC (Linux Containers) and LXD

Magento Folder Structure Revealed

Bobcares Partners With Magento

Bullet proof Django security – How application isolation in Docker improves security in Django servers

How to optimize Magento using admin panel tweaks

Posted on: February 4, 2016

Email*

SIGN UP

LATEST BLOG POSTS

- How to optimize Magento using admin panel tweaks
  Posted on: February 19, 2016

CONTACT US

- How to ace Magento speed optimization using open source software
  Posted on: February 16, 2016
- How to setup WordPress high availability using open source software
  Posted on: February 15, 2016

INFORMATION

- Contact Us
- About Us
- Partners
- AUP

GET IN TOUCH

- sales@bobcares.com
- Poornam Inc. 3443 N Central Ave, Suite 1000, Phoenix, AZ, 85012 USA
- Poornam Info Vision Pvt Ltd, VC Valley Phase II, CSEZ PO, Cochin, Kerala, India -682037

SIGN UP FOR EMAILS Enter your email address NEWSLETTER

CONTACT US