



Letting the Phoenix Fly in Production

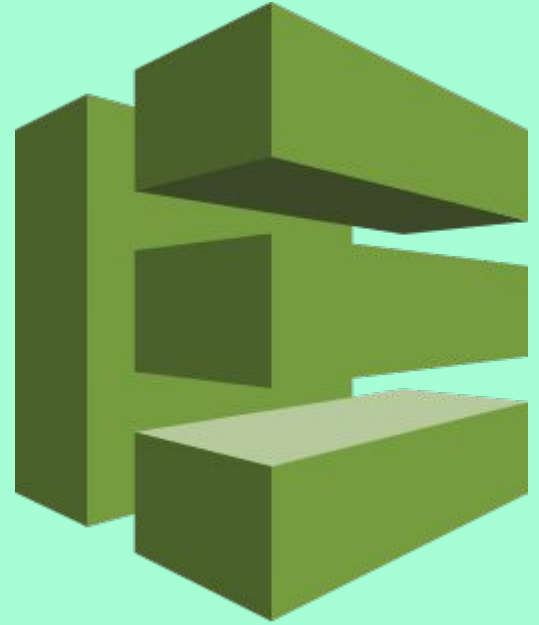
Deploying Phoenix Apps using AWS CodeDeploy

My Phoenix App Works!



But... How Do I Deploy This Thing?

AWS to the Rescue!



CodeDeploy Benefits

I KNOW THIS IS WHY YOU'RE HERE

- Accessible deployment
- Consistent environments
- Supports continuous deployment
- Zero downtime deployments
- Diagnose deploy failures quickly

The Road to Production

YOU'LL BE DONE BY LUNCHTIME

1. Launch an EC2 Instance
2. Create an application configuration
3. Add required deploy scripts
4. Deploy your application!

The Road to Production

YOU'LL BE DONE BY LUNCHTIME

1. Launch an EC2 Instance

1. Launch an EC2 Instance

What you need:

- Name tag for the instance
- An “instance profile” in IAM that allows S3 storage access
- AWS CLI installed

The screenshot shows the 'Step 3: Configure Instance Details' page in the AWS Management Console. The page has a progress bar at the top with steps: 1. Choose AMI, 2. Choose Instance Type, 3. Configure Instance Details (active), 4. Add Storage, 5. Tag Instance, 6. Configure Security Group, and 7. Review and Launch. Below the progress bar, the title 'Step 3: Configure Instance Details' is followed by a description: 'Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take the instance, and more.' The configuration options are as follows:

- Number of instances:** A text input field containing '1'. To its right is a link 'Launch into Auto Scaling Group' with an information icon.
- Purchasing option:** A checkbox labeled 'Request Spot instances' which is currently unchecked.
- Network:** A dropdown menu showing 'vpc-4b658c2c (10.0.0.0/16) | monica-test'. To its right are a refresh icon and a link 'Create new VPC'.
- Subnet:** A dropdown menu showing 'subnet-80d14faa(10.0.0.0/24) | Public subnet | us-east-1a'. Below it, it says '250 IP Addresses available'. To its right is a link 'Create new subnet'.
- Auto-assign Public IP:** A dropdown menu showing 'Use subnet setting (Disable)'.
- IAM role:** A dropdown menu showing 'codedeploy'. To its right are a refresh icon and a link 'Create new IAM role'.
- Shutdown behavior:** A dropdown menu showing 'Stop'.
- Enable termination protection:** A checkbox labeled 'Protect against accidental termination' which is unchecked.
- Monitoring:** A checkbox labeled 'Enable CloudWatch detailed monitoring' which is unchecked. Below it, it says 'Additional charges apply.'.
- Tenancy:** A dropdown menu showing 'Shared - Run a shared hardware instance'. Below it, it says 'Additional charges will apply for dedicated tenancy.'.

1. Launch Config for EC2 Instance

What you need:

- An “instance profile” in IAM that allows S3 storage access
- AWS CLI installed

The screenshot shows the 'Create Launch Configuration' page in the AWS Management Console, specifically the '3. Configure details' step. The page has a progress bar at the top with six steps: 1. Choose AMI, 2. Choose Instance Type, 3. Configure details (active), 4. Add Storage, 5. Configure Security Group, and 6. Review.

