

JENKINS

- CI/CD tools written in Java. It is an automation server basically.
- **Continuous Integration:** means merging all the developer working copies into a mainline several times a day. Software is produced in short cycles.
- All developers must commit at least once a day and Jenkins pushes their changes to a version control system.
- Jenkins itself does not merge code.
- Jenkins can publish every build of your software.
- Tests: Unit, Integration, Regression, User Acceptance et cetera.
- SDLC Code → Build → Test → Release → Deploy

Alternatives: Drone CI, TeamCity, Wercker, Travis, Gitlab, Github, CircleCI, CodeShip, AWS CI/CD tools ...

- Ubuntu is preferred here. In the course the guy uses DigitalOcean droplet and installs docker and jenkins inside there.
- Can you simulate it on the local though?
- Docker isolates the binary with all the dependencies
- No more “works on my machine!”
- If you create an image of an app you can use it pretty much anywhere.
- Go is pretty much immature and circlejerky with no robust libraries at all, learn node.js instead!

Node.js App on Jenkins CI/CD

You first install the dependencies and run tests “npm install && npm test”

Docker basically does what go does when you create a shippable binary.

Jenkins has a lot of plugins for binary creation from a sample node.js app.

Jenkins has a great UI which you can use to do all the docker deployment of images.

However if you use the UI you can not track the history of changes so one should use CLI to create an audit trail.


There are Jenkins administrators and developers need to ask for permission.

Developers should do version control of their own build files.

Jenkins Job DSL(Domain specific language)

Plugin uses groovylike scripting language. When you have a lot of jobs(apps) you need to use this.

```
job('NodeJS example') {  
    scm {  
        git('git://github.com/wardviaene/docker-demo.git') { node ->  
            node / gitConfigName('DSL User')  
            node / gitConfigEmail('jenkins-dsl@newtech.academy')  
        }  
    }  
  
    triggers {  
        scm('H/5 * * * *')  
    }  
  
    wrappers {  
        nodejs('nodejs') // this is the name of the NodeJS installation in  
                           // Manage Jenkins -> Configure Tools -> NodeJS Installations -> Name  
    }  
  
    steps {  
        shell("npm install")  
    }  
}
```



Jenkins DSL

you need to approve scripts beforehand.

You can integrate it with Slack, Github, Bitbucket, Emails etc.

Jenkins Pipelines

1. Build steps are written in code build ---> test ---> deploy
2. Basically you automate the build test deploy cycle.
3. Probably Gitlab CI/CD is better!.
4. You can write this in Jenkins DSL or Groovy.

```

node {

    def mvnHome

    stage('Preparation') {
        git 'https://github.com/wardviaene/java-demo.git'
        // Get the Maven tool.
        // ** NOTE: This 'M3' Maven tool must be configured
        // **          in the global configuration.
        mvnHome = tool 'M3'
    }

    stage('Build') {
        // Run the maven build
        if (isUnix()) {
            sh "${mvnHome}/bin/mvn" -Dmaven.test.failure.ignore
        } else {
            bat("${mvnHome}\bin\mvn" -Dmaven.test.failure.ignore
        }
    }

    stage('Results') {
        junit '**/target/surefire-reports/TEST-*.xml'
        archive 'target/*.jar'
    }
}


```

Example Jenkins Pipeline

Node: Worker node which node will run this pipeline.

Stage: Build,test,deploy stages.

20 lines (19 sloc) | 542 Bytes



```
1 node {
2   def commit_id
3   stage('Preparation') {
4     checkout scm
5     sh "git rev-parse --short HEAD > .git/commit-id"
6     commit_id = readFile('.git/commit-id').trim()
7   }
8   stage('test') {
9     nodejs(nodeJSInstallationName: 'nodejs') {
10      sh 'npm install --only=dev'
11      sh 'npm test'
12    }
13  }
14  stage('docker build/push') {
15    docker.withRegistry('https://index.docker.io/v1/', 'dockerhub') {
16      def app = docker.build("wardviaene/docker-nodejs-demo:${commit_id}", '.').push()
17    }
18  }
19 }
```

