

# CONTINUOUS INTEGRATION

*Jenkins*

15. Februar, 2016



## Jenkins

Jenkins



Michael Bruckner



Meine Ansichten



Element anlegen



Benutzer



Build-Verlauf

# INHALTSVERZEICHNIS

---

ANGABE.....	3
ZEITAUFWAND .....	4
AUFSETZEN VON JENKINS .....	4
BUILD KONFIGURATION.....	7
PROBLEME .....	8

# ANGABE

---

## A10 - CONTINUOUS INTEGRATION

*"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. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly. This article is a quick overview of Continuous Integration summarizing the technique and its current usage." M.Fowler*

Schreibe fünf Testfälle für dein CSV-Projekt und lass diese mithilfe von Jenkins automatisch bei jedem Build testen!

- Installiere auf deinem Rechner bzw. einer virtuellen Instanz das Continuous Integration System Jenkins
- Installiere die notwendigen Plugins für Jenkins (Git Plugin, Violations, Cobertura)
- Installiere Nose und Pylint (mithilfe von pip)
- Integriere dein CSV-Projekt in Jenkins, indem du es mit Git verbindest
- Schreibe fünf Unit Tests für dein CSV-Projekt
- Konfiguriere Jenkins so, dass deine Unit Tests automatisch bei jedem Build durchgeführt werden inkl. Berichte über erfolgreiche / fehlgeschlagene Tests und Coverage
- Protokolliere deine Vorgehensweise (inkl. Zeitaufwand, Konfiguration, Probleme) und die Ergebnisse (viele Screenshots!)

Viel Spaß!

## ZEITAUFWAND

---

	Geschätzte Zeit	Benötigte Zeit	Betroffene Punkte
<i>Konfiguration von Jenkins</i>	3 Stunden	½ Stunde	1-4 (siehe Angabe)
<i>Testfälle</i>	1 Stunde	¾ Stunde	5 (siehe Angabe)
<i>Build Konfiguration</i>	2 Stunden	8 ½ Stunden	6 (siehe Angabe)
<i>Gesamt</i>	6 Stunden	9 ¾ Stunden	

## AUFSETZEN VON JENKINS

---

### Installation des Continuous Integration System Jenkins

Es gibt mehr als eine Möglichkeit Jenkins zu installieren. Ich habe diese Variante gewählt:

- `wget -q -O - http://pkg.jenkins-ci.org/debian/jenkins-ci.org.key | apt-key add -`
- `'echo deb http://pkg.jenkins-ci.org/debian binary/>/etc/apt/sources.list.d/jenkins.list' -`
- `apt-get update`
- `apt-get install jenkins`

Nachdem die Installation geklappt hat, startet man mit dem Befehl `service jenkins start` Jenkins und kann schon loslegen. Der Server ist über den Port **8080** erreichbar.

Um den Zugriff von anderen Usern im selben Netzwerk zu verhindern, können die Sicherheitseinstellungen geändert werden. Da wir aber mit Jenkins nur unsere Tests checken wollen ist die Sicherheit eher unwichtig, kann aber unter dem Punkt **Manage Jenkins > Setup Security** eingestellt werden.

## Installation der notwendigen Plugins

Laut der Angabe, benötigen wir folgende Plugins:

- *Git Plugin*
- *Cobertura*
- *Violations*

Die Plugins können direkt in Jenkins installiert werden unter dem Punkt **Manage Jenkins > Plugins**.

Filter:

