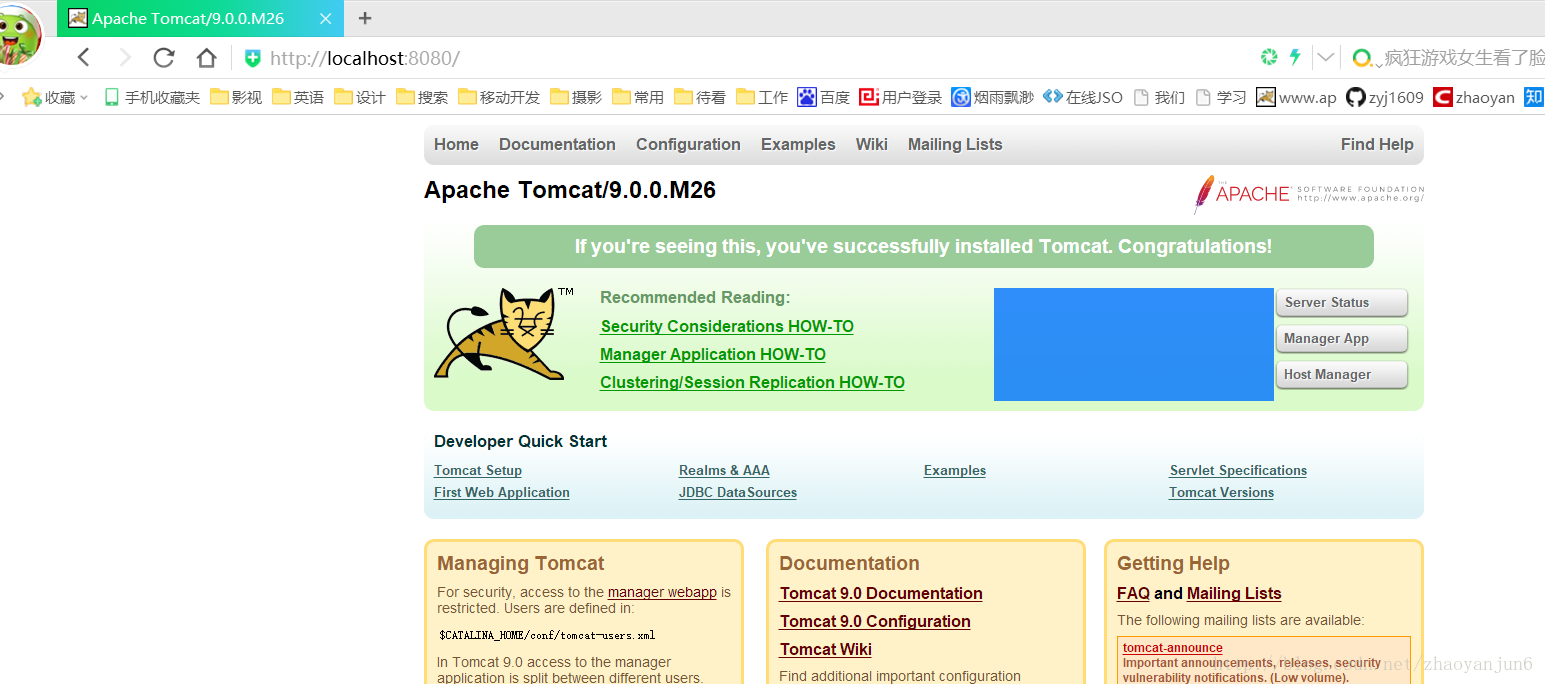
**Jenkins 自动打包（Android）**

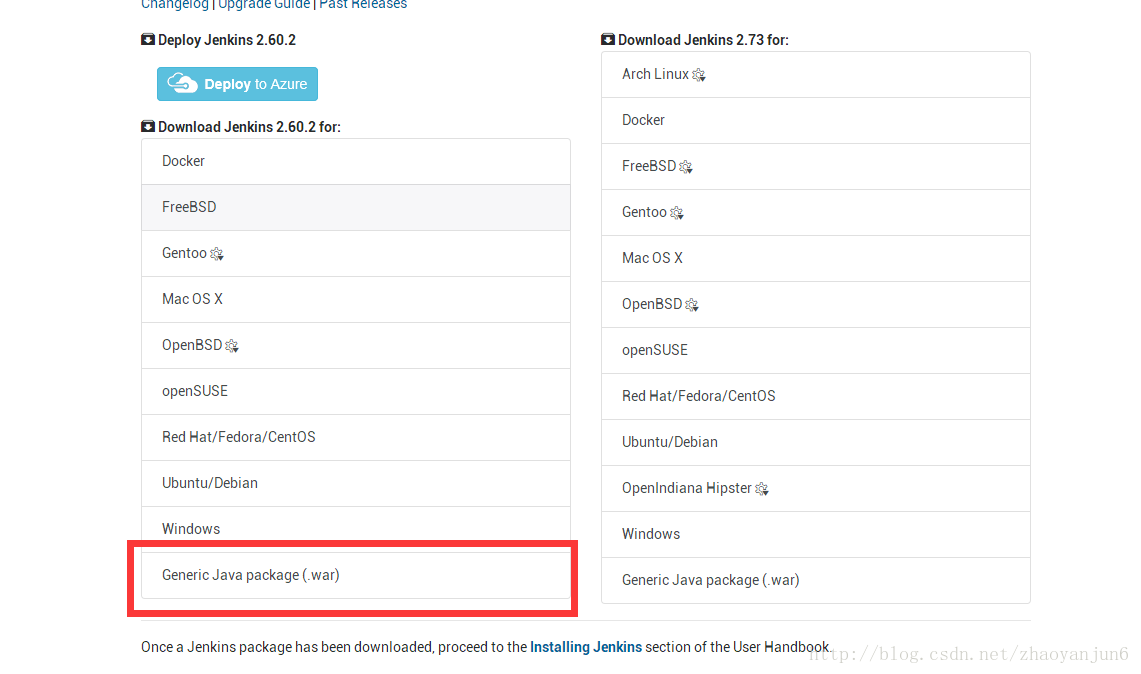
**1、Tomcat**

进入 <https://tomcat.apache.org/> 官网，下载最新的 tomcat 安装包并且安装。安装完成后，启动 tomcat 后，在浏览器输入 <http://localhost:8080/> ， 如果看到下面的界面，那么tomcat 就安装配置好了。

