https://kefeng.wang/2017/01/06/jenkins/

[**Jenkins 持续集成综合实战**](https://kefeng.wang/2017/01/06/jenkins/)

 发表于 2017-01-06

Jenkins 是一款流行的开源持续集成（Continuous Integration）工具，广泛用于项目开发，具有自动化构建、测试和部署等功能。本文以 CentOS7 环境为例，总结了 Jenkins 的安装与配置、邮件功能使用，并接入阿里巴巴的著名开源项目 fastjson，以此演示 Java 项目（SVN+Maven）中 FindBugs/CheckStyle/PMD 等常用插件的使用、单元测试及其覆盖率报告等，力求实战性强。  
  
**作者：**王克锋  
**出处：**<https://kefeng.wang/2017/01/06/jenkins/>  
**版权：**[自由转载-非商用-非衍生-保持署名](https://creativecommons.org/licenses/by-nc-nd/3.0/deed.zh)，转载请标明作者和出处。

**1 安装 Jenkins**

官网: <https://jenkins.io/>

**1.1 安装 JDK/Maven**

Jenkins 自身采用 Java 开发，所以要必须安装 JDK；  
本文集成的项目基于 Maven 构架，所以 Maven 也必须安装；  
两者的安装方法请参考我的另一篇文章：[Tomcat 安装及其单机多实例部署](https://kefeng.wang/2016/12/26/tomcat-installation/)

|  |  |
| --- | --- |
| 1 2 | export JAVA\_HOME=/usr/java/jdk1.8.0\_111 export MAVEN\_HOME=/opt/apache-maven-3.3.9 |

**1.2 安装 Jenkins**

以下四种方法任选一种，推荐第一种。

**1.2.1 离线安装**

|  |  |
| --- | --- |
| 1 2 3 4 | ## http://pkg.jenkins-ci.org/redhat/ wget http://pkg.jenkins-ci.org/redhat/jenkins-2.39-1.1.noarch.rpm ## 下载(也可以Windows下载再转过来) sudo rpm --import http://pkg.jenkins-ci.org/redhat/jenkins.io.key ## 公钥 sudo yum -y install jenkins-\*.noarch.rpm |

**1.2.2 在线安装**

|  |  |
| --- | --- |
| 1 2 3 4 | ## http://pkg.jenkins-ci.org/redhat/ sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenkins.repo sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key sudo yum -y install jenkins |

**1.2.3 基于 Tomcat 安装**

* 安装并启动 Tomcat；
* 从官网下载 [jenkins.war](http://mirrors.jenkins.io/war/latest/jenkins.war) 至 $CATALINA\_BASE/webapps，Tomcat 会自动部署；
* 浏览器访问：<http://centos:8080/jenkins/>

**1.2.4 免安装方式**

|  |  |
| --- | --- |
| 1 2 3 | wget http://mirrors.jenkins.io/war-stable/latest/jenkins.war sudo java -jar jenkens.war ## 启动服务，直至看到日志 `Jenkins is fully up and running` curl http://localhost:8080/ ## Jenkins 已就绪 |

**1.3 调整配置文件**

|  |  |
| --- | --- |
| 1 2 3 4 | ## sudo vim /etc/sysconfig/jenkins JENKINS\_USER="root" ## 原值 "jenkins" 必须修改，否则权限不足 JENKINS\_PORT="8080" ## 原值 "8080" 可以不修改 ## 还有开启 HTTPS 相关参数，此处不打算开启，故不作说明 |

安装目录: /usr/lib/jenkins  
工作目录: /var/lib/jenkins(对应于环境变量 JENKINS\_HOME)

**1.4 启动**

|  |  |
| --- | --- |
| 1 2 | sudo systemctl enable jenkins sudo systemctl restart jenkins |

查看日志文件: sudo tail -f /var/log/jenkins/jenkins.log

启动后会生成文件 hudson.model.UpdateCenter.xml，需要修改它，  
否则浏览器首次进入时会卡在“Jenkins 正在启动，请稍后…”

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 | ## 原值: http://updates.jenkins-ci.org/update-center.json ## 新值: http://mirror.xmission.com/jenkins/updates/update-center.json ## 或者: http://mirror.xmission.com/jenkins/updates/current/update-center.json sudo cat /var/lib/jenkins/hudson.model.UpdateCenter.xml sudo sed -i 's/updates.jenkins-ci.org/mirror.xmission.com\/jenkins\/updates/g' /var/lib/jenkins/hudson.model.UpdateCenter.xml sudo cat /var/lib/jenkins/hudson.model.UpdateCenter.xml sudo systemctl restart jenkins |

**1.5 防火墙放行**

|  |  |
| --- | --- |
| 1 2 3 | ### sudo vim /etc/sysconfig/iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport 8080 -j ACCEPT ### 重启生效: sudo systemctl restart iptables |

