Continuous integration with Jenkins CI

Vojtěch Juránek

JBoss - a division by Red Hat

17. 2. 2012, Developer conference, Brno

Outline

- What continuous integration (CI) is and why it’s useful.
- Show you, that CI with Jenkins is easy (Python and Ruby examples).
- Show you, that CI can be even more easy.

Continuous integration

When you are developing a piece of code, you probably

- compile the sources from time to time (if the code is compiled)
- check the functionality (run tests)

If something fails (compilation, tests etc.) you start to look for a wrong commit. . .

Is it better and more easy to try to find a mistake in one commit (several/several dozen changed lines of code) or many commits (hundreds/thousands changed lines)?

Continuous integration

When you are developing a piece of code, you probably

- compile the sources from time to time (if the code is compiled)
- check the functionality (run tests)

If something fails (compilation, tests etc.) you start to look for a wrong commit. . .

Is it better and more easy to try to find a mistake in one commit (several/several dozen changed lines of code) or many commits (hundreds/thousands changed lines)?

Continuous integration

When you are developing a piece of code, you probably

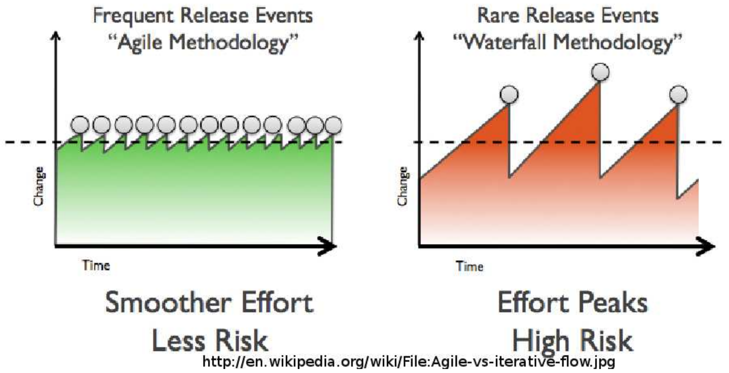
- compile the sources from time to time (if the code is compiled)
- check the functionality (run tests)

If something fails (compilation, tests etc.) you start to look for a wrong commit. . .

Is it better and more easy to try to find a mistake in one commit (several/several dozen changed lines of code) or many commits (hundreds/thousands changed lines)?

Continuous integration

Yup, as small change as possible is better!



Checking the status of the project very often (after each commit, if possible) is roughly what we call **continuous integration**.
CI is necessary when you use agile development methodology, but very useful even if you use any other development methodology.

Advantages

- Immediate feedback to any change.
- Immediate overview of the status of your project - test results, test coverage, performance etc. (helps with decisions, planning).
- Complete history of the project (test results, build artifacts, etc.).
- Can improve your workflow (e.g. gerrit integration).
- Can be very easily extended to continuous deployment and eventually continuous delivery.

Don’t I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

Don't I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

Don't I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

Don’t I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

Don't I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

Don't I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

Don't I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

Don't I spend too much time by running compilation and checking the results?

No, let the computer work instead of you. Hopefully, you time is more valuable than machine time:-)

Well, I can write some bash scripts which compile the code and run the tests. . .

Later on:
I should also automate

- *. . . checkout from SVN/git . . .*
- *. . . and setup some post commit hook to run it only after a commit . . .*
- *. . . analysis of test results . . .*
- *. . . some notification, to get alert only when something fails . . .*

. . . and I can also . . .

. . . wait, don't I re-invent the wheel??

HE SHOUTED OUT ABOUT SOME NEW INVENTION & THEN THAT THING HIT HIM!
http://blogs.technet.com/b/andrew/archive/2009/06/09/reinventing-the-wheel.aspx
DeepFat '09

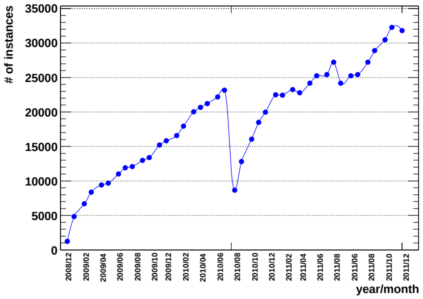
Yes, you do! There are several good CI servers. And probably the most popular is Jenkins.

Jenkins

- Kohsuke Kawaguchi started Hudson project in 2006 while working in Sun.
- Trademark and other issues after acquisition of Sun by Oracle.
- Community decided to rename Hudson to Jenkins in January 2011 (Hudson is still developed by Oracle and Sonatype, now moving under Eclipse foundation).
- Jenkins now affiliated with Software in the Public Interest (SPI) NPO
- All important information can be found at

  `https://wiki.jenkins-ci.org/display/JENKINS/Governance+Document`

- Jenkins usage grows steeply - more then 30k instances in anonymous usage statistics (i.e. actual # of instances is probably higher as not all instances send the stats.)
- 670+ repositories and 350 developer on GitHub!

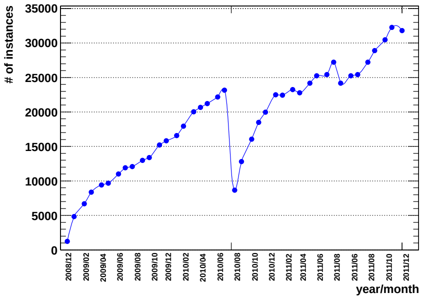


Jenkins

- Kohsuke Kawaguchi started Hudson project in 2006 while working in Sun.
- Trademark and other issues after acquisition of Sun by Oracle.
- Community decided to rename Hudson to Jenkins in January 2011 (Hudson is still developed by Oracle and Sonatype, now moving under Eclipse foundation).
- Jenkins now affiliated with Software in the Public Interest (SPI) NPO
- All important information can be found at

`https://wiki.jenkins-ci.org/display/JENKINS/Governance+Document`

- Jenkins usage grows steeply - more then 30k instances in anonymous usage statistics (i.e. actual # of instances is probably higher as not all instances send the stats.)
- 670+ repositories and 350 developer on GitHub!



