



DevOps World



Jenkins World

“Look ma, no hands”
Jenkins Configuration-as-Code

Who are we?

Name: Ewelina Wilkosz

Work: IT Consultant @ Praqma



Jenkins World 2018 Awards

Most Valuable Jenkins Contributor



@ewelinawilkosz



@ewelinawilkosz



ewe@pragma.com

Who are we?

Name: Nicolas De Loof

Work: Hacker @ CloudBees

Jenkins contributor & Docker Captain

Conference organizer



Nicolas De loof @ndeloof

Seems I broke the CI (again). Let me force push to master...



@ndeloof



@ndeloof

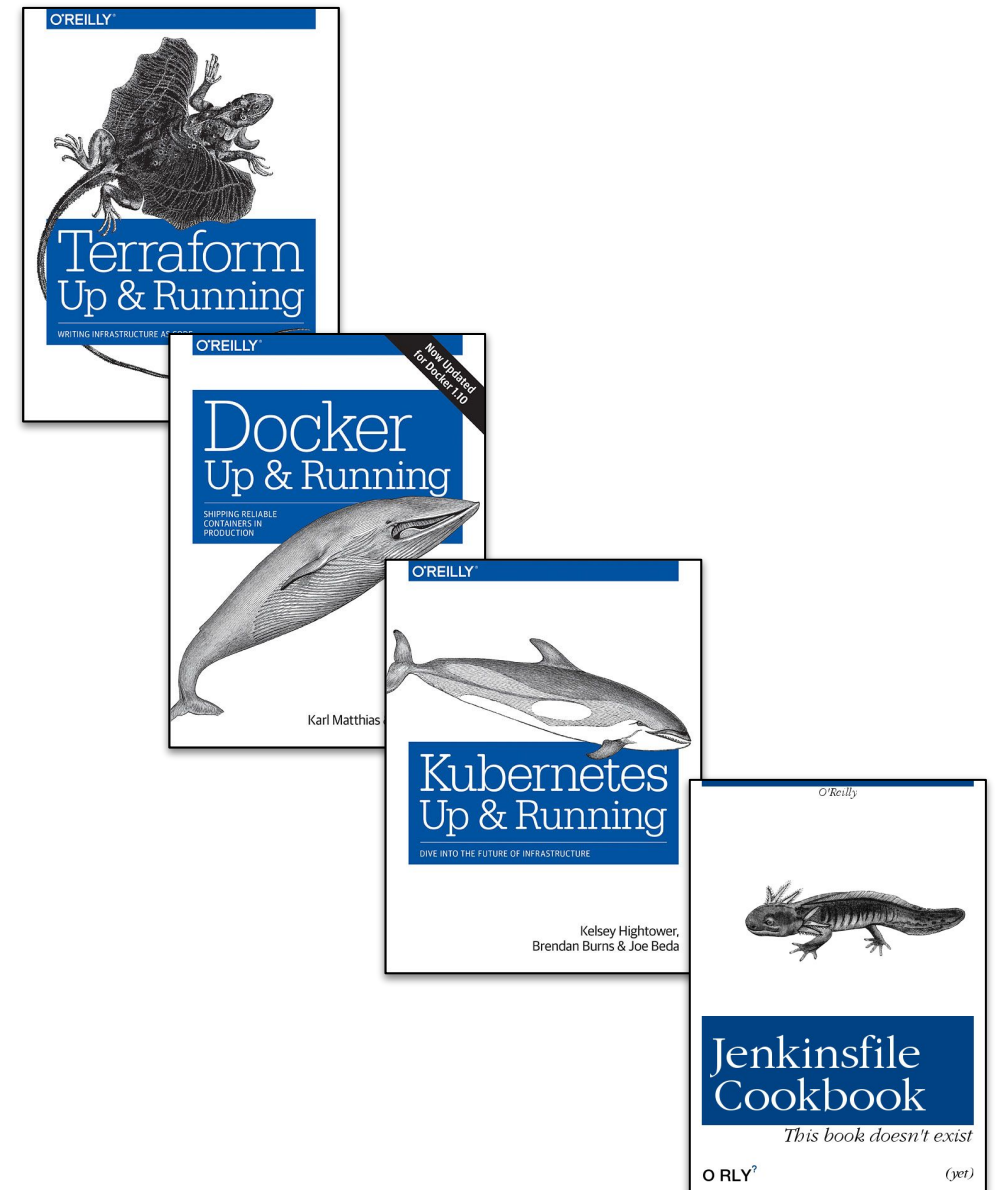


ndeloof@cloudbees.com



2018 is “* as code”