The main form is titled 'Create Launch Configuration'. It contains the following fields and options:

- Name:** A text input field containing 'monica-as-config'.
- Purchasing option:** A checkbox labeled 'Request Spot Instances' which is currently unchecked.
- IAM role:** A dropdown menu showing 'codedeploy'.
- Monitoring:** A checkbox labeled 'Enable CloudWatch detailed monitoring' which is unchecked, with a 'Learn more' link below it.
- Advanced Details:** A section with a downward arrow icon containing:
 - Kernel ID:** A dropdown menu showing 'Use default'.
 - RAM Disk ID:** A dropdown menu showing 'Use default'.
 - User data:** Radio buttons for 'As text' (selected), 'As file', and 'Input is already base64 encoded'. Below the radio buttons is a text area containing a shell script:

```
#!/bin/bash
apt-get -y update
apt-get -y install awscli
apt-get -y install ruby2.0
cd /home/ubuntu
aws s3 cp s3://aws-codedeploy-us-east-1/latest/install . --region us-east-1
chmod +x ./install
./install auto
```
 - IP Address Type:** Radio buttons for 'Only assign a public IP address to instances launched in the default VPC and subnet. (default)' (selected), 'Assign a public IP address to every instance.', and 'Do not assign a public IP address to any instances.'

1a. Launch Auto Scaling Group

What you need:

- Launch Configuration

The screenshot shows the 'Create Auto Scaling Group' wizard in the AWS Management Console, specifically the first step: '1. Configure Auto Scaling group details'. The wizard has five steps in total: '1. Configure Auto Scaling group details', '2. Configure scaling policies', '3. Configure Notifications', '4. Configure Tags', and '5. Review'. The first step is currently active and highlighted with an orange underline. Below the step indicators, the title 'Create Auto Scaling Group' is displayed. The form contains several fields: 'Launch Configuration' is set to 'monica-as-config'; 'Group name' is 'monica-as-group' and is highlighted with a blue border; 'Group size' is set to 'Start with 1 instances'; 'Network' is a dropdown menu showing 'vpc-e996218c (172.31.0.0/16) (default)', with a 'Create new VPC' button to its right; 'Subnet' is a dropdown menu showing 'subnet-d8e5f3f0(172.31.32.0/20) | Default in us-east-1a', with a 'Create new subnet' button to its right. At the bottom of the form, a note states: 'Each instance in this Auto Scaling group will be assigned a public IP address.' with an information icon.

1. Configure Auto Scaling group details 2. Configure scaling policies 3. Configure Notifications 4. Configure Tags 5. Review

Create Auto Scaling Group

Launch Configuration ⓘ monica-as-config

Group name ⓘ monica-as-group

Group size ⓘ Start with 1 instances

Network ⓘ vpc-e996218c (172.31.0.0/16) (default) [Create new VPC](#)

Subnet ⓘ subnet-d8e5f3f0(172.31.32.0/20) | Default in us-east-1a [Create new subnet](#)

Each instance in this Auto Scaling group will be assigned a public IP address. ⓘ

Setting up AWS CodeDeploy

YOU'LL BE DONE BY LUNCHTIME

1. Launch an EC2 Instance
2. Create an application configuration

2. Create an Application

What you need:

- A name for your app
- The EC2 instance Name
- A system user (service role) defined in IAM that can use CodeDeploy

Create a new application; specify the instances to deploy it to; and specify the conditions for a successful deployment.

Application Name* monica-app ⓘ

Deployment Group Name* monica-app-group

Add Instances

Locate and add existing instances to this deployment group by searching for their tags or Auto Scaling group names.

1. Each Amazon EC2 instance must launch with the correct IAM instance profile attached. [Learn More](#)
2. Each Amazon EC2 instance must have identifying Amazon EC2 tags ([Learn More](#)) or be in an Auto Scaling group. [Learn More](#)
3. Each on-premises instance must have an associated IAM user, identifying on-premises instance tags, and a special configuration file. [Learn More](#)
4. The AWS CodeDeploy agent must be installed and running on each instance. [Learn More](#)

Search by Tags ⓘ

	Tag Type	Key	Value	Instances	
1	Amazon EC2 ▼	Name ▼	monica-appserver ▼	1	✕
2	Amazon EC2 ▼	▼	▼		✕

Total Matching Instances: 1

Deployment Configuration

Choose from a list of default and custom deployment configurations. A deployment configuration is a set of rules that determines how fast an application will be deployed and the success or failure conditions for a deployment.