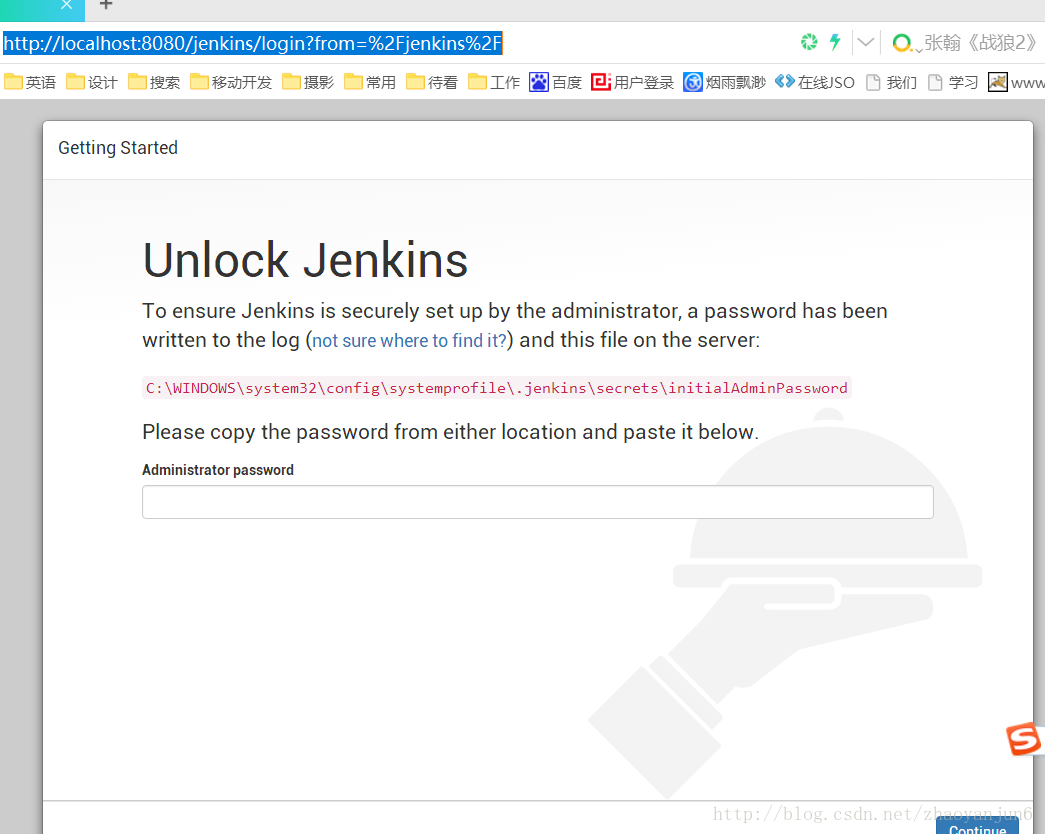


**2、Jenkins 部署到 Tomcat**

进入jenkins 的官网 <https://jenkins.io/> ， 根据你搭建平台的不同，下载 jenkins war 包。

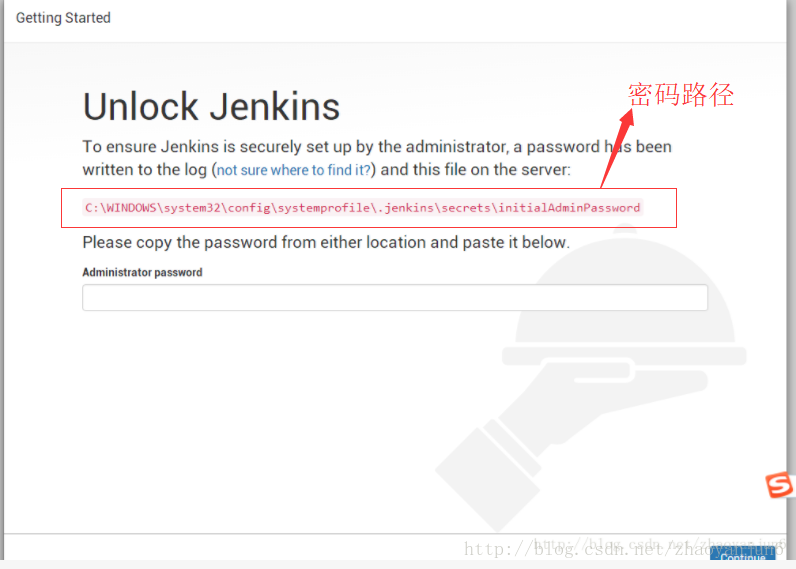


把 war 包放在 tomcat 的 webapps 目录下，然后在浏览器中访问 ： <http://localhost:8080/jenkins/> ， 你将会看到如下界面 , 代表 Jenkins 已经部署完成了，下一步安装 Jenkins.