Releases, packages

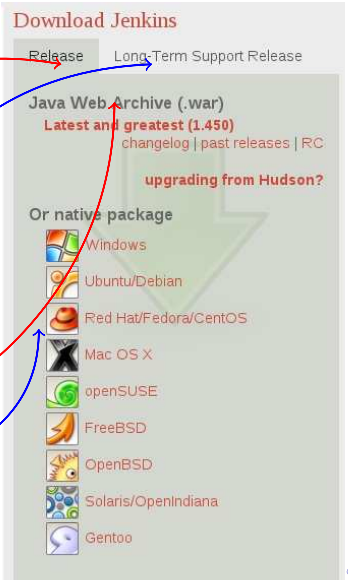
`http://jenkins-ci.org`

Release cycle:

- Released usually weekly - release early, release often.
- Long term support (LTS) release - every 3 months, every month minor release with backports of major bug fixes.

Distribution (Jenkins is a Java servlet):

- Web archive (WAR).
- Native package.


Download Jenkins

Release | Long-Term Support Release

Java Web Archive (.war)

Latest and greatest (1.450)

changelog | past releases | RC

upgrading from Hudson?

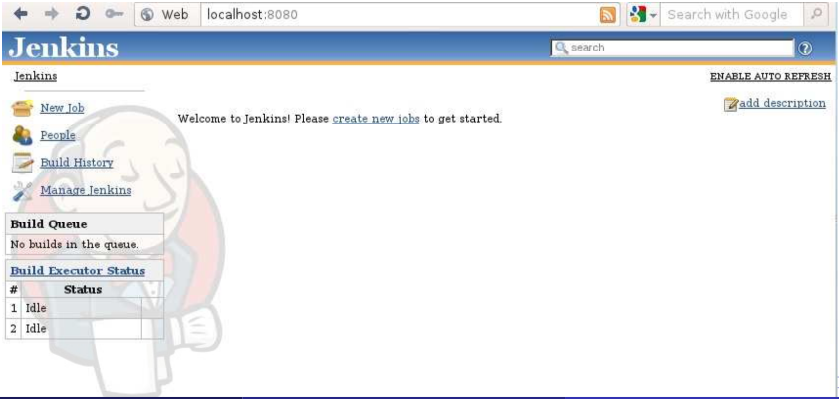
Or native package

- Windows
- Ubuntu/Debian
- Red Hat/Fedora/CentOS
- Mac OS X
- openSUSE
- FreeBSD
- OpenBSD
- Solaris/OpenIndiana
- Gentoo

Running Jenkins

- Download war file,
  - deploy it on your favorite servlet container like JBoss AS 7 or Tomcat,
  - or just simply run `java -jar jenkins.war`.
- Install native package e.g. via `yum` and start it as a service

```
wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins-ci.org/redhat/jenkins.repo
rpm --import http://pkg.jenkins-ci.org/redhat/jenkins-ci.org.key
yum -y install jenkins
service jenkins start
```



Welcome to Jenkins! Please create new jobs to get started.

Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

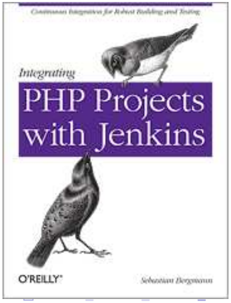
Installing plug-ins

Plug-ins are one of the strong point of Jenkins, around 400 plug-ins, plug-ins for almost everything.
Visit `Manage Jenkins -> Manage Plugins -> Available`, see also `https://wiki.jenkins-ci.org/display/JENKINS/Plugins`
Few tips for language agnostic plugins:

- SCM:
  - CVS and Subversion are already pre-installed.
  - Git plugin - integrates Jenkins with git.
  - GitHub plugin - better integration with GitHub, e.g. browse changelog on GitHub etc.
  - Plugins for most of the SCMs are available.
- Issue tracker integration:
  - Jira plugin.
  - Bugzilla plugin.
- Ingratiation with other tools / services:
  - EC2 / Delta cloud plugins - using machines from cloud for builds.
  - Gerrit plugin.
- Notifications / post-build actions - almost whatever you like :-)

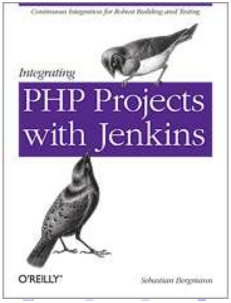
Few tips for language specific plug-ins

- Java is very well supported, plugins for all commonly used tools.
- C++:
  - xUnit plugins - supports (besides other) UnitTest++, CppUnit, Boost Test Library
  - qmakebuilder plugin
- Python:
  - Python plugin
  - Shining Panda plugin
- Ruby:
  - Ruby plugin
  - Rake plugin
  - Ruby metrics plugin
- PHP:
  - Check `http://jenkins-php.org/` or book `PHP projects with Jenkins` by O'Reilly
- Far not a complete list, list above is just my personal selection of some interesting plugins! Also plugins for other languages exists.


Continuous Integration for Robust Building and Testing
Integrating
PHP Projects with Jenkins
O'REILLY
Sebastian Bergmann

Few tips for language specific plug-ins

- Java is very well supported, plugins for all commonly used tools.
- C++:
  - xUnit plugins - supports (besides other) UnitTest++, CppUnit, Boost Test Library
  - qmakebuilder plugin
- Python:
  - Python plugin
  - Shining Panda plugin
- Ruby:
  - Ruby plugin
  - Rake plugin
  - Ruby metrics plugin
- PHP:
  - Check `http://jenkins-php.org/` or book `PHP projects with Jenkins` by O'Reilly
- Far not a complete list, list above is just my personal selection of some interesting plugins! Also plugins for other languages exists.


Continuous Integration for Robust Building and Testing
Integrating
PHP Projects with Jenkins
O'REILLY
Sebastian Bergmann

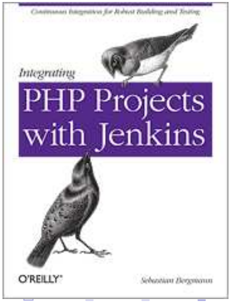
Few tips for language specific plug-ins

- Java is very well supported, plugins for all commonly used tools.
- C++:
  - xUnit plugins - supports (besides other) UnitTest++, CppUnit, Boost Test Library
  - qmakebuilder plugin

- Python:
  - Python plugin
  - Shining Panda plugin

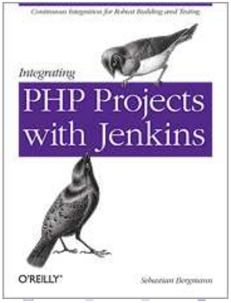
- Ruby:
  - Ruby plugin
  - Rake plugin
  - Ruby metrics plugin

- PHP:
  - Check `http://jenkins-php.org/` or book `PHP projects with Jenkins` by O'Reilly
- Far not a complete list, list above is just my personal selection of some interesting plugins! Also plugins for other languages exists.



