## 基于Jenkins2.0 Pipeline的持续交付实践

**一、Jenkins2.0 介绍**

Jenkins是世界上使用最广泛的持续集成平台之一，它被应用于代码检测、单元测试、环境部署等研发各个环节，相信大多数工程师对它都不会陌生。

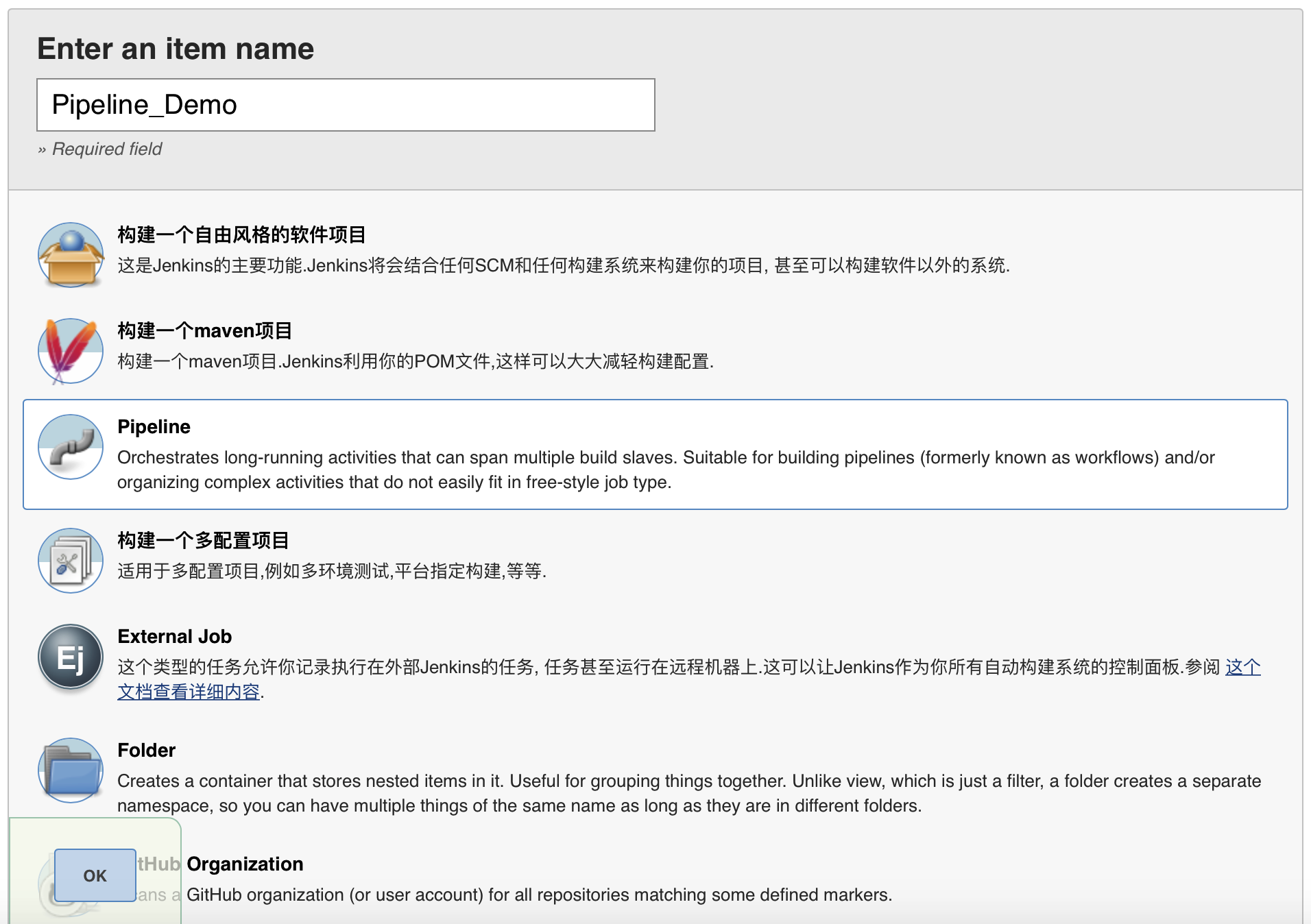
2016年4月20日，包含了全新的Pipeline功能的Jenkins2.0正式上线。Jenkins2.0跟1.0有什么不同呢？