**3、填写 Jenkins 超级管理员密码**

在上图的输入框内要输入超级管理员权限的密码，这个 **密码在红色路径的文件里** ，输入密码后，点击 continue , 如下图所示：

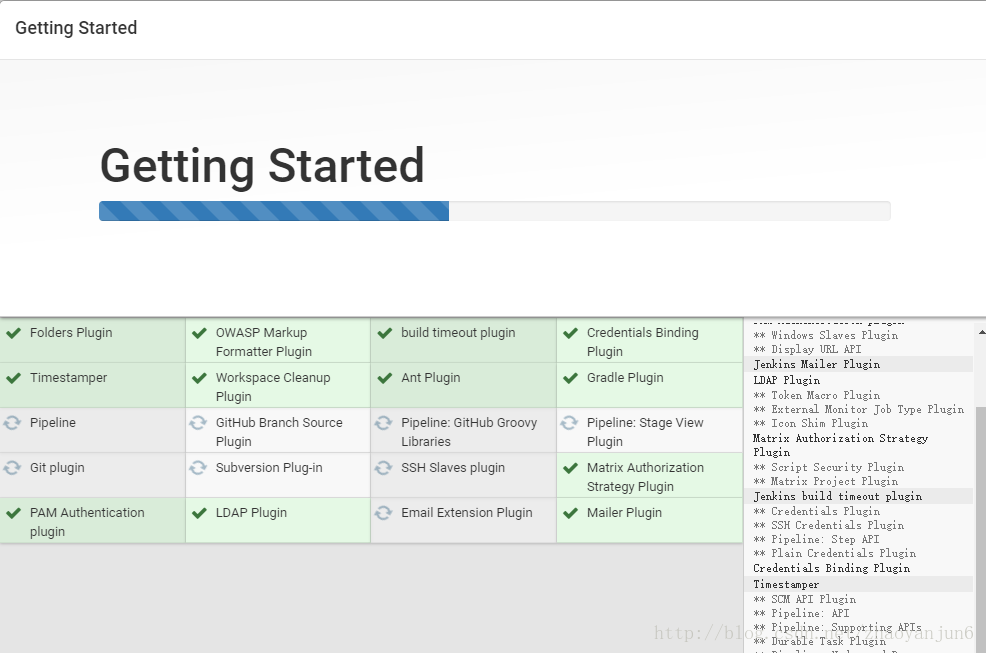


**4、Jenkins 插件安装和超级管理员配置**

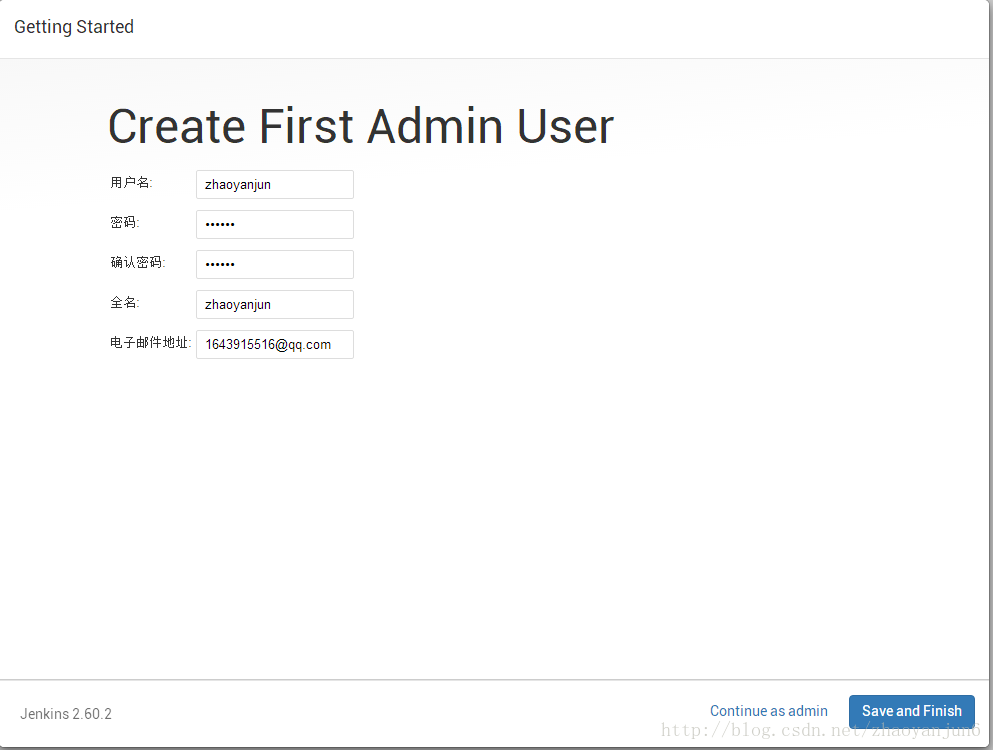
填写完密码后，将会看到插件安装界面



我们选择第一项，安装默认Jenkins插件。安装插件过程如下图所示：



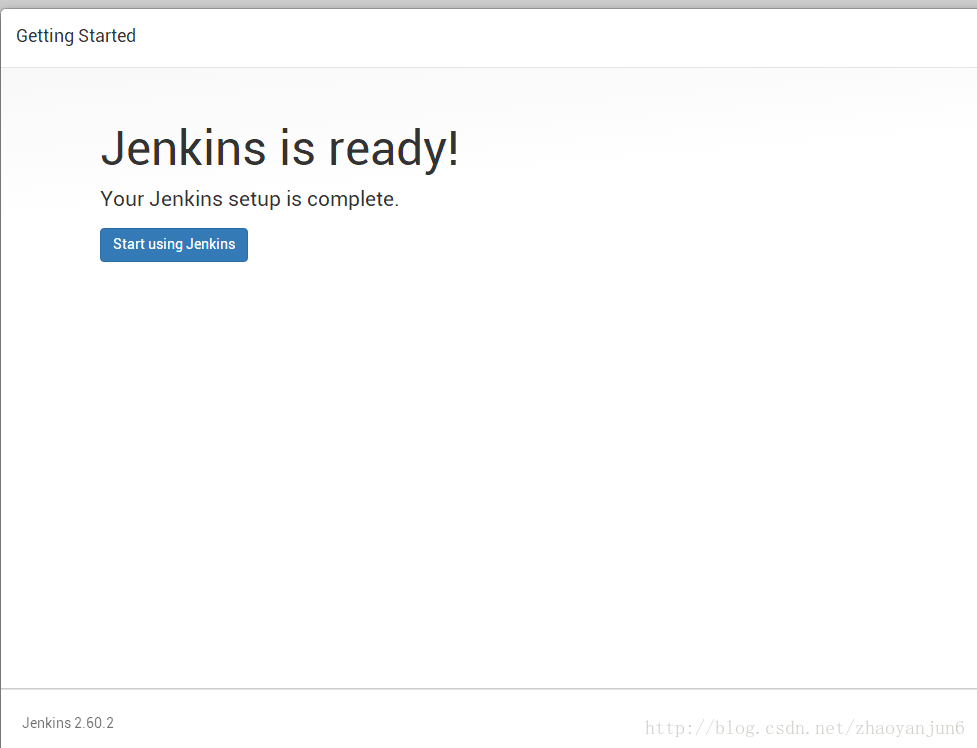
等所有的插件都安装完成，我们将会看到用户名设置界面。



这个界面你有两个选择可以操作，

* 选择一：直接点击 Continue as Admin 安装，默认此时的超级管理员为 admin .
* 选择二：输入用户名，密码，确认密码，全名，电子邮件地址。然后点击 Save and Finish . 注意你输入的用户名就是 Jenkins 的超级管理员。需要注意的是，上面的每一项都需要填写，否则点击 Save and Finish 按钮没有效果。