Infrastructure *as Code*
Environment *as Code*
Architecture *as Code*
CI/CD *as Code*



Manage Jenkins *as Code*



- Jenkins infrastructure
- Jenkins job configuration
- Jenkins system configuration



Jenkins infrastructure



Jenkins infrastructure

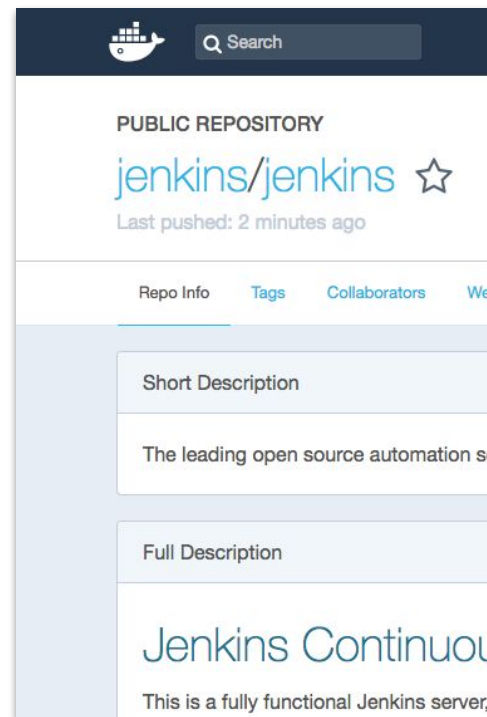
Using external tools

- Jenkins CLI
- REST API
- Python-jenkins
- Jenkins-client (Java, golang)
- ...