Kawaguchi（Jenkins的创始人）在一份阐述Jenkins2.0的愿景的presentation（<https://docs.google.com/presentation/d/12ikbbQoMvus_l_q23BxXhYXnW9S5zsVNwIKZ9N8udg4/edit#slide=id.p>）中提出了2.0的三个目标：

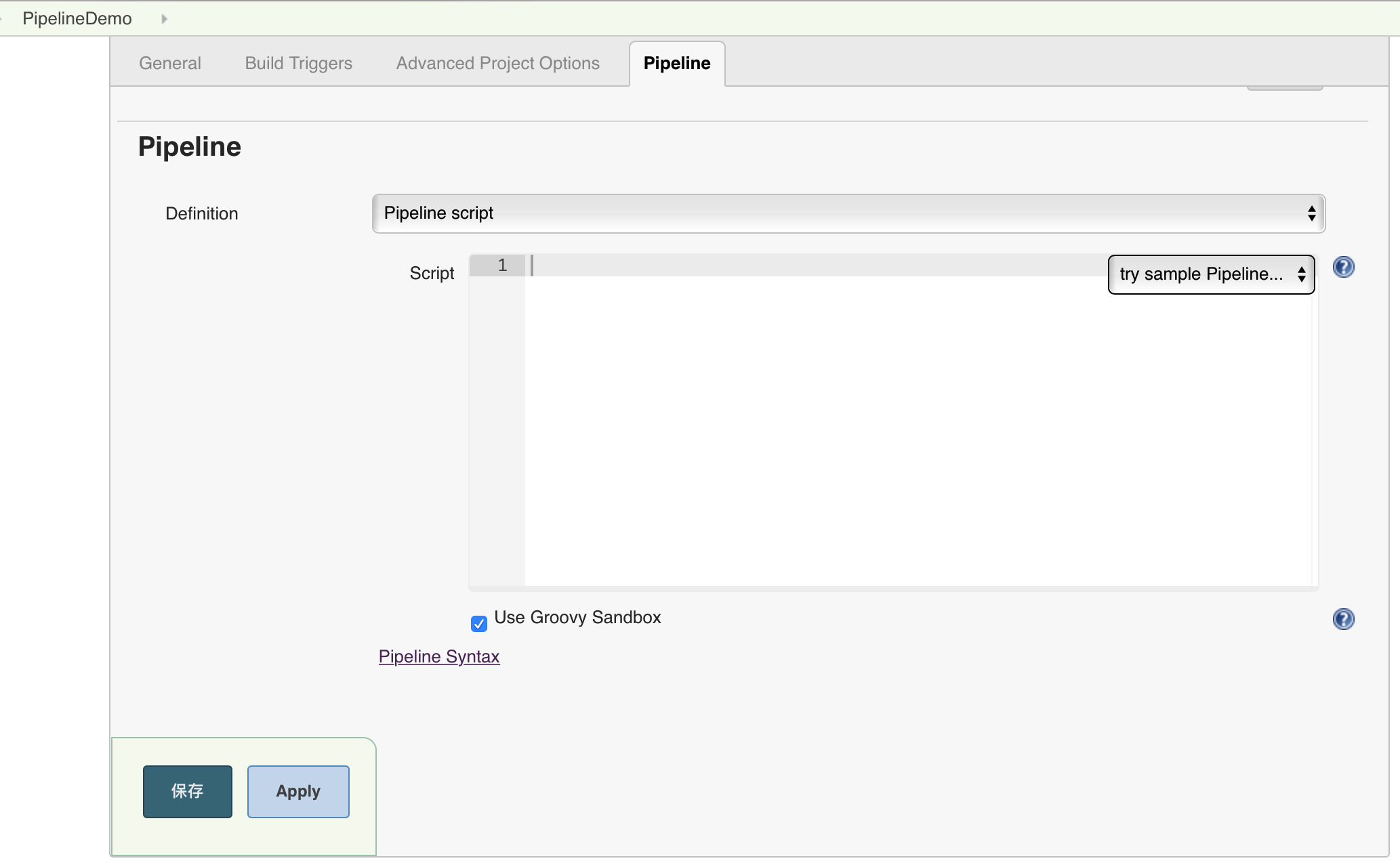
1. 2.0需要支持CD（Continuous delivery 持续交付）
2. 2.0需要给社区带来新能量
3. 加快完成一些高风险、高收益的feature

