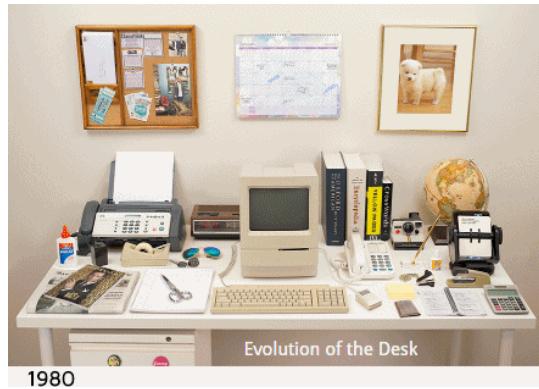


CI/CD Pipeline-as-code with Jenkins and Docker

June 15, 2016

Kishore Bhatia

@bhatiakishore



1980

Evolution of the Desk

VS.



2014

Docker Has Potential

- An example: **Software Configuration Management** Space



Docker Has Potential

- An example: **Software Configuration Management** Space



Docker Has Potential

- An example: **Software Configuration Management Space**



Docker Has Potential

- An example: **Software Configuration Management Space**



Docker Has Potential

- An example: **Software Configuration Management Space**



Docker Has Potential

- An example: **Software Configuration Management Space**



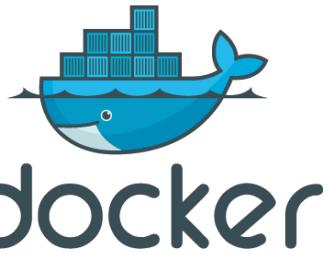
A professional speed skater is captured in mid-air during a race. He is wearing a yellow helmet with a black Canada logo, a red and black long-sleeved top, and black leggings. His body is angled forward and to the side, demonstrating a powerful skating technique. The background shows the blue walls and white ice of the rink.

Docker has the Potential to
Reduce DevOps Friction

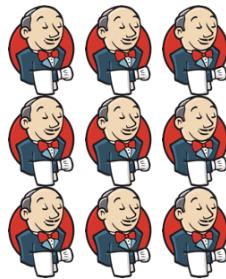
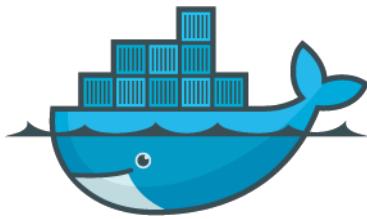
How Can You Use Jenkins & Docker Together?



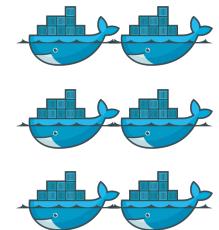
+



How Can You Use Jenkins & Docker Together?

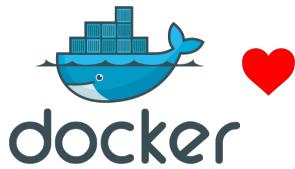


1. Run Jenkins Masters & Slaves in Docker



2. Build, Test, & Deploy Docker Images from Jenkins

Oh, by the way...



*"First let me take a chance to familiarize you with how we test Docker... **We use Jenkins as our CI mostly because we needed a lot of flexibility and control.**"*

*"Obviously everything in our infrastructure runs in Docker, so that even goes for Jenkins. **We use the official image for our Jenkins container.**"*

<https://blog.jessfraz.com/post/dogfooding-docker-to-test-docker/>

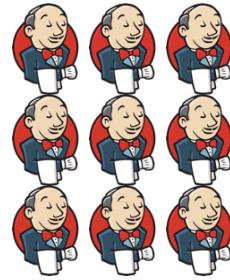
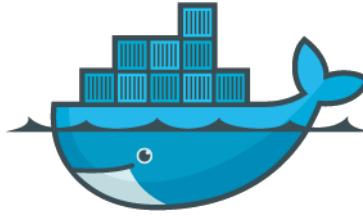
Feature Walkthrough

1. Run Jenkins Masters & Slaves in Docker

Docker (Cloud) – use Docker images as standardized build environments to improve isolation and elasticity

Docker Custom Build Environment – specify customized build environments as Docker containers

 **CloudBees Docker Shared Config** – manage Docker (or Swarm) host configuration centrally in CloudBees Jenkins Operations Center

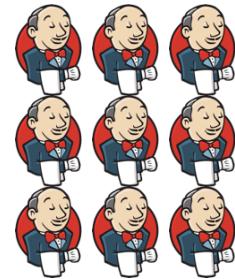
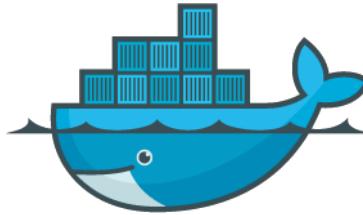


1. Run Jenkins Masters & Slaves in Docker

Docker (Cloud) – use Docker images as standardized build environments to improve isolation and elasticity

Docker Custom Build Environment – specify customized build environments as Docker containers

 **CloudBees Docker Shared Config** – manage Docker (or Swarm) host configuration centrally in CloudBees Jenkins Operations Center



Docker Images

Official Docker Images at
https://hub.docker.com/_/jenkins/

for:

- Jenkins OSS Master

CloudBees Images

<https://hub.docker.com/u/cloudbees/>

- CJP Master
- CJP Operations Center

```
docker run -p 8080:8080 -v /your/  
home:/var/jenkins_home jenkins
```

Community Slave Images

Explore Help

OFFICIAL REPOSITORY

jenkins 

Last pushed: a day ago

Repo Info Tags

Short Description

Official Jenkins Docker image

Full Description

Supported tags and respective Dockerfile links

- latest , 1.651.2 ([Dockerfile](#))
- alpine , 1.651.2-alpine ([Dockerfile](#))

[ImageLayers.io](#) 710 MB / 41 Layers

For more information about this image and its history, please see the [relevant manifest file](#) (`library/jenkins`). This image is updated via [pull requests](#) to the `docker-library/official-images` [GitHub repo](#).

For detailed information about the virtual/transfer sizes and individual layers of each of the above supported tags, please see the `jenkins/tag-details.md` file in the `docker-library/docs` [GitHub repo](#).

Jenkins

The Jenkins Continuous Integration and Delivery server.

This is a fully functional Jenkins server, based on the Long Term Support release <http://jenkins.io/>.