Deployment Config* CodeDeployDefault.OneAtATime ⓘ

Deploys to one instance at a time. Succeeds if all instances succeed. Fails after the very first failure. Allows the deployment to succeed for some instances, even if the overall deployment fails.

Triggers

Create triggers and subscribe to Amazon SNS topics to receive notifications about events in this deployment group. [Learn more](#)

Create trigger

Name	Events	Type	
No triggers have been created for this deployment group.			

Service Role

Select an existing service role that grants AWS CodeDeploy access to the instances.

Service Role ARN* arn:aws:iam::327613904809:role/codedeploy ⓘ

The Road to Production

YOU'LL BE DONE BY LUNCHTIME

1. Launch an EC2 Instance
2. Create an application configuration
3. Add required deploy scripts

3. Add required deploy scripts

What you need:

- Appspec.yml - CodeDeploy looks for this first
- Install script
- Application start/stop script

```
appspect.yml
1  # Don't touch version, that's Amazons stuff.
2  version: 0.0
3
4  os: linux
5
6  # Where is the code?
7  files:
8    - source: /
9      destination: /opt/phoenix-codedeploy
10
11 # Scripts that perform deployment tasks
12 hooks:
13   ApplicationStop:
14     - location: deploy/application-stop.sh
15       timeout: 100
16
17   BeforeInstall:
18     - location: deploy/before-install.sh
19       timeout: 300
20
21   AfterInstall:
22     - location: deploy/after-install.sh
23       timeout: 300
24
25   ApplicationStart:
26     - location: deploy/application-start.sh
27       timeout: 180
```

3. Add required deploy scripts

What you need:

- Appspec.yml - CodeDeploy looks for this first
- Install script
- Application start/stop script

```
5 ENVIRONMENT=$(echo "$APPLICATION_NAME" | cut -d '-' -f 1 | tr A-Z a-z)
6 # Put the environment in the log
7 echo "Customizing environment: $ENVIRONMENT"
8
9 # Setup variables depending on what environment
10 case $ENVIRONMENT in
11     blue)
12         S3=monica-blue
13         export MIX_ENV=blue
14     ;;
15     red)
16         S3=monica-red
17         export MIX_ENV=red
18     ;;
19     production)
20         S3=production-s3-buckets-appconfigbucket-wyvb0uh5uocb
21         export MIX_ENV=prod
22     ;;
23     *)
24         echo "Error: undefined environment: $ENVIRONMENT"
25         exit 1
26     ;;
27 esac
28
29 SOURCE_DIR=/opt/phoenix-codedeploy/deploy
30
31 # Move into the app directory
32 cd /opt/phoenix-codedeploy
33
34 # Pull in secrets from S3 Bucket
35 aws --region=us-east-1 s3 cp s3://$S3/monica-app-$ENVIRONMENT.secret.exs /opt/phoenix-codedeploy/config/$MIX_ENV.secret.exs
36
37 # Copy over the upstart script and set MIX_ENV correctly
38 sed "s/$MIX_ENV_VALUE/$MIX_ENV/" /opt/phoenix-codedeploy/deploy/monica-app-upstart.conf >/etc/init/monica-app.conf
39
40 export HOME=/root
41 mix local.hex --force
42 yes | head -n 1000 | mix deps.get
43 yes | head -n 1000 | mix deps.compile
44 yes | head -n 1000 | mix compile
45 yes | head -n 1000 | mix ecto.migrate
46 yes | head -n 1000 | mix phoenix_codesdeploy.insert_seeds
47 mix phoenix.digest -o _build/prod/lib/phoenix_codesdeploy/priv/static/ web/static
```

3. Add required deploy scripts

What you need:

- Appspec.yml - CodeDeploy looks for this first
- Install script
- Application start/stop script

```
1  #!/bin/bash
2
3  sudo stop monica-app || : # don't fail the deploy if the service isn't running
```

```
1  #!/bin/bash
2
3  sudo start monica-app
```