而2.0的核心就是名为'Pipeline'的Workflow（工作流），以及支持CD的全新界面。 下面我们就来看看2.0的Pipeline到底有何不同。

打开2.0版本的Jenkins，点击新建，选择Pipeline后点击OK，即可进入Pipeline的配置界面。



以下就是Pipeline的配置界面，与之前版本最大的不同，是去掉了大部分界面配置，转而使用了一个名为Pipeline的脚本配置栏。

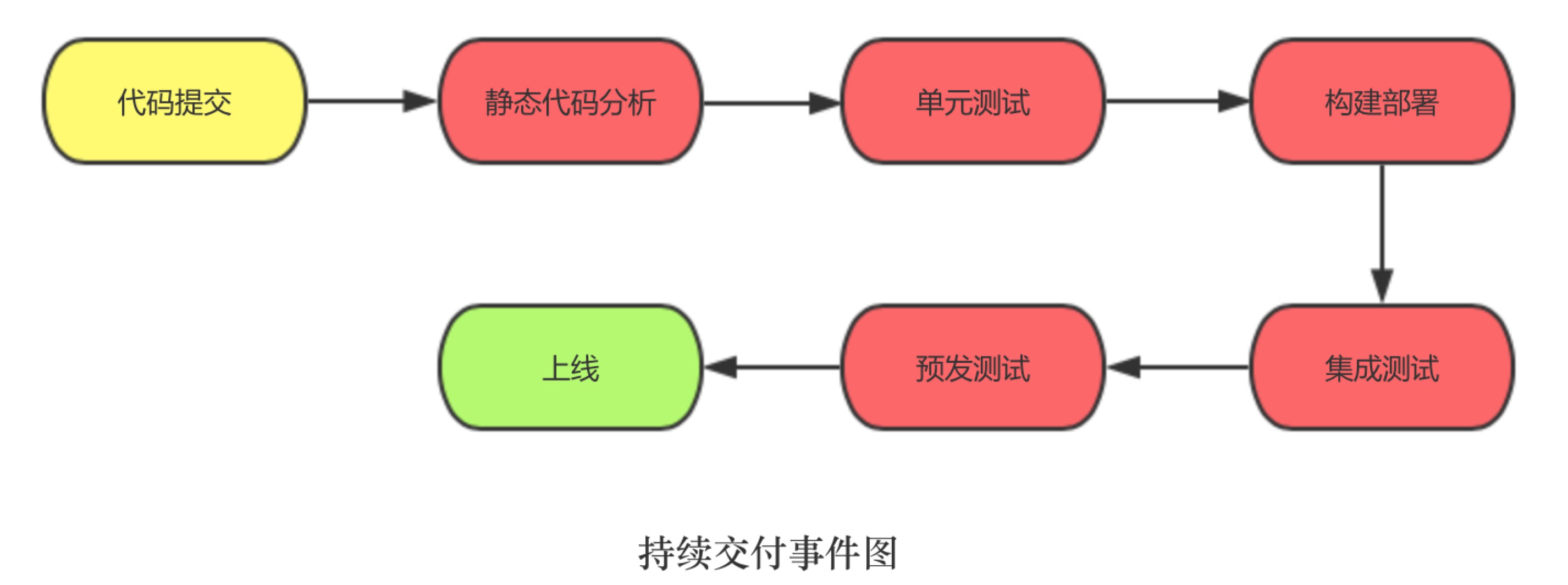


在这里，我们看到有两种使用Pipeline脚本的方式：Pipeline script 和 Pipeline script from SCM。前者可以直接在界面上编写Pipeline代码，后者可以通过读取代码库中的Pipeline脚本文件间接获取配置。以下主要以Pipeline script为例子来讲述Pipeline的使用。