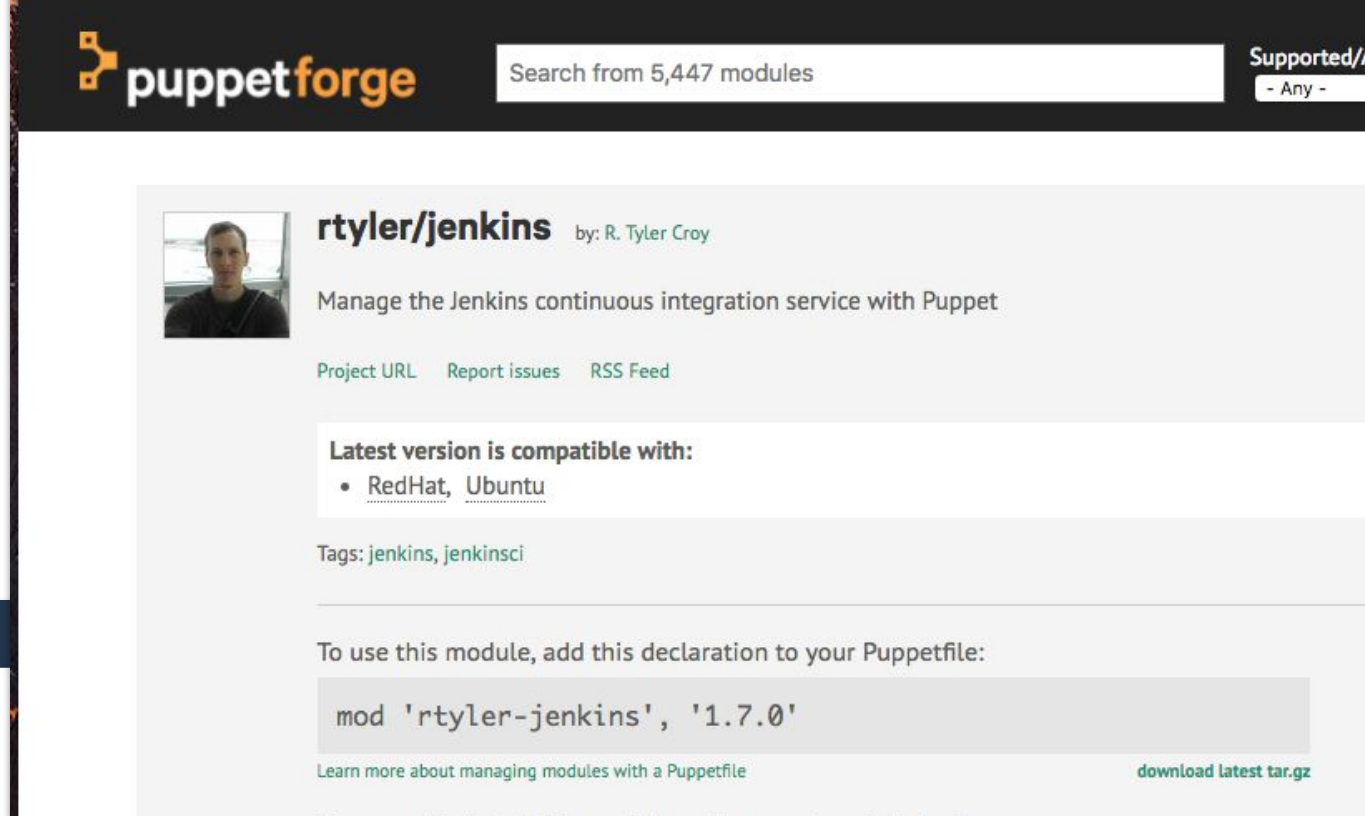
Jenkins infrastructure

Ansible, Chef, Puppet

Docker



A screenshot of the Jenkins/jenkins public repository page on Docker Hub. The page features a dark blue header with a search bar and a 'PUBLIC REPOSITORY' label. The repository name 'jenkins/jenkins' is displayed in blue with a star icon, and it notes 'Last pushed: 2 minutes ago'. Below the header, there are tabs for 'Repo Info', 'Tags', 'Collaborators', 'Webhooks', and 'Settings'. The 'Short Description' section states 'The leading open source automation server'. The 'Full Description' section describes it as a 'Jenkins Continuous Integration and Delivery server' based on weekly and LTS releases.



A screenshot of the puppetforge module page for 'rtyler/jenkins' by R. Tyler Croy. The page includes a search bar at the top with 'Search from 5,447 modules' and a 'Supported/' dropdown. The module description states 'Manage the Jenkins continuous integration service with Puppet'. It provides links for 'Project URL', 'Report issues', and 'RSS Feed'. A section titled 'Latest version is compatible with:' lists 'RedHat' and 'Ubuntu'. The 'Tags' section shows 'jenkins' and 'jenkinsci'. A code block shows the Puppetfile declaration: `mod 'rtyler-jenkins', '1.7.0'`. A link to 'download latest tar.gz' is also present.

jobs configuration



Jenkins job configuration

- JobDSL plugin (groovy)
- Job builder plugin (yaml)
- ...
- Jenkins Pipeline
 - Multibranch
 - Organizations folders

JobDSL

```
job('gr8 example') {  
    scm {  
        github 'sheehan/job-dsl-gradle-example'  
    }  
    triggers {  
        scm 'H/5 * * * *'  
    }  
    steps {  
        gradle 'clean test'  
    }  
    publishers {  
        archiveJUnit 'build/test-results/**/*.*xml'  
        extendedEmail 'mr.sheehan@gmail.com'  
    }  
}
```



Jenkins master configuration



“Jenkins can be installed through native system packages, Docker, or run standalone by any machine with a Java Runtime Environment (JRE) installed...”

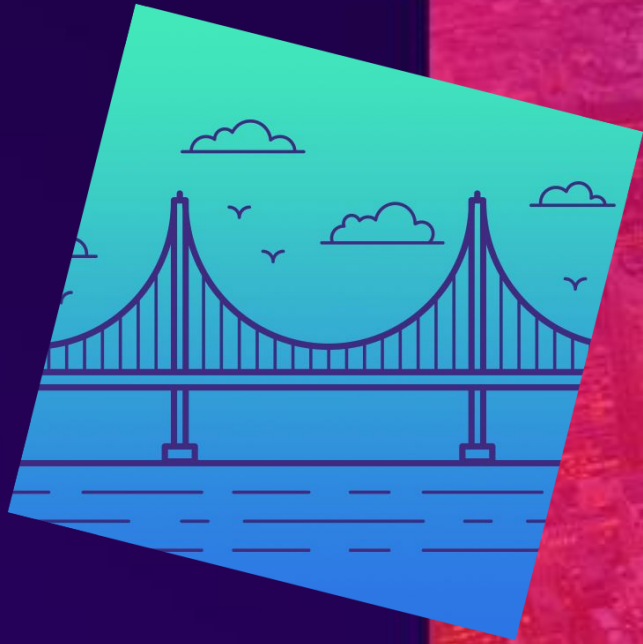
--- an enthusiast Jenkins user



“... but it has to be
configured
manually”

--- a not so enthusiast Jenkins user

loong scroll down



And we don't
(always) like that



- New Item
- People
- Build History
- Manage Jenkins
- My Views
- Credentials
- New View

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Home directory

/var/jenkins_home

System Message

Advanced...

