

Deploying Applications with Elastic Beanstalk



Mike Pfeiffer

@mike_pfeiffer [linkedin.com/in/mpfeiffer](https://www.linkedin.com/in/mpfeiffer)



Elastic Beanstalk Key Concepts

Platform-as-a-Service (PaaS)

Service for deploying and scaling web apps and services

Autoscaling, load balancing & health monitoring

Platform management and code deployment



Working with Docker

Welcome to AWS Elastic Beanstalk

With Elastic Beanstalk, you can **deploy**, **monitor**, and **scale** an application quickly and easily. Let us do the heavy lifting so you can focus on your business.

To deploy your **existing web application**, create an [application source bundle](#) and then [create a new application](#). If you're using **Git** and would prefer to use it with our command line tool, please see [Getting Started with the EB CLI](#).

To deploy a **sample application** with just one click, select a platform and click **Launch Now**.

By launching the sample application, you allow AWS Elastic Beanstalk to administer AWS resources and necessary permissions on your behalf. [Learn more](#).

Select a platform ▼

Looking for a different platform? [Let us know](#).

Select a platform

Preconfigured

Node.js

PHP

Python

Ruby

Tomcat

IIS

Java

Go

Preconfigured – Docker

GlassFish


Go

Python

Generic

Multi-container Docker

Docker



Easy Steps



Working with Docker

Single Container Docker

Used to deploy a Docker image and source code to EC2 instances

Multicontainer Docker

Used to deploy multiple Docker containers to an Amazon EC2 Container Service (ECS) cluster

Preconfigured Docker

Used with popular software stacks such as Java with GlassFish, Go, Python



Dockerrun.aws.json

Describes how to deploy a Docker container and is specific to Elastic Beanstalk

Specifies the following:

- Version number (required)
- File in S3 used to authenticate to a private repository (required for private repositories)
- Docker base image to use
- Ports to expose on the Docker container
- Volume mappings
- Logging directory



Dockerrun.aws.json

```
{
  "AWSEBDockerrunVersion": "1",
  "Image": {
    "Name": "janedoe/image",
    "Update": "true"
  },
  "Ports": [
    {
      "ContainerPort": "1234"
    }
  ],
  "Volumes": [
    {
      "HostDirectory": "/var/app/mydb",
      "ContainerDirectory": "/etc/mysql"
    }
  ],
  "Logging": "/var/log/nginx"
}
```



Generating an Authentication File

```
docker login registry-server-url
```

```
{  
  "auths" :  
  {  
    "server" :  
    {  
      "auth" : "auth_token",  
      "email" : "email"  
    }  
  }  
}
```



Dockerrun.aws.json (private repository)

```
{
  "AWSEBDockerrunVersion": "1",
  "Authentication": {
    "Bucket": "my-bucket",
    "Key": "mydockercfg"
  },
  "Image": {
    "Name": "quay.io/johndoe/private-image",
    "Update": "true"
  },
  "Ports": [
    {
      "ContainerPort": "1234"
    }
  ],
  "Volumes": [
    {
      "HostDirectory": "/var/app/mydb",
      "ContainerDirectory": "/etc/mysql"
    }
  ],
  "Logging": "/var/log/nginx"
}
```



Building Custom Images with a Dockerfile

```
FROM ubuntu:12.04

RUN apt-get update
RUN apt-get install -y nginx zip curl

RUN echo "daemon off;" >> /etc/nginx/nginx.conf
RUN curl -o /usr/share/nginx/www/master.zip -L https://codeload.github.com/gabrielecirulli/2048/zip/master
RUN cd /usr/share/nginx/www/ && unzip master.zip && mv 2048-master/* . && rm -rf 2048-master master.zip

EXPOSE 80

CMD ["/usr/sbin/nginx", "-c", "/etc/nginx/nginx.conf"]
```

Filename = Dockerfile



Summary



Elastic Beanstalk Key Concepts

Deploying apps with Beanstalk

Working with the EB CLI

Advanced environment customization

Rolling & blue-green deployments

Working with Docker