**二、什么是Pipeline**

Pipeline的原意是“管道”，在Jenkins2.0中，Pipeline是一套工作流框架，将原本独立运行于单个或者多个节点的任务连接起来，从而实现从代码开发、测试，到部署，上线的持续交付流程。

我们先来看一个基础的持续交付模型包含了哪些事件：



持续交付的过程，就是把单一的事件任务关联起来，形成一条管道。当前置任务成功时，会自动进入下一任务的执行，否则，此任务做失败处理。当所有任务依次执行成功时，代码会发布到线上，开始提供线上服务。这个过程非常类似相互连接的管道，所以不难理解，业界喜欢用Pipeline这个词来形容持续交付的过程。

Jenkins2.0提供了一套Pipeline框架，用来描述、执行这个持续交付过程（其他支持持续交付功能的平台同样会定义一套自己的Pipeline框架，其基本原理、概念都是一致的）。

当然，在现实项目中，会有更多、更复杂的因素影响持续交付的过程，对于每一个事件本身，也有不同的实现版本，甚至事件之间的顺序也并非一成不变——但这不妨碍我们使用Jenkins2.0的Pipeline框架来帮助我们提高生产效能。

**三、Pipeline框架介绍**

1. **基础概念**

在Jenkins2.0中，我们使用的是一种名为Scripted Pipeline的语法，用来描述Pipeline框架。这种语法支持Groovy，意味着我们可以在其中直接使用Groovy本身的功能。

Scripted Pipeline包含了几个基础概念：

1. node. 指定运行的节点。通过node指定在哪个节点上运行。
2. stage. 阶段，我们可以把持续交付中所有的执行命令归类到一个或者多个stage，比如，我们可以把持续交付事件图中的每一个事件当做一个stage。
3. step. 就是一系列具体的执行命令，比如构建命令，单测命令等。

举一个简单的例子，来看看node，stage，step是如何关联起来的：



我们在名为master的node节点上，执行了两个stage：'checkout' 和 'unit test’，stage里面包含了一些step。根据命名，我们可以判断出这两个stage分别完成了下载代码，和运行单元测试的任务。我们可以通过不同的stage，来完成持续交付的整体流程。