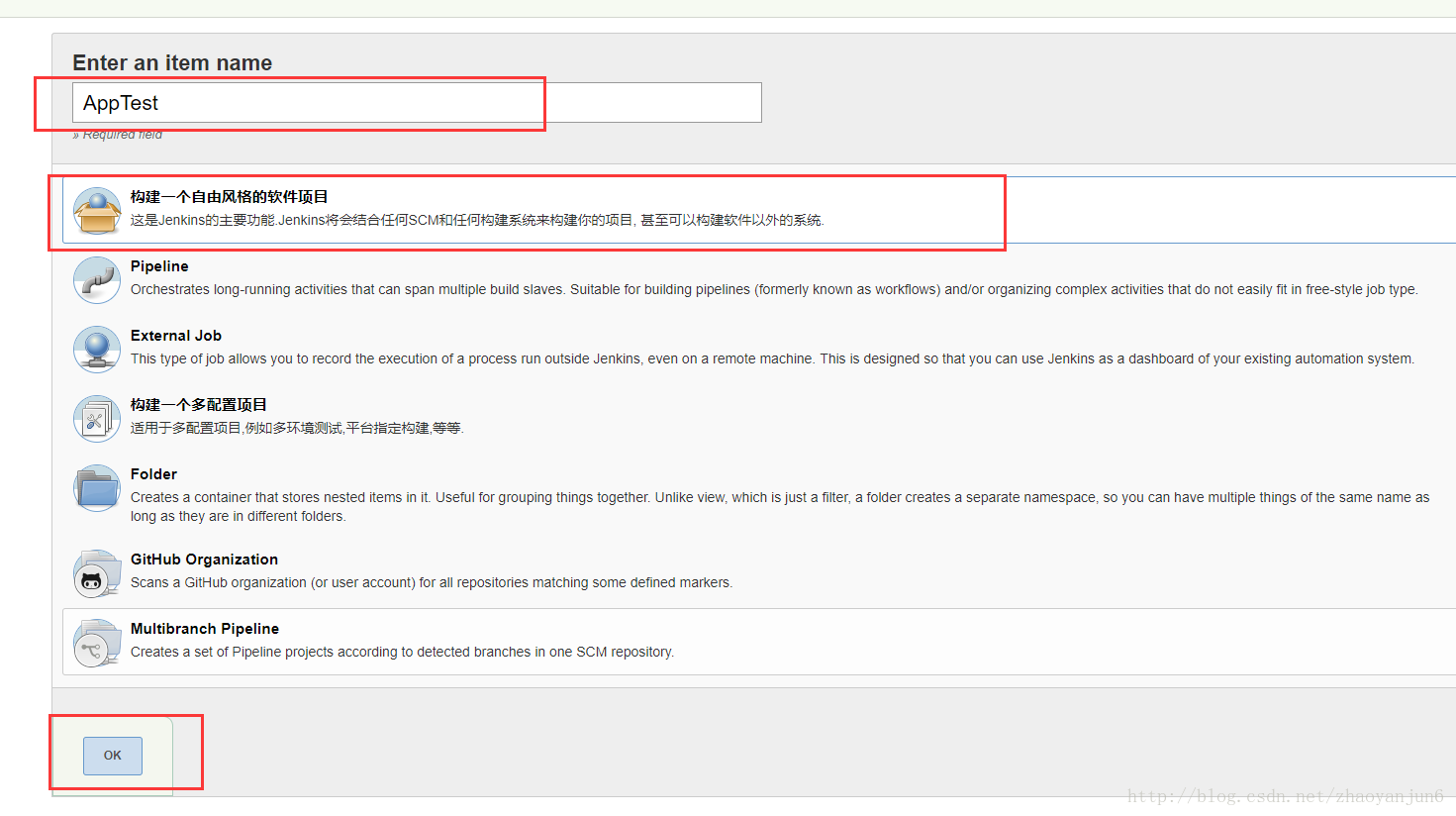
等上面的步骤都操作完毕，你将会看到如下界面，代表 Jenkins 的插件安装和超级管理员已经配置好了。



**5、创建任务**



点击创建一个新的任务，如下图所示：



**6、项目仓库配置**

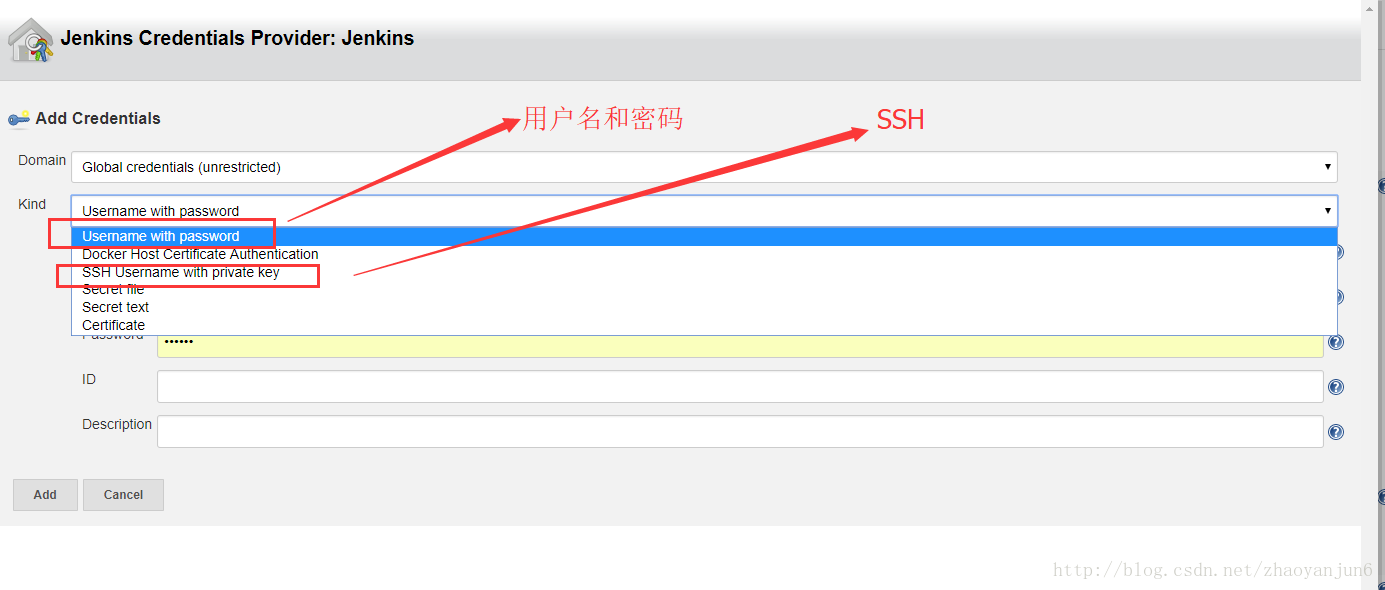


1、选择源码管理工具，Git 或者 Subversion . 下面我们的演示选择 Git.   
2、填写 Git 仓库地址 https://github.com/hu0514/test.git

3、选择 Git 证书凭证, 如果没有创建过证书，显示 none , 可以点击右边的 Add 按钮添加一个证书凭证。   
4、添加证书凭证，添加的方法详见下面的图。   
5、选择 Git 仓库的分支   
6、源码浏览器，Git 有很多管理器，我们这里选择自动就好了。