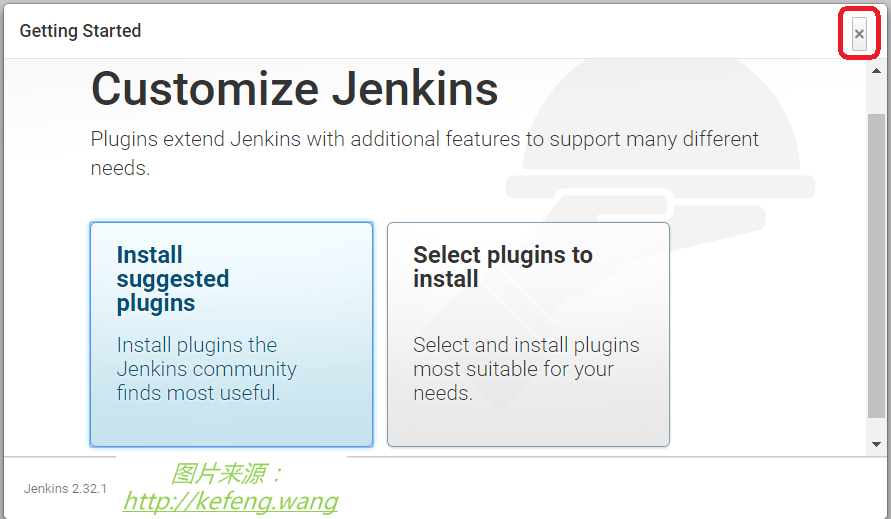
**1.6 卸载**

如果要重装，或者本机不需要 Jenkins，可卸载：

|  |  |
| --- | --- |
| 1 2 3 4 5 | sudo systemctl stop jenkins sudo systemctl disable jenkins sudo yum -y remove jenkins sudo rm -rf /var/{lib,log,cache}/jenkins /usr/lib/jenkins /root/.jenkins sudo rm -rf `sudo find /{etc,var,run} -name "jenkins\*"` |

**2 基本配置**

**2.1 登录**

浏览器进入 <http://centos:8080/>  
首次进入需要输入初始密码来解锁，初始密码查看方法: sudo cat /var/lib/jenkins/secrets/initialAdminPassword  
[](https://kefeng.wang/images/jenkins/unlock.png)jenkins unlock  
进入 Jenkins 定制界面，让选择 Install suggested plugins 或 Select plugins to install 时，两者都不要选，直接关闭，下一个界面点击“Start using Jenkins”；  
[](https://kefeng.wang/images/jenkins/customize.png)jenkins customize  
修改登录密码：主界面右上角“注销”链接的左边的用户名下拉菜单中点“设置”。

**2.2 安全设置**

默认情况下，任何用户都可以使用 Jenkins 进行发布。  
可以进入相关设置：系统管理 / Configure Global Security,  
选择 Jenkins专有用户数据库，不要选中 允许用户注册；  
选择 登录用户可以做任何事，选中 Allow anonymous read access

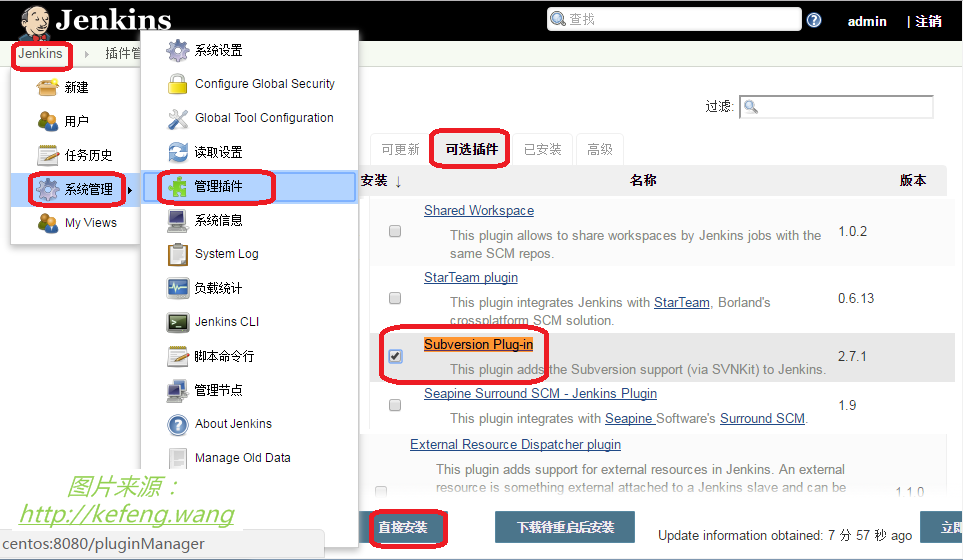
**2.3 插件安装方法**

如果安装插件失败率很高，可设置翻墙，请参考我之前的文章：[翻墙代理 Shadowsocks 使用详解](https://kefeng.wang/2016/10/05/shadowsocks/)

**2.3.1 离线安装**

手工下载(\*.hpi): <http://updates.jenkins-ci.org/download/plugins/>  
进入：系统管理 / 管理插件 / 高级，然后上传插件进行安装。  
无需重启 Jenkins 插件即生效。

**2.3.2 在线安装**

进入：系统管理 / 管理插件 / 可选插件  
查找并勾选所需插件，点击“直接安装”；  
无需重启 Jenkins 插件即生效。如遇失败可重试或离线安装。  
[](https://kefeng.wang/images/jenkins/plugins.png)jenkins plugins

**2.4 工具设置**

进入：系统管理 / Global Tool Configuration  
JDK 下不勾选“自动安装”，指定别名=JDK-1.8.0.111, JAVA\_HOME=/usr/java/jdk1.8.0\_111  
Maven 下不勾选“自动安装”，指定别名=Maven-3.3.9, MAVEN\_HOME=/opt/apache-maven-3.3.9  
点击 Save 按钮。  
重启 Jenkins 后生效：sudo systemctl restart jenkins  
[](https://kefeng.wang/images/jenkins/tools.png)jenkins tools