Welcome to the demo setup for Jenkins Configuration as Code plugin. For more information look in the official repo with our demo setup: <https://github.com/ewelinawilkosz/pragma-jenkins-casc>

[Plain text] [Preview](#)

Maven Project Configuration

Global MAVEN_OPTS

Local Maven Repository

of executors

Labels

Usage

Quiet period

SCM checkout retry count

☐ Restrict project naming

Global properties

☐ Environment variables

☐ Tool Locations

SonarQube servers

Environment variables

SonarQube installations

☐ Enable SonarQube server configuration as build environment variables

If checked, SonarQube servers will be able to inject a SonarQube server configuration as environment variables in the build.

Name

URL

Server version

Server authentication token

SonarQube account login

SonarQube account password

Sonar566

http://SERVER_URL/

Server version is http://localhost:9000

higher

Fields depend on the SonarQube server version.

SonarQube authentication token. Mandatory when anonymous access is disabled.

SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.

SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.

Advanced...

Delete SonarQube

Add SonarQube

List of SonarQube installations

Compiler Warnings

Parsers

Name

Link name

Example parser

Example parser link

Display name of the parser. If the name already exists then this parser is added to the set of parsers that share the same name.

Text of the hyper link that references the results of this parser.



So how do we solve it?

Jenkins system configuration

- `init.groovy`
- `scriptler`
- `scm-sync-configuration` 🤖


```
1 // Initializes the Development folder, which is fully configurable by the user
2
3 import groovy.io.FileType
4 import com.synopsys.arc.jenkins.plugins.ownership.OwnershipDescription
5 import hudson.plugins.filesystem_scm.FSSCM
6 import jenkins.model.Jenkins
7 import com.cloudbees.hudson.plugins.folder.Folder
8 import org.jenkinsci.plugins.ownership.model.folders.FolderOwnershipHelper
9 import org.jenkinsci.plugins.workflow.cps.CpsFlowDefinition
10 import org.jenkinsci.plugins.workflow.cps.CpsScmFlowDefinition
11 import org.jenkinsci.plugins.workflow.job.WorkflowJob
12 import org.jenkinsci.plugins.workflow.libs.FolderLibraries
13 import org.jenkinsci.plugins.workflow.libs.LibraryConfiguration
14 import org.jenkinsci.plugins.workflow.libs.SCMRetriever
15
16 println("=== Initialize the Development folder")
17 if (Jenkins.instance.getItem("Development") != null) {
18     println("Development folder has been already initialized, skipping the step")
19     return
20 }
21
22 // Admin owns the root Development folder
23 def folder = Jenkins.instance.createProject(Folder.class, "Development")
24 FolderOwnershipHelper.setOwnership(folder, new OwnershipDescription(true, "admin"))
25
26 // Users get their own sandboxes
27 def folder2 = folder.createProject(Folder.class, "User")
28 FolderOwnershipHelper.setOwnership(folder2, new OwnershipDescription(true, "user"))
29
30 // Create a library for local Jenkins Pipeline Library Development
31 // if the Env Var is set and the directory is mapped
32 println("==== Initializing local Pipeline Library development dir")
33 File file = new File("/var/jenkins_home/pipeline-library/vars")
34 if (!file.exists()) {
35     println("/var/jenkins_home/pipeline-library is not mapped, skipping")
36     return
37 } else {
38     println("/var/jenkins_home/pipeline-library is mapped, initializing the directory")
39 }
40 }
```

We're not alone



- JENKINS-31094 (system-config-dsl)
- XML templating (seen at JenkinsWorld 2017)
- Various Groovy bindings
- Praqma's "JenkinsAsCodeReference"
- CloudBees CTO Office's prototype



to **join forces**

- Both had working prototypes last summer
- Pragma focusing on:
 - **real world** usage by customers
- CloudBees focusing on:
 - **community adoption**
 - out-of-the box support for our products



⇒ <https://github.com/jenkinsci/configuration-as-code-plugin>



Let's make it as
easy as possible

jenkins.yaml

```
jenkins:
  systemMessage: "JCasC Demo"
  numExecutors: 1
  scmCheckoutRetryCount: 4
  mode: NORMAL
  securityRealm:
    local:
      allowsSignup: false
      users:
        - id: demoAdmin
          password: ${adminpw}
```