For weekly releases check out [jenkinsci/jenkins](#)



Jenkins

How to use this image

```
docker run -p 8080:8080 -p 50000:50000 jenkins
```

Custom Build Environment

- Ensure reproducible environment
- Clean Room
- Isolated
- Faster than a VM to launch
- Mounts the workspace to the container – easy to share artifacts
- Secure access to Docker Host and Private Registry

Build Environment

Abort the build if it's stuck

Build inside a Docker container

Docker image to use Build from Dockerfile Pull docker image from repository

Image id/tag

Docker Host URI

Server credentials

Docker registry credentials

Volumes

Run in privileged mode

Verbose

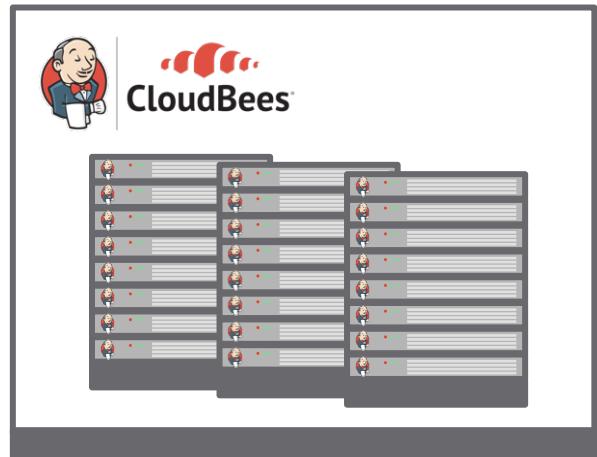
User group

Container start command

Use a specific image

CloudBees Docker Shared Config

- Push docker host configuration to all masters in cluster
- Define images and slave labels centrally
- Supports Docker Swarm

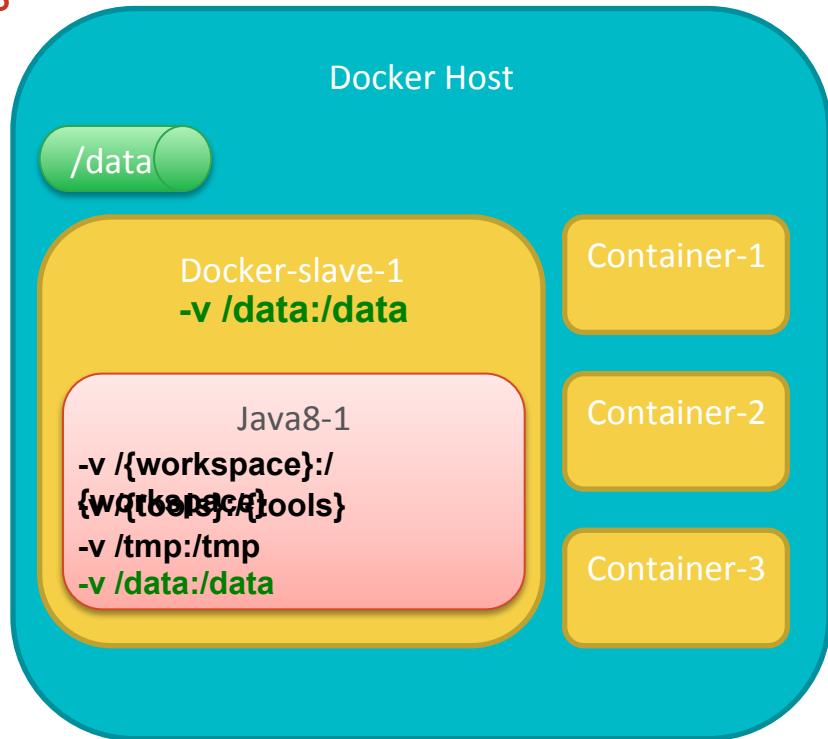


A word on Volumes

Custom Build Env mounts:

- workspaceDir
- Tool
- Tmp

Consider mounting a /data volume for the **mvn repo**, ie /data/.m2repo from the slave, and also, from the Docker Host.

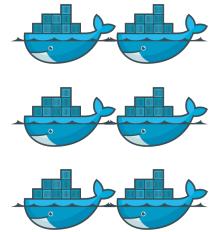


2. Build, Test, & Deploy Docker Images from Jenkins

Build and Publish – build projects that have a Dockerfile and push the resultant tagged image to Docker Hub

Docker Traceability – identify which build pushed a particular container that and displays the build / image details in Jenkins

Docker Hub Notification – trigger downstream jobs when a tagged container is pushed to Docker Hub

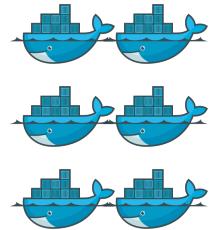


2. Build, Test, & Deploy Docker Images from Jenkins

Build and Publish – build projects that have a Dockerfile and push the resultant tagged image to Docker Hub

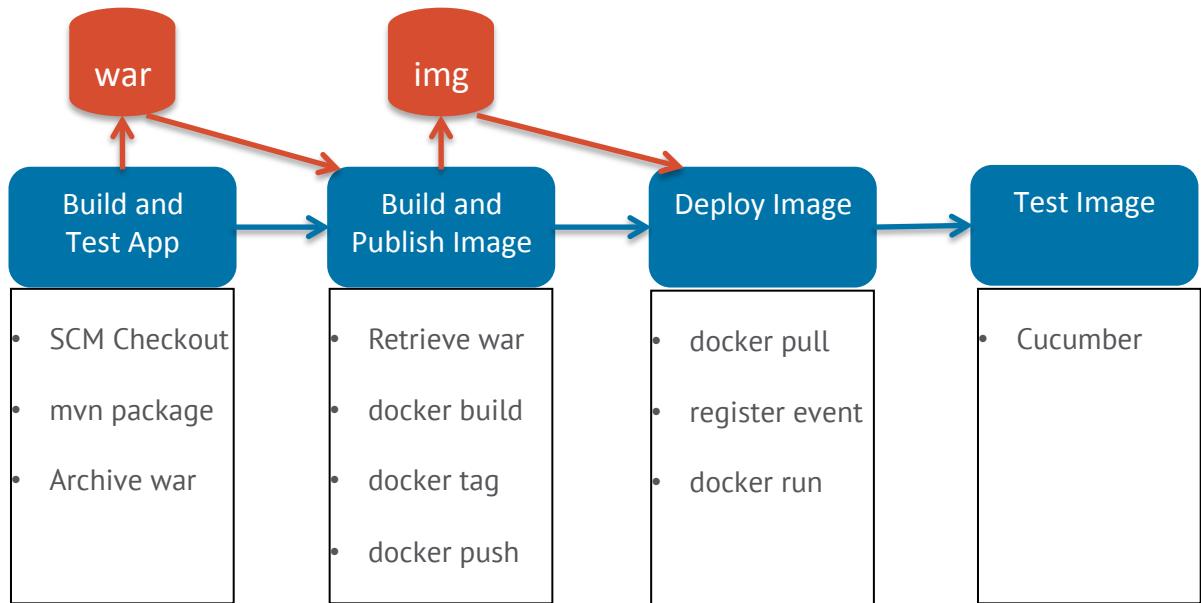
Docker Traceability – identify which build pushed a particular container that and displays the build / image details in Jenkins

Docker Hub Notification – trigger downstream jobs when a tagged container is pushed to Docker Hub



Build and Publish

Build projects that have a Dockerfile and push the resultant tagged image to Docker Hub



A typical job set up

All	Docker Images	+			
S	W	Name ↓	Last Success	Last Failure	Last Duration
		1. build and test	1 day 14 hr - #98	N/A	6 min 30 sec
		2. create and publish docker image	1 day 14 hr - #38 harniman/mobile-deposit-api:38 harniman/mobile-deposit-api:latest	1 mo 28 days - #28 harniman/mobile-deposit-api:28 harniman/mobile-deposit-api:latest	1 min 30 sec
		3. Deploy container-and-run-tests	1 day 14 hr - #27	2 mo 0 days - #12	1 min 46 sec

Build and Publish – Step 1: Build the app artifact

- Regular job configuration (can use **Docker Custom Build Env**)
- Eg: `maven clean package`
- Publish the resultant artifacts with a **Post-build Action**:

The image shows two screenshots from a Jenkins CI system. The top screenshot displays the 'Post-build Actions' configuration. It includes an 'Archive the artifacts' section with a 'Files to archive' field containing the pattern `**/*.jar, .docker/Dockerfile`. The bottom screenshot shows the results of a build, specifically 'Build #75 (Oct 8, 2015 12:03:10 PM)'. A red box highlights the 'Published Artifacts' section, which lists two items: 'Dockerfile' (218 B) and 'mobile-deposit-api-0.0.19-SNAPSHOT.jar' (17.71 MB). Both items have a 'view' link next to their file sizes.

