Hudson Plugin Categories and Usage Analysis

Winston Prakash

Jan 2013

The slides in this document present Hudson Plugin Categories based on their importance in a Continuous Integration System and usage by community.

See also
http://wiki.hudson-ci.org/display/HUDSON/All+Plugins+by+Topic

Successful Agile Team

The three main concepts,

- Test Driven Development
- Continuous Integration
- Continuous Delivery or Deployment

are the solid supporting pillars of a Successful Agile Team.

In a Test Driven Development build pipeline, Continuous Integration is the first step and the end result is the Continuous Delivery.

Best tool of the trade for all the above is **HUDSON**

Continuous Integration

Martin Fowler in his landmark article described Continuous Integration as

"*Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible*."

Ten commandments of continuous integration

- *Maintain a Single Source Repository.*
- *Automate the Build*
- *Make Your Build Self-Testing*
- *Everyone Commits To the Mainline Every Day*
- *Every Commit Should Build the Mainline on an Integration Machine*
- *Keep the Build Fast*
- *Test in a Clone of the Production Environment*
- *Make it Easy for Anyone to Get the Latest Executable*
- *Everyone can see what's happening*
- *Automate Deployment*

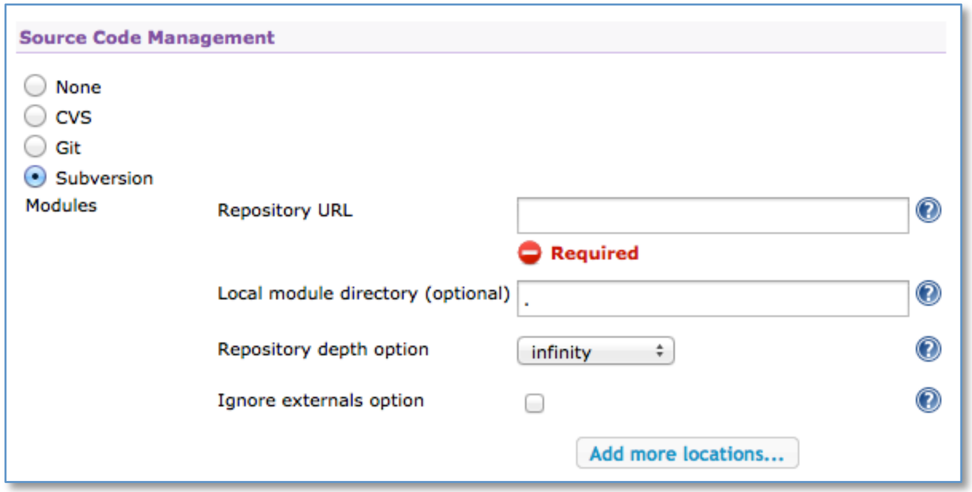
Rest of the slides categorize Hudson Plugins based on these guidelines

Maintain a Single SCM

This principle encourages the project team to use SCM to maintain their source code. Hudson supports various SCMs via plugins.

99% of Hudson users use one of

- Git
- CVS
- SVN
- Perforce
- Clearcase
- Mercurial


Source Code Management

None [ ]
CVS [ ]
Git [ ]
Subversion [✓]

Modules

Repository URL	Required
Local module directory (optional)	.
Repository depth option	infinity
Ignore externals option	☐

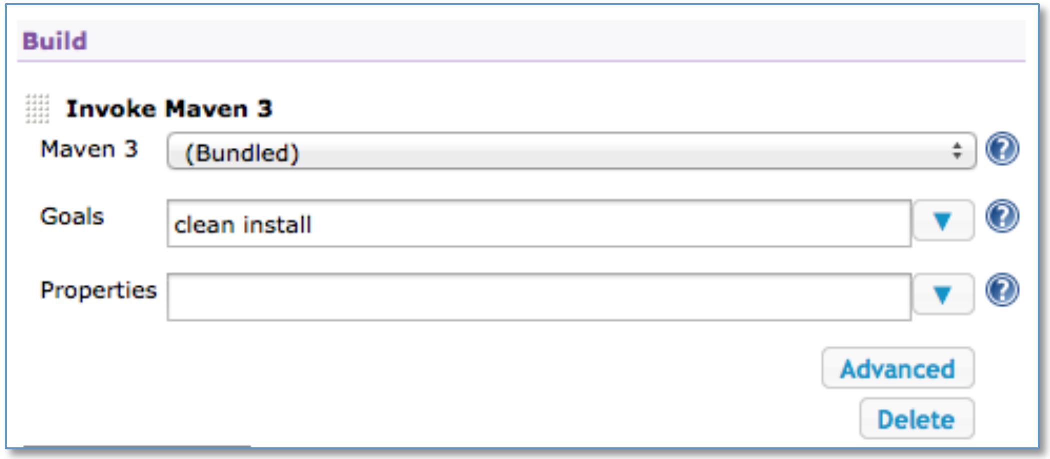
Hudson supports ~20 additional SCM which are used by less than 1% of the users