**3 Maven(SVN) 项目接入**

**3.1 准备 SVN 源码**

使用 alibaba 的 fastjson 来演示：  
下载: <https://github.com/alibaba/fastjson/archive/master.zip>  
SVN地址: svn://centos/repo1/fastjson

**3.2 安装相关插件**

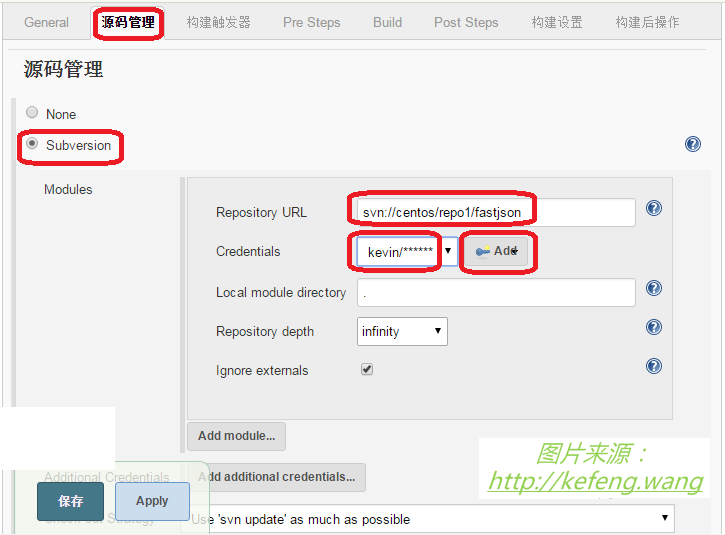
* [Subversion Plug-in](http://updates.jenkins-ci.org/latest/subversion.hpi) ## 版本管理 SVN 的插件
* [Git plugin](http://updates.jenkins-ci.org/latest/git.hpi) ## 版本管理 GIT 的插件
* [Maven Integration plugin](http://updates.jenkins-ci.org/latest/maven-plugin.hpi) ## 项目构建 Maven 的插件
* [Gradle Plugin](http://updates.jenkins-ci.org/latest/gradle.hpi) ## 项目构建 Gradle 的插件  
  无需重启 Jenkins 插件即生效。如遇失败可重试或离线安装。

**3.3 新建构建项目**

**3.3.1 开始创建**

主页：点击“新建”；  
项目类型：输入项目名称 fastjson，类型选择“构建一个Maven项目”(如果没有此项可选，请检查插件“Maven Integration plugin”是否已安装)；  
[](https://kefeng.wang/images/jenkins/create.png)jenkins create

**3.3.2 源码管理**

选择“Subversion”(如果没有此项可选，请检查插件“Subversion Plug-in”是否已安装)，  
仓库URL可以指定 svn/http/https 之一，这里指定 svn://centos/repo1/fastjson  
Credentials: 添加 SVN 用户名密码，并从下拉列表中选择它。  
[](https://kefeng.wang/images/jenkins/source.png)jenkins source

**3.3.3 定时触发构建**

构建触发器: 只选中 Build periodically, “日程表”格式与 crontab 相似但有细微差别，示例如下：

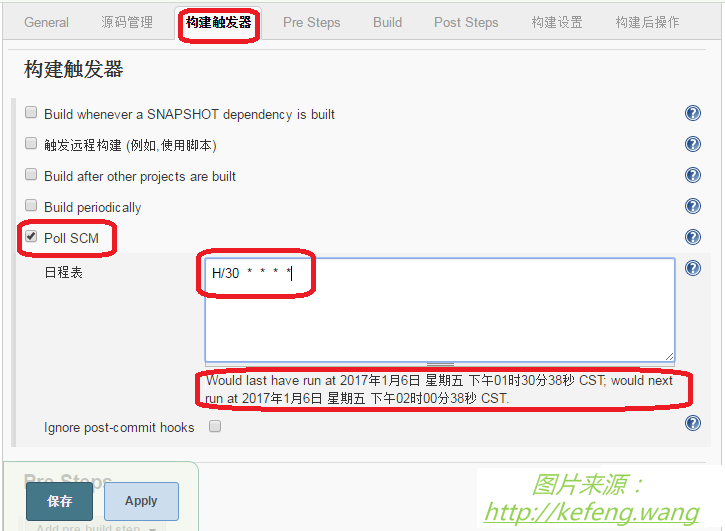
|  |  |
| --- | --- |
| 1 2 3 4 5 6 | ## 每行由 5 个值组成(空格或TAB分隔)，分别表示分(0-59)、时(0-23)、日(1-31)、月(1-12)、周(0-7, 0/7=周日) ## "M,N" 表示M和N；"M-N" 表示范围[M,N]；"M-N/X" 表示范围[M,N]内每隔X；"\*/X" 表示整个范围内每隔X ## 前面提到的M/N/X的值都可以用H(意为Hash)代替，散列值起到随机值的效果，且同一项目取值稳定，这对于项目多时分散压力很有用。 H/10 H(0-8) \* \* 1-5 ## 触发时间: 工作日、Hour为0~8按哈希随机、Minute以10为间隔 H/10 H \* \* 0,6,7 ## 触发时间: 周末、Hour为全天按哈希随机、Minute以10为间隔 ## “日程表”修改后，下方会给出下次执行时间点的预告。 |

