

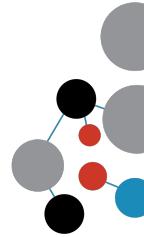
The Game of DevOps: Applying Jenkins Best Practices in a Dynamic Industry

Victor Martinez



Jenkins World
A global DevOps event

2017



Who am I



- Victor Martinez
- Configuration Engineer @ARM
- Twitter: @aninfinite
- Active user in the Jenkins community
- JAM Co-Organiser





Jenkins World
A global DevOps event
2017

Dynamic Industry

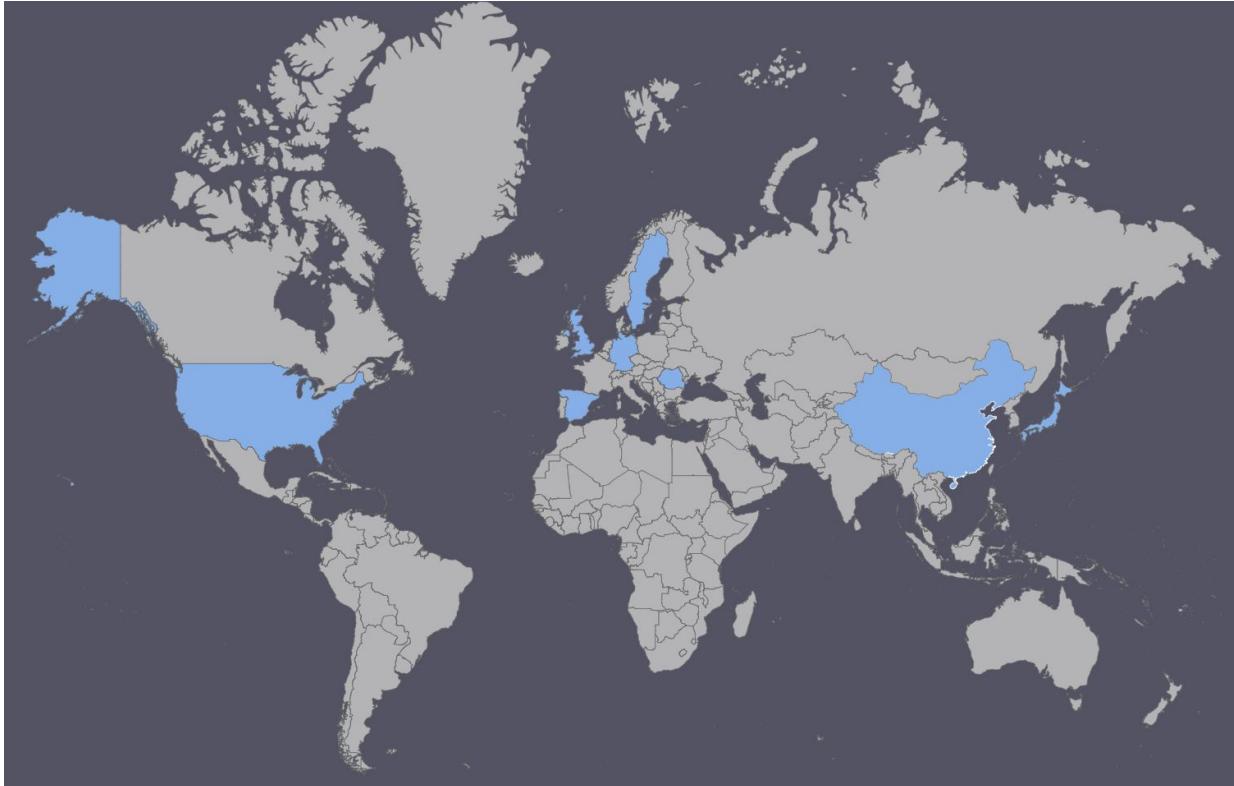




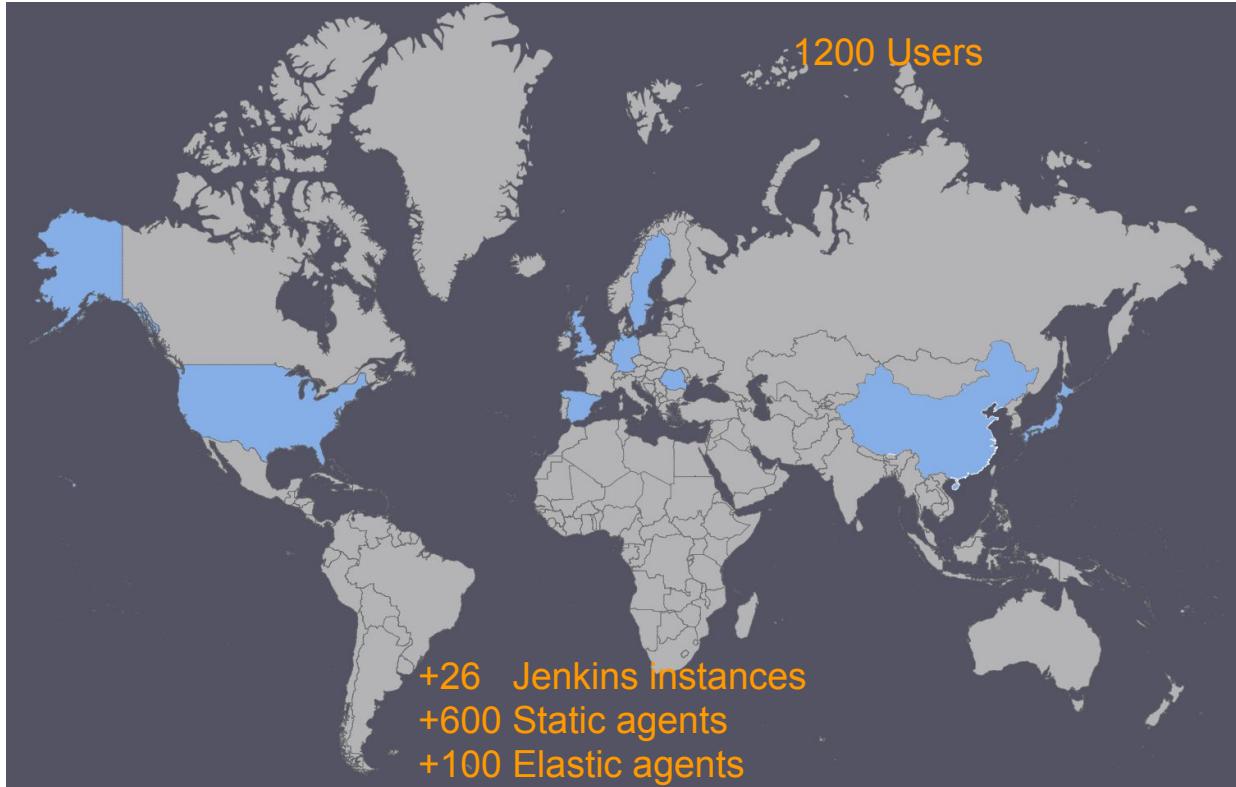
Jenkins World
A global DevOps event
2017