Automate the Build

Automating the build using a single command is an important principle of a CI build. Hudson supports various Build Tools via Plugins.

99% of Hudson users use one of

- Ant
- maven
- gradle
- MSBuild
- Nant
- Rake


Build

Invoke Maven 3

Maven 3	(Bundled)
Goals	clean install
Properties	

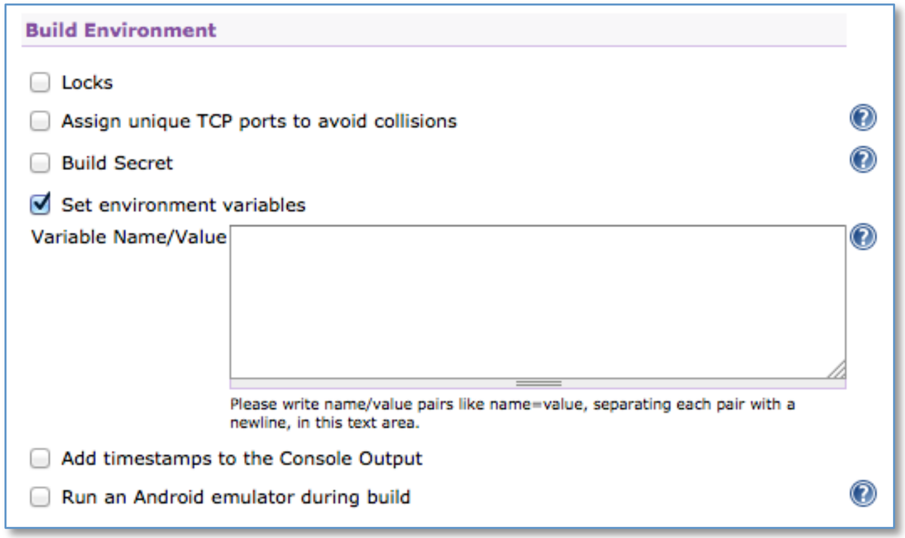
Hudson supports ~40 additional build tools which are used by less than 1% of the users

Automate the Build (Cont..)

Before the build starts, the build must be prepared for proper build. Hudson must support various type Build Wrappers. They can be used to post process the build results. Hudson supports various Build wrappers via Plugins. Build wrappers are usually very specific to the Software Project.

Few useful plugins used by 99% of the users are

- Lock & Latch
- Setenv
- EnvInject


Build Environment

- [ ] Locks
- [ ] Assign unique TCP ports to avoid collisions
- [ ] Build Secret
- [x] Set environment variables

Variable Name/Value

Please write name/value pairs like name=value, separating each pair with a newline, in this text area.

- [ ] Add timestamps to the Console Output
- [ ] Run an Android emulator during build

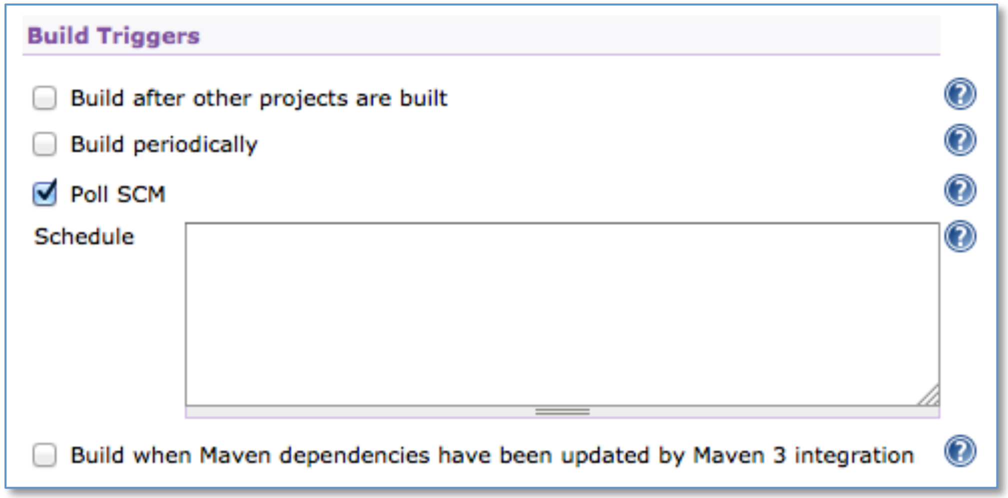
Hudson supports ~25 additional build wrappers which are used by less than 1% of the users