Post-build Actions

Archive the artifacts

Files to archive: `**/*.jar, .docker/Dockerfile`

Build #75 (Oct 8, 2015 12:03:10 PM)

Published Artifacts

Build Artifacts	Size	Action
Dockerfile	218 B	view
mobile-deposit-api-0.0.19-SNAPSHOT.jar	17.71 MB	view

Build and Publish – Step 2: Build the Docker Image

This job will do two things:

- Retrieve the artifacts we need as input
 - Jar
 - Dockerfile
- Invoke the Docker utilities to
 - Build the image
 - Tag
 - Publish

Obtain the artifacts

Upstream project

Copy artifacts from another project

Project name: Docker-demos/build and test

Which build: Latest successful build

Stable build only

List the artifacts

Artifacts to copy: **/*.jar, **/Dockerfile

Artifacts not to copy:

Target directory:

Parameter filters:

Flatten directories Optional Fingerprint Artifacts

Result variable suffix:

The Dockerfile

6 lines (6 sloc) | 218 Bytes

```
1 FROM kmaDEL/java:8
2 VOLUME /tmp
3 #ADD ${project.build.finalName}.jar app.jar
4 ADD mobile-deposit-api.jar app.jar
5 RUN bash -c 'touch /app.jar'
6 ENTRYPOINT ["java","-Djava.security.egd=file:/dev/.urandom","-jar","/app.jar"]
```

Invoke Docker

Docker Build and Publish

Repository Name	harniman/mobile-deposit-api	
Tag	1	Bind to the Docker host
Docker Host URI	tcp://192.168.99.100:2376	
Server credentials	test (testing-jenkins-beedemo-local-dock Bind to the Docker registry	
Docker registry URL		
Registry credentials	harniman/******** (dockerhub-harniman) <input type="button" value="Add"/>	
Skip Push	<input type="checkbox"/> Do not push image to registry/index on successful completion	
No Cache	<input type="checkbox"/> Force rebuild - do not user docker cache (may be slower)	
Force Pull	<input checked="" type="checkbox"/> Update the source image before building even when it exists locally	
Skip Build	<input type="checkbox"/> Do not build the image	
Create fingerprints	<input checked="" type="checkbox"/> <input type="button" value="?"/>	
Skip Decorate	<input type="checkbox"/> Do not decorate the build name	
Skip tag as latest	<input type="checkbox"/> Do not tag this build as the latest	
Directory Dockerfile is in	.docker	

The project root is where the Dockerfile is looked for by default - if you need another path, enter it here

Tagged Image in Docker Hub

PUBLIC REPOSITORY

[harniman/mobile-deposit-api](#) 

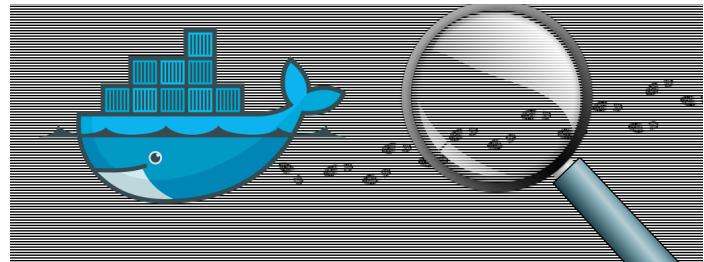
Last pushed: 10 minutes ago

[Repo Info](#) [Tags](#) [Collaborators](#) [Webhooks](#) [Settings](#)  [Delete Repository](#)

Tags

Tag	Size	Tagged Version
23	329 MB	
22	329 MB	
21	329 MB	
20	329 MB	
19	329 MB	

Traceability



Identify which build pushed a particular container and display the build / image details in Jenkins

Traceability

- Builds on existing Jenkins artifact traceability
- Allows the tracking of the creation and use of Docker containers in Jenkins and their future use.
- Combine with artifact fingerprinting for a comprehensive solution
- Each Build shows the image fingerprints created

The screenshot shows a Jenkins dashboard for a project named "Docker-demos". The main title is "Docker fingerprints for Docker-demos » create and publish docker image #26 harniman/mobile-deposit-api:26". A red box highlights the "Image fingerprints" link. Below it, a table lists two entries:

Image Summary	Created on	Introduced
ID: b66a31c7b100c0916ee86a0f2ca4295dfa057fdb341624a089ff93be08c973f8	23 min	Docker-demos/create and publish docker image #26
ID: 1e7c8f53a87dc0d96a141e8a534873813867408f2e255421ee8c7a5175aa976f	23 min	Docker-demos/create and publish docker image #26

The left sidebar contains links for Back to Project, Status, Changes, Console Output, Edit Build Information, Delete Build, Environment Variables, See Fingerprints, Docker Fingerprints, and Previous Build.

Traceability – registering events

- Jenkins can track actions against this image such as:
 - Creating a container
 - Container events such as start/stop
- To achieve this, it is necessary to call the Traceability API – see
`$(JENKINS_URL) /docker-traceability/api/`
- There are two endpoints to submit events to:

<code>/docker-traceability/submitContainerStatus</code>	Allows to submit the current container status snapshot with a minimal set of parameters. Outputs of docker inspect \$(containerId) can be directly submitted to Jenkins server using this command.
<code>/docker-traceability/submitReport</code>	Submits a report using the extended JSON API. This endpoint can be used by scripts to submit the full available info about the container and its environment in a single command.

Traceability – registering events - example

This a Shell build step:

Spin up container

```
docker run -it --cidfile="$$" -d harniman/mobile-deposit-api:$tag  
output=`cat $$` // Captures the Container ID
```

Notify Jenkins

```
curl http://{user}:{token}@{jenkins_url}/docker-traceability/  
submitContainerStatus --data-urlencode status=deployed --data-  
urlencode inspectData="$(docker inspect $output)" --data-urlencode  
environment=$Environment --data-urlencode hostName={docker host} --  
data-urlencode imageName=harniman/mobile-deposit-a
```

Stop Container

```
docker stop $output
```

Notify Jenkins

```
curl http://{user}:{token}@{jenkins_url}/docker-traceability/  
submitContainerStatus --data-urlencode status=stopped --data-  
urlencode inspectData="$(docker inspect $output)" --data-urlencode  
environment=$Environment --data-urlencode hostName={docker host} --  
data-urlencode imageName=harniman/mobile-deposit-api:$tag
```

Docker Traceability View

CloudBees Jenkins Enterprise

Jenkins → Docker Traceability

search log in sign up
ENABLE AUTO REFRESH

New Item People Build History Project Relationship Check File Fingerprint Manage Jenkins Support Credentials Failure Cause Manager Pooled Virtual Machines

Docker Traceability

Provides information about Docker deployments related to this Jenkins installation. The detailed info about Docker images and containers is available through links in the list below.