The Game of DevOps









Guidelines

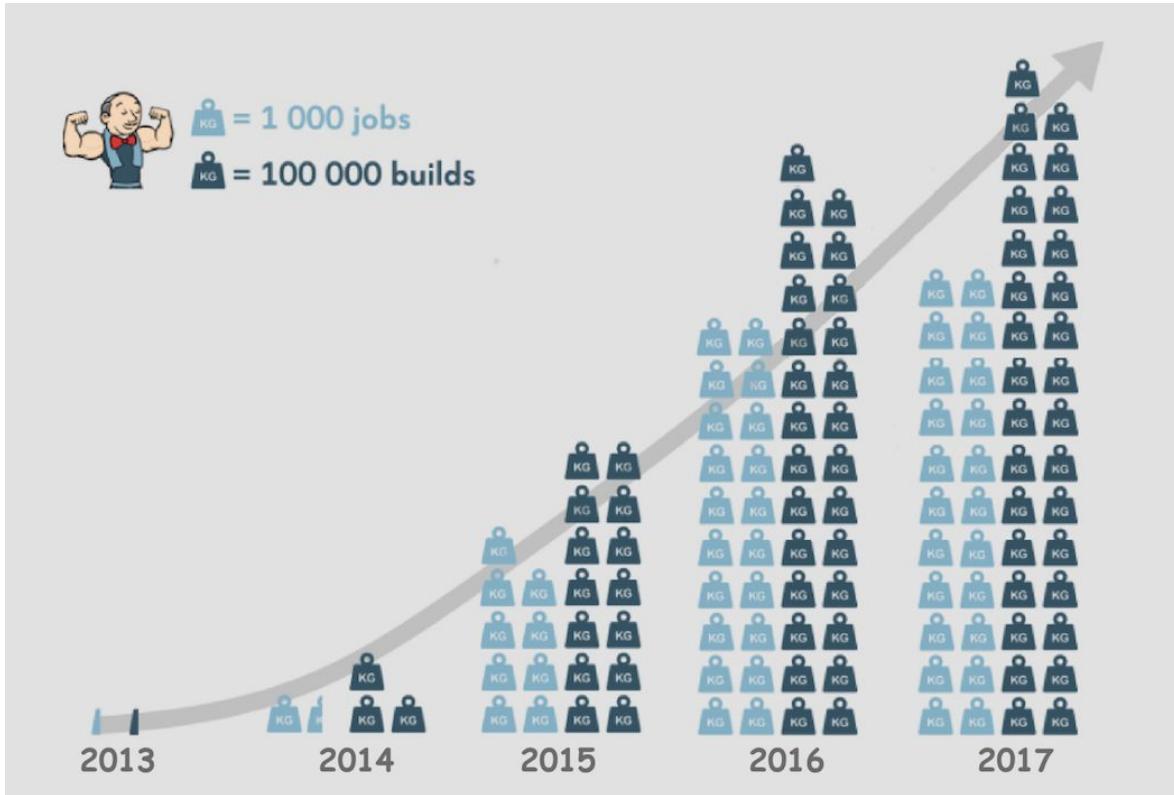


Jenkins World
A global DevOps event
2017

Recommended Action	Benefits	More Information
Name the jobs using the convention [game].[branch].[action].[platform] , as detailed here .	Better organisation and project view in Jenkins.	Jenkins Job Naming Convention
Organise jobs in tabs using [game] as the tab name.	Jobs that are more traceable.	
Set up a different job/project for each maintenance or development branch you create.	The build flow becomes more understandable and has a better high-level view.	
Use different jobs for building, testing, and deploying.		
Set a label on your jobs to build on the slaves and, if possible, avoid using the name of the node as the label.	Avoiding Jenkins memory outages.	
Use the label that is geographically closest to you (for example, bcn-osx-pool if you are in Barcelona).	Sharing the the build load between nodes. Shorter build times and faster downloads from repositories.	
Use a policy for log rotation. We recommend a maximum of 90 builds, 30 days for keeping build logs, 3 days for keeping artifacts, and 7 builds for keeping artifacts.	This allows Jenkins and its slaves to save space in the workspaces and delete unused artifacts.	<input checked="" type="checkbox"/> ES-3901 - JENKINS: Set reasonable Log Rotation automatically REOPENED
Avoid polling the repository and use hooks instead.	Polling must die, as it is a waste of resources to ask repositories for changes every minute.	
When polling, use randomised polling and set an appropriate time for polling (not every minute).	If every job uses polling, Jenkins can	Polling must die: triggering Jenkins builds from hooks



But what if there is an exponential usage?



Best Practices





Jenkins World
A global DevOps event
2017

#1 Description





#1 Description

```
1 jenkins.model.Jenkins.instance?.
2 getAllItems()?.
3 findAll { !it.getDescription() }?.
4 each {
5     println "'${it.fullName}' doesn't explain what it does."
6 }
```

#2 Spaces