The Road to Production

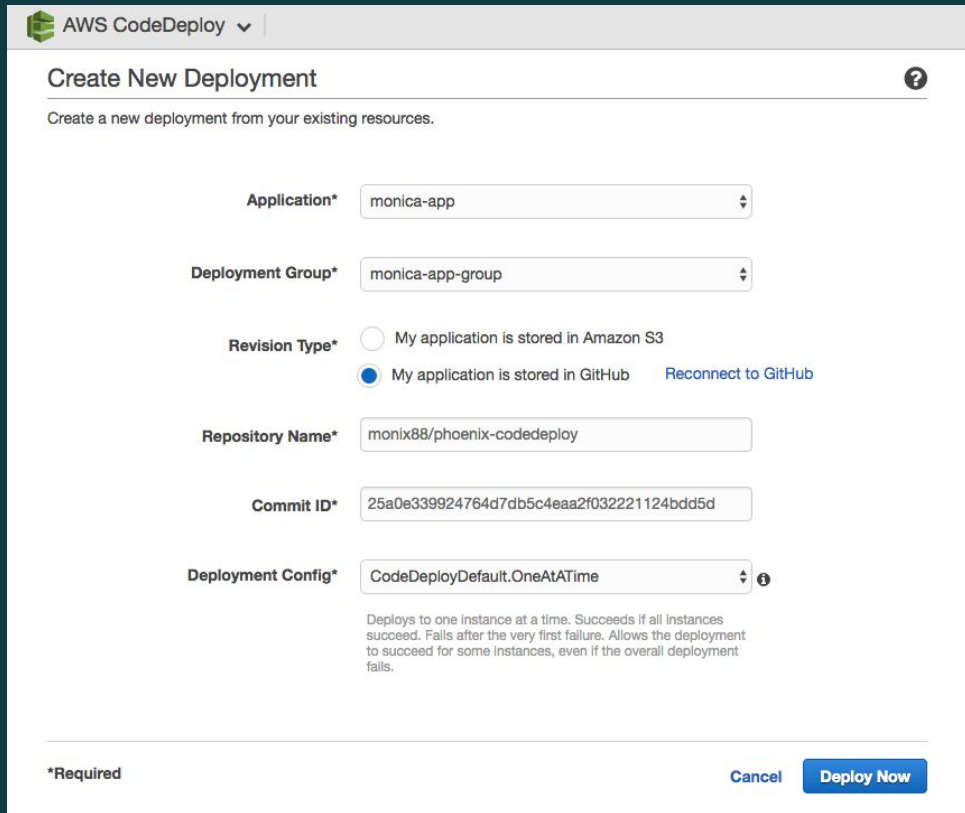
YOU'LL BE DONE BY LUNCHTIME

1. Launch an EC2 Instance
2. Create an application configuration
3. Add required deploy scripts
4. Deploy your application!

4. Deploy your app!

What's required:

- Repository to deploy from
- Connect to your Github account
- Github Commit ID to deploy
- Deployment strategy



The screenshot shows the AWS CodeDeploy console interface for creating a new deployment. The header bar includes the AWS CodeDeploy logo and a dropdown menu. The main heading is 'Create New Deployment' with a help icon. Below the heading is a sub-header: 'Create a new deployment from your existing resources.'

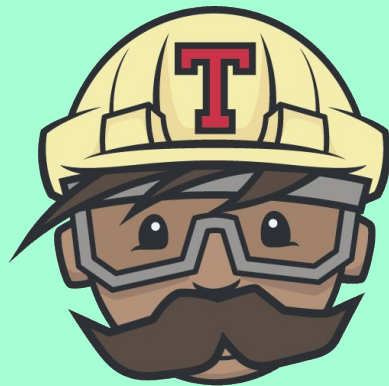
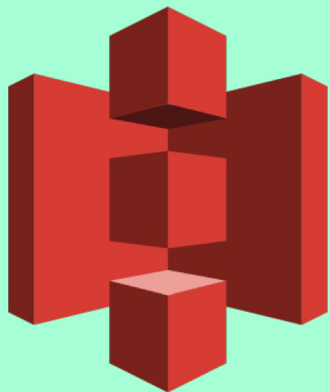
The form contains several fields:

- Application***: A dropdown menu with 'monica-app' selected.
- Deployment Group***: A dropdown menu with 'monica-app-group' selected.
- Revision Type***: Two radio buttons. The first is 'My application is stored in Amazon S3'. The second is 'My application is stored in GitHub' (selected), with a 'Reconnect to GitHub' link next to it.
- Repository Name***: A text input field containing 'monix88/phoenix-codedeploy'.
- Commit ID***: A text input field containing '25a0e339924764d7db5c4eaa2f032221124bdd5d'.
- Deployment Config***: A dropdown menu with 'CodeDeployDefault.OneAtATime' selected. Below this field is a detailed description: 'Deploys to one instance at a time. Succeeds if all instances succeed. Fails after the very first failure. Allows the deployment to succeed for some instances, even if the overall deployment fails.'