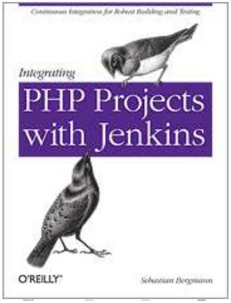
Few tips for language specific plug-ins

- Java is very well supported, plugins for all commonly used tools.
- C++:
  - xUnit plugins - supports (besides other) UnitTest++, CppUnit, Boost Test Library
  - qmakebuilder plugin
- Python:
  - Python plugin
  - Shining Panda plugin
- Ruby:
  - Ruby plugin
  - Rake plugin
  - Ruby metrics plugin
- PHP:
  - Check `http://jenkins-php.org/` or book `PHP projects with Jenkins` by O'Reilly
- Far not a complete list, list above is just my personal selection of some interesting plugins! Also plugins for other languages exists.


Continuous Integration for Robust Building and Testing
Integrating
PHP Projects with Jenkins
O'REILLY
Sebastian Bergmann

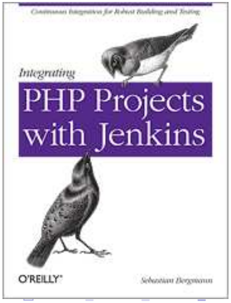
Few tips for language specific plug-ins

- Java is very well supported, plugins for all commonly used tools.
- C++:
  - xUnit plugins - supports (besides other) UnitTest++, CppUnit, Boost Test Library
  - qmakebuilder plugin
- Python:
  - Python plugin
  - Shining Panda plugin
- Ruby:
  - Ruby plugin
  - Rake plugin
  - Ruby metrics plugin
- PHP:
  - Check `http://jenkins-php.org/` or book `PHP projects with Jenkins` by O'Reilly
- Far not a complete list, list above is just my personal selection of some interesting plugins! Also plugins for other languages exists.


Continuous Integration for Robust Building and Testing
Integrating
PHP Projects with Jenkins
O'REILLY
Sebastian Bergmann

Few tips for language specific plug-ins

- Java is very well supported, plugins for all commonly used tools.
- C++:
  - xUnit plugins - supports (besides other) UnitTest++, CppUnit, Boost Test Library
  - qmakebuilder plugin
- Python:
  - Python plugin
  - Shining Panda plugin
- Ruby:
  - Ruby plugin
  - Rake plugin
  - Ruby metrics plugin
- PHP:
  - Check `http://jenkins-php.org/` or book `PHP projects with Jenkins` by O'Reilly
- Far not a complete list, list above is just my personal selection of some interesting plugins! Also plugins for other languages exists.


Continuous Integration for Robust Building and Testing
Integrating
PHP Projects with Jenkins
O'REILLY
Sebastian Bergmann

Plug-ins roadmap

- Java plugins more mature, e.g. most Java tools plugins has auto-installer - if the tool is not installed, plugin is able to install and set it up (very useful when running in the cloud), such features in plug-ins for other languages (Python, Ruby) will hopefully follow.
- Moving Java specific features into the plug-ins.
- Some originally Java features can be used also for other languages, e.g. JUnit test results can be used for any JUnit compatible format (see examples), JavaDoc can publish arbitrary documentation etc.
- If you are missing something, you can always use shell task to execute command you want . . .

. . . or implement missing plugin/functionality.
Support for other languages - not only plug-ins for different languages but **even possibility to develop plugin in different languages!** Already done for Ruby, support for other languages should follow.

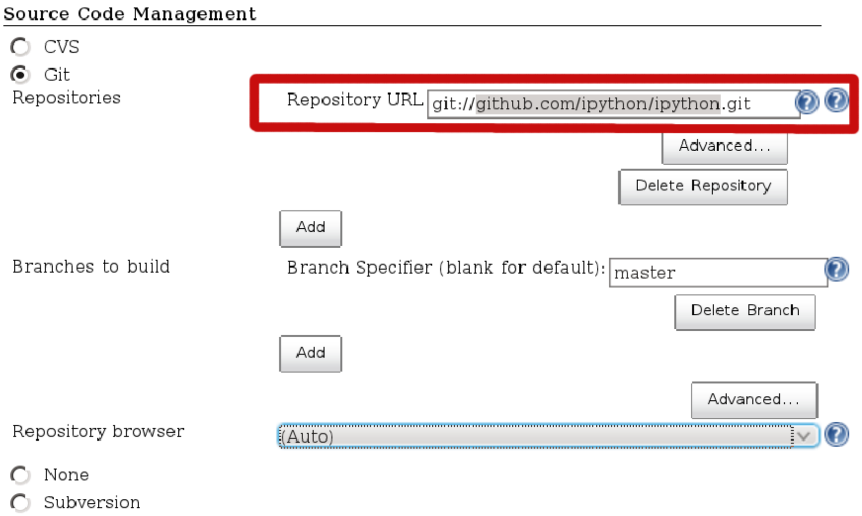
Python example: IPython

Fresh Fedora 16 installation

```
# Install Java
yum -y install java-1.6.0-openjdk-devel

# Python setup
yum -y install python-pip python-nose python-zmq python-virtualenv
pip-python install nose-cov


# Install Jenkins
wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins-ci.org/redhat/jenkins.repo
rpm --import http://pkg.jenkins-ci.org/redhat/jenkins-ci.org.key
yum -y install jenkins
service jenkins start
```

Via Jenkins UI install Git, Cobertura and ShiningPanda plug-ins.