Jenkins World
A global DevOps event
2017





#2 Spaces

```
1 jenkins.model.Jenkins.instance?.
2     getAllItems()?.
3         findAll { it.name.contains(' ') }?.
4             each {
5                 println "'${it.fullName}' contains spaces!"
6 }
```





Jenkins World
A global DevOps event
2017

#3 Label





#3 Label

```
1 jenkins.model.Jenkins.instance?.
2 getAllItems()?.
3 findAll { !it.getAssignedLabel() }?.
4 each {
5     println "'${it.fullName}' doesn't have any assigned agent/label"
6 }
```





#4 Workspace





```
1 jenkins.model.Jenkins.instance?.
2 getAllItems()?.
3 findAll {
4     !(it.getPublishersList()?.find {it.getClass().getName().contains('WsCleanup')}) &&
5     !(it.getBuildWrappers()?.find{ k,v -> k.getClass().getName().contains('PreBuildCleanup') } != null ) &&
6     !(it.scm && it.scm.getClass().getName().contains('hudson.plugins.git.GitSCM') ) &&
7         it.scm.getExtensions()?.find { it instanceof hudson.plugins.git.extensions.impl.WipeWorkspace } != null )
8     }?.
9 each {
10     println "${it.fullName}" is not cleaning up its workspace"
11 }
```



Jenkins World
A global DevOps event
2017

#5 Log rotator





#5 Log rotator

```
1 jenkins.model.Jenkins.instance?.
2 getAllItems(Job.class)?.
3 each {
4     found = true
5     if (it.getBuildDiscarder() )
6         found = it.getBuildDiscarder().getArtifactDaysToKeep() < 0 && it.getBuildDiscarder().getArtifactNumToKeep() < 0 &&
7             it.getBuildDiscarder().getDaysToKeep() < 0 && it.getBuildDiscarder().getNumToKeep() < 0
8     if (found)
9         println "'${it.fullName}' doesn't have any log rotator configuration"
10 }
```



#6 Pushing





#6 Pushing

```
1 jenkins.model.Jenkins.instance?.
2     getAllItems()?.
3     each {
4         try {
5             if (it.getTrigger(hudson.triggers.SCMTrigger.class) ) {
6                 println "'${it.fullName}' is polling!"
7             }
8             // Skip items without build discarding method
9         } catch (groovy.lang.MissingMethodException mme) { }
10    }
```