Aktualisierungen	Verfügbar	Installiert	Erweiterte Einstellungen			
Aktiviert	Name ↓	Version	Vorher installierte Version	Gesperrt	Deinstallieren	
<input checked="" type="checkbox"/>	<a href="#">Ant Plugin</a> This plugin adds <a href="#">Apache Ant</a> support to Jenkins.	<a href="#">1.2</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Cobertura Plugin</a> This plugin integrates <a href="#">Cobertura coverage reports</a> to Jenkins.	<a href="#">1.9.7</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Credentials Plugin</a> This plugin allows you to store credentials in Jenkins.	<a href="#">1.24</a>	<a href="#">1.18 wiederherstellen</a>	<a href="#">Entsperren</a>	<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">CVS Plug-in</a> Integrates Jenkins with CVS version control system using a modified version of the Netbeans cvsclient.	<a href="#">2.11</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">External Monitor Job Type Plugin</a> Adds the ability to monitor the result of externally executed jobs.	<a href="#">1.4</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Git client plugin</a> Shared library plugin for other Git related Jenkins plugins.	<a href="#">1.19.2</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Git plugin</a> This plugin integrates <a href="#">Git</a> with Jenkins.	<a href="#">2.4.1</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Javadoc Plugin</a> This plugin adds Javadoc support to Jenkins.	<a href="#">1.1</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">JUnit Plugin</a> Allows JUnit-format test results to be published.	<a href="#">1.10</a>	<a href="#">1.2-beta-4 wiederherstellen</a>	<a href="#">Entsperren</a>	<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">LDAP Plugin</a> Adds LDAP authentication to Jenkins	<a href="#">1.11</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">pyenv plugin</a> This plugin runs your jobs in the pyenv	<a href="#">0.0.7</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Python Plugin</a> Adds the ability to execute python scripts as build steps.	<a href="#">1.3</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">ruby-runtime</a> Provides the Ruby runtime and bindings required to implement <a href="#">plugins in Ruby</a> .	<a href="#">0.12</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">SCM API Plugin</a> This plugin provides a new enhanced API for interacting with SCM systems.	<a href="#">1.0</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Script Security Plugin</a> Allows Jenkins administrators to control what in-process scripts can be run by less-privileged users.	<a href="#">1.13</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">SSH Credentials Plugin</a> This plugin allows you to store SSH credentials in Jenkins.	<a href="#">1.11</a>	<a href="#">1.10 wiederherstellen</a>	<a href="#">Entsperren</a>	<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">SSH Slaves plugin</a> This plugin allows you to manage slaves running on \nix machines over SSH.	<a href="#">1.9</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Subversion Plug-in</a> This plugin adds the Subversion support (via SVNKit) to Jenkins.	<a href="#">1.54</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Translation Assistance plugin</a> This plugin adds an additional dialog box in every page, which enables people to contribute localizations for the messages they are seeing in the current page.	<a href="#">1.10</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Violations plugin</a> This plugin does reports on checkstyle, csslint, pmd, cpd, txcop, pylint, JcReport, findbugs, and pericritic violations.	<a href="#">0.7.11</a>			<a href="#">Deinstallieren</a>	
<input checked="" type="checkbox"/>	<a href="#">Windows Slaves Plugin</a> Allows you to connect to Windows machines and start slave agents on them.	<a href="#">1.0</a>			<a href="#">Deinstallieren</a>	

## Installation von Nose und Pylint

Die Installation von nose und pylint über pip funktioniert so:

- *pip install nosetests-3.4*
- *pip install noseexcover*
- *pip install pylint*

## Integration des CSV-Projektes

Mithilfe des Git Plugins können wir nun das CSV-Projekt importieren.

### Source-Code-Management

- ☐ Keines  
☐ CVS  
☐ CVS Projectset  
☒ Git  
Repositories

Repository URL

Credentials

# BUILD KONFIGURATION

Die Build Konfiguration wird unter *Configuration* im Projekt eingestellt.

## Buildverfahren

### Shell ausführen

Befehl

```
set +e
PYTHONPATH=/usr/bin/env python3.4
cd /var/lib/jenkins/workspace/CSV-Projekt
chmod -R 777 .
nosetests-3.4 --with-xunit --all-modules --traverse-namespace --with-coverage --cover-package=csv_tool --cover-inclusive
chmod -R 777 .
python3 -m coverage xml --include=csv_tool*
chmod -R 777 .
pylint -f parseable -d I0011,R0801 csv_tool
chmod -R 777 .|
```

[Liste der verfügbaren Umgebungsvariablen](#)

Löschen

## Post-Build-Aktionen

--&gt;

### Veröffentliche JUnit-Testergebnisse.

Testberichte in XML-Format

nosetests.xml

Es sind reguläre Ausdrücke wie z.B. 'myproject/target/test-reports/\*.xml' erlaubt. Das genaue Format können Sie [der Spezifikation für @includes eines Ant-Filesets](#) entnehmen. Das Ausgangsverzeichnis ist der [Arbeitsbereich](#).