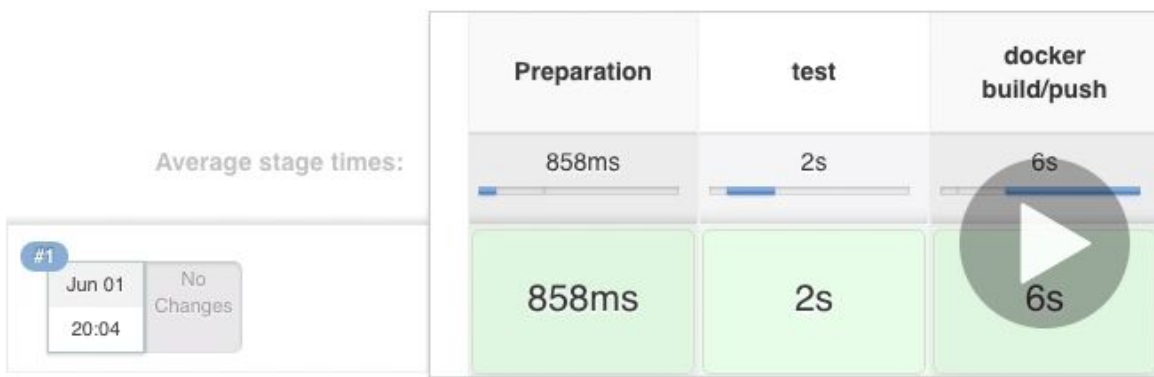
For a Node.js project a Jenkins Pipeline

Pipeline nodejs docker pipeline



[Recent Changes](#)

Stage View



Permalinks

- [Last build \(#1\), 1 min 25 sec ago](#)
- [Last stable build \(#1\), 1 min 25 sec ago](#)
- [Last successful build \(#1\), 1 min 25 sec ago](#)
- [Last completed build \(#1\), 1 min 25 sec ago](#)

A visual representation of a Jenkins Pipeline

- You can run an already built docker container in Jenkins.
- You can also test database integration with Jenkins pipelines.
- For example in the below example you pull a MySQL image and do an integration test and then remove the container if the test succeeds.

```

node {
    def commit_id
    stage('Preparation') {
        checkout scm
        sh "git rev-parse --short HEAD > .git/commit-id"
        commit_id = readFile('.git/commit-id').trim()
    }
    stage('test') {
        def myTestContainer = docker.image('node:4.6')
        myTestContainer.pull()
        myTestContainer.inside {
            sh 'npm install --only=dev'
            sh 'npm test'
        }
    }
    stage('test with a DB') {
        def mysql = docker.image('mysql').run("-e MYSQL_ALLOW_EMPTY_PASSWORD=yes")
        def myTestContainer = docker.image('node:4.6')
        myTestContainer.pull()
        myTestContainer.inside("--link ${mysql.id}:mysql") { // using linking, mysql will be available at host: mysql, port: 3306
            sh 'npm install --only=dev'
            sh 'npm test'
        }
        mysql.stop()
    }
    stage('docker build/push') {
        docker.withRegistry('https://index.docker.io/v1/', 'dockerhub') {
            def app = docker.build("wardviaene/docker-nodejs-demo:${commit_id}", '.').push()
        }
    }
}

```

Docker Images used in Jenkins Pipelines

- For every pull and push you can set up notifications via email, slack et cetera.
- You can do this with Jfrog, Custom APIs and Sonarqube as well.
- You use webhooks for Slack and Custom API integrations
- Sonarqube is the “Code Quality Checker” that basically enforces best practices and you can also integrate this to your pipelines.

Advanced Jenkins Topics

- One Droplet = One node
- Jenkins web UI on one node and worker nodes to do the work(Jenkins Slaves)
- You can scale manually during working hours when a lot of the code is being developed.
- You can also use automatic plugins for scaling. (Uses AWS EC2 API or Docker Plugin or DigitalOcean plugin and many others for this purpose)
- Builds can only run on a specific node.
- Always use slaves for economical practices.
- Master node connects to the slave using SSH
- or Slave connects to master using JNLP

- **Blue Ocean is a frontend for Jenkins to replace the Web UI. Better Visualization this is a plugin.**
- ssh-agent ---> this is used so the slaves can be configured automatically.
- use a firewall for Jenkins to protect business tools no need for external access.
- Whitelist IPs for buckets
- If you use it as exposed to the public networks you can get HACKED!
- **Onelogin is used for authentication to Jenkins.** Matrix based authentication or there are other plugins too.
- SAML is used for authentication. Onelogin is not cheap!!