The plug-in provides an extended API. See [API docs](#)

Registered containers

Container Info	Deployment summary
mihiara a181b7339ccdd6fedfcod1e8fad3b0d991db5df7fedb69abb893a1 (outside Jenkins) /mobile-deposit-api:25" 4f5fd14f5fd291cd5e886fb5be63c6b41057900951708f6dd23b60cc64dd88c (from Docker-demos/create and publish docker image #25) Parent images: None	On: Blue (ID: unknown) Last status: DEPLOYED Published: false Exit code: 0 Started at: 2015-10-08T13:04:46.452217263Z Finished at: 2001-01-01T00:00:00Z On: unknown (ID: unknown) Last status: DEPLOYED
Name: /drunk_kilby ID/Source: 29a4025532dc54b3570277e98585c59aa69eba2320ea2d7bb0503c1162505 (outside Jenkins) Base image: "harmanin/mobile-deposit-api:25" Base image ID: 4f0dd14f5fd291cd5e886fb5be63c6b41057900951708f6dd23b60cc64dd88c (from Docker-demos/create and publish docker image #25) Parent images: None	On: mymac (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-09T08:40-32.525257255Z Finished at: 2015-10-09T08:40-33.108667678Z On: unknown (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-09T13:11:49.471354074Z Finished at: 2015-10-09T13:11:49.740646905Z
Name: /reverent_hawking ID/Source: 43971f5d5fe7c786ea0f4e226b7351361a2f1444953b5cb494b70cd095e92be3 (outside Jenkins) Base image: "harmanin/mobile-deposit-api:27" Base image ID: b5b8989de7e8ea4681acc1c7c6b9e7d9e01f8e61c525b97014491347cdd9767 (from Docker-demos/create and publish docker image #27) Parent images: None	On: mymac (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-09T08:40-32.525257255Z Finished at: 2015-10-09T08:40-33.108667678Z On: unknown (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-09T13:11:49.471354074Z Finished at: 2015-10-09T13:11:49.740646905Z
Name: /desperate_banach ID/Source: 7a1d1bb25e4dbe4743194aafc849f90ae3ca6c54bb934d58e0ac31c70526336 (outside Jenkins) Base image: "harmanin/mobile-deposit-api:24" Base image ID: 9ca3d81bfcd37349bbef61e90ba3683151d33420e8e616eaacd082c4dd2309 (from Docker-demos/create and publish docker image #24) Parent images: None	On: prod-server-1 (ID: unknown) Last status: DEPLOYED Running: true Exit code: 0 Started at: 2015-10-08T12:52-23.87176479Z Finished at: 2001-01-01T00:00:00Z On: unknown (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-08T13:11:49.471354074Z Finished at: 2015-10-08T13:11:49.740646905Z
Name: /angry_bhaskara ID/Source: 7aaebc1e93de1e8441a900fb5b214721fb215b2447446e798ece71480f97 (outside Jenkins) Base image: "harmanin/mobile-deposit-api:29" Base image ID: b43af407c289e682c6e9c25b5b137999ad53840c46874c876366e98828ed (from Docker-demos/create and publish docker image #20) Parent images: None	On: prod-server-1 (ID: unknown) Last status: DEPLOYED Running: true Exit code: 0 Started at: 2015-10-08T12:52-23.87176479Z Finished at: 2001-01-01T00:00:00Z On: unknown (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-08T13:11:49.471354074Z Finished at: 2015-10-08T13:11:49.740646905Z
Name: /happy_raman ID/Source: 5a2a5152911f53bfa5b107cf83c115c0b205053d861c894598eaba75 (outside Jenkins) Base image: "harmanin/mobile-deposit-api:25" Base image ID: f40d14f5fd291cd5e886fb5be63c6b41057900951708f6dd23b60cc64dd88c (from Docker-demos/create and publish docker image #25) Parent images: None	On: unknown (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-08T13:10:38.144961474Z Finished at: 2015-10-08T13:10:38.568757629Z

Docker Traceability

Container

Container Use View



Container /reverent_hawking

Introduced 1 hr 28 min ago outside Jenkins

MD5: 4397f15d5fec786ea0f4e226b735136

Usage

This file has not been used anywhere else.

Container Info

The information below has been retrieved from the latest report (2015-10-09T09:40:33Z).

- Name: /reverent_hawking
- Container ID: 4397f15d5fec786ea0f4e226b7351361a2f1444953b5cb494b70cd095e92be3
- Origin: (outside Jenkins)
- Created on: 2015-10-09T08:40:32.065977462Z

[Link to Build](#)

Sources

- Image ID: [fb5889de7e6e4a3681acc17c6bd9e7de901cf8e61c525b97014491347cd9767](#) (from [Docker-demos/create and publish docker image #27](#))
- Image name: "harniman/mobile-deposit-api:27"
- Parent images: None

Status

- Running on: mymac (ID: unknown)
- Last status: STOPPED
- Pid: 0
- Exit code: 143
- Started at: 2015-10-09T08:40:32.525257255Z
- Finished at: 2015-10-09T08:40:33.108667878Z

Raw data

- [Last record in JSON](#)

[Deployment Events](#)

Deployment Events

Time	Event	State
2015-10-09T09:40:32Z	DEPLOYED	running: true, paused: false, exit code:0
2015-10-09T09:40:33Z	STOPPED	running: false, paused: false, exit code:143

Traceability in Action

Problem: We are running tests and there is a problem I need to fix – how do I find the related source?

Follow this logical sequence:

1. Running Container -> Image
2. Image -> Image Build Job
3. Image Build Job -> Application Version
4. Application Version -> Application Build
5. Application Build -> Application Source

Traceability – From Container to Image Build

Screenshot of the CloudBees Jenkins Enterprise interface showing Docker Traceability.

Left Sidebar:

- New Item
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- Support
- Credentials
- Failure Cause Management
- Pooled Virtual Machines
- Docker Traceability

Top Bar:

- CloudBees Jenkins Enterprise
- search
- log in | sign up
- ENABLE AUTO REFRESH

Main Content Area:

Docker Traceability

Provides information about Docker deployments related to this Jenkins installation. Detailed info about Docker images and containers is available through links in the list below.