* 添加证书

从 Git 上拉取代码，有两种方式：用户名+密码、SSH . 根据自己的需要选择。



填写 Git 仓库的用户名和密码

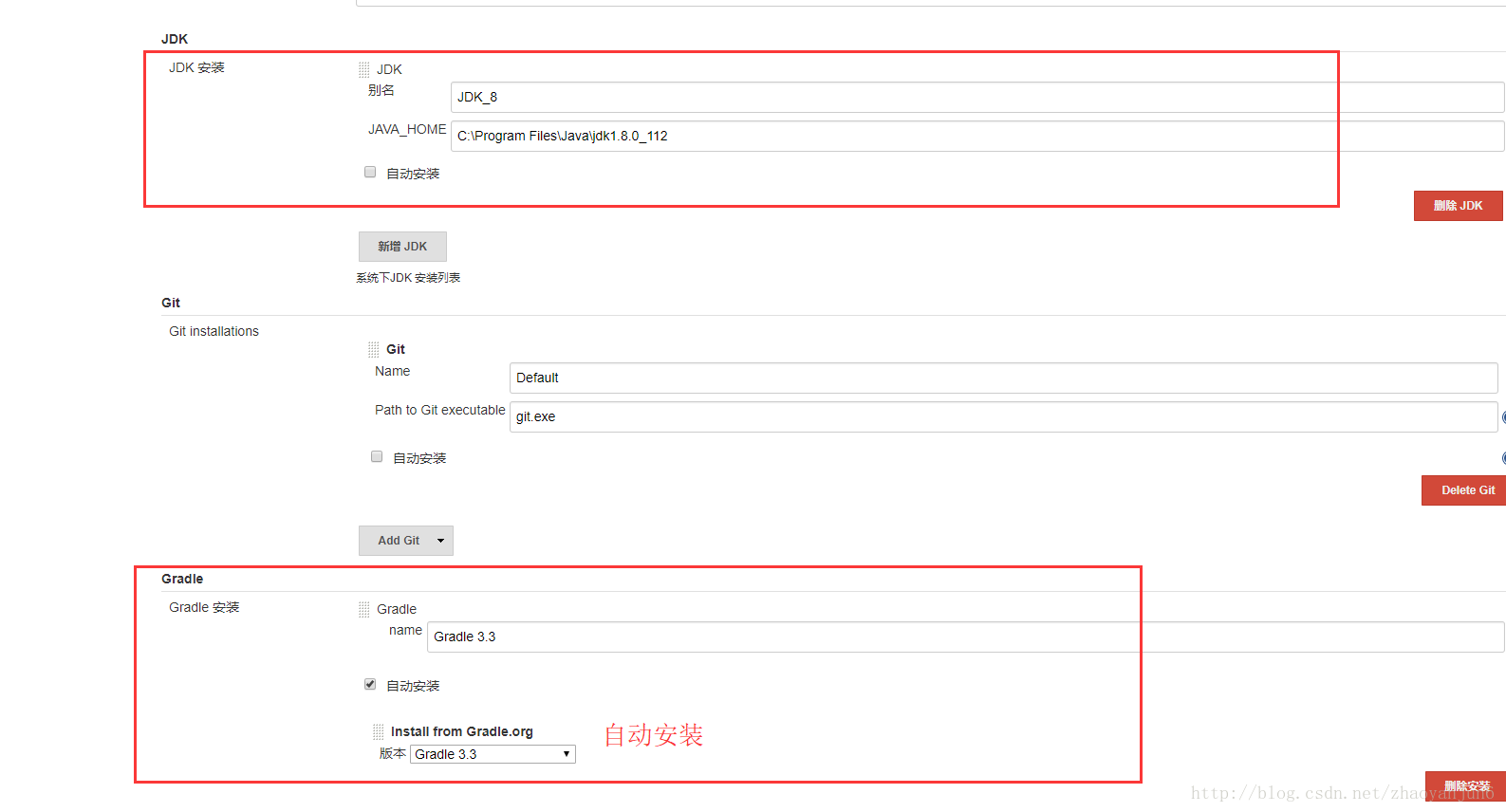


**7、Jenkins 全局配置**

**JDK 和 Gradle 配置**

由于Android 是基于 java 的，我们需要配置 JDK；由于我们用 Gradle 构建Android 项目，我们需要设置 Gradle 的环境 。

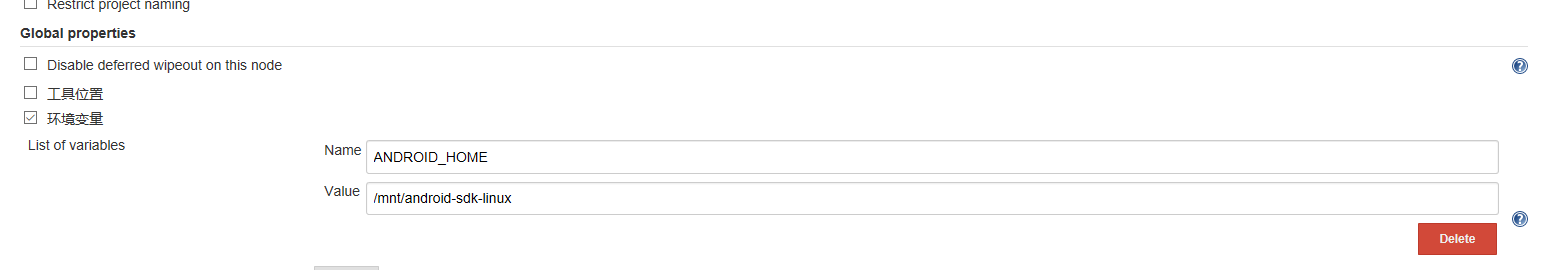
在 系统管理 –> Global Tool Configuration



**Android SDK 配置**

系统管理 –> 系统设置 – 全局属性

在 Environment variables 里面增加一个键值对.   
键：ANDROID\_HOME   
值：SDK 路径



Linux系统安装sdk方法：

下载 wget <http://dl.google.com/android/android-sdk_r24.4.1-linux.tgz>

解压 tar zxvf android-sdk\_r24.4.1-linux.tgz

设置环境变量

在/etc/profile文件最后添加

export ANDROID\_HOME=/usr/local/android-sdk-linux

export PATH=$ANDROID\_HOME/tools:$PATH

然后执行命令

Source /etc/profile

根据需要更新sdk插件

# 查看可以用的组件

android list sdk --all

# 通过filter指定下载的版本(filter接的数字，来自于list查找的工具序号)

android update sdk -u --all --filter 2,22,23,24,25,26

**8、创建 Jenkins 构建任务**



1、代表配置的项目   
2、构建模块   
3、选择 Gradle 构建工具的版本号，可以选择默认构建版本号，也可以选择自己自定的版本号。   
4、填写 Gradle 构建任务