#6 Pushing

```
1 def message = '''<div class="alert alert-warning">
2             <strong>Warning!</strong> Git+Polling is deprecated. This job
3             will be disabled from 1st, May
4             </div>''
5
6 jenkins.model.Jenkins.instance?.
7   getAllItems()?.
8   each {
9     try {
10       if (it.getTrigger(hudson.triggers.SCMTrigger.class) ) {
11         println "INFO '${it.fullName}' is polling!"
12         if (!it.getDescription().toLowerCase().contains('git+polling')) {
13           it.setDescription(it.getDescription() + message)
14         }
15     }
16   }
17
18 }
```

Project editor-master.win.release

Warning! Git+Polling is deprecated. This job will be disabled from 1st, May if git polling is still enabled. Further details in [this page](#)



#6 Pushing

```
1 def message = '''<div class="alert alert-warning">
2             <strong>Important! Git+Polling is not supported</strong>.
3             </div>'''
4
5 jenkins.model.Jenkins.instance?.
6     getAllItems()?.
7     each {
8         try {
9             if (it.getTrigger(hudson.triggers.SCMTrigger.class) ) {
10                 println "INFO '${it.fullName}' is polling!"
11                 if (!it.getDescription().toLowerCase().contains('git+polling')) {
12                     it.setDescription(it.getDescription() + message)
13                     it.disable()
14                 }
15             }
16             // Skip items without build discorder me
17         } catch (groovy.lang.MissingMethodException e)
18     }
```

Project editor-master.win.release

Important! Git+Polling is not supported. Please read this [page](#)

⚠ This project is currently disabled

Enable



Jenkins World
A global DevOps event
2017

#7 Shallow





#7 Shallow

```
1 import hudson.plugins.git.extensions.impl.CloneOption
2 jenkins.model.Jenkins.instance?.
3 getAllItems()?.
4 findAll { try { it.getScm() } catch (groovy.lang.MissingMethodException mme) {} }?.
5 findAll { it.getScm() instanceof hudson.plugins.git.GitSCM }?.
6 each {
7     isShallow = false
8     if (it.getScm().getExtensions() && it.getScm().getExtensions()?.
9             find {
10                 it instanceof CloneOption &&
11                 it.isShallow()
12             }
13         ) {
14             isShallow = true
15         }
16     if (!isShallow)
17         println "'${it.fullName}' is not using shallow cloning"
18 }
```



#8 Reference



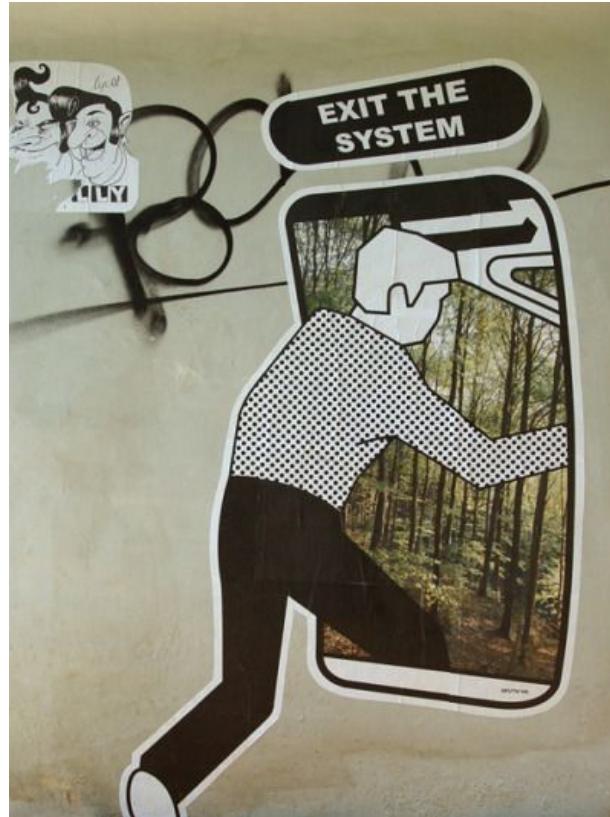


#8 Reference

```
1 import hudson.plugins.git.extensions.impl.CloneOption
2 jenkins.model.Jenkins.instance?.
3 getAllItems()?.
4 findAll { it.getScm().getType().toLowerCase().contains('git') }?.
5 each { job ->
6     if (job.scm.getExtensions().empty) { // When no options
7         println "'${job.fullName}' is not using Git Reference Repo"
8     }
9     job.scm.getExtensions()?.findAll { it instanceof CloneOption }.each { p ->
10        if (!p.getReference().trim()){
11            println "'${job.fullName}' is not using Git Reference Repo"
12        }
13    }
14 }
```



#9 System exit





```
1 jenkins.model.Jenkins.instance?.
2 getAllItems(AbstractProject.class)?.
3 findAll { it.getPublishersList() }?.each {
4     if (it.getPublishersList()?.
5         findAll { it.getClass().getName().endsWith("GroovyPostbuildRecorder") } &&
6             it.getScript()?.getScript()?.toLowerCase().contains("system.exit"))
7     ) {
8         println "${it.fullName}" is using System Exit"
9     }
10 }
```