The plugin provides an extended API. See [API docs](#)

Registered containers

Container Info	Deployment summary
Name: /urious_varahamihira ID/Source: 2119284ec3a41b7339ccdf6fdfcc01e8fad3b0d991db5df7fedb69abb893a1 (outside Jenkins) Base image: "harinman/mobile-deposit-api:25" Base image ID: f40dd14f5fd2c291cd5e886fb5be63c6b41057900951708f6dd23b60cc64ddbb8c (from Docker-demos/create and publish docker image #25) Parent images: None	On: Blue (ID: unknown) Last status: DEPLOYED Running: true Exit code: 0 Started at: 2015-10-08T13:04:46.45227226Z Finished at: 2001-01-01T00:00:00Z
Name: /drunk_kilby ID/Source: 29a4025532dcf54b3570277ee985f85c59aa69eba2320ea2d7bb0503c1162505 (outside Jenkins) Base image: "harinman/mobile-deposit-api:25" Base image ID: f40dd14f5fd2c291cd5e886fb5be63c6b41057900951708f6dd23b60cc64ddbb8c (from Docker-demos/create and publish docker image #25) Parent images: None	On: unknown (ID: unknown) Last status: DEPLOYED Running: true Exit code: 0 Started at: 2015-10-08T13:08:45.79075068Z Finished at: 2001-01-01T00:00:00Z
Name: /reverent_hawking ID/Source: 4397f15d5fefc786ea0f4e226b7351361a2f1444953b5cb494b70cd095e92be3 (outside Jenkins) Base image: "harinman/mobile-deposit-api:27" Base image ID: h5989de07e6a4c681aca17c7b9a7de901f8e61c525b97014491347cd9767 (from Docker-demos/create and publish docker image #27) Parent images: None	On: mymac (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-09T08:40:32.52525725Z Finished at: 2015-10-09T08:40:33.10866787Z
Name: /desperate_banach ID/Source: 7a1d1bb25e4dbe4743194affc849f90ae3ca6c54bb93d58e0ac31c70526336 (outside Jenkins) Base image: "harinman/mobile-deposit-api:24" Base image ID: 9ca3d81bfcd7349bbef6f1e90be3683151d33420e8e616eaacd082c4dd2309 (from Docker-demos/create and publish docker image #24) Parent images: None	On: unknown (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-08T13:11:49.471354074Z Finished at: 2015-10-08T13:11:49.740646905Z
Name: /angry_bhaskara ID/Source: 7aa5bcb1e93dfe1a441a900fb5b214721fb21582447446e798ece71480197 (outside Jenkins) Base image: "harinman/mobile-deposit-api" Base image ID: b43af407c289e6d82c6e9c25b5b137999ad53840c48674c876366e98828ed (from Docker-demos/create and publish docker image #20) Parent images: None	On: prod-server-1 (ID: unknown) Last status: DEPLOYED Running: true Exit code: 0 Started at: 2015-10-08T12:52:23.87176479Z Finished at: 2001-01-01T00:00:00Z
Name: /happy_raman ID/Source: 7aa5bcb1e93dfe1a441a900fb5b214721fb21582447446e798ece71480197 (outside Jenkins) Base image: "harinman/mobile-deposit-api:25" Base image ID: f40dd14f5fd2c291cd5e886fb5be63c6b41057900951708f6dd23b60cc64ddbb8c (from Docker-demos/create and publish docker image #25) Parent images: None	On: unknown (ID: unknown) Last status: STOPPED Running: false Exit code: 143 Started at: 2015-10-08T13:10:38.1449261474Z Finished at: 2015-10-08T13:10:38.56875762Z

Bottom Right Callout: Link to Image Build

Traceability – Image Build Job

[Back to Project](#)

[Status](#)

[Changes](#)

[Console Output](#)

[Edit Build Information](#)

[Delete Build](#)

[Environment Variables](#)

[See Fingerprints](#)

[Docker Fingerprints](#)

[Previous Build](#)



Build #27 harniman/mobile-deposit-api:27 (Oct 9, 2015 9:38:51 AM)

Started 1 hr 40 min ago
Took 1 min 10 sec on jenkins-beedemo-local-a37896b67a6d

[add description](#)

No changes. Changes in dependency

[jenkins-beedemo-local-a37896b67a6d → #81 \(detail\)](#)

Fingerprints

originally caused by:
[jenkins-beedemo-local-a37896b67a6d → Docker-demos > build and test build number 81](#)

- Started by anonymous user



This run spent:

- 5 sec waiting in the queue;
- 1 min 10 sec building on an executor;
- 1 min 15 sec total from scheduled to completion.

Upstream Builds

[build and test → #81](#)

Traceability – Image Build Job - Fingerprints



Recorded Fingerprints

File ↓	Original owner	Age	
Dockerfile	Docker-demos/build and test #72	23 hr old	more details
mobile-deposit-api-0.0.19-SNAPSHOT.jar	Docker-demos/build and test #81	1 hr 42 min old	more details

The builds that
provided the artifacts

Traceability – Application Build



Build #81 (Oct 9, 2015 9:38:23 AM)



Build Artifacts

Dockerfile

mobile-deposit-api-0.0.19-SNAPSHOT.jar

218 B view

17.71 MB view



No changes.

Change Details



Started by anonymous user



This run spent:

- 1 ms waiting in the queue;
- 22 sec building on an executor;
- 22 sec total from scheduled to c

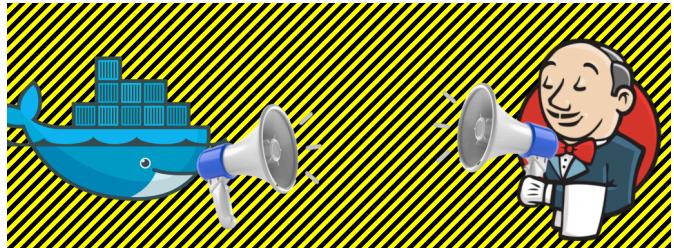
SCM Commit



Revision: f214af3806834fc003d7a5b8c64cf8b630825212

- refs/remotes/origin/master

Notification



Trigger downstream jobs when a tagged container is pushed to Docker Hub

Docker Hub Notification

Trigger downstream jobs when a tagged container is pushed to Docker Hub

The Docker Hub Notification Trigger plugin lets you configure Jenkins to trigger builds when an image is pushed to Docker Hub. E.g. to run verification for the container.

What are the steps

- Set up a WebHook Account for Notification
- Set up your Docker Registry to make callbacks on Image events
- Set up your builds

Docker Hub Notification – Docker Registry Webhook

In the format:

`http://<user>:<token>@<jenkins_url>/dockerhub-webhook/notify`

Add Webhook

The screenshot shows a user interface for adding a webhook. It includes fields for 'Webhook Name' (containing 'my-jenkins') and 'Hook URL 0' (containing 'http://my-jenkins/dockerhub-webhook/notify'). A large orange button with a white 'x' is present. At the bottom, there are two buttons: a blue 'Create' button and a grey 'Add URL' button.