Every commit should build the mainline on an integration machine

Automating the build based on user commit is part of CI. Hudson supports various Build Triggers via plugins.

99% of Hudson users use one of

- SCM trigger
- Upstream/Downstream
- Gerrit Trigger
- URL Change
- Scheduled
- Parameterized Trigger



Build Triggers

- [ ] Build after other projects are built
- [ ] Build periodically
- [x] Poll SCM

Schedule

- [ ] Build when Maven dependencies have been updated by Maven 3 integration

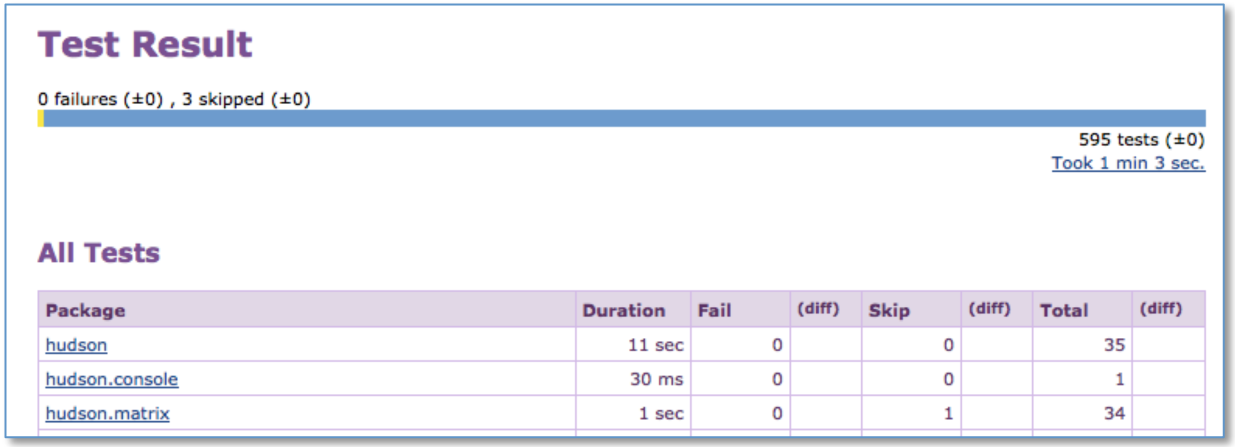
Hudson supports ~15 additional build triggers which are used by less than 1% of the users

Make your build self-testing

CI build is not about catching compilation error but also catching bugs more quickly and efficiently. Hudson supports various Unit Testing Frameworks via Plugins.

99% of Hudson users use one of

- jUnit
- nUnit
- Selenium
- CppUnit
- TestNg
- xUnit



Test Result

0 failures (±0) , 3 skipped (±0)

595 tests (±0)
Took 1 min 3 sec.

All Tests

Package	Duration	Fail	(diff)	Skip	(diff)	Total	(diff)
hudson	11 sec	0		0		35	
hudson.console	30 ms	0		0		1	
hudson.matrix	1 sec	0		1		34	