Setup git repository:



Source Code Management

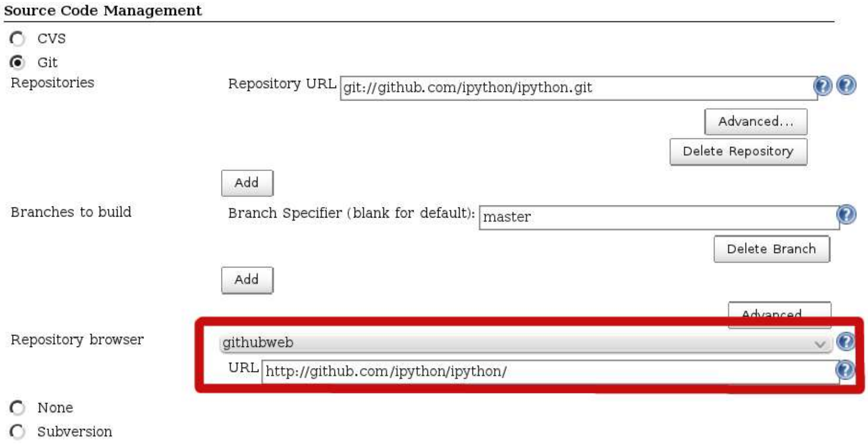
CVS [✓ Git]

Repositories	Repository URL: git://github.com/ipython/ipython.git
Branches to build	Branch Specifier (blank for default): master
Repository browser	(Auto)

None

Subversion

Optionally set up repository browser if you want to browse commits directly on GitHub:



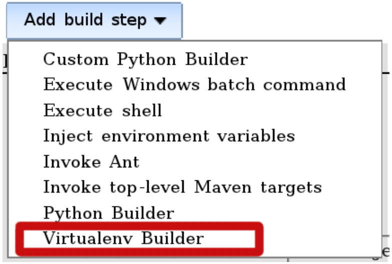
Source Code Management

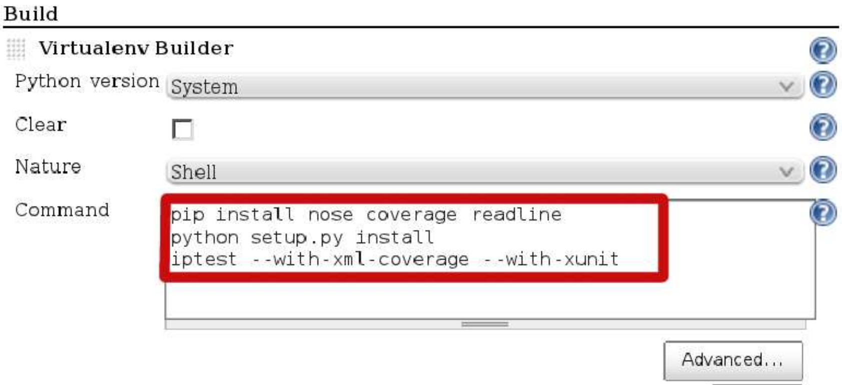
- ○ CVS
- ● Git

Repositories	Repository URL: git://github.com/ipython/ipython.git
Branches to build	Branch Specifier (blank for default): master
Repository browser	githubweb
	URL: http://github.com/ipython/ipython/

- ○ None
- ○ Subversion

Add a build step: execute shell command (install tools, ipython and run tests) in virtualenv:




Build

Virtualenv Builder

Python version: System

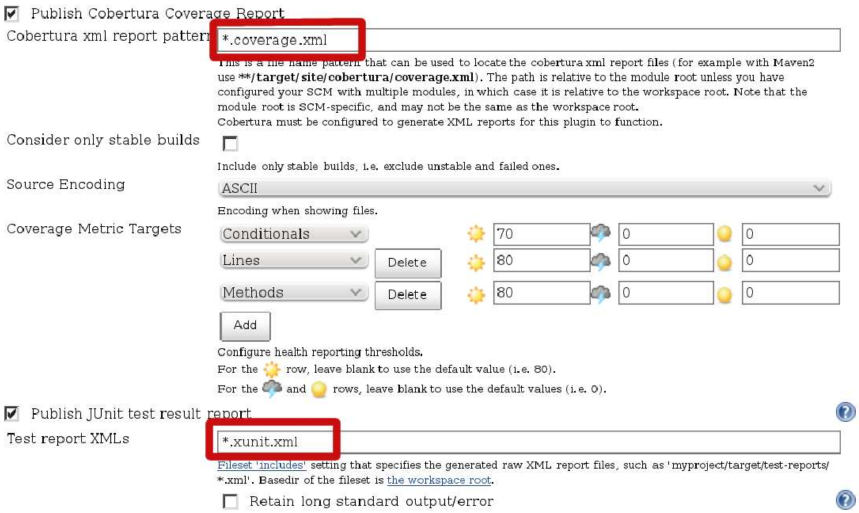
Clear: ☐

Nature: Shell

Command:

```
pip install nose coverage readline
python setup.py install
iptest --with-xml-coverage --with-xunit
```

Setup paths to unit and coverage reports:



☑ Publish Cobertura Coverage Report

Cobertura xml report pattern	*.coverage.xml

This is a file name pattern that can be used to locate the cobertura xml report files (for example with Maven2 use **/target/site/cobertura/coverage.xml). The path is relative to the module root unless you have configured your SCM with multiple modules, in which case it is relative to the workspace root. Note that the module root is SCM-specific, and may not be the same as the workspace root.
Cobertura must be configured to generate XML reports for this plugin to function.

Consider only stable builds	☐

Include only stable builds, i.e. exclude unstable and failed ones.

Source Encoding	ASCII

Encoding when showing files.

Coverage Metric Targets

Conditionals		70	0	0
Lines	Delete	80	0	0
Methods	Delete	80	0	0

Add

Configure health reporting thresholds.
For the row, leave blank to use the default value (i.e. 80).
For the and rows, leave blank to use the default values (i.e. 0).

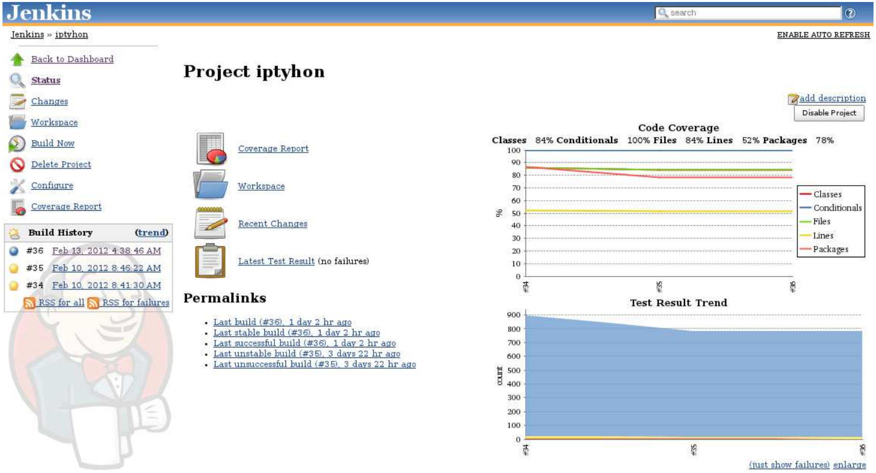
☑ Publish JUnit test result report

Test report XMLs	*.xunit.xml

Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/*.xml'. Basedir of the fileset is the workspace root.