Webhook Name:
my-jenkins

Hook URL 0
http://my-jenkins/dockerhub-webhook/notify

x

Create Add URL

Docker Hub Notification – Job Set up

Build Triggers

- Trigger builds remotely (e.g., from scripts)
- Build after other projects are built
- Build periodically
- Build pull requests to the repository
- Build when a change is pushed to GitHub
- Build when another project is promoted
- Monitor Docker Hub for image changes

- Any referenced Docker image can trigger this job
- Specified repositories will trigger this job

Repositories

harniman/mobile-deposit-api

Configure Trigger



Best of All: Jenkins Pipeline + Docker

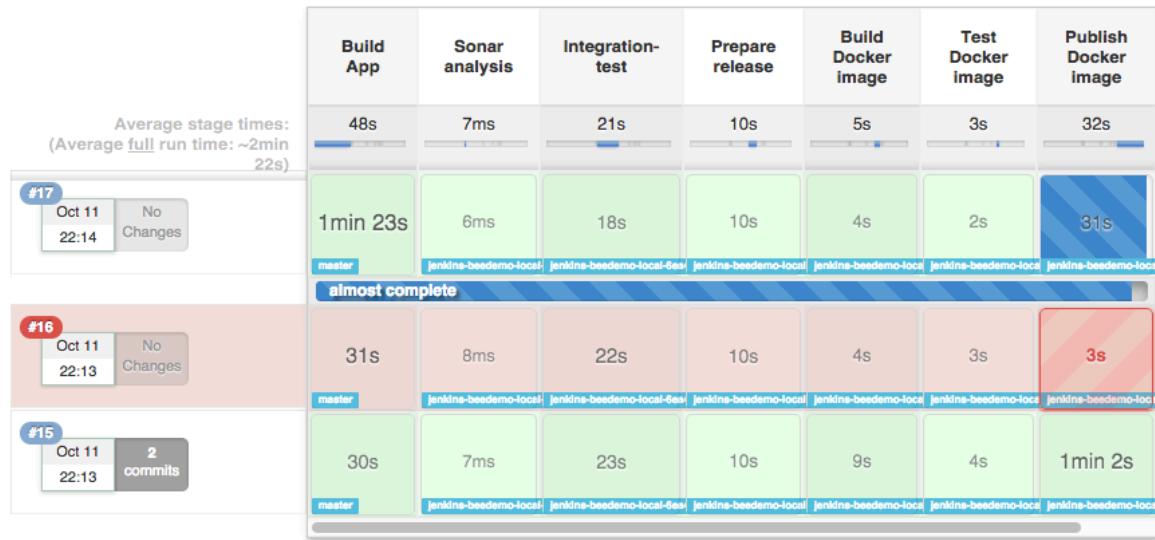
Workflow mobile-deposit-api-workflow

Full project name: Docker-demos/mobile-deposit-api-workflow

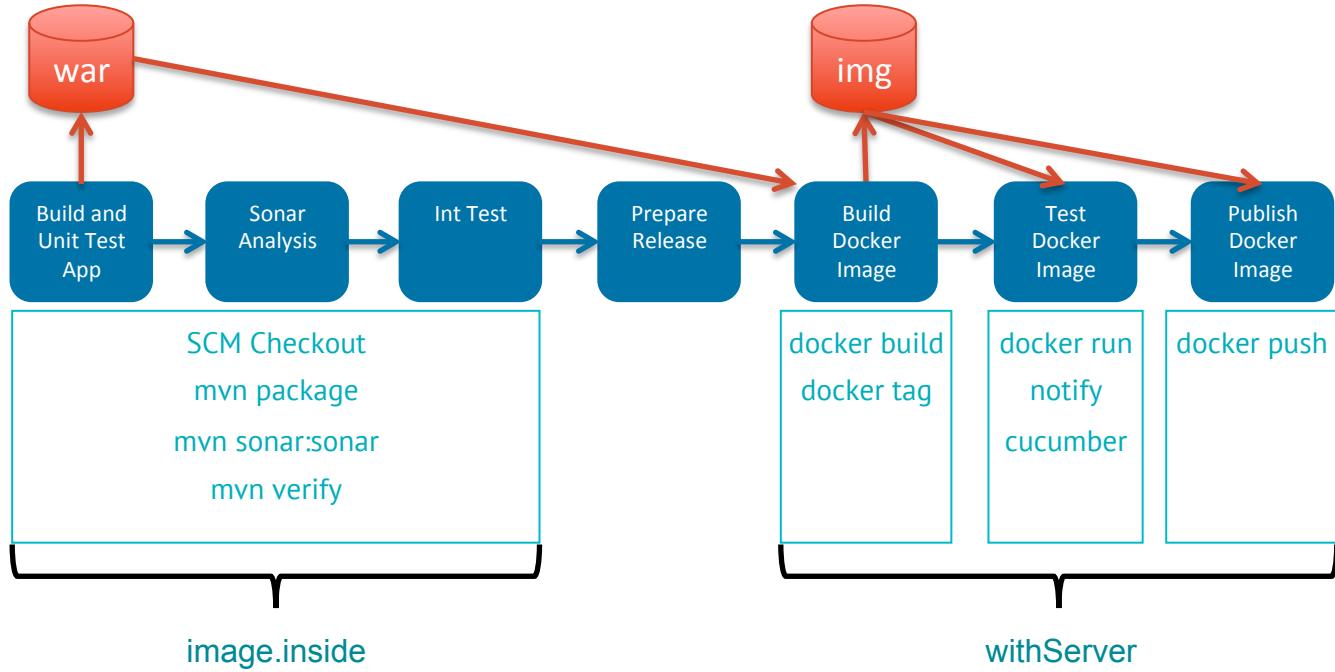


[Recent Changes](#)