The screenshot shows the Jenkins web interface. At the top, there's a navigation bar with the Jenkins logo and a search bar. Below it, a sidebar contains links: 'New Item', 'People', 'Build History', 'Manage Jenkins' (highlighted), 'My Views', 'Credentials', and 'New View'. The main content area is titled 'Manage Jenkins' and features a blue banner announcing the new version 2.107.2. Below the banner are several configuration sections: 'Configure System', 'Configure Global Security', 'Configure Credentials', 'Global Tool Configuration', 'Reload Configuration from Disk', and 'Manage Plugins'. The 'System Message' section is expanded, showing a text area with 'JCasC Demo' and a 'Preview' button. The 'Maven Project Configuration' section is also visible, showing fields for 'Global MAVEN_OPTS', 'Local Maven Repository', '# of executors' (set to 1), 'Labels' (set to 'master-label'), 'Usage' (set to 'Use this node as much as possible'), 'Quiet period' (set to 5), 'SCM checkout retry count' (set to 4), and a checkbox for 'Restrict project naming'.

Main benefits

- Safety
- Traceability
- Speed
- Easy to use
- Easy to reuse

There are challenges

- *Manage configuration as **human-readable** config file(s)*
- *Self-describing model to **reflect Web UI***
- *Configure **all** jenkins initial setup (including plugins)*
- *Support most (*) plugins **without** extra development effort*
- *Generate **documentation and validation** tools (schema)*

human-readable config file(s)

- Structured content
- Nothing language centric
 - No groovy / ruby / xx
- Readable and commentable

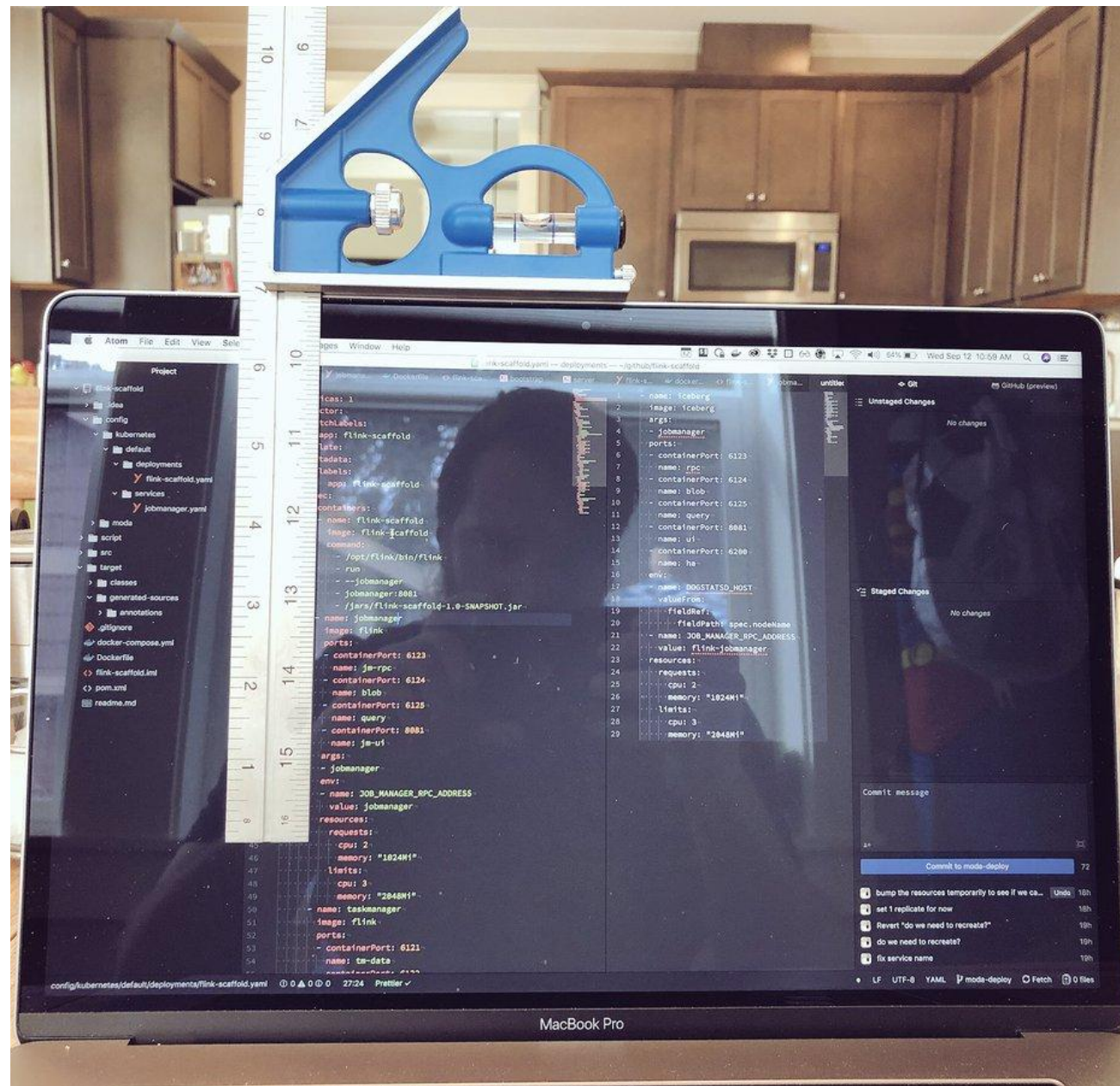


YAML ...