☐ Retain long standard output/error

Run a build and check the results:



Jenkins » iptyhon

ENABLE AUTO REFRESH

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now
- Delete Project
- Configure
- Coverage Report

Build History (trend)

#36 Feb 13, 2012 4:38:46 AM
#35 Feb 10, 2012 8:46:22 AM
#34 Feb 10, 2012 8:41:30 AM

RSS for all RSS for failures

Project iptyhon

add description

Disable Project

Coverage Report

Workspace

Recent Changes

Latest Test Result (no failures)

Permalinks

- Last build (#36), 1 day 2 hr ago
- Last stable build (#36), 1 day 2 hr ago
- Last successful build (#36), 1 day 2 hr ago
- Last unstable build (#35), 3 days 22 hr ago
- Last unsuccessful build (#35), 3 days 22 hr ago

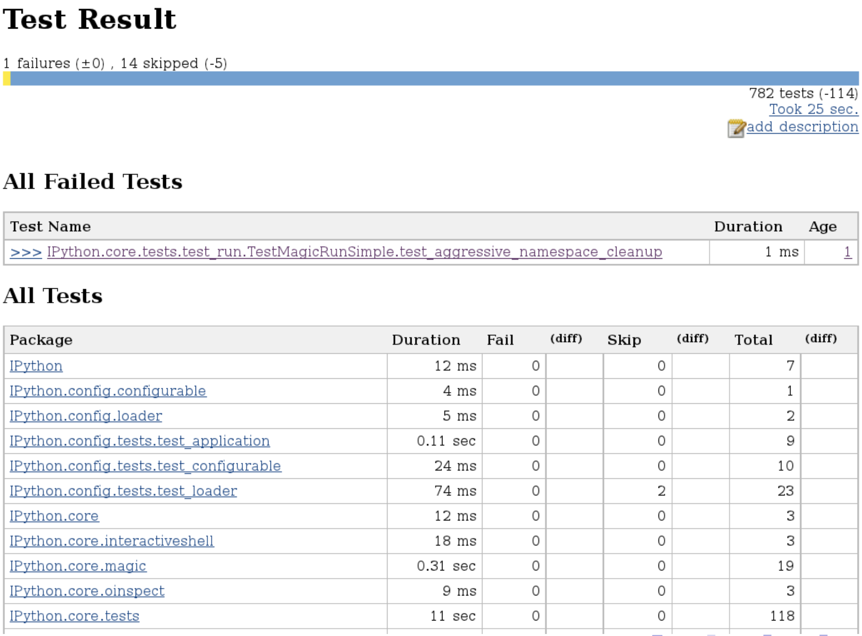
Code Coverage

Classes 84% Conditionals 100% Files 84% Lines 52% Packages 78%

Test Result Trend

(just show failures) enlarge

Unit test results



Test Result

1 failures (±0) , 14 skipped (-5)

782 tests (-114)
Took 25 sec.
add description

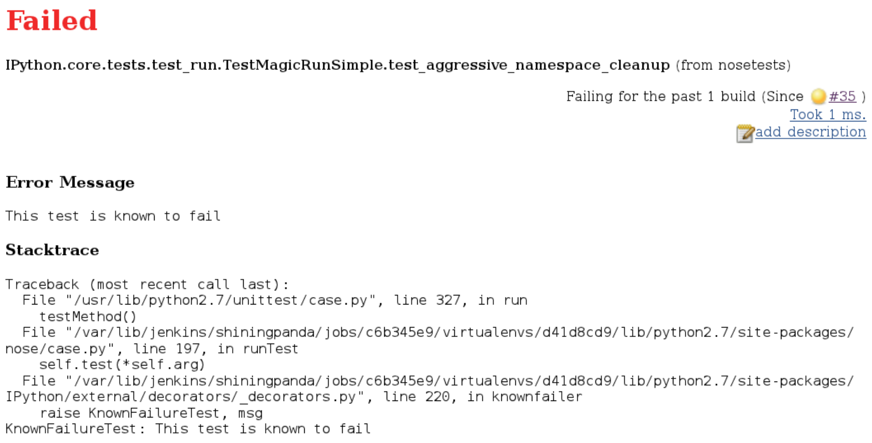
All Failed Tests

Test Name	Duration	Age
>>> IPython.core.tests.test_run.TestMagicRunSimple.test_aggressive_namespace_cleanup	1 ms	1

All Tests

Package	Duration	Fail	(diff)	Skip	(diff)	Total	(diff)
IPython	12 ms	0		0		7	
IPython.config.configurable	4 ms	0		0		1	
IPython.config.loader	5 ms	0		0		2	
IPython.config.tests.test_application	0.11 sec	0		0		9	
IPython.config.tests.test_configurable	24 ms	0		0		10	
IPython.config.tests.test_loader	74 ms	0		2		23	
IPython.core	12 ms	0		0		3	
IPython.core.interactiveshell	18 ms	0		0		3	
IPython.core.magic	0.31 sec	0		0		19	
IPython.core.oinspect	9 ms	0		0		3	
IPython.core.tests	11 sec	0		0		118	

...and eventually easily check failed tests:



Failed

IPython.core.tests.test_run.TestMagicRunSimple.test_aggressive_namespace_cleanup (from nosetests)