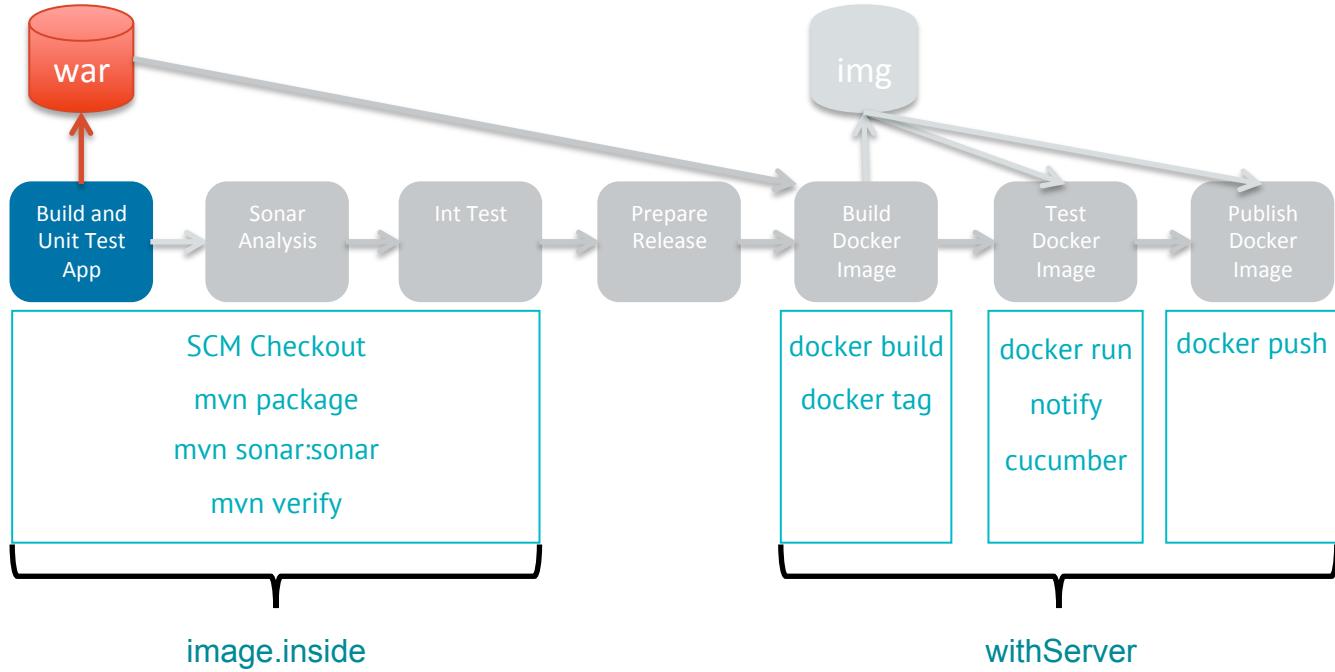
Stage View



Pipeline Stages



Build, unit test and package



Build, unit test and package

```
Specify the Stage Name  
stage 'Build A' Specify the slave label  
node('docker') {  
    docker.image('maven:3.3.3-jdk-8').inside({  
        Custom Build Env  
        Mount volume from slave  
        '-v /data:/data'  
        {  
            .m2 repo location  
            co and build  
            settings.xml', text: "<.....,>  
            git 'https://github.com/cloudbees/mobile-deposit-api.git'  
            sh 'mvn -s settings.xml clean package'  
            ...  
        }  
    })  
}
```

Defining a Docker Slave

Specify Image as template

Assign labels

Docker Template

ID: kmadel/dind-jenk

Labels: docker

Usage: Only build jobs with label restrictions matching this node

Credentials: jenkins

Add

Remote Filing System Root: /home/jenkins

Remote FS Root Mapping:

Instance Cap: 4

DNS:

Port bindings: 1234

Bind all declared ports:

Hostname:

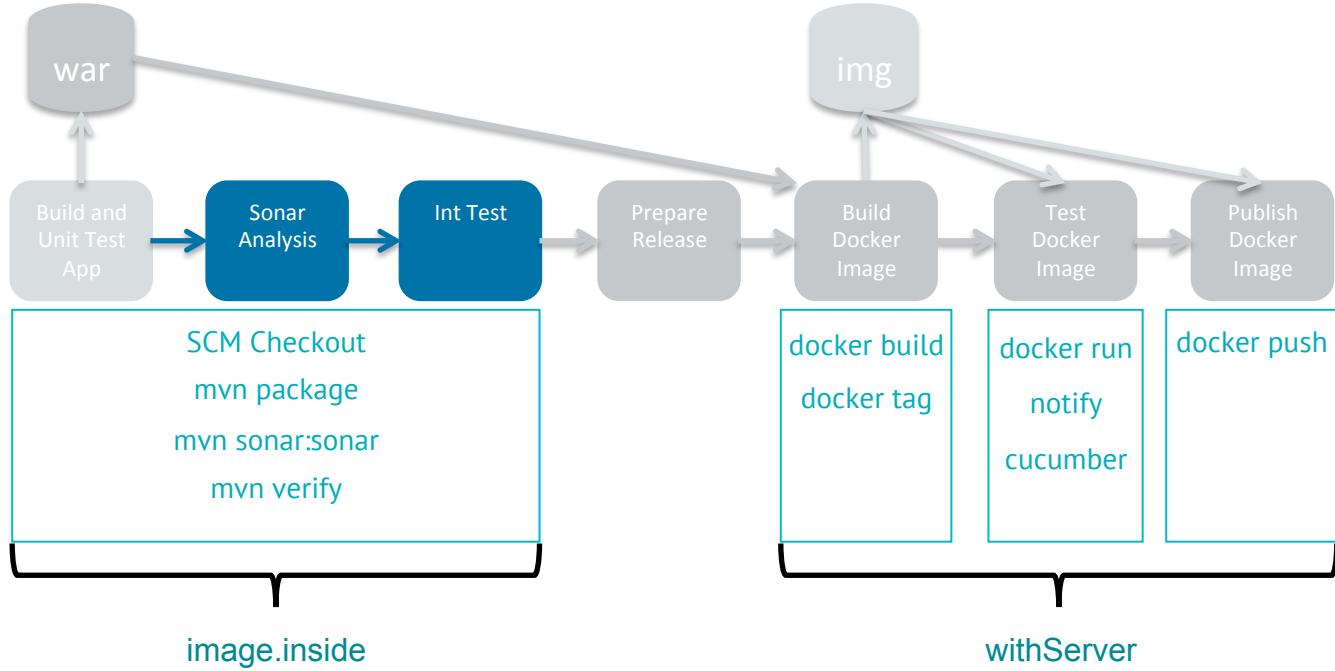
Advanced...

Delete Docker Template

Add Docker Template ▾

List of Images to be launched as slaves

Test the app



Test the app

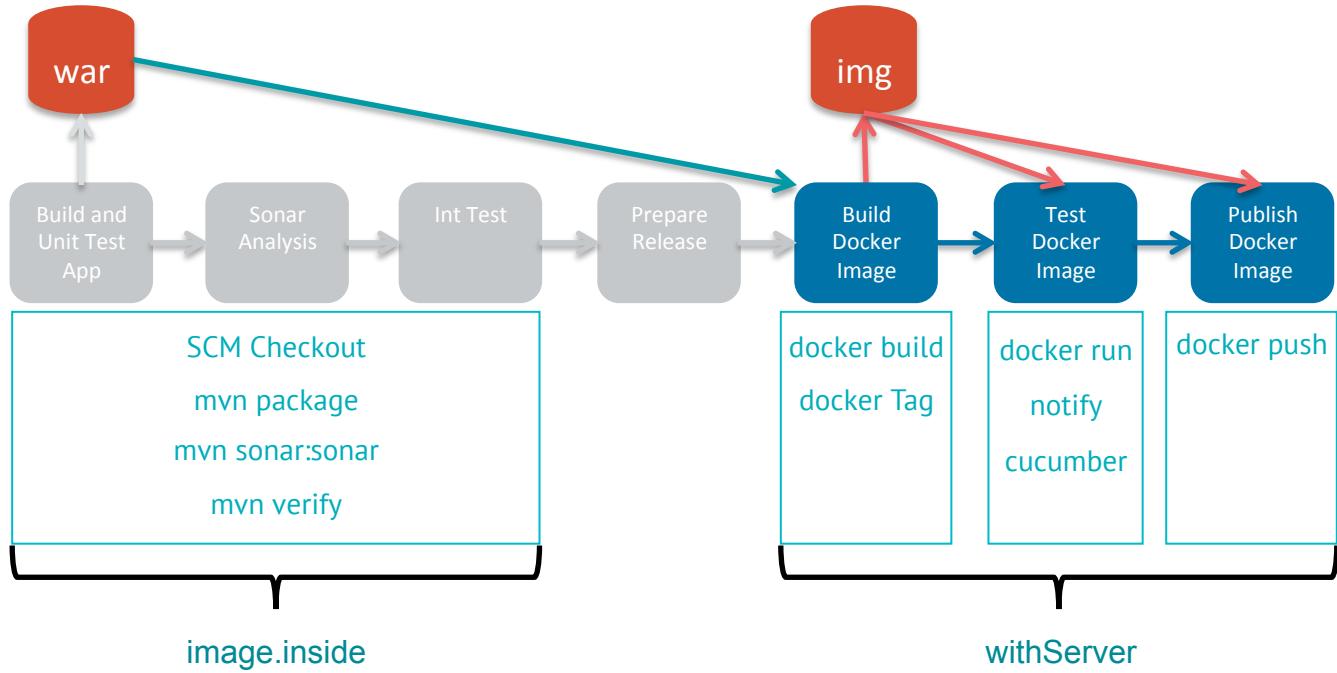
In same env as build

```
node('docker') {  
    docker.image('maven:3.3.3-jdk-8').inside('-v /data:/data') {  
        stage 'Sonar analysis'  
        sh 'mvn -s settings.xml sonar:sonar'  
        stage 'Integration-test'  
        sh 'mvn -s settings.xml verify'  
        step([$class: 'JUnitResultArchiver', testResults: '**/  
target/surefire-reports/TEST-*.xml'])  
    }  
}
```