Jenkins World
A global DevOps event
2017

#10 Groovy Sandbox



<http://chicagokidsisland.com/wp-content/uploads/2013/11/indoor-sandbox.jpg>



#10 Groovy Sandbox

```
1 jenkins.model.Jenkins.instance?.
2     getAllItems(org.jenkinsci.plugins.workflow.job.WorkflowJob.class)?.
3         findAll { !it.getDefinition()?.isSandbox() }?.
4             each {
5                 println "Pipeline -> '${it.fullName}' doesn't use Groovy Sandbox"
6 }
```



Jenkins World
A global DevOps event
2017

#11 Maven Jobs





#11 Maven Jobs

```
1 jenkins.model.Jenkins.instance?.
2     getAllItems(hudson.maven.MavenModuleSet.class)?.
3     each {
4         println "${it.fullName}" is a Maven Project!
5     }
```



Jenkins World
A global DevOps event
2017

#12 Abort building





#12 Abort building

```
1 import hudson.plugins.build_timeout.BuildTimeoutWrapper
2 jenkins.model.Jenkins.instance?.
3 getAllItems()?.
4 findAll {
5     try {
6         !it.getBuildWrappers()?.find { k,v -> v instanceof BuildTimeoutWrapper }
7     } catch (groovy.lang.MissingMethodException mee) {}
8 }?.each {
9     println "'${it.fullName}' doesn't use Build Timeout Plugin"
10 }
```



Jenkins World
A global DevOps event
2017

#13 Build Failure Plugin





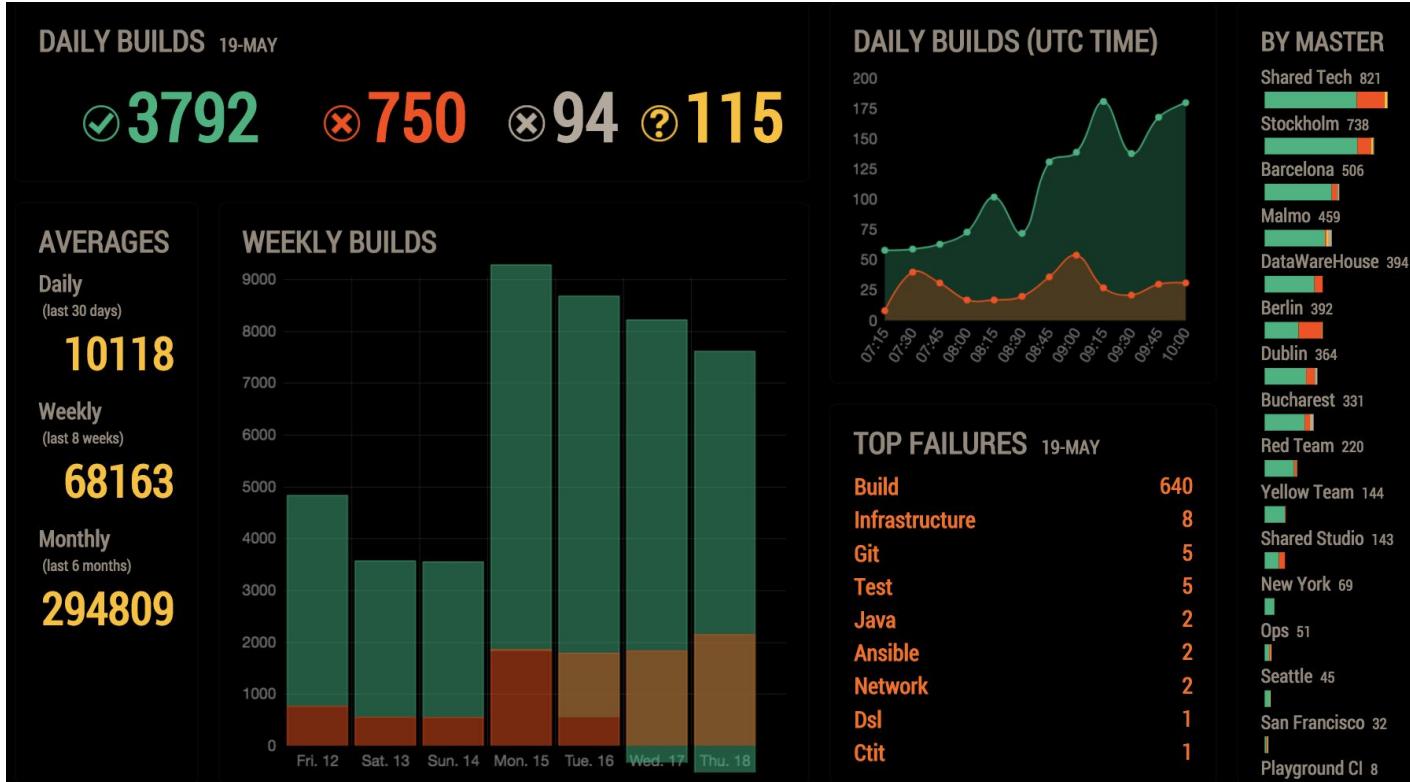
#13 Build Failure Plugin

```
1 import com.sonyericsson.jenkins.plugins.bfa.model.ScannerJobProperty
2 jenkins.model.Jenkins.instance?.
3 getAllItems()?.
4 each {
5     def bfa = it.getAllProperties()?.find { prop -> prop instanceof ScannerJobProperty }
6     if (!bfa || bfa?.isDoNotScan()) {
7         println "'${it.fullName}' is not using BFA"
8     }
9 }
```