BEWARE...

Indentation matters

photo credit :
Justin Palmer @Caged



Web UI as implicit documentation

Config element in web UI

==

Config element in YAML



“No need to be a Jenkins expert to do it right”

-- Obi Wan Kenobi

Configure Jenkins in yaml

Obvious, isn't it ?

```
jenkins:
  securityRealm:
    ldap:
      configurations:
        - server: ldap.acme.com
          rootDN: dc=acme,dc=fr
          managerPasswordSecret: ${LDAP_PASSWORD}
      cache:
        size: 100
        ttl: 10
      userIdStrategy: CaseSensitive
      groupIdStrategy: CaseSensitive

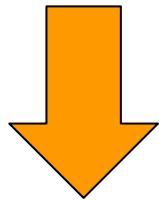
  tool:
    git:
      installations:
        - name: git
          path: /bin/git
```


Configure ALL jenkins initial setup

No hand on keyboard

No click on web UI

to deploy



a fully working Jenkins master



Support ALL plugins

- No need to write glue code for **every** supported plugin
- Most(*) plugins supported out of the box
- Others can bundle adapter code

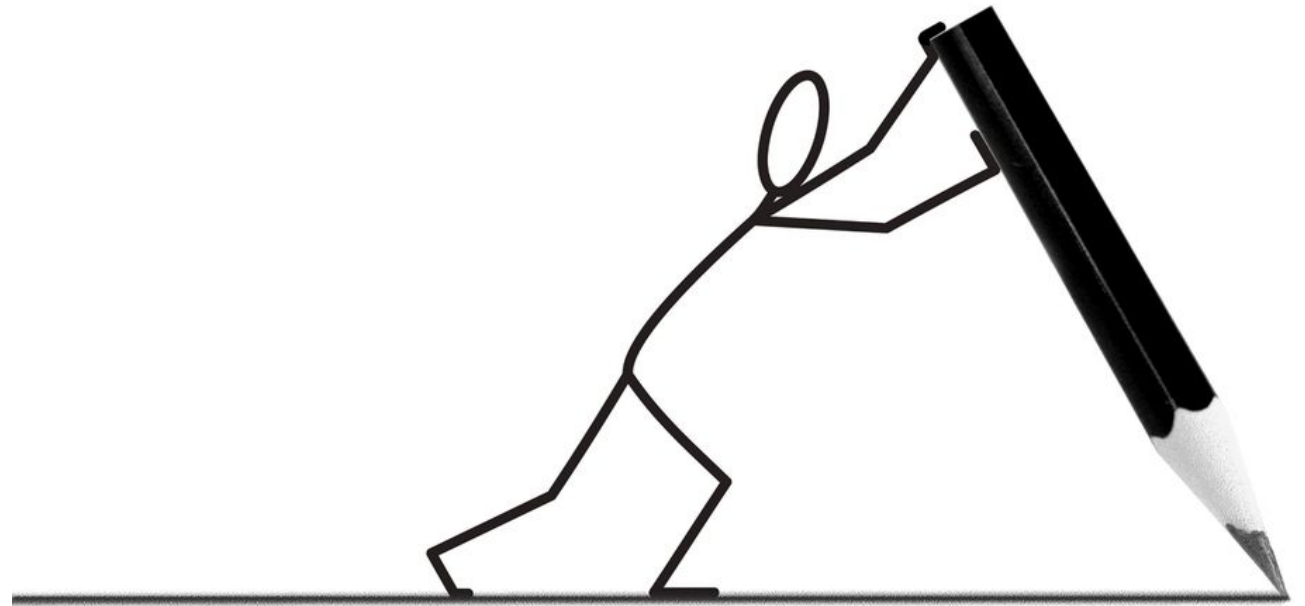
!! we require configuration-as-code-support plugin to be installed, for now !!

(*) could require some minor changes



Generate documentation and validation tools

- Can validate without running a test master
- IDE support





Here comes JCasC

Where to find it?!