[](https://kefeng.wang/images/jenkins/trigger1.png)jenkins trigger

**3.3.4 版本提交触发构建**

构建触发器: 只选中 Poll SCM，可指定检查 SVN 代码是否有提交的时间：

|  |  |
| --- | --- |
| 1 2 | ## 格式与 "Build periodically" 相同 H/10 \* \* \* \* ## 任何时候，每隔 30 分钟就检测一次 SVN，如果有提交就启动构建 |

[](https://kefeng.wang/images/jenkins/trigger2.png)jenkins trigger

**3.3.5 Build 设置**

Root POM: 指定 pom.xml 相对路径  
Goals and options: 指定 mvn 执行目标或选项，设置为 package -Dmaven.test.skip=true

**3.3.6 Post Steps 设置**

定制构建后的操作，通常选择“Run only if build succeeds”，支持 shell 等脚本；

|  |  |
| --- | --- |
| 1 2 3 4 5 | ## 脚本工作目录为: /var/lib/jenkins/workspace/fastjson # sudo systemctl stop tomcat # sudo cp -f target/\*.war /opt/apache-tomcat-8.5.9/webapps/ # sudo systemctl start tomcat ls -l target/\*.?ar |

[](https://kefeng.wang/images/jenkins/poststep.png)jenkins poststep

**3.4 手工触发构建**

进入 My Views, 右侧看到各个项目；  
点击进入关注的项目，点击左侧的“立即构建”；  
[](https://kefeng.wang/images/jenkins/build.png)jenkins build  
开始构建或构建完毕后，左下方列出每次构建的链接，点击进入某次构建；  
点击左侧的“Console Output”，可查看构建日志，如有报错可以看到；  
[](https://kefeng.wang/images/jenkins/console.png)jenkins console  
纠正错误后，返回到工程，再次点击“立即构建”，直至构建成功；  
如有网络相关报错，重试几次也会成功。

**4 开启 Java 代码静态检查**

**4.1 安装相关插件**

如果是离线安装，需要先依次安装被依赖的插件：

* [Script Security Plugin](http://updates.jenkins-ci.org/latest/script-security.hpi)
* [Matrix Project Plugin](http://updates.jenkins-ci.org/latest/matrix-project.hpi)
* [Static Analysis Utilities](http://updates.jenkins-ci.org/latest/analysis-core.hpi)

再安装直接被使用的插件：

* [FindBugs Plug-in](http://updates.jenkins-ci.org/latest/findbugs.hpi)
* [Checkstyle Plug-in](http://updates.jenkins-ci.org/latest/checkstyle.hpi)
* [PMD Plug-in](http://updates.jenkins-ci.org/latest/pmd.hpi)

插件官方资料：  
FindBugs: <https://wiki.jenkins-ci.org/display/JENKINS/FindBugs+Plugin>  
Checkstyle: <http://maven.apache.org/plugins/maven-checkstyle-plugin/usage.html>  
PMD: <http://maven.apache.org/plugins/maven-pmd-plugin/usage.html>

**4.2 Maven 中设置插件(pom.xml)**

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 | <project ...>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  </properties>   <!-- ...... -->   <reporting>  <plugins>  <plugin>  <groupId>org.codehaus.mojo</groupId>  <artifactId>findbugs-maven-plugin</artifactId>  <version>3.0.4</version>  <configuration>  <xmlOutput>true</xmlOutput>  <findbugsXmlOutput>true</findbugsXmlOutput>  <findbugsXmlWithMessages>true</findbugsXmlWithMessages>  </configuration>  </plugin>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-checkstyle-plugin</artifactId>  <version>2.17</version>  <configuration>  <linkXRef>false</linkXRef>  <failsOnError>true</failsOnError>  <consoleOutput>true</consoleOutput>  <configLocation>checkstyle.xml</configLocation>  </configuration>  </plugin>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-pmd-plugin</artifactId>  <version>3.7</version>  <configuration>  <linkXref>false</linkXref>  </configuration>  </plugin>  </plugins>  </reporting> </project> |

**4.3 修改 Jenkins 构建配置**

[](https://kefeng.wang/images/jenkins/findbugs-config.png)jenkins findbugs config

**4.3.1 设置 Maven**

“Build”标签页，Goals and options 设置为：  
package -Dmaven.test.skip=true findbugs:findbugs checkstyle:checkstyle pmd:pmd