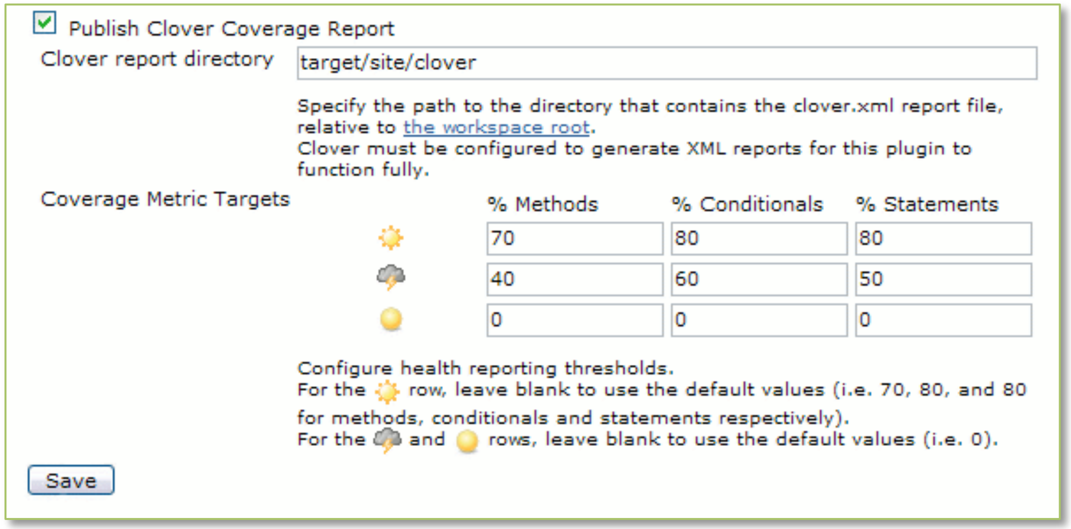
Hudson supports ~10 additional Unit Test which are used by less than 1% of the users

Make your build self-testing (Code Coverage)

Self testing is best achieved if there is uniform code coverage. Hudson supports various Code Coverage Tools via Plugins

99% of Hudson users use one of

- Clover
- Cobertura
- Emma
- Serenity
- Sonar
- NCover



✓ Publish Clover Coverage Report

Clover report directory: target/site/clover

Specify the path to the directory that contains the clover.xml report file, relative to the workspace root.
Clover must be configured to generate XML reports for this plugin to function fully.

Coverage Metric Targets

	% Methods	% Conditionals	% Statements
☀	70	80	80
⛈	40	60	50
●	0	0	0

Configure health reporting thresholds.
For the ☀ row, leave blank to use the default values (i.e. 70, 80, and 80 for methods, conditionals and statements respectively).
For the ⛈ and ● rows, leave blank to use the default values (i.e. 0).

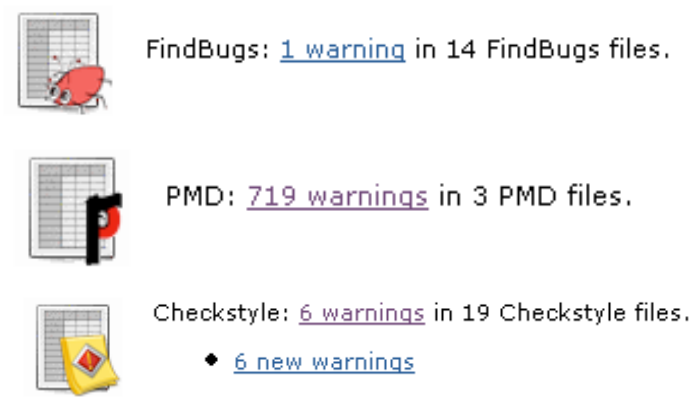
Hudson supports ~2 additional Code Coverage which are used by less than 1% of the users

Make your build self-testing (Code Analysis)

Static Analysis improves the confidence of Self testing. Hudson supports various Static Code Analysis Tools via Plugins

99% of Hudson users use one of

- Checkstyle
- PMD
- Dry
- Findbugs
- Crap4J
- Warnings
- CCM
- Violations


FindBugs: 1 warning in 14 FindBugs files.

PMD: 719 warnings in 3 PMD files.

Checkstyle: 6 warnings in 19 Checkstyle files.