Sonar tests

Run API Tests

Build, test and publish Docker image



Build, test and publish Docker image

```
docker.withServer('tcp://192.168.99.100:2376', 'slave-docker-us-east-1-tls') {
```

Bind to docker host

```
    stage 'Build Docker image'
```

```
    def mobileDepositApiImage
```

Change directory

```
        dir('.docker') {
```

```
            sh "mv ..../target/*-SNAPSHOT.jar mo" Build docker image
```

```
            mobileDepositApiImage = docker.build "harniman/mobile-deposit-api:${buildVersion}"
```

```
        }
```

...

Build, test and publish Docker image

```
...  
stage 'Test Docker image'  
container=mobileDepositApiImage.run("--name mobile-deposit-  
api -p 8080:8080")  
sh "curl http://<user>:<token>@<host>:8080/docker-  
traceability/submitContainerStatus \....."  
// insert cucumber tests here  
stage 'Publish Docker image'  
withDockerRegistry(registry: [credentialsId: 'dockerhub-  
harniman']) {  
    mobileDepositApiImage.push()  
}
```

Launch container

Submit traceability report

Run some Tests

Bind to registry

Push image

Tagged Image in Docker Hub

PUBLIC REPOSITORY

[harniman/mobile-deposit-api](#) 

Last pushed: 10 minutes ago

[Repo Info](#) [Tags](#) [Collaborators](#) [Webhooks](#) [Settings](#)  [Delete Repository](#)

Tags

Tag	Size	Tagged Version
23	329 MB	
22	329 MB	
21	329 MB	
20	329 MB	
19	329 MB	

Best of All: Jenkins Pipeline + Docker

```
stage 'Build Source'
node('docker') {
    docker.image('maven:3.3.3-jdk-8') {
        git 'https://github.com/cloudbees/mobile-deposit-api.git'
        sh 'mvn clean package'
    }
}

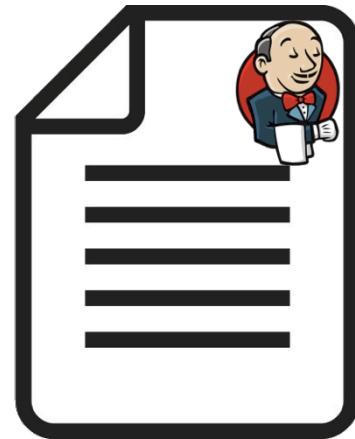
node('docker') {
    docker.withServer('tcp://docker.beedemo.net:2376', 'docker-beedemo-creds'){
        stage 'Build Docker Image'
        def image = docker.build "cloudbees/mobile-deposit-api:${buildVersion}"

        stage 'Publish Docker Image'
        docker.withRegistry('https://registry.beedemo.net/', 'docker-registry-login') {
            image.push()
        }

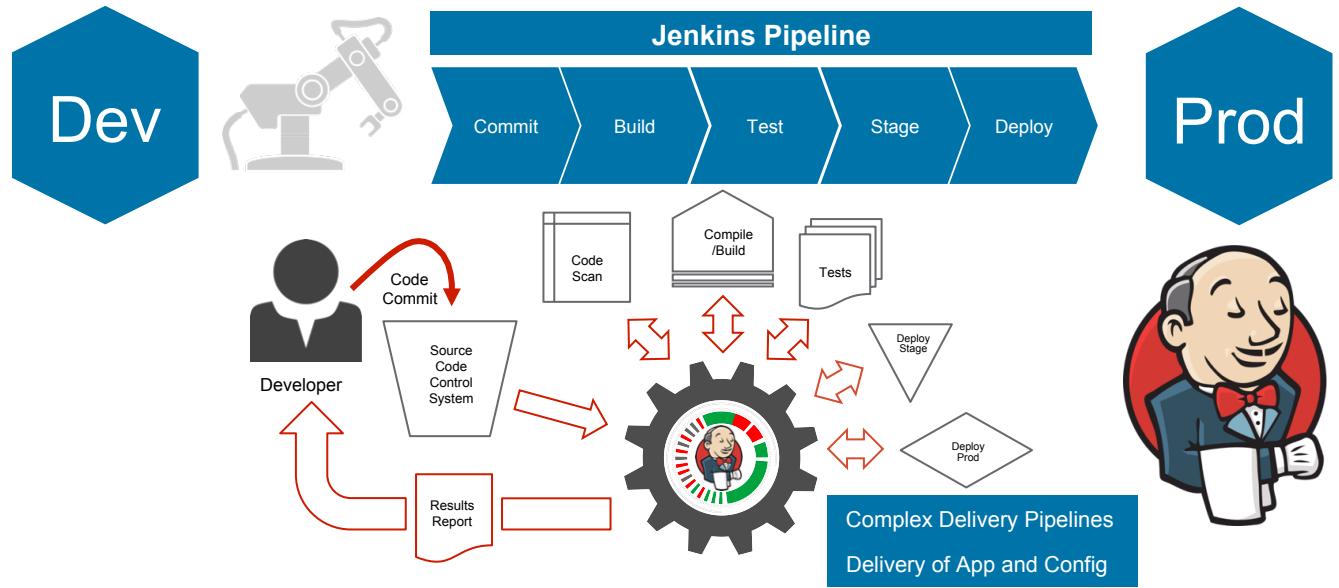
        stage 'Deploy Docker Image'
        def container = image.run('--name mobile-deposit-api -p 8080:8080')
    }
}
```

OSS - Pipeline-as-Code Jenkinsfile

- Define complete Workflow within Code
- Edit Workflow Job in IDE
- Workflow Job automatically incorporates changes
- Version Workflow Jobs
- Supports: Git, Mercurial, SVN



Jenkins, with Pipeline, is the Proven CD Platform



Jenkins World 2016

The event for everything Jenkins:
community, CloudBees and ecosystem.

- Santa Clara Convention Center
- September 13-15, 2016



**Jenkins World
2016**

Learn more and register now!
www.jenkinsworld.com

Resources

- @BhatiaKishore
- Github: <https://github.com/kishorebhatia>
- Mobile-deposit-api example:
<https://github.com/NewGithubOrg/mobile-deposit-api>
- Jenkins BlueOcean: <https://jenkins.io/projects/blueocean/>
- Jenkins Pipeline-as-code: <https://jenkins.io/doc/pipeline/>



Thank You!