Failing for the past 1 build (Since #35)
Took 1 ms.
add description

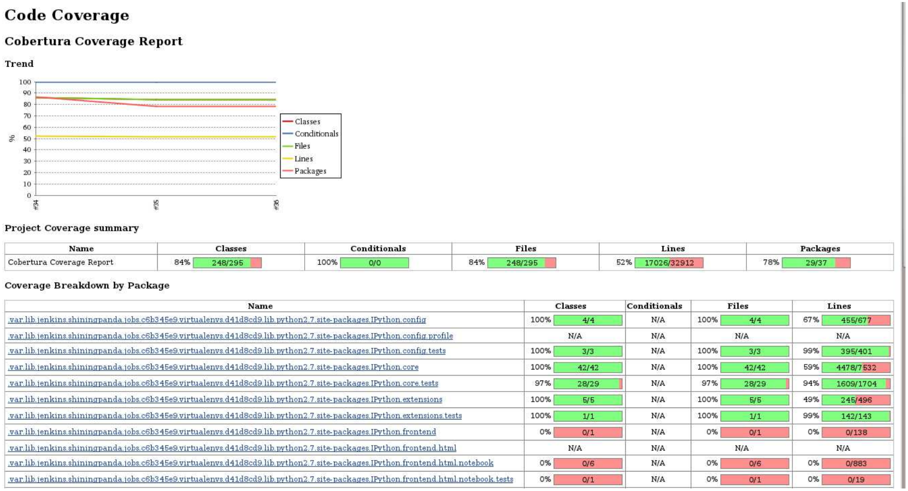
Error Message

```
This test is known to fail
```

Stacktrace

```
Traceback (most recent call last):
  File "/usr/lib/python2.7/unittest/case.py", line 327, in run
    testMethod()
  File "/var/lib/jenkins/shiningpanda/jobs/c6b345e9/virtualenvs/d41d8cd9/lib/python2.7/site-packages/
nose/case.py", line 197, in runTest
    self.test(*self.arg)
  File "/var/lib/jenkins/shiningpanda/jobs/c6b345e9/virtualenvs/d41d8cd9/lib/python2.7/site-packages/
IPython/external/decorators/_decorators.py", line 220, in knownfailer
    raise KnownFailureTest, msg
KnownFailureTest: This test is known to fail
```

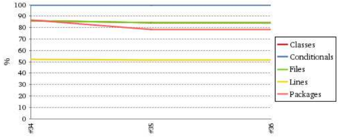
Test coverage report:



Code Coverage

Cobertura Coverage Report

Trend



Project Coverage summary

Name	Classes	Conditionals	Files	Lines	Packages
Cobertura Coverage Report	84% 248/295	100% 0/0	84% 248/295	52% 17026/32912	78% 29/37

Coverage Breakdown by Package

Name	Classes	Conditionals	Files	Lines
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.config	100% 4/4	N/A	100% 4/4	67% 455/677
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.config.profile	N/A	N/A	N/A	N/A
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.config.tests	100% 3/3	N/A	100% 3/3	99% 395/401
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.core	100% 42/42	N/A	100% 42/42	59% 4478/7532
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.core.tests	97% 28/29	N/A	97% 28/29	94% 1609/1704
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.extensions	100% 5/5	N/A	100% 5/5	49% 245/496
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.extensions.tests	100% 1/1	N/A	100% 1/1	99% 142/143
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.frontend	0% 0/1	N/A	0% 0/1	0% 0/138
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.frontend.html	N/A	N/A	N/A	N/A
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.frontend.html.notebook	0% 0/6	N/A	0% 0/6	0% 0/883
.var.lib.jenkins.shiningpanda.jobs.c6b345e9.virtualenvs.d41d8cd9.lib.python2.7.site-packages.IPython.frontend.html.notebook.tests	0% 0/1	N/A	0% 0/1	0% 0/19

Ruby example: Rails tutorial blog project

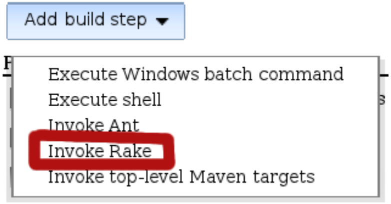
```
# Install Ruby and Rails
yum -y install ruby ruby-devel rubygems sqlite sqli-devel js
gem install rails
gem install ci_reporter

# Create tutorial app
rails new blog
rails generate scaffold Post name:string title:string content:text
```

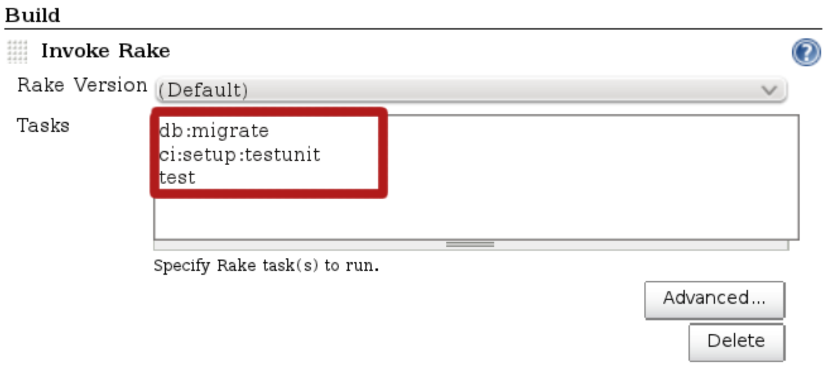
Add into `Rakefile`

```
require 'rubygems'
require 'ci/reporter/rake/test_unit'
```

Push your new app into git repo.
Via Jenkins UI install Ruby metrics plugin (Rake plugin will be installed as dependency).

Add build step: execute Rake tasks (db migration, test setup and test execution):



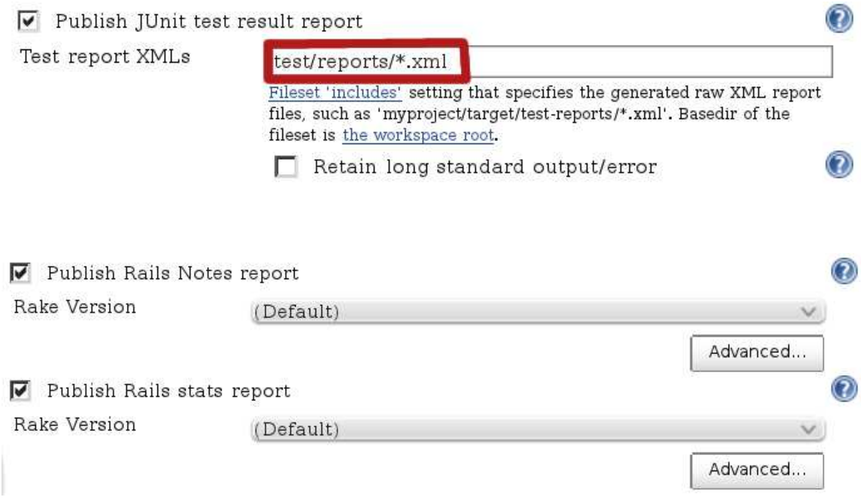


Tasks:

```
db:migrate
ci:setup:testunit
test
```

Specify Rake task(s) to run.