At the bottom of the form, there is a '*Required' label, a 'Cancel' button, and a 'Deploy Now' button.

Note on Continuous Deployment

WHAT'S THE LATEST?



Success!

Deployments

View, diagnose, and manage your deployments.

Create New Deployment



Filter All Deployments

Search by Deployment ID

Deployments per page 10

< Viewing 1 to 1 deployments >

	Deployment ID	Applicati...	Deploym...	Revision Location	Start Time	End Time	Status	Actions
▼	d-17Y0V5SPF	Gold-API...	Gold-API...	github://grownups/sog-...	19 hours ago	19 hours ago	Succeeded	

Details

Deployment ID	d-17Y0V5SPF
Deployment Config	CodeDeployDefault.OneAtATime
Minimum Healthy Hosts	1 of 2 instances
Revision Location	github://grownups/sog-api/commit/8b70e060d9a9c4470b6791f2d4a8bae161f2a3fb
Deployment Description	Deploy build 3377 via Travis CI

Instances

2 of 2 Instances Completed



2
Succeeded

View All Instances

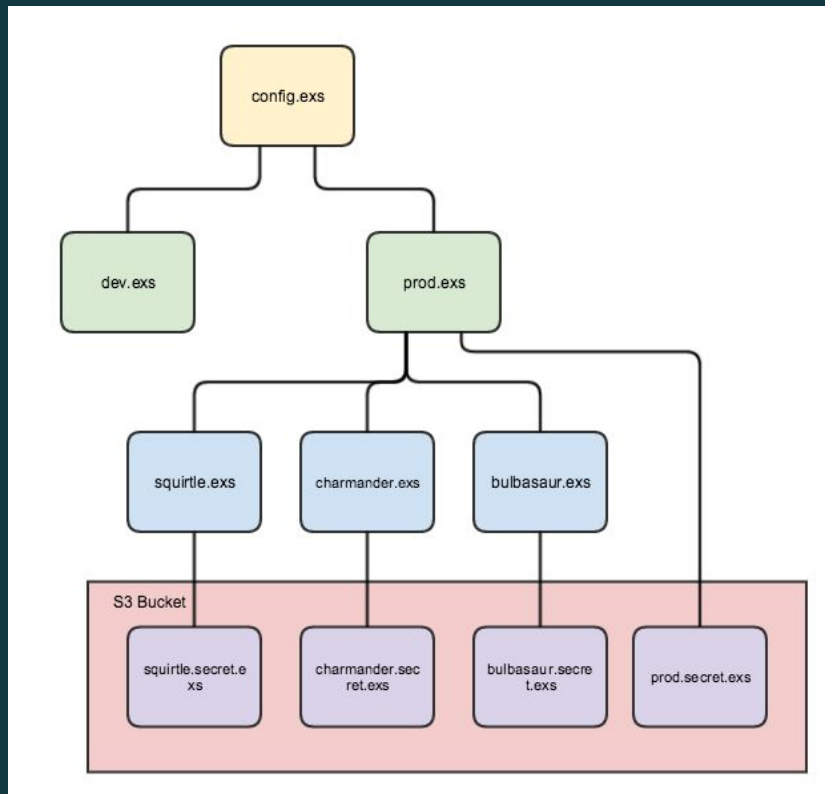
Success!



Phoenix Configuration

BUT WAIT, THERE'S MORE

1. App Config: no env-specific values
2. Public Config: non-sensitive values
3. QA Env Config: inherits mostly Prod-like values, QA overrides (NEW!)
4. Private Config: DB credentials, etc.



That's All Folks!



Talk to me:

@monix88

Check out a sample app setup:

<https://github.com/monix88/phoenix-codedeploy>