* 打 release 包

1. app:clean
2. app:assembleRelease

* 1
* 2

解释：

1. app:clean : 清除....../app目录下的build文件夹
2. app:assembleRelease : 编译并打Release的包

* 1
* 2
* 打 debug 包

1. app:clean
2. app:assembleDebug

* 1
* 2

解释：

1. app:clean : 清除....../app目录下的build文件夹
2. app:assembleDebug: 编译并打Debug的包

* 1
* 2
* 同时打 debug 包和 release 包

1. app:clean
2. app:assembleDebug
3. app:assembleRelease

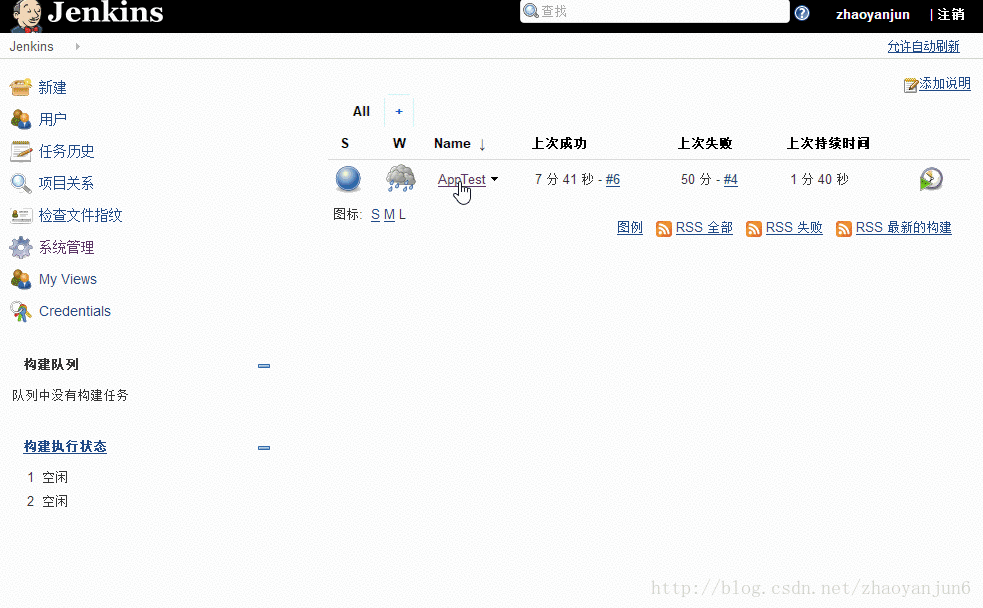
**9、执行构建任务**

点击立即构建按钮，开始执行构建任务。



1. - 【1】、点击立即构建，开始执行构建任务
2. - 【2】、进度条，表示正在构建的项目。
3. - 【3】、蓝色的图标表示构建成功
4. - 【4】、红色的图标表示构建失败

动图演示如下图所示：



点击正在构建的条目，可以进入正在构建的信息。



* 状态集：构建人用户名、构建git 节点
* 变更记录：
* Console Output：构建过程中的输出控制台，会输出构建过程中的详细信息，这个功能很重要，如果构建失败，可以从这里找到错误的原因。
* 编辑编译信息：每次构建任务的时候，构建名字是数字形式叠加，这里可以自定义构建任务的名字。
* Git Build Data：本次构建 Git 仓库的相关信息。

**10、构建结果**

在 Console Output 面板中，出现了 BUILD SUCCESSFUL , 代表任务构建成功。



**11、构建后操作**

**构建结果发布到 fir 平台**

fir 是国内提供测试应用极速发布,应用崩溃实时分析、用户反馈收集等一系列开发测试效率工具服务,帮助开发者将更多精力放在产品的开发与应用的优化上。

官网： <https://fir.im/>

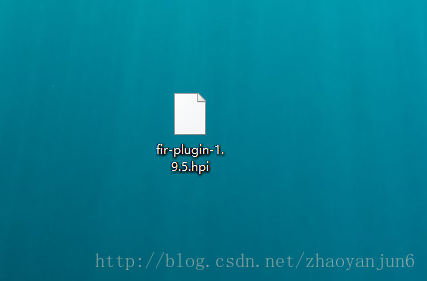
登录后，在 fir 官网的个人资料中， 可以找到 API Token , 这个 token 我们在后面会用到。