Setup path to unit test results and also you can turn on generation of Rails reports (annotations and statistics):



☑ Publish JUnit test result report

Test report XMLs: test/reports/*.xml

Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/*.xml'. Basedir of the fileset is the workspace root.

☐ Retain long standard output/error

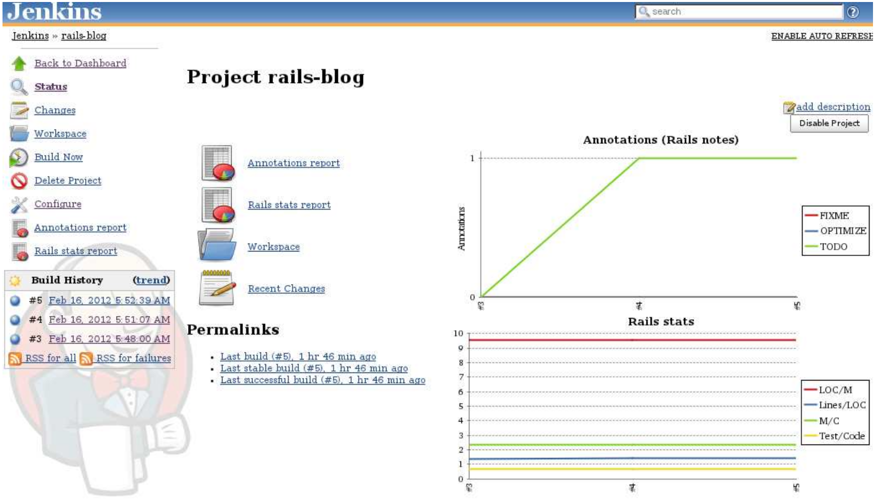
☑ Publish Rails Notes report

Rake Version: (Default)

☑ Publish Rails stats report

Rake Version: (Default)

Run a build and check the results:



Jenkins » rails-blog

ENABLE AUTO REFRESH

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now
- Delete Project
- Configure
- Annotations report
- Rails stats report

Build History (trend)

- #5 Feb 16, 2012 5:52:39 AM
- #4 Feb 16, 2012 5:51:07 AM
- #3 Feb 16, 2012 5:48:00 AM

RSS for all RSS for failures

Project rails-blog

add description

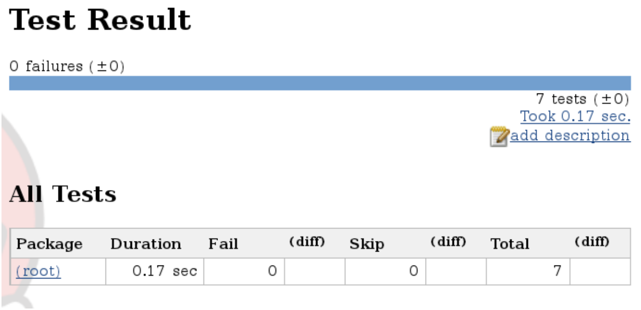
Disable Project

- Annotations report
- Rails stats report
- Workspace
- Recent Changes

Permalinks

- Last build (#5), 1 hr 46 min ago
- Last stable build (#5), 1 hr 46 min ago
- Last successful build (#5), 1 hr 46 min ago

Unit tests results and Rails annotations report



Test Result

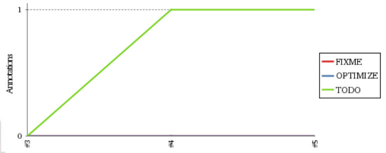
0 failures (±0)

7 tests (±0)
Took 0.17 sec.
add description

All Tests

Package	Duration	Fail	(diff)	Skip	(diff)	Total	(diff)
(root)	0.17 sec	0		0		7	

Annotations (Rails notes) report



Filename	TODO	FIXME	OPTIMIZE
Total	1	0	0
app/controllers/posts_controller.rb	1	0	0

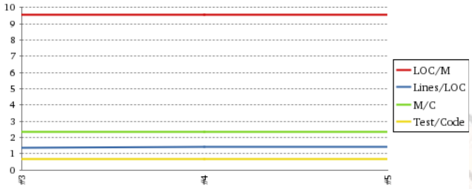
Output

```
[workspace] $ rake --silent notes
app/controllers/posts_controller.rb:
  * [  2] [TODO] improve this class
```

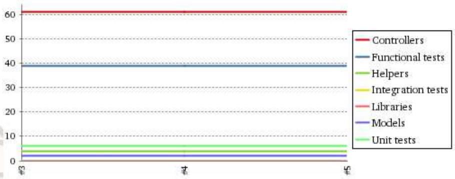
Rails statistics report

Rails stats report

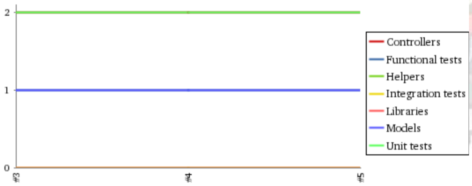
Ratios



Lines of Code



Classes



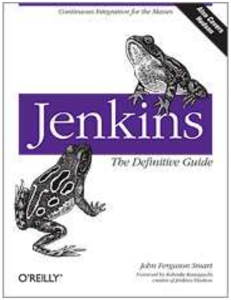
Stats

Name	Lines	LOC	Classes	Methods	M/C	LOC/M
Controllers	88	61	2	7	3	
Helpers	4	4	0	0	0	
Models	2	2	1	0	0	
Libraries	0	0	0	0	0	
Integration tests	0	0	0	0	0	
Functional tests	49	39	1	0	0	
Unit tests	11	6	2	0	0	
Total	154	112	6	7	1	1

Code LOC: 67 Test LOC: 45 Code to Test Ratio: 1:0.7

- Only basic examples.
- Real use cases can be much more sophisticated. Jenkins has many advanced interesting features which allows you to create very robust builds.
- Check `Jenkins: The Definitive Guide`