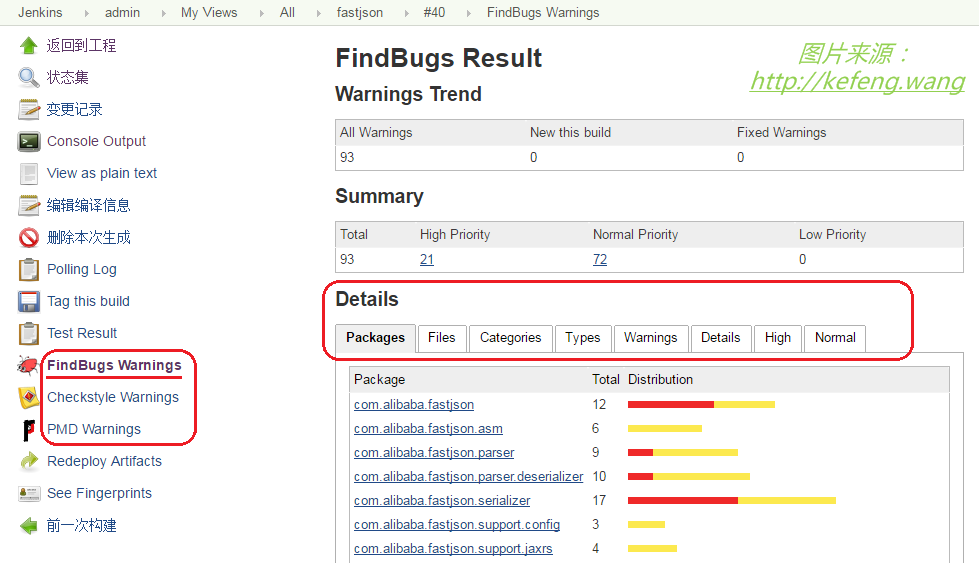
**4.3.2 开启相关报告**

“构建设置”标签页，打开以下三个选项：

* Publish FindBugs analysis results
* Publish Checkstyle analysis results
* Publish PMD analysis results

**4.4 构建后查看报告**

构建后进入构建页面，从左边点击查看报告：

* FindBugs Warnings
* Checkstyle Warnings
* PMD Warnings  
  [](https://kefeng.wang/images/jenkins/findbugs-report.png)jenkins findbugs report

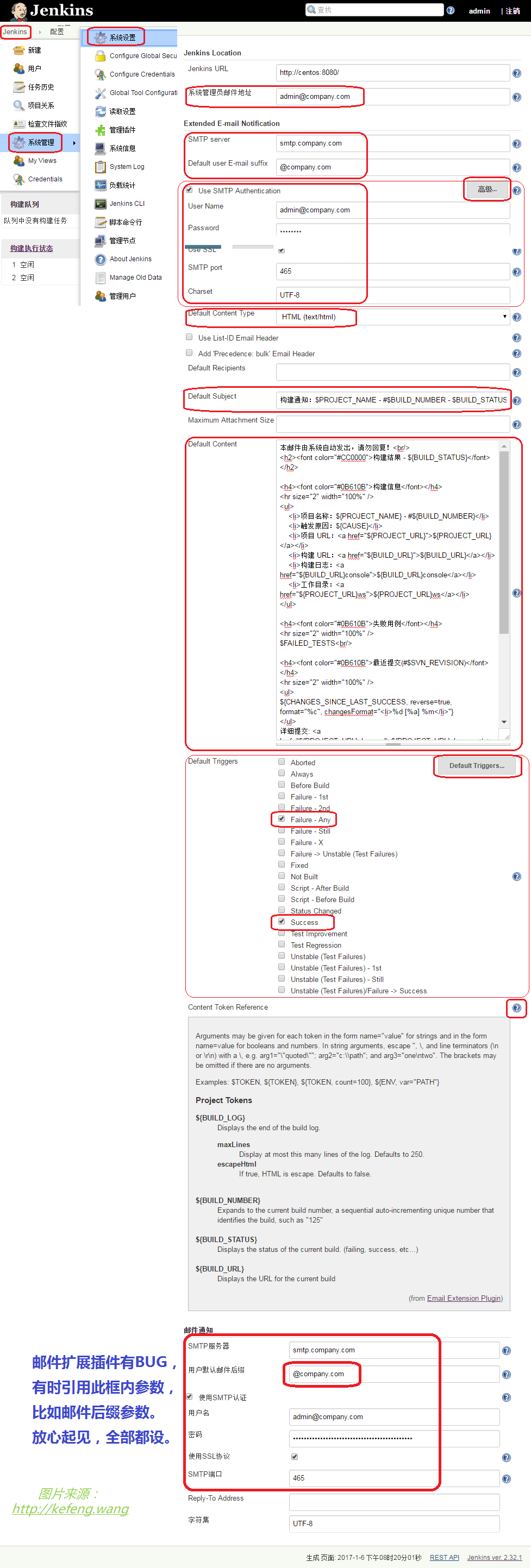
**5 开启邮件报告**

Jenkins 的邮件发送，原理与邮件客户端一样：把发件人的邮箱帐号、密码、SMTP服务器告诉 Jenkins，Jenkins 需要时使用该帐号向指定的收件人发送邮件报告。

**5.1 安装相关插件**

Jenkins 自带邮件插件(Mailer Plugin)的功能很弱且无法自定制，  
需要安装扩展邮件插件: Email Extension Plugin

**5.2 全局统一设置**

以 [admin@company.com](mailto:admin@company.com) 为例，如下图设置：  
[](https://kefeng.wang/images/jenkins/email-global.png)jenkins email-global

**5.2.1 设置管理员邮箱**

进入：系统管理 / 系统设置 / Jenkins Location  
系统管理员邮件地址 设为 [admin@company.com](mailto:admin@company.com)

**5.2.2 设置发件人帐号**

进入：系统管理 / 系统设置 / Extended E-mail Notification  
“SMTP server” 设为 smtp.company.com  
“Default user E-mail suffix” 设为 @company.com  
“Use SMTP Authentication” 选中  
“User Name” = [admin@company.com](mailto:admin@company.com)(必须与管理员邮箱相同)  
“Password” = *\*\**  
“Use SSL” = on  
“SMTP port” = 465  
“Charset” = UTF-8  
“Default Content Type” = HTML (text/html)