<https://github.com/jenkinsci/configuration-as-code-plugin>

[Implementation details](#) and guide for [plugin developers](#) available in plugin's github repository



DEMO 🥰💧

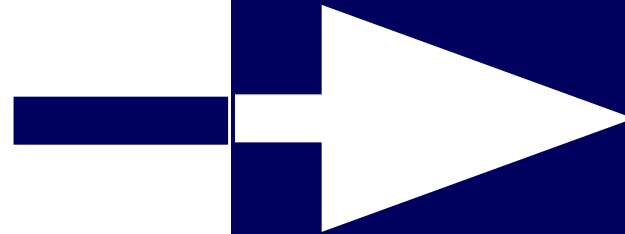




How it works

Live Jenkins instance

Core + plugins



Data model

- Yaml parser
- Doc generator
- Schema validator

Introspection

Jenkins-core 2.xx + plugins [git:3.7.0, ...]

- Jenkins root instance
- Descriptors (global configuration)
- + Special component with CasC support

=> hierarchical data model, trying to mimic Jenkins UI

Requirements

Target components need to follow some basic design rules

We rely on UI data binding mechanism (@DataBound)

Component to directly parse StaplerRequest / JsonObject can't be introspected

- Recommendations to plugin developers
github.com/jenkinsci/configuration-as-code-plugin/blob/master/PLUGINS.md
- Pull requests on major plugins we want to support
github.com/jenkinsci/mailer-plugin/pull/39

Doc/Schema Generation

JENKINS/plugin/configuration-as-code/

Jenkins Configuration as Code Reference

jenkins

agentProtocols

list of String

authorizationStrategy

AuthorizationStrategy

- [loggedInUsersCanDoAnything](#)
- [legacy](#)
- [projectMatrix](#)
- [globalMatrix](#)
- [unsecured](#)

clouds

list of Cloud

crumbIssuer

CrumbIssuer

- [standard](#)

disableRememberMe

boolean

labelString

String

markupFormatter

In such places as project description, user description, view description, and build description, Jenkins configuration determines how such free-form text is converted to HTML. By default, Jenkins treats the of the backward compatibility.)

While this is convenient and people often use it to load <iframe>, <script>, and so on to mash up data

JENKINS/plugin/configuration-as-code/schema

```
{
  $schema: "http://json-schema.org/draft-06/schema#",
  id: "http://jenkins.io/configuration-as-code#",
  description: "Jenkins Configuration as Code",
  type: "object",
  - properties: {
    - jenkins: {
      type: "object",
      ref: "#/definitions/jenkins.model.Jenkins"
    },
    - credentials: {
      type: "object",
      ref: "#/definitions/com.cloudbees.plugins.credentials.CredentialsStore"
    },
    - : {
      type: "object",
      ref: "#/definitions/com.cloudbees.plugins.credentials.GlobalCredentialsConfiguration$Category"
    },
    - security: {
      type: "object",
      ref: "#/definitions/jenkins.model.GlobalConfigurationCategory$Security"
    },
    - unclassified: {
      type: "object",
      ref: "#/definitions/jenkins.model.GlobalConfigurationCategory$Unclassified"
    },
    - Tool: {
      type: "object",
      ref: "#/definitions/jenkins.tools.ToolConfigurationCategory"
    },
    - mavenmoduleset: {
      type: "object",
      ref: "#/definitions/hudson.maven.MavenModuleSet$DescriptorImpl"
    },
    - masterBuild: {
      type: "object",
      ref: "#/definitions/jenkins.model.MasterBuildConfiguration"
    },
    - quietPeriod: {
      type: "object",
      ref: "#/definitions/jenkins.model.GlobalQuietPeriodConfiguration"
    },
  },
}
```

Corner cases

Some components hardly fit this model

For those we can develop dedicated Configurator adapter classes.



Under the hood



Root Elements → RootElementConfigurator

```
jenkins:
  securityRealm:
    ldap:
      configurations:
        - server: ldap.acme.com
          rootDN: dc=acme,dc=fr
          managerPasswordSecret: ${LDAP_PASSWORD}
      cache:
        size: 100
        ttl: 10
        userIdStrategy: CaseSensitive
        groupIdStrategy: CaseSensitive
  tool:
    git:
      installations:
        - name: git
          path: /bin/git
```

Root Element

- JenkinsConfigurator
“jenkins” → Jenkins.instance root object
- GlobalConfigurationCategoryConfigurator
“tools”, “security”, ... → Descriptors grouped by categories
- DescriptorRootElementConfigurator
Uncategorized Descriptors with a global configuration page
“mailer”, ...
- CredentialsRootConfigurator
“credentials” → Glue code for credentials plugin (more on this later)