fir.im Jenkins 插件使用方法: <http://blog.fir.im/jenkins/>

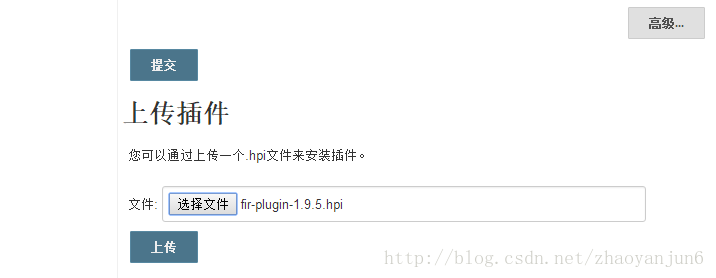
**下载插件**

Jenkins 构建完成后，生成的 apk 文件，我们通过插件的方式上传到 fir . 通过上面的连接，我们下载 Jenkins 的 fir 插件到本地。



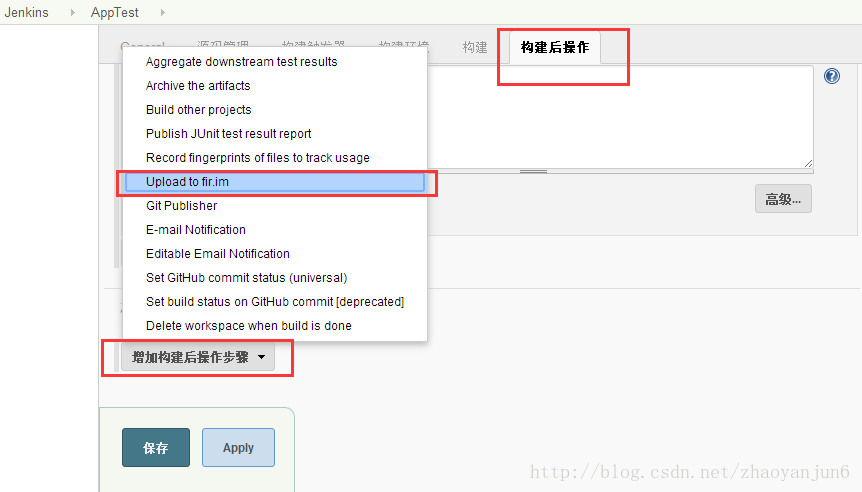
**安装插件**

系统管理 –> 插件管理 –> 高级 –> 上传插件

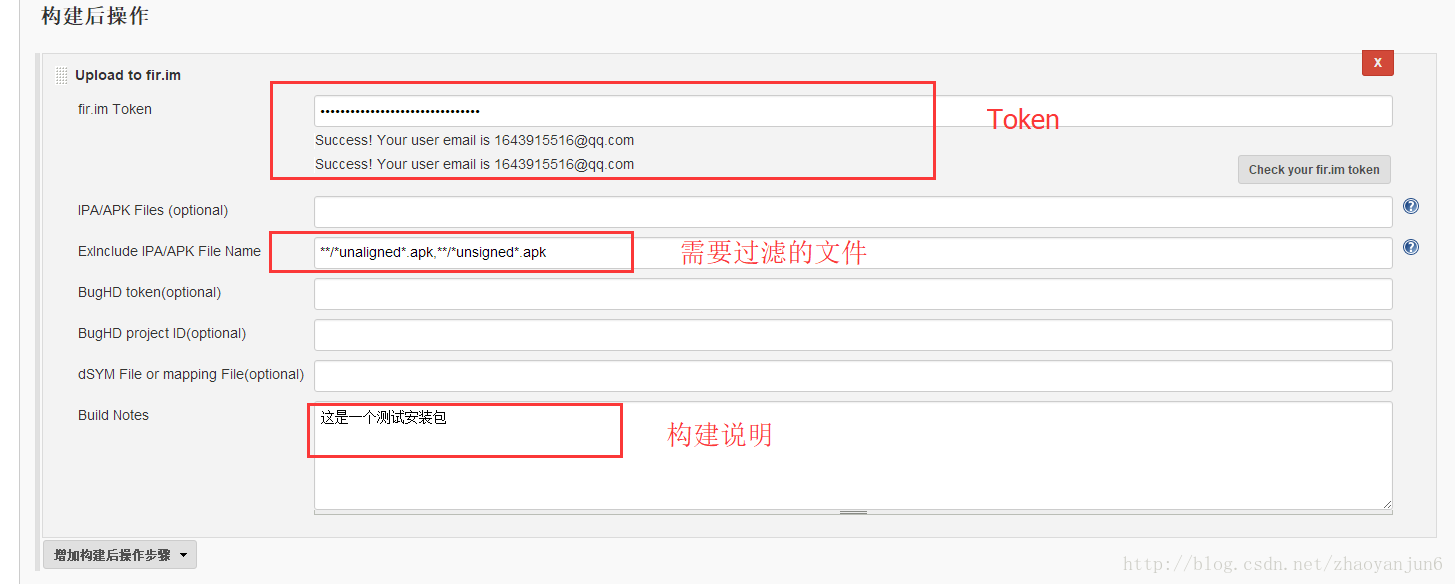


**使用插件**

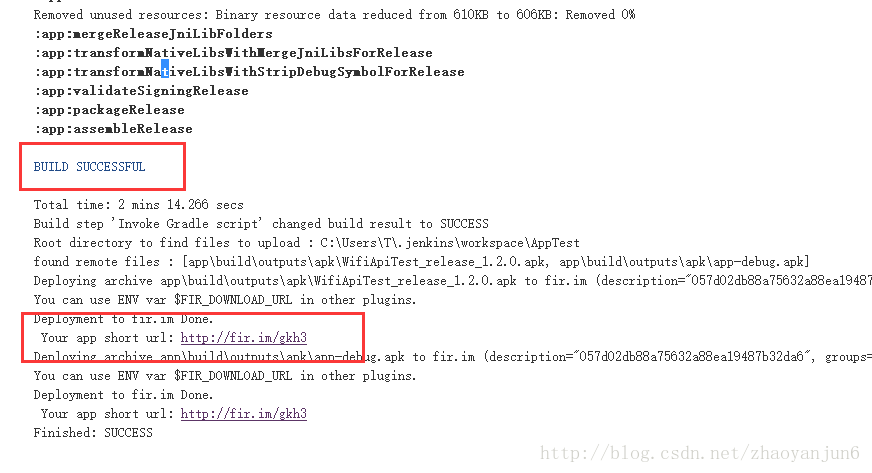
配置 –> 构建后操作 –> 增加构建后操作步骤 –> Upload to fir.im



填写 API Token



**构建结果**



通过日志看到上传 fir 成功。

**构建结果通知**[**钉钉**](https://www.baidu.com/s?wd=%E9%92%89%E9%92%89&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)

钉钉，[阿里巴巴](https://www.baidu.com/s?wd=%E9%98%BF%E9%87%8C%E5%B7%B4%E5%B7%B4&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)出品，专为中国企业打造的免费智能[移动办公平台](https://www.baidu.com/s?wd=%E7%A7%BB%E5%8A%A8%E5%8A%9E%E5%85%AC%E5%B9%B3%E5%8F%B0&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)。

* **第一步：创建钉钉群聊机器人**



选择自定义机器人



点击添加

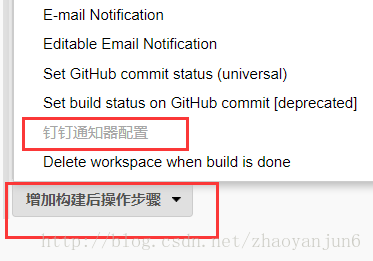


输入机器人的名字

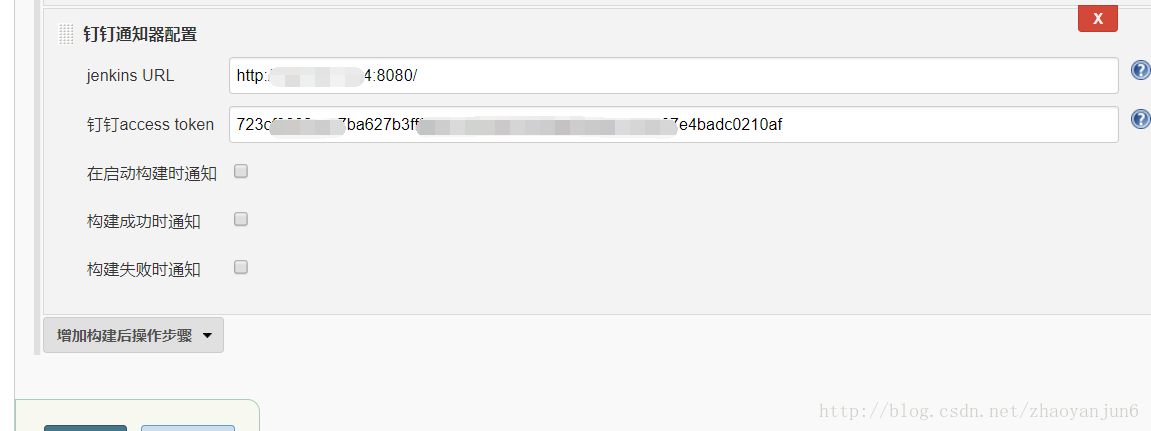


复制 Hook 地址

* **第二步：Jenkins 后台添加钉钉插件**
* **第三步：添加构建后操作**



* **第四步：添加钉钉插件参数**



**总结**

到这里关于 Jenkins 的自动化打包的主要内容已经介绍完了。下面将会介绍一些附加的操作，能够更好的使用 Jenkins .