- 6 new warnings

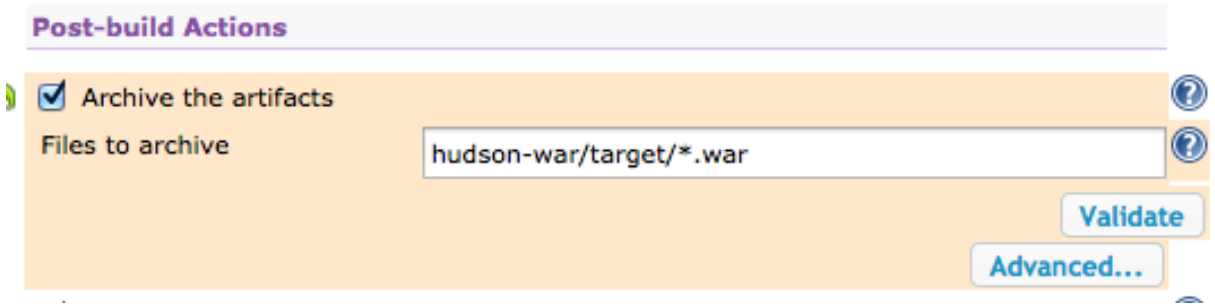
Hudson supports ~2 additional Code Analysis which are used by less than 1% of the users

Make it easy for everyone to get latest executable

Make the build artifacts to stakeholders is important in CI. Hudson supports various Artifact Uploaders via Plugins.

95% of Hudson users use one of

- CopyArtifacts
- SCP
- FTP Publisher
- Artifactory
- Maven Release
- HTML Publisher


Post-build Actions

✓ Archive the artifacts

Files to archive: hudson-war/target/*.war

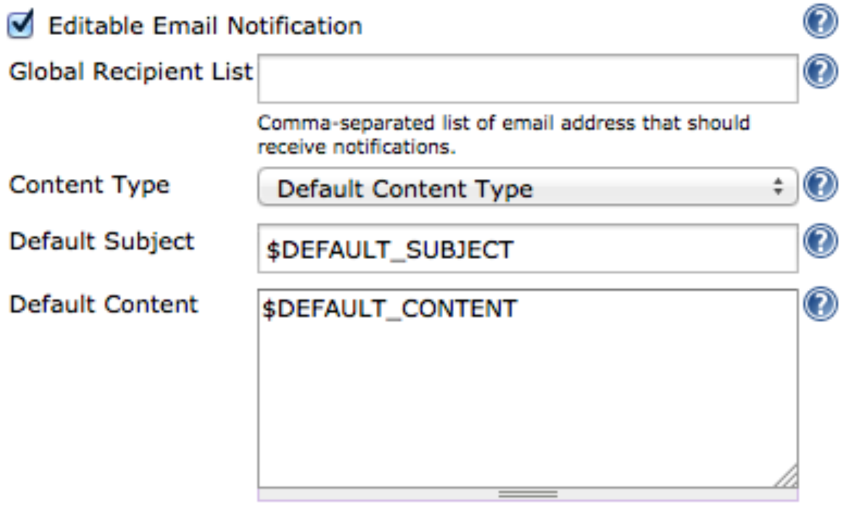
Hudson supports ~10 additional Artifact Uploaders used by less than 5% of the users

Everyone can see what is happening

Communicate the state of the build especially if it is broken . Hudson supports various Build Notifiers via plugins.

95% of Hudson users use one of

- Email
- Email-Ext
- IRC
- Jabber
- SMS



[✓] Editable Email Notification

Global Recipient List	
	Comma-separated list of email address that should receive notifications.
Content Type	Default Content Type
Default Subject	$DEFAULT_SUBJECT
Default Content	$DEFAULT_CONTENT

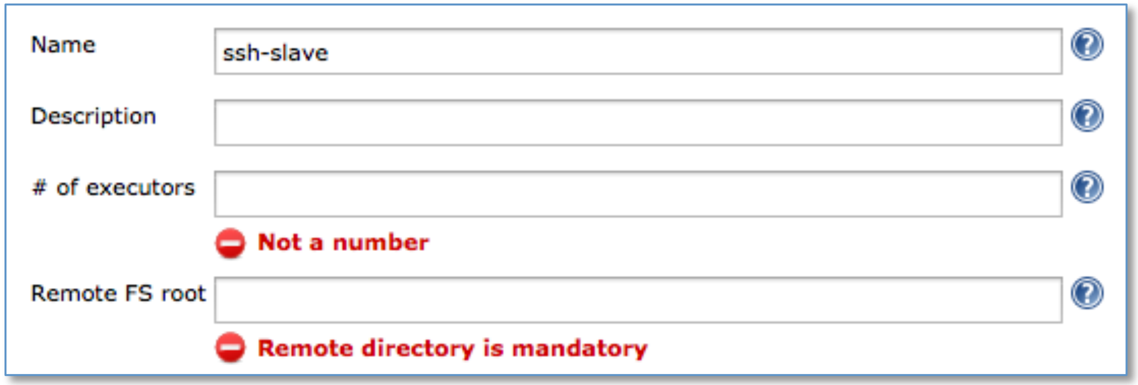
Hudson supports ~20 additional Build Notifiers used by less than 5% of the users

Test in a clone of the production environment

The build must happen in various slaves that clone the production environment. Hudson supports various kind of Slave Management via plugins so builds can happen in clones of production environment.