#13 Build Failure Plugin



Jenkins World
A global DevOps event
2017





Control Comments

Project MyFreestyleJob

This is only for testing purpose

```
disable:git:shallow_cloning
```





Control Comments

Project MyFreestyleJob

This is only for testing purpose

```
disable:git:shallow_cloning
```

```
1  ∵ jenkins.model.Jenkins.instance?.
2    getAllItems()?.
3      findAll { try { it.getScm() } catch (groovy.lang.MissingMethodException mme) {} }?.
4      findAll { it.getScm() instanceof hudson.plugins.git.GitSCM }?.
5      findAll { !it.description.contains("disable:git:shallow_cloning") }?.
6      each {
7        isShallow = false
8        if (it.getScm().getExtensions() && it.getScm().getExtensions()?.
9          find {
10            it instanceof hudson.plugins.git.extensions.impl.CloneOption &&
11              it.isShallow()
12            }
13            ) {
14              isShallow = true
15            }
16        if (!isShallow)
17          println "'${it.fullName}' is not using shallow cloning"
18      }
```



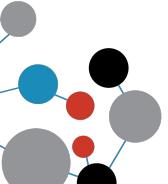
How to evangelise those practices?

- They are coded practices
- Easy to automate/collaborate
- DryRun
- Visualise
- Poke Owners (JobOwnership Plugin helped)
- Levels:
 - Deprecated policy
 - Disable policy
 - Modify policy

Automated Linting



Jenkins World
A global DevOps event
2017

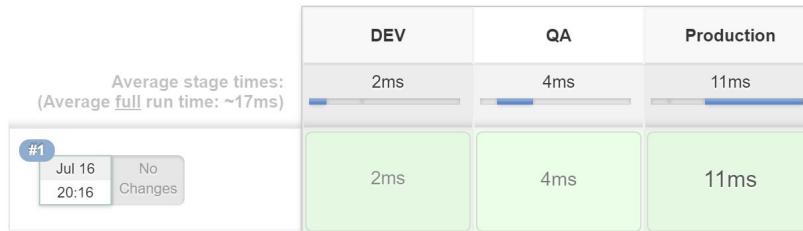


Jenkins Lint: Items supported



Jenkins World
A global DevOps event
2017

Stage View



Permalinks

- [Last build \(#1\), 2 hr 14 min ago](#)
- [Last stable build \(#1\), 2 hr 14 min ago](#)
- [Last successful build \(#1\), 2 hr 14 min ago](#)

Enter an item name

Your_Task

» Required field



Freestyle project

This is the central feature of Jenkins. Jenkins will build you



Pipeline

Orchestrates long-running activities that can span multiple



External Job

This type of job allows you to record the execution of a pro



Multi-configuration project

Suitable for projects that need a large number of different c



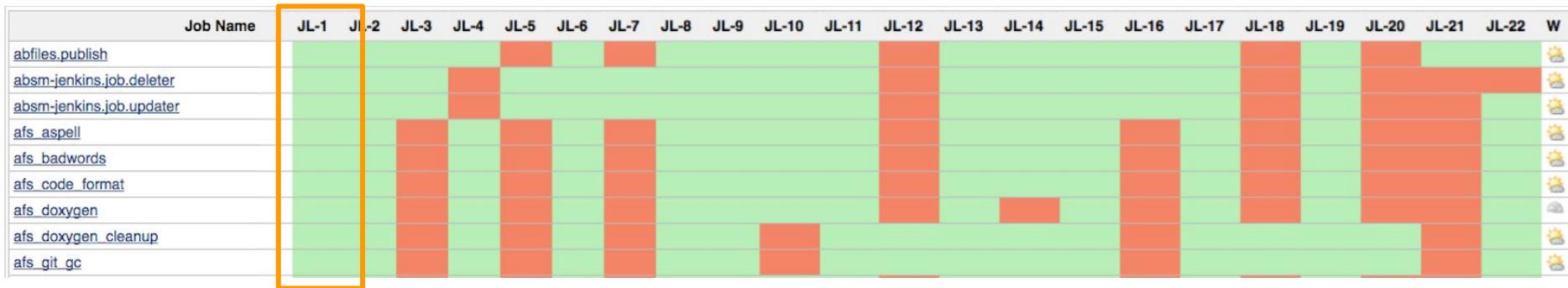
Folder

Creates a container that stores nested items in it. Useful fo long as they are in different folders.