**5.2.3 设置邮箱模板**

Default Subject: 构建通知：$PROJECT\_NAME - #$BUILD\_NUMBER - $BUILD\_STATUS  
Default Content:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 | 本邮件由系统自动发出，请勿回复！<br/> <h2><font color="#CC0000">构建结果 - ${BUILD\_STATUS}</font></h2>  <h4><font color="#0B610B">构建信息</font></h4> <hr size="2" width="100%" /> <ul>  <li>项目名称：${PROJECT\_NAME} - #${BUILD\_NUMBER}</li>  <li>触发原因：${CAUSE}</li>  <li>项目 URL：<a href="${PROJECT\_URL}">${PROJECT\_URL}</a></li>  <li>构建 URL：<a href="${BUILD\_URL}">${BUILD\_URL}</a></li>  <li>构建日志：<a href="${BUILD\_URL}console">${BUILD\_URL}console</a></li>  <li>工作目录：<a href="${PROJECT\_URL}ws">${PROJECT\_URL}ws</a></li> </ul>  <h4><font color="#0B610B">失败用例</font></h4> <hr size="2" width="100%" /> $FAILED\_TESTS<br/>  <h4><font color="#0B610B">最近提交(#$SVN\_REVISION)</font></h4> <hr size="2" width="100%" /> <ul> ${CHANGES\_SINCE\_LAST\_SUCCESS, reverse=true, format="%c", changesFormat="<li>%d [%a] %m</li>"} </ul> 详细提交: <a href="${PROJECT\_URL}changes">${PROJECT\_URL}changes</a><br/>  <h4><font color="#0B610B">代码检查</font></h4> <hr size="2" width="100%" /> FindBugs：<a href="${BUILD\_URL}findbugsResult">${BUILD\_URL}findbugsResult</a><br/> CheckStyle：<a href="${BUILD\_URL}checkstyleResult">${BUILD\_URL}checkstyleResult</a><br/> PMD：<a href="${BUILD\_URL}pmdResult">${BUILD\_URL}pmdResult</a><br/> |

也可以自定义 Jelly 模板，并在 Default Content 中引用 ${JELLY\_SCRIPT, template="html\_gmail"}：

|  |  |
| --- | --- |
| 1 2 3 4 | sudo mkdir -p /var/lib/jenkins/email-templates sudo cd /var/lib/jenkins/email-templates sudo wget https://github.com/jenkinsci/email-ext-plugin/raw/master/src/main/resources/hudson/plugins/emailext/templates/html\_gmail.jelly # 更多模板列表: https://github.com/jenkinsci/email-ext-plugin/tree/master/src/main/resources/hudson/plugins/emailext/templates |

**5.2.4 设置邮件触发时机**

点击 Default Triggers，比较好的方案是选中两项：

* Failure - Any ## 可定义“编译失败时”才向哪些人员发邮件(比如最近代码提交者)
* Success ## 可定义“构建成功时”才向哪些人员发邮件(比如所有产品关注者)