99% of Hudson users use one of

- SSH Slave
- Windows Slave
- Slave Status
- EC2
- Virtual Box
- JCloud



Name	ssh-slave
Description	
# of executors	
	Not a number
Remote FS root	
	Remote directory is mandatory

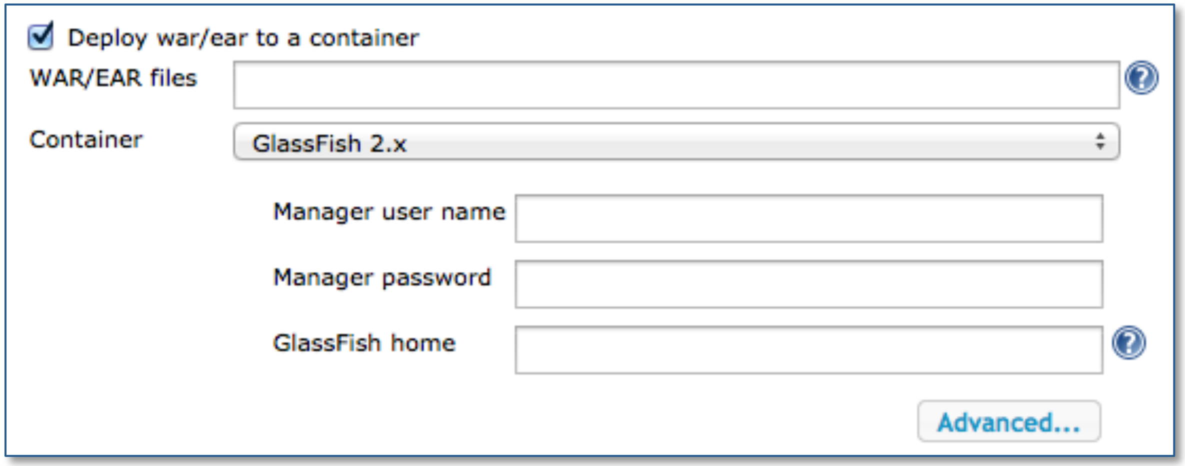
Hudson supports ~15 additional Slave Management Tools used by less than 1% of the users

Automate deployment

One of the CI best practices is to automate the deployment. Hudson supports various type of Deployment or External Tool Integration via plugins.

95% of Hudson users use one of

- Tomcat
- Websphere
- Weblogic
- Jboss
- JRebel
- IIS



☑ Deploy war/ear to a container

WAR/EAR files	
Container	GlassFish 2.x
Manager user name	
Manager password	
GlassFish home	

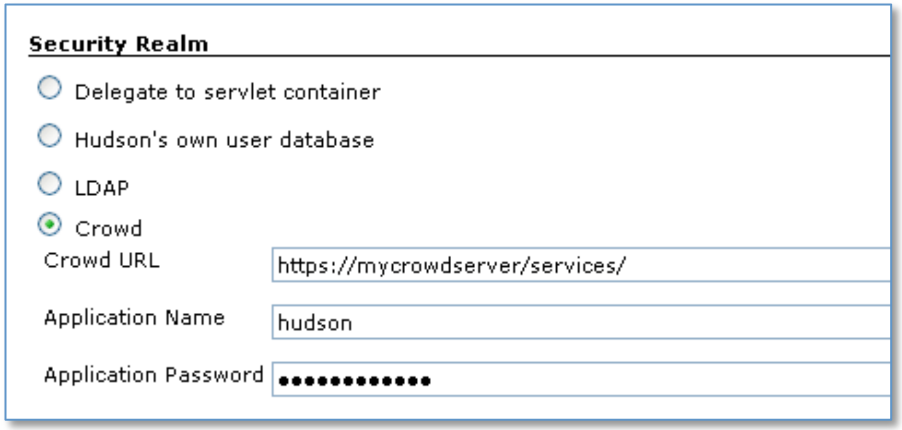
Hudson supports ~40 additional External Tool Integration used by less than 5% of the users

Hudson Security

Hudson supports various type of Authentication & Authorization mechanisms via plugins.