Jenkins Lint: jobs view

Jenkins World
A global DevOps event
2017



Lint Defects

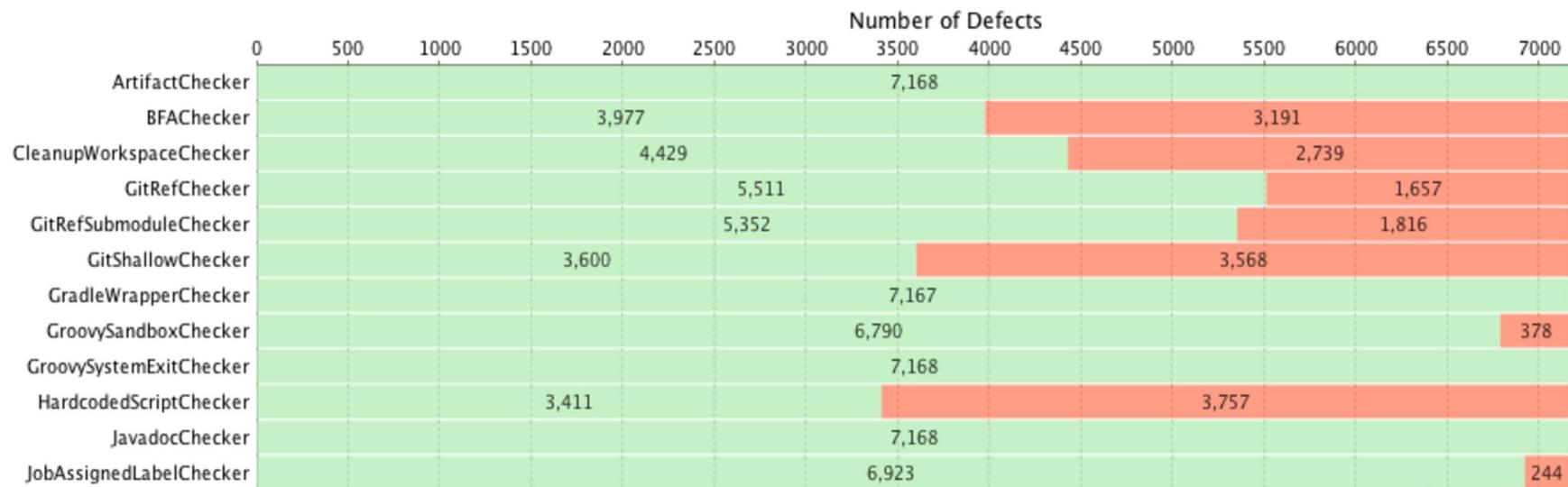


Jenkins World

A global DevOps event

2017

Jenkins Lint: defects view





Jenkins Lint: job view

Jenkins > MyPipeline > Jenkins Lint

[Back to Dashboard](#)
[Status](#)
[Changes](#)
[Build Now](#)
[Delete Pipeline](#)
[Configure](#)
[Open Blue Ocean](#)
[Full Stage View](#)
[Jenkins Lint](#) **(selected)**
[Pipeline Syntax](#)

Build History [trend](#) [X](#)

JenkinsLint: Project MyPipeline

The [JenkinsLint Plugin](#) has mainly two goals:

- To make it easier to detect issues in your Jenkins configuration that will cause problems.
- To encourage discussion within the Jenkins community on the more subjective aspects of what good Jenkins configuration looks like.

Status	Defect Name	Description
Green	JobNameChecker	When creating Jenkins Jobs you must give them a name.
Red	JobDescriptionChecker	Jenkins project description might help you to identify the job.
Green	JobAssignedLabelChecker	When setting Jenkins Jobs you should assign them a label.
Green	MasterLabelChecker	When setting Jenkins Jobs you shouldnt give them the same label as Jenkins master.
Red	JobLogRotatorChecker	When setting Jenkins Jobs with some amount of disk space and speed up Jenkins.
Green	MavenJobTypeChecker	Maven job type builds considerably slower than other jobs. Besides of that it has its own set of build steps.



Jenkins Lint: overall lint job view

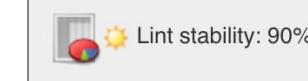
Pipeline MyPipeline



[Recent Changes](#)

[add description](#)

[Disable Project](#)





Jenkins Lint: agent view

Back to List

Status

Delete Agent

Configure

Build History

Load Statistics

Log

Jenkins Lint

Open Blue Ocean

Build Executor Status

JenkinsLint: Computer Terminator

The [JenkinsLint Plugin](#) has mainly two goals:

- To make it easier to detect issues in your Jenkins configuration that will break your build.
- To encourage discussion within the Jenkins community on the more subtle ways to write Jenkins jobs in good style.