**12、构建触发器**

在上面部分我们发起构建的命令是点击 **立即构建** 的按钮，Jenkins 里面有构建触发器的方式，提供远程构建任务。



* 1、触发远程构建 (例如,使用脚本)：填入身份验证令牌，可以随便填写，我们这里填入 123456 ， 填写完成后，在输入框的下方有提醒

1. Use the following URL to trigger build remotely: JENKINS\_URL/job/AppTest/build?token=TOKEN\_NAME 或者 /buildWithParameters?token=TOKEN\_NAME
2. Optionally append &cause=Cause+Text to provide text that will be included in the recorded build cause.

* 1
* 2

根据提示，我们自浏览器中访问地址： http://127.0.0.1:9999/jenkins/job/AppTest/build?token=123456 ， 这样就可以触发构建任务。

* 2、Build after other projects are built ： 在其他任务结束后，开始本次构建任务。

示例：在 AppTest1 项目构建成功后，开始本次构建任务。



①Trigger only if build is stable：其他项目构建成功   
②Trigger even if the build is unstable：其他项目不稳定   
③Trigger even if the build fails：其他项目构建失败

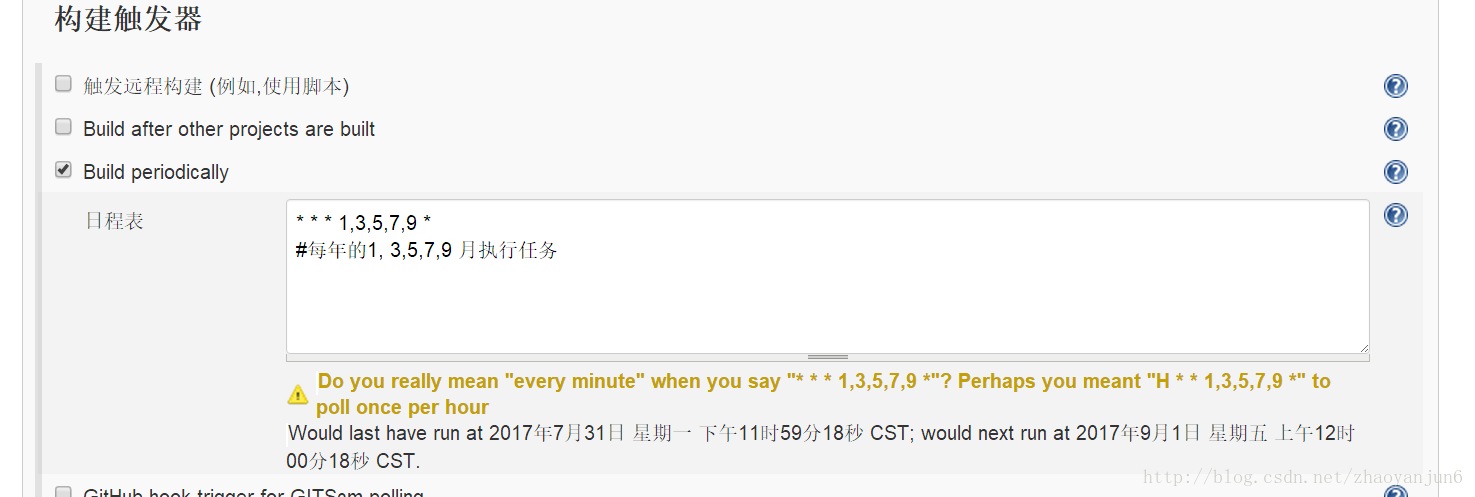
* Build periodically：周期进行构建（它不关心源码是否发生变化）

日程表：

1. H 2 \* \* \* *//每天凌晨2天构建*
3. H 16 \* \* \* *//每天下午4点构建,16 代表下午4点*
5. 0 16 15 \* \* *//每个月15号下午4点构建，15代表天，16代表下午4点，0代表0分钟*
7. H 6,12,18 \* \* \* *//每天的 6时，12时，下午6时构建*
9. 15,30,45 \* \* \* \* *//每个小时 15分，30分，45分 发起构建*
11. \* \* \* 1,3,5,7,9 \* *//每年的1, 3,5,7,9 月发起构建*

* 1

示例：



**注意**

1、在日程表中，# 可以代表注释，但是需要单独写一行。

2、官方日程表语法：<https://en.wikipedia.org/wiki/Cron#CRON_expression>

3、日程表的语法字段遵循cron的语法（略有不同）。 具体来说，每行由5个由TAB或空格分隔的字段组成：MINUTE HOUR DAY MONTH WEEK

4、时间顺序和: MINUTE (0-59), HOUR (0-23), DAY (1-31), MONTH (1-12), DAY OF THE WEEK (0-7)

其他的日程表：

1. 30 08 \* \* 1-6 *//周一至周六 早上08:30开始构建，最后一个字段是 WEEK ，1-6 代表周一到周五，0 代表周日*
3. 00 0,12 \* \* 0-5 *//周日到周五，每天建成两次，中午12:00和午夜00:00*
5. H 16 \* \* 1-5 *//工作日 下午4点左右开始每天开始建设,构建时间 16:00 -16：59，具体的时间取决于项目哈希：*

* 1
* GitHub hook trigger for GITScm polling：hookplugin 检测到源码的 push 操作触发构建 , 感觉Poll SCM 更方便些,如果提交频繁,则这个触发就会频繁,看业务需要设置。
* Poll SCM ： 定时检查源码变更（根据SCM软件的版本号），如果有更新就 checkout 最新code下来，然后执行构建动作。

日程表示例：

1. H/5 \* \* \* \* *//每5分钟检查一次*
3. H/60 \* \* \* \* *//每1个小时检查一次*
5. H H/2 \* \* \* *//每2个小时检查一次*
7. H \* H/10 \* \* *//每10天检查一次*
9. H \* \* H/3 \* *//每3个月检查一次*
11. H \* \* \* 1 *//每一周检查一次*

* 1
* 2

示例：