99% of Hudson users use one of

- LDAP
- Active Directory
- Crowd
- OpenId
- PwAuth (Unix)



Security Realm

- Delegate to servlet container
- Hudson's own user database
- LDAP
- [✓ Crowd]

Crowd URL	https://mycrowdserver/services/
Application Name	hudson
Application Password	●●●●●●●●●●●●●

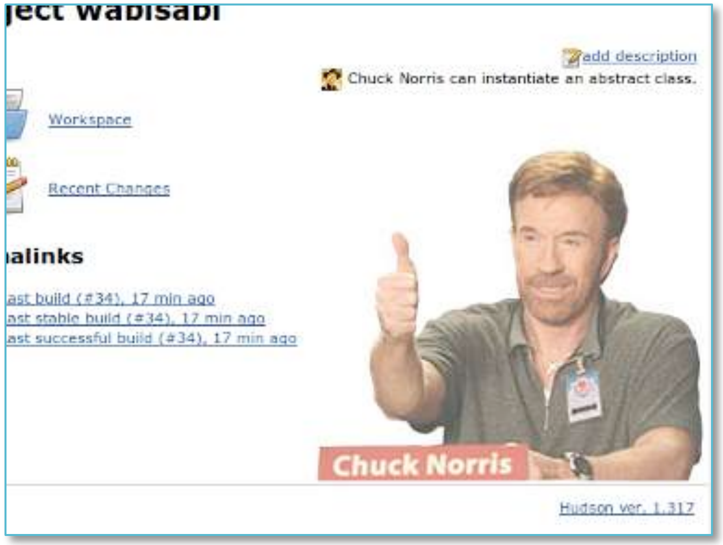
Hudson supports ~15 additional Authentication used by less than 5% of the users

Hudson UI Configuration

Various parts of Hudson UI can be configured or Augmented. These UI configurations are supported via Plugins.

95% of Hudson users use one of

- Chuck Norris
- Disk Usage
- Plot
- Build Dependency
- Radiator View
- Xfpanel
- Green Balls
- Build Pipeline
- Nested View
- Downstream Build View
- Dashboard View


ject wabisabi

add description

Chuck Norris can instantiate an abstract class.

Workspace

Recent Changes

alinks

ast build (#34), 17 min ago

ast stable build (#34), 17 min ago

ast successful build (#34), 17 min ago

Hudson ver. 1.317

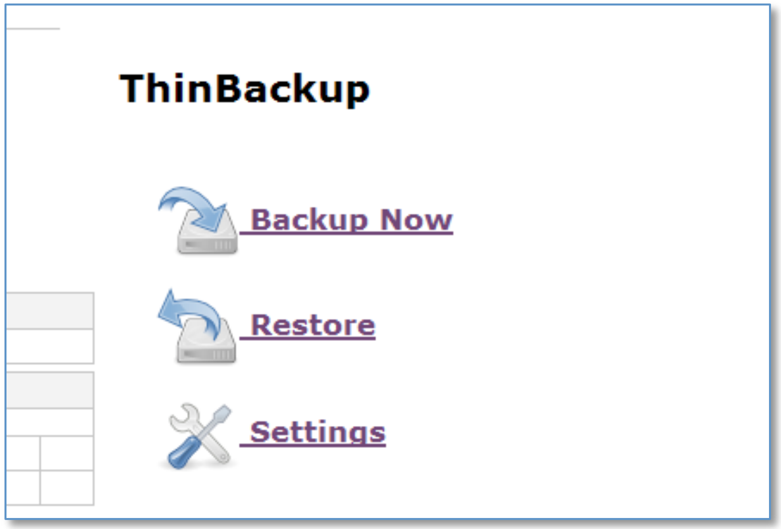
Hudson supports ~50 additional UI configurations used by less than 5% of the users

Hudson Utilities

Various Utilities, supported via Plugins can be added to Hudson to improve day to day CI activities.

95% of Hudson users use one of

- Translation
- Build timeout
- backup
- Log parser
- Audit Trail
- Status Monitor
- Global Build Stats
- Project Stats
- Slave Status



Hudson supports ~20 additional Utilities used by less than 5% of the users