Status	Defect	Description
	SlaveDescriptionChecker	Jenkins slave description might have been copied from another slave.
	SlaveVersionChecker	Jenkins slave version should match the master's.
	SlaveLabelChecker	When setting Jenkins Slaves you must use the same label.
	WindowsSlaveLaunchChecker	This launch method relies on DC/OS to handle the slave registration.
		Start instead, which also permits installation as a service.



Jenkins Lint: docs

JL-1: JobNameChecker

Description: When creating Jenkins Jobs you must avoid whitespace. In order to comply with the style guide.

Severity: High

Control Comment: To disable this checker you can add `lint:ignore:JobNameChecker` to your Jenkins project.

JL-2: JobDescriptionChecker

Description: Jenkins project description might help you to know what it does and further details.

Severity: Medium

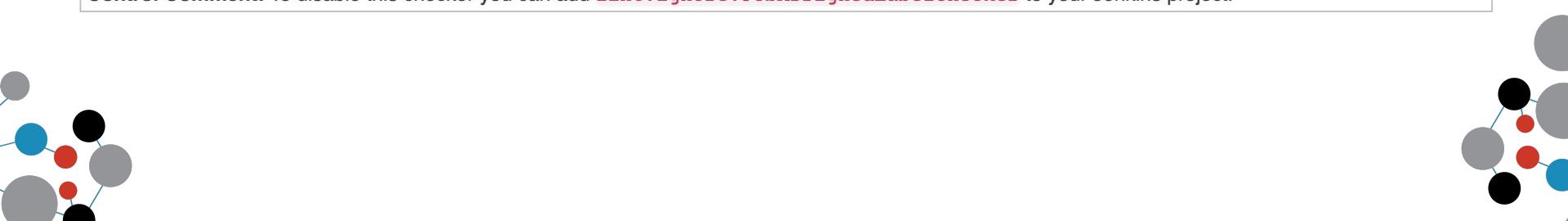
Control Comment: To disable this checker you can add `lint:ignore:JobDescriptionChecker` to your Jenkins project.

JL-3: JobAssignedLabelChecker

Description: When setting Jenkins Jobs you should set where those Jobs can run.

Severity: Medium

Control Comment: To disable this checker you can add `lint:ignore:JobAssignedLabelChecker` to your Jenkins project.





Jenkins Lint: disable lints

Project My FreeStyle

```
lint:ignore:JobLogRotatorChecker  
lint:ignore:JobAssignedLabelChecker
```

 [edit description](#)



Jenkins Lint: disable lints

Project My FreeStyle

```
lint:ignore:JobLogRotatorChecker  
lint:ignore:JobAssignedLabelChecker
```

[edit description](#)

Jenkins > My FreeStyle > Jenkins Lint

- [Back to Dashboard](#)
- [Status](#)
- [Changes](#)
- [Workspace](#)
- [Build Now](#)
- [Delete Project](#)
- [Configure](#)
- [Jenkins Lint](#)
- [Failure Cause Management](#)
- [Failure Scan Options](#)

JenkinsLint: Project My FreeStyle

The [JenkinsLint Plugin](#) has mainly two goals:

- To make it easier to detect issues in your Jenkins configuration that will cause Jenkins to blow up when triggered.
- To encourage discussion within the Jenkins community on the more subjective stuff. Having a set of checks that can be easily disabled.

Status ↓	Defect Name	Description
High	JobNameChecker	When creating Jenkins Jobs you must avoid whitespace in the name.
Medium	TimeoutChecker	There are some builds which might hang forever on the build timeout plugin .
Low	NullSCMChecker	Jenkins works fine with cron/batch tasks, Its strong on CI.
Low	CleanupWorkspaceChecker	There are some builds which demand a lot of disk space. Its recommended to wipe out those workspaces at regular intervals.
Low	JobLogRotatorChecker	When setting Jenkins Jobs with some Log Rotator, its good to have a large amount of disk space and speed up Jenkins UI.
Low	JobAssignedLabelChecker	When setting Jenkins Jobs you should set where Jenkins can assign labels.



Jenkins World
A global DevOps event
2017

Jenkins ➤ configuration

Jenkins Lint

Enabled



If Jobs should be scanned or not.

Enable analysis of disabled Jobs



Whether to lint disabled jobs or not.

Enable Job floating box



Whether to display JobAction floatingBox or not.

Jenkins Lint Checkers

Artifact Checker



BFA Checker



Cleanup Workspace Checker



Git Ref Checker





Future

- Fully Pipeline support.
- Jenkins lint command line tool to lint Pipelines and JobDSL.
- Auto-fix trivial lint defects, such as:
 - Log rotator
 - Groovy sandbox
 - Workspace cleanup
- Disable projects automatically.
- Code Quality integration as a single source of truth.



Want to try Jenkins Lint?

<https://plugins.jenkins.io/jenkinslint>



Q&A



Jenkins World
A global DevOps event
2017





Jenkins World

A global DevOps event

2017