Child element → Attribute

```
jenkins:
  securityRealm:
    ldap:
      configurations:
        - server: ldap.acme.com
          rootDN: dc=acme,dc=fr
          managerPasswordSecret: ${LDAP_PASSWORD}
      cache:
        size: 100
        ttl: 10
      userIdStrategy: CaseSensitive
      groupIdStrategy: CaseSensitive
```

Attribute

Configurator do describe a target component as a set of Attributes

Attribute handle :

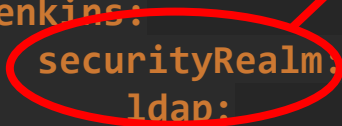
- Name
- Type (inferred by reflection on generics)
- Multiplicity (Collection<x>)
- Setting value

Generic Attribute

writable JavaBean property | DataBound constructor parameter

```
public void setSecurityRealm(SecurityRealm securityRealm) {
```

```
jenkins:  
  securityRealm:  
    ldap:
```



SecurityRealm is an ExtensionPoint (abstract)

Configuration-as-Code need to identify implementation

Extension point implementation

SecurityRealm is an ExtensionPoints => candidates implementations:

LegacySecurityRealm

→ @Symbol("legacy") → **legacy**

HudsonPrivateSecurityRealm

→ @Symbol("local") → **local**


ActiveDirectorySecurityRealm

→ ActiveDirectory → **activedirectory**

LDAPSecurityRealm

→ LDAP → **ldap**

jenkins:
securityRealm:
 ldap:




Build target Component

```
@DataBoundConstructor public LDAPSecurityRealm(  
    List<LDAPConfiguration> configurations,  
    boolean disableMailAddressResolver,  
    CacheConfiguration cache,  
    IdStrategy userIdStrategy,  
    IdStrategy groupIdStrategy)
```

+ DataBoundSetters

```
jenkins:  
  securityRealm:  
    ldap:  
      configurations:  
        ...  
      cache:  
        size: 100  
        ttl: 10  
      userIdStrategy: CaseSensitive  
      groupIdStrategy: CaseSensitive
```



Corner cases

- Setter method defined for internal needs / backward compatibility
We exclude `@Deprecated` and `@Restricted`
- [WiP] Technical facing Property name : “labelString”
We support `@Symbol` on setters
- Not a Describable / Internal model is ... weird for end-user
Custom Configurator | Attribute implementation

Custom Configurator, a.k.a “Glue Code”

Sample : expose a **user-friendly** credentials model

```
credentials:
  system:
    domainCredentials:
      # global credentials
      - credentials:
        - certificate:
          scope: SYSTEM
          id: ssh_private_key
          password: ${SSH_KEY_PASSWORD}
          keyStoreSource:
            fileOnMaster:
              keyStoreFile: /docker/secret/id_rsa
```

CredentialsRootConfigurator
custom code

A fake **Attribute** "system"
to expose **DomainCredentials** (List)
with custom setter implementation:

```
target.setDomainCredentialsMap(
    DomainCredentials.asMap(value)
)
```



Status

1.0 is there ! ... even 1.3 (released last week)

We welcome Feedback !

- [jenkins-users mailing list](#)
- [jenkinsci/configuration-as-code gitter](#)
- [github issues](#)



Features

- Read configuration from local drive or url, REST API or CLI
- Reload configuration (Manage Jenkins → Configuration as Code → Reload)
- Export existing jenkins instance configuration into yaml (*here be dragons*)
- Compatibility dashboard :

<https://issues.jenkins-ci.org/secure/Dashboard.jspa?selectPageId=17346>

Please report issues with “`jcasc-compatibility`” label

- + Additionally docker demo setup (which can be easily adapted for different than demo purpose): <https://github.com/Praqma/pragma-jenkins-casc>

JEP-201

Make this **THE** configuration component for Jenkins community

<https://github.com/jenkinsci/jep/blob/master/jep/201/README.adoc>





Give it a try

Report missing plugin support / broken features

Contribute test cases (easy) or fixes (not so easy :P)

[chat](#) [on gitter](#)

How to talk to us?

- github issues working well for reporting problems
- we're monitoring Jenkins Users, Jenkins Developers mailing lists

but...

- **gitter** channel is a place to go to:
<https://gitter.im/jenkinsci/configuration-as-code-plugin>



Questions?



DevOps World



Jenkins World

Thank you!