  `http://www.wakaleo.com/books/jenkins-the-definitive-guide`

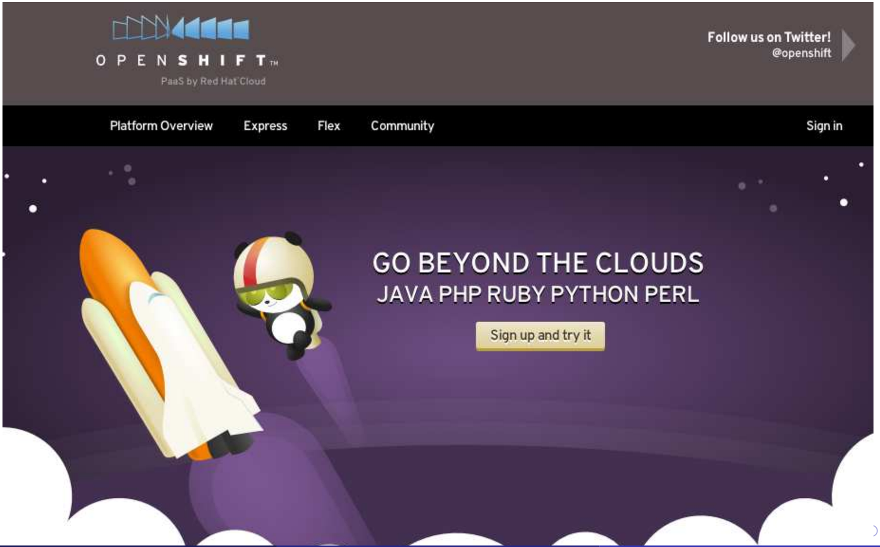


Still not persuaded?

- Don't want to setup a build machine for Jenkins server/builds (or slow down you dev machine)?
- Still think it's too difficult to setup CI?
- Just lazy to do the setup?
- Java hater (*Java never ever on my server*)?
- $<$Any other reason$>$?

OpenShift

. . . then consider some cloud PaaS (platform as a service) offering, e.g. OpenShift by Red Hat!



OpenShift

`https://openshift.redhat.com`

Still some initial set up (like registration) needs to be done, but you needn't

- find suitable machine for Jenkins server
- set up Jenkins server and build environment
- maintain CI environment

Support for

- Java
- PHP
- Python
- Ruby
- Perl

... and all this is **for free!** (in case of OpenShift Express)

Setup OpenShift app

Install OpenShift tools:

```
wget https://openshift.redhat.com/app/repo/openshift.repo
     -O /etc/yum.repos.d/openshift.repo
yum -y install rhc
```

If you don’t use `yum`, check OpenShift tutorial how to install it using `gem`.
Register your domain:

```
rhc-create-domain -n mydomain -l rhlogin
```

Create your app with Jenkins CI enabled:

- `rhc-create-app -a jbosstest -t jbossas-7.0 --enable-jenkins ci`
- `rhc-create-app -a rubytest -t ruby-1.8 --enable-jenkins ci`
- `rhc-create-app -a pythontest -t python-2.6 --enable-jenkins ci`

Add existing app under Jenkins CI by

```
rhc-ctl-app -a myapp -e add-jenkins-client-1.4
```

What OpenShift does

OpenShift will

- create git repo for your application,
- create Jenkins instance it already doesn't exists,
- create CI job for your app and do some basic setup,
- after each commit run CI build,
- deploy your application.

You have admin login for your Jenkins instance so you can install arbitrary plugin and modify jobs as in examples above.

See also video tutorial on
`https://openshift.redhat.com/app/express`.
Some OpenShift example apps: `https://github.com/openshift`.

Sample output

After each commit OpenShift starts CI builds

```
vjuranek@localhost rubytest\$ git push
Enter passphrase for key '/home/vjuranek/.ssh/libra_id_rsa':
Counting objects: 133, done.
Delta compression using up to 2 threads.
Compressing objects: 100\% (89/89), done.
Writing objects: 100\% (124/124), 90.96 KiB, done.
Total 124 (delta 20), reused 120 (delta 18)
remote: Executing Jenkins build.
remote:
remote: You can track your build at http://ci-vjuranek.rhcloud.com/job/rubytest-buil
remote:
remote: Waiting for build to schedule....Done
remote: Waiting for job to complete.......Done
remote: SUCCESS
remote: New build has been deployed.
To ssh://fef2974bd2e1403f92f873358cf360c5@rubytest2-vjuranek.rhcloud.com/~/git/rubyte
   8e4a040..3fb2bd3  master -> master
```

Other PaaS offerings providing Jenkins

- CloudBees
  - `http://www.cloudbees.com/`
  - For FOSS projects free 2,000 minutes/month of m1.small and 500/month minutes of m1.large build/test capacity
  - For more details check `http://www.cloudbees.com/foss/index.cb`
  - Examples (FOSS projects): `http://www.cloudbees.com/foss/foss-projects.cb`
- Shining Panda
  - `https://www.shiningpanda.com`
  - For FOSS projects free 1 hour/day
  - For more details check `https://www.shiningpanda.com/pricing`
  - Examples (FOSS projects): `https://www.shiningpanda.com/public`

Other (possibly) interesting stuff

- Mobile Jenkins
  `http://www.jenkins-ci.mobi/`
- Integration with Eclipse
  `http://www.cloudbees.com/eclipse-plugin.cb`
  `http://tasktop.com/connectors/hudson-jenkins`
- KDE tray app
  `https://gitorious.org/fargies-misc-tools/jenkins-tray`

- Python API
  `http://packages.python.org/python-jenkins`
- Ruby API
  `https://github.com/cowboyd/jenkins.rb`

Want to know more about Jenkins?

Something unclear? Doesn’t work? Need some help with setup CI for your project?
Room **B411, tomorrow 12:45-13:15**, I’ll be there for tomorrow and we can discuss your issues.

Jenkins channels:

- Mailing users lists: `http://groups.google.com/group/jenkinsci-users`
- Mailing dev list: `http://groups.google.com/group/jenkinsci-dev`
- IRC channel: #jenkins on `http://www.freenode.net/`
- Twitter: `http://twitter.com/jenkinsci`

Interested in Jenkins and also virtualization? RHEV team is searching Jenkins specials!
Check `https://careers.redhat.com/` in near future.