**5.2.5 邮件可用变量**

点击 Content Token Reference 右侧的问号图标，可以看到各邮件模板中可引用变量的含义。

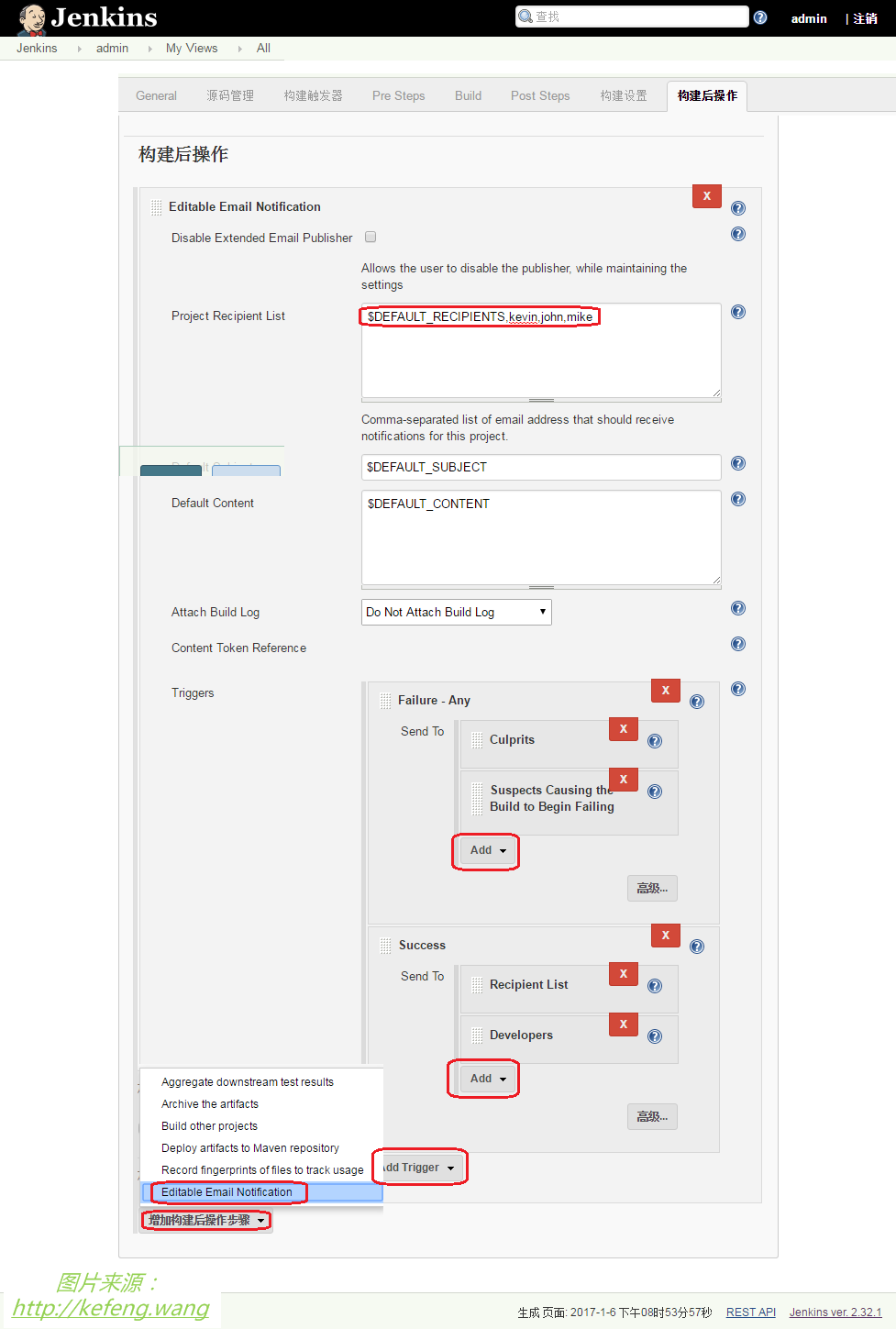
**5.2.6 邮件未发送问题排查**

构建设置里指定的收件人只是 kevin，没有后续，希望采用插件 Email Extension Plugin 中的邮箱后缀，  
却发现构建日志中报错: Failed to send e-mail to kevin because no e-mail address is known, and no default e-mail domain is configured  
原来，扩展邮件插件有BUG，使用的是自带邮件插件的后缀设置，而后者之前没有设置。  
点击 保存，保存邮件全局设置。

**5.3 项目设置中启用邮件**

构建后操作步骤: 增加 Editable Email Notification(来自插件 Email Extension Plugin)。下图设置的效果是：

* 构建失败时，向上次成功构建之后导致构建失败的 SVN 提交的人员发送邮件；
* 构建成功时，向上次成功构建之后有 SVN 提交的人员、还有“Project Recipient List”中的人员发送邮件；

[](https://kefeng.wang/images/jenkins/email-project.png)jenkins email-project

**5.4 验证构建报告邮件**

分别按以下三种情形触发构建，都能看到构建报告邮件：

* 手工触发：进入工程后，点击“立即构建”；
* 定时触发构建：“构建触发器”只选中 Build periodically；
* 版本提交触发构建：“构建触发器”只选中 Poll SCM。  
  [](https://kefeng.wang/images/jenkins/email-report.png)jenkins email-report

**6 Java/JUnit 单元测试**

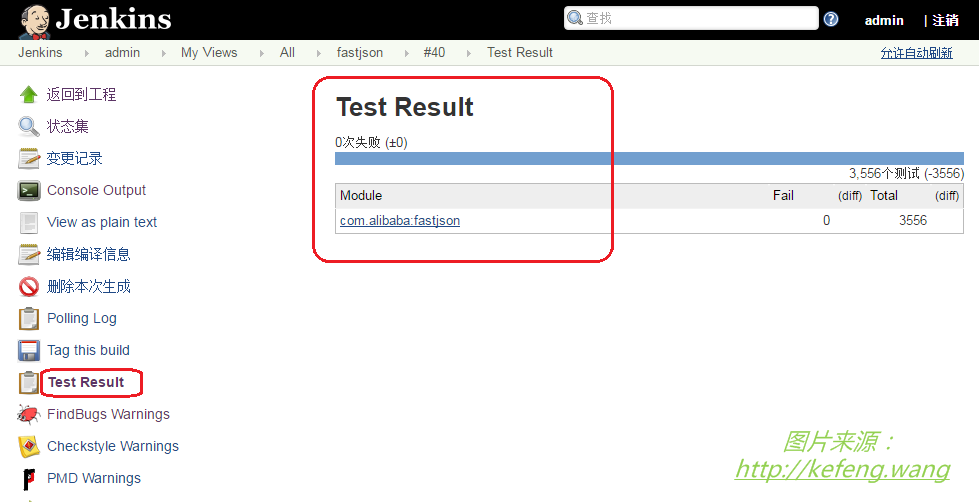
**6.1 Maven 中设置插件(pom.xml)**

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 | <project ...>  <reporting>  <plugins>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-surefire-plugin</artifactId>  <version>2.19.1</version>  </plugin>  </plugins>  </reporting> </project> |