☐ Lange Standard-Out/Error Ausgaben aufbewahren

Health report amplification factor

1,0

1% failing tests scores as 99% health. 5% failing tests scores as 95% health

Erweitert...

Löschen

### Veröffentliche die Cobertura Testabdeckung

Dateien Cobertura XML Ergebnisse

\*\*/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.

Erweitert...

Löschen

# PROBLEME

---

Ich hatte zuerst Problem damit die Tests zum Laufen zu bringen, da meine Konfiguration fehlerhaft war. Nachdem ich mir die Konfiguration von dem Herrn Melichar angesehen habe, habe ich meinen Input in der Build-Shell verändert.

Nun erhalte ich folgenden Fehler:

## Fehler behoben!

```
Started by user Michael Bruckner
Building in workspace /var/lib/jenkins/workspace/CSV-Projekt
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/brucki2811/CSV-Projekt.git # timeout=10
Fetching upstream changes from https://github.com/brucki2811/CSV-Projekt.git
> git --version # timeout=10
> git -c core.askpass=true fetch --tags --progress https://github.com/brucki2811/CSV-Projekt.git
+refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 3eedfcc3ae5322376d36bf697ad2c2054ac53e21 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 3eedfcc3ae5322376d36bf697ad2c2054ac53e21
> git rev-list 3eedfcc3ae5322376d36bf697ad2c2054ac53e21 # timeout=10
[CSV-Projekt] $ /bin/sh -xe /tmp/hudson2396529561406396371.sh
+ set +e
+ PYTHONPATH=/usr/bin/env python3.4
+ cd /var/lib/jenkins/workspace/CSV-Projekt
+ chmod -R 777 .
+ nosetests-3.4 --with-xunit --all-modules --traverse-namespace --with-coverage --cover-package=csv_tool --cover-inclusive

Name                               Stmts   Miss  Cover   Missing
-----
csv_tool/CSVUebung.py              23      23     0%    1-39
-----
Ran 0 tests in 0.007s

OK
+ chmod -R 777 .
+ python3 -m coverage xml --include=csv_tool*
+ chmod -R 777 .
+ pylint -f parseable -d I0011,R0801 csv_tool
No config file found, using default configuration
*****
csv_tool/__init__.py:1: [F0010(parse-error), ] error while code parsing: Unable to load file
'csv_tool/__init__.py' ([Errno 2] No such file or directory: 'csv_tool/__init__.py')
+ chmod -R 777 .
Recording test results
ERROR: Step 'Publish JUnit test result report' failed: None of the test reports contained any result
Skipping Cobertura coverage report as build was not UNSTABLE or better ...
Finished: FAILURE
```

## Fehler behoben!

Nun funktionieren einige Zeilen im CSV-Code nicht, aber im Großen und Ganzen sollte nun eine vollständige und überwiegend korrekte Konfiguration zu Stande gekommen sein.

Meldung:

## Fehler behoben!



```

Started by user Michael Bruckner
Building in workspace /var/lib/jenkins/workspace/CSV-Projekt
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/brucki2811/CSV-Projekt.git # timeout=10
Fetching upstream changes from https://github.com/brucki2811/CSV-Projekt.git
> git --version # timeout=10
> git -c core.askpass=true fetch --tags --progress https://github.com/brucki2811/CSV-Projekt.git
+refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 1c621c612113f8aa7225acdc781b5efc0e90de9c (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 1c621c612113f8aa7225acdc781b5efc0e90de9c
> git rev-list 5b62b634d2e101bba64cd3fff529123d5195e0b7 # timeout=10
[CSV-Projekt] $ /bin/sh -xe /tmp/hudson8538853897619250250.sh
+ set +e
+ PYTHONPATH=
+ cd /var/lib/jenkins/workspace/CSV-Projekt
+ chmod -R 777 .
+ nosetests-3.4 --with-xunit --all-modules --traverse-namespace --with-coverage --cover-package=csv_tool --cover-inclusive

Name                Stmts  Miss  Cover   Missing
-----
csv_tool.py          23      15    35%    15-16, 22-33, 39-41
-----
Ran 0 tests in 0.015s

OK
+ chmod -R 777 .
+ python3 -m coverage xml --include=csv_tool*
No source for code: '/var/lib/jenkins/workspace/CSV-Projekt/csv_tool/CSVUebung.py'.
+ chmod -R 777 .
+ tee pylint.out
+ pylint -f parseable -d I0011,R0801 csv_tool
No config file found, using default configuration
***** Module csv_tool
csv_tool/_init_.py:21: [C0326(bad-whitespace), ] Exactly one space required after comma
def read(self,filename):
    ^
csv_tool/_init_.py:38: [C0326(bad-whitespace), ] Exactly one space required after comma
def write(self,filename):
    ^

csv_tool/_init_.py:1: [C0111(missing-docstring), ] Missing module docstring
csv_tool/_init_.py:10: [C0413(wrong-import-position), ] Import "import csv" should be placed at the top of the
module
csv_tool/_init_.py:12: [C0103(invalid-name), csvwr] Invalid class name "csvwr"
csv_tool/_init_.py:12: [C0111(missing-docstring), csvwr] Missing class docstring
csv_tool/_init_.py:20: [W0105(pointless-string-statement), csvwr] String statement has no effect
csv_tool/_init_.py:21: [C0111(missing-docstring), csvwr.read] Missing method docstring
csv_tool/_init_.py:22: [C0103(invalid-name), csvwr.read] Invalid variable name "f"
csv_tool/_init_.py:25: [W0702(bare-except), csvwr.read] No exception type(s) specified
csv_tool/_init_.py:30: [C0103(invalid-name), csvwr.read] Invalid variable name "n"
csv_tool/_init_.py:37: [W0105(pointless-string-statement), csvwr] String statement has no effect
csv_tool/_init_.py:38: [C0111(missing-docstring), csvwr.write] Missing method docstring
csv_tool/_init_.py:39: [C0103(invalid-name), csvwr.write] Invalid variable name "f"

```

## Report

=====

26 statements analysed.

## Statistics by type

-----

type	number	old number	difference	%documented	%badname
module	1	NC	NC	0.00	0.00
class	1	NC	NC	0.00	100.00
method	3	NC	NC	33.33	0.00
function	0	NC	NC	0	0

## Raw metrics

-----

type	number	%	previous	difference
code	31	70.45	NC	NC
docstring	4	9.09	NC	NC
comment	1	2.27	NC	NC
empty	8	18.18	NC	NC

## Duplication

-----

	now	previous	difference
nb duplicated lines	0	NC	NC
percent duplicated lines	0.000	NC	NC

## Messages by category

-----

type	number	previous	difference
convention	11	NC	NC
refactor	0	NC	NC
warning	3	NC	NC
error	0	NC	NC

## Messages

-----

message id	occurrences
missing-docstring	4
invalid-name	4
pointless-string-statement	2
bad-whitespace	2
wrong-import-position	1
bare-except	1

## Global evaluation

-----

Your code has been rated at 4.62/10

+ chmod -R 777 .

Recording test results

ERROR: Step 'Publish JUnit test result report' failed: None of the test reports contained any result

Skipping Cobertura coverage report as build was not UNSTABLE or better ...

Finished: FAILURE

**Fehler behoben!****Lösung:**

Ich habe das File `__init__.py` falsch verwendet. In dem eben genannten File soll nichts stehen und der Code, den ich darin hatte, habe ich in das File `CSVUebung.py` geschrieben. Nun habe ich noch eine Kleinigkeit in der Build-Shell verändert und die Tests zum Laufen gebracht.

## Build-Shell:

## Buildverfahren

## Shell ausführen

Befehl

```
set +e
PYTHONPATH=''
cd /var/lib/jenkins/workspace/CSV-Projekt/
nosetests-3.4 --with-xunit --all-modules --traverse-namespace --with-coverage --cover-package=csv_tool --cover-inclusive
python3 -m coverage xml
pylint -f parseable -d I0011,R0801 csv_tool | tee pylint.out
```

## Ergebnis:

Name	Stmts	Miss	Cover	Missing
------	-------	------	-------	---------

csv_tool.py	0	0	100%	
-------------	---	---	------	--

Ran 0 tests in 0.010s