

Jenkins 使用指南

一、安装及启动

1. 直接从 <http://mirrors.jenkins-ci.org/war/latest/jenkins.war> 下载 war 包

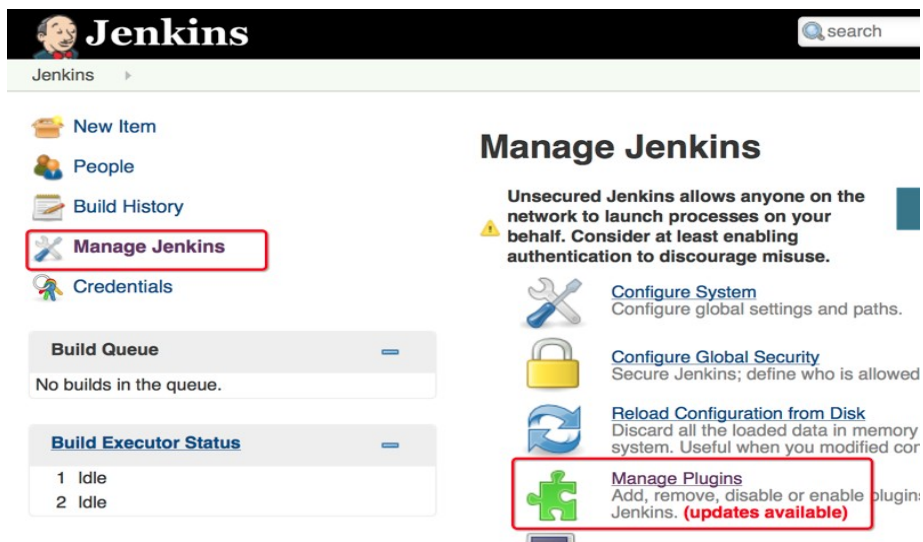
2. 启动

(1) java -jar jenkins.war, 启动成功后, 浏览 <http://localhost:8080/>

(2) 将 jenkins.war 复制到 tomcat-webapps 目录下, 启动 tomcat, 然后打开网页 <http://localhost:8080/jenkins>

二、下载插件

1. 打开 jenkins 网页后, 可以修改登录名和登陆密码, 然后从左侧选 Manage Jenkins → Manage Plugins (如下图所示), 下载必要插件并安装。



2. Jenkins 的很多功能都是借助 plugin 来完成的, 首次启动时很多插件会提示已经过时, 需要更新, 建议升级成最新版本, 同时建议安装以下插件, 以方便支持 Maven, git 项目。同时, 若需要发送 Email, 需要下载 Email 插件。



三、系统配置

1. 从左侧选 Manage Jenkins → Global Tool Configuration (如下图所示) 进行配

置。

(1) JDK

不勾选“Install automatically”，JAVA_HOME 填写安装 JDK 的路径

(2) Maven

不勾选“Install automatically”，MAVEN_HOME 填写安装 Maven 的路径

注意：保存设置

 New Item

 People

 Build History

 Project Relationship

 Check File Fingerprint

 **Manage Jenkins**

 My Views

 Credentials

Manage Jenkins

 [Configure System](#)
Configure global settings and paths.

 [Configure Global Security](#)
Secure Jenkins; define who is allowed to access/use the system.

 [Configure Credentials](#)
Configure the credential providers and types

 [Global Tool Configuration](#)
Configure tools, their locations and automatic installers.

JDK

JDK installations

 **JDK**

Name


JDK1.7

JAVA_HOME

/usr/lib/jvm/jdk1.7.0

☐ Install automatically

Maven installations

 **Maven**

Name

Maven3.3.9

MAVEN_HOME

/home/beaver/Documents/java/apache-maven-3.3.9

☐ Install automatically

四、新建一个 Maven 项目

1. 选择左侧任务栏的 New Item，输入项目名称，选择 Maven project，点击 OK。

Enter an item name

» Required field



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system or something other than software build.



Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

2. 项目配置

(1) General

可以在 Description 中对项目进行介绍。

General Maven Info Plugin Configuration Source Code Management Build Tool Configuration

Post Steps Build Settings Post-build Actions

Maven project name

Test

Description

This is a test project for jenkins.

[Plain text] [Preview](#)

(2) Source Code Management

选择代码的来源（Git，CSV，SVN），这里我们选择 Git。填写版本库的 URL 和分支名称。

Git

Repositories

Repository URL

Credentials

- none -

[Add](#)

[Advanced...](#)

[Add Repository](#)

Branches to build

Branch Specifier (blank for 'any')

(3) Build Triggers

设置项目运行的触发条件。这里，我们选择 Build periodically，设置在每天晚上 8 点运行，点击右边的“问号”可以查看如何设置时间。

Build Triggers

☐ Build whenever a SNAPSHOT dependency is built

☐ Trigger builds remotely (e.g., from scripts)

☐ Build after other projects are built

☒ Build periodically

Schedule

H 20 * * *

Would last have run at Tuesday, July 19, 2016 8:48:15 PM CST; would next run at Wednesday, July 20, 2016 8:48:15 PM CST.

☐ Build when a change is pushed to GitHub

☐ GitHub Pull Request Builder

☐ Poll SCM

注：Schedule 的配置规则是有 5 个空格隔开的字符组成，从左到右分别代表：分 时 天 月 年。
*代表所有，H 20 * * * 表示“在任何年任何月任何天的 20 点进行构建。

(4) Build

Goals and options 填写命令，不加 mvn

Build

Root POMpom.xml

Goals and optionsclean package

(5) Build Settings

设置邮件通知，填写需要通知的邮箱地址

Build Settings

☒ E-mail Notification

Recipients

jiajia@cloudbeaver.com;shujie@cloudbeaver.com

☒ Send e-mail for every unstable build


☐ Send separate e-mails to individuals who broke the build






☒ Send e-mail for each failed module



(6) 点击保存

五、运行项目

选择左侧任务栏的 Build Now，左下侧 Build History 会显示进度，点击进度可以看到 Console Output。

 **Build History** [trend](#)

 #6	Jul 19, 2016 12:58 PM
 #5	Jul 19, 2016 11:39 AM
 #4	Jul 19, 2016 11:22 AM
 #3	Jul 19, 2016 10:49 AM
 #1	Jul 18, 2016 6:25 PM

 [RSS for all](#)  [RSS for failures](#)

 [Back to Project](#)

 [Status](#)

 [Changes](#)

 **Console Output**

 [View as plain text](#)

 [Edit Build Information](#)


 [Delete Build](#)

 [Git Build Data](#)

 [No Tags](#)

 [Redeploy Artifacts](#)

 [See Fingerprints](#)

 [Previous Build](#)

Console Output

```
Started by user admin
Building in workspace /home/beaver/.jenkins/works
> git rev-parse --is-inside-work-tree # timeout=
Fetching changes from the remote Git repository
> git config remote.origin.url git@github.com:Be
Fetching upstream changes from git@github.com:Bea
> git --version # timeout=10
> git -c core.askpass=true fetch --tags --progre
/origin/*
> git rev-parse refs/remotes/origin/test-cases^{
> git rev-parse refs/remotes/origin/origin/test-
Checking out Revision e5ef1491a856ce4ab3a03bcd03b
> git config core.sparsecheckout # timeout=10
> git checkout -f e5ef1491a856ce4ab3a03bcd03b94c
> git rev-list e5ef1491a856ce4ab3a03bcd03b94c18e
Parsing POMs
```