**6.2 修改 Jenkins 构建配置**

“Build”标签页，Goals and options：确保去掉-Dmaven.test.skip=true即可。

**6.3 构建后查看单元测试报告**

[](https://kefeng.wang/images/jenkins/testing.png)jenkins findbugs report

**7 单元测试覆盖率报告**

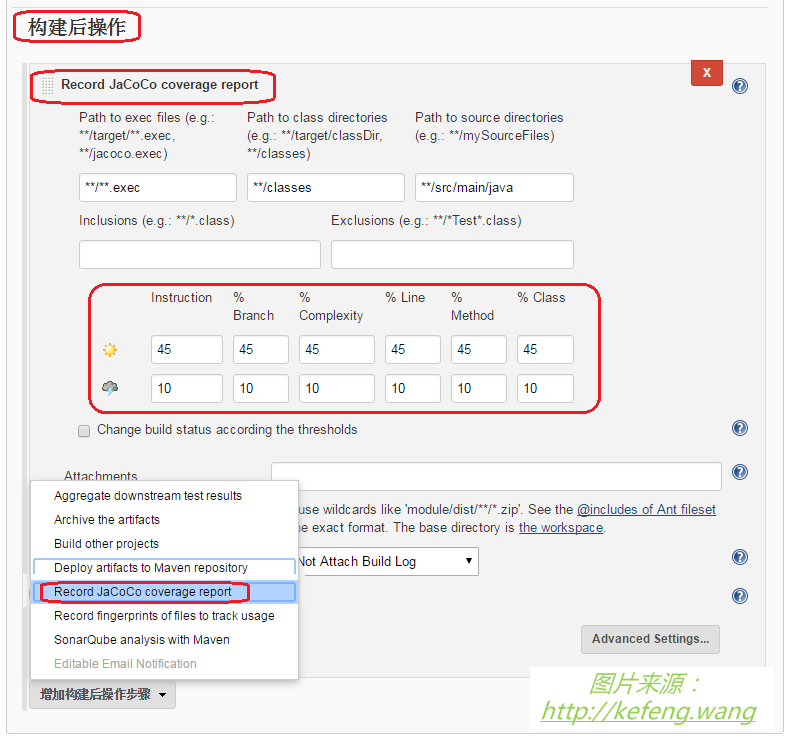
**7.1 安装插件**

[JaCoCo plugin](http://updates.jenkins-ci.org/latest/jacoco.hpi)：与插件 Cobertura 一样，用于生成覆盖率报告，但比 Cobertura 更易用。

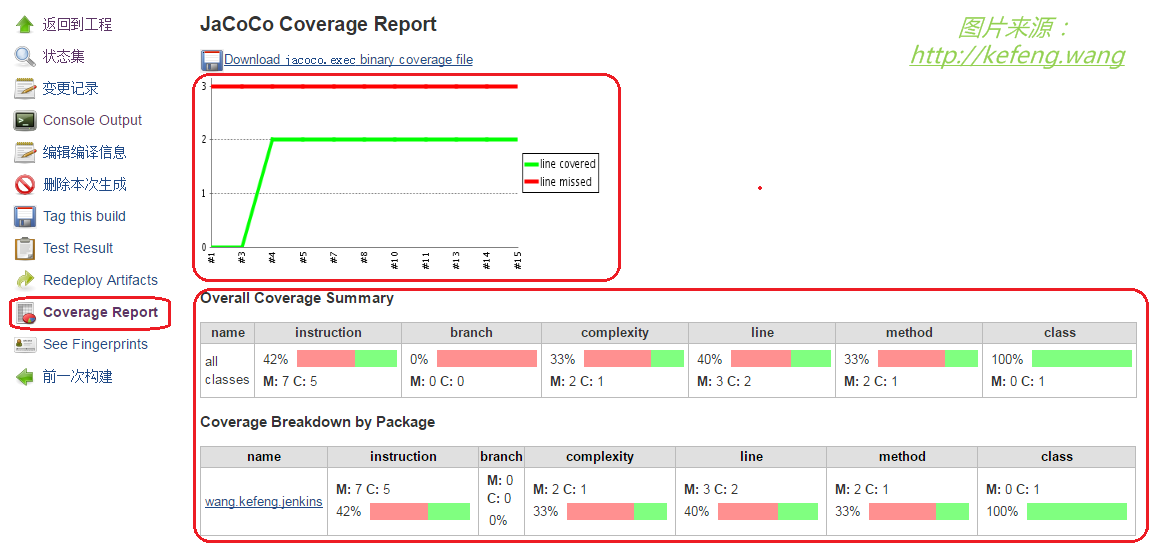
**7.2 调整工程 pom.xml**

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 | <build>  <plugins>  <plugin>  <groupId>org.jacoco</groupId>  <artifactId>jacoco-maven-plugin</artifactId>  <version>0.7.8</version>  <executions>  <execution>  <id>prepare</id>  <goals>  <goal>prepare-agent</goal>  </goals>  </execution>  <execution>  <id>report</id>  <phase>package</phase>  <goals>  <goal>report</goal>  </goals>  </execution>  </executions>  </plugin>  </plugins> </build> |

**7.3 调整项目的构建设置**

项目的构建设置 Post Build Action 中，增加 “Record JaCoCo coverage report”，并设置期望的覆盖率阈值。  
[](https://kefeng.wang/images/jenkins/jacoco-config.png)jenkins jacoco-config

**7.4 构建并查看报告**

构建完成后，点击本次构建页的“Coverage Report”查看报告：  
[](https://kefeng.wang/images/jenkins/jacoco-report.png)jenkins jacoco-report

**8 更多有用插件**

* Folders Plugin：可嵌套地定义文件夹来级别 views / jobs；
* build timeout plugin：允许用户中止耗时太久的构建；
* Dingding[钉钉] Plugin：允许向阿里钉钉发送构建通知；
* TestNG Results Plugin: 用 TestNG 代替 JUnit；
* Publish Over SSH: 通过SSH发布文件或执行命令(使用SFTP的SCP)。