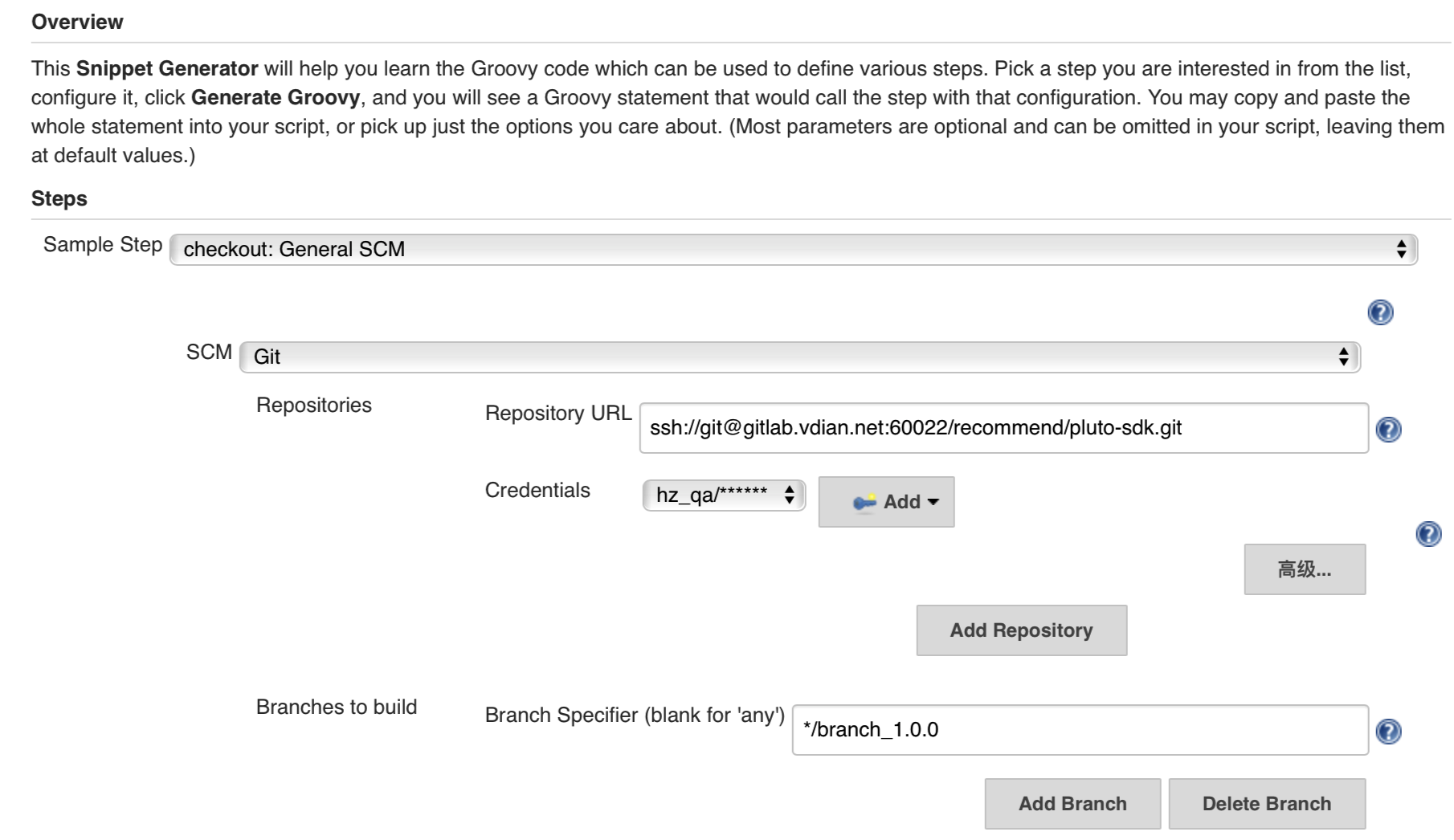
1. **功能强大的step**

当我们定义好node和stage之后，我们需要做的就是把命令通过step的形式放到合适的stage中。我们可能会对上面示例中checkout和sh等关键字感到陌生，这里就讲讲如何使用这些关键字。

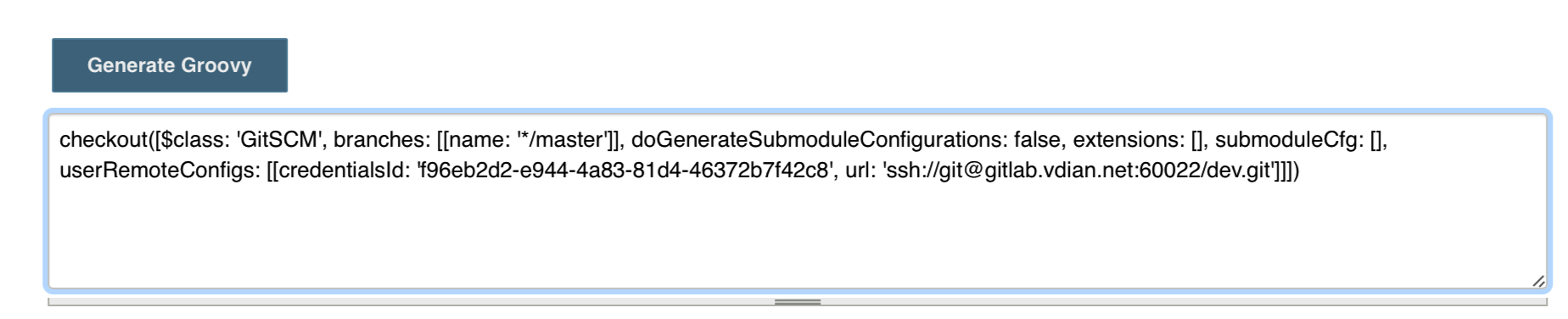
checkout后面跟着的一连串的参数很容易令人迷惑，好在Jenkins给我们提供了辅助工具帮我们简化配置。找到Pipeline script一栏，点击左下角的Pipeline Syntax。



这时我们会进入Snippet Generator的页面，在Sample Step中选择checkout，根据界面的选项填入相应的参数。



填完之后点击Generate Groovy，即可生成对应的step。



我们需要做的就是把这部分代码填入对应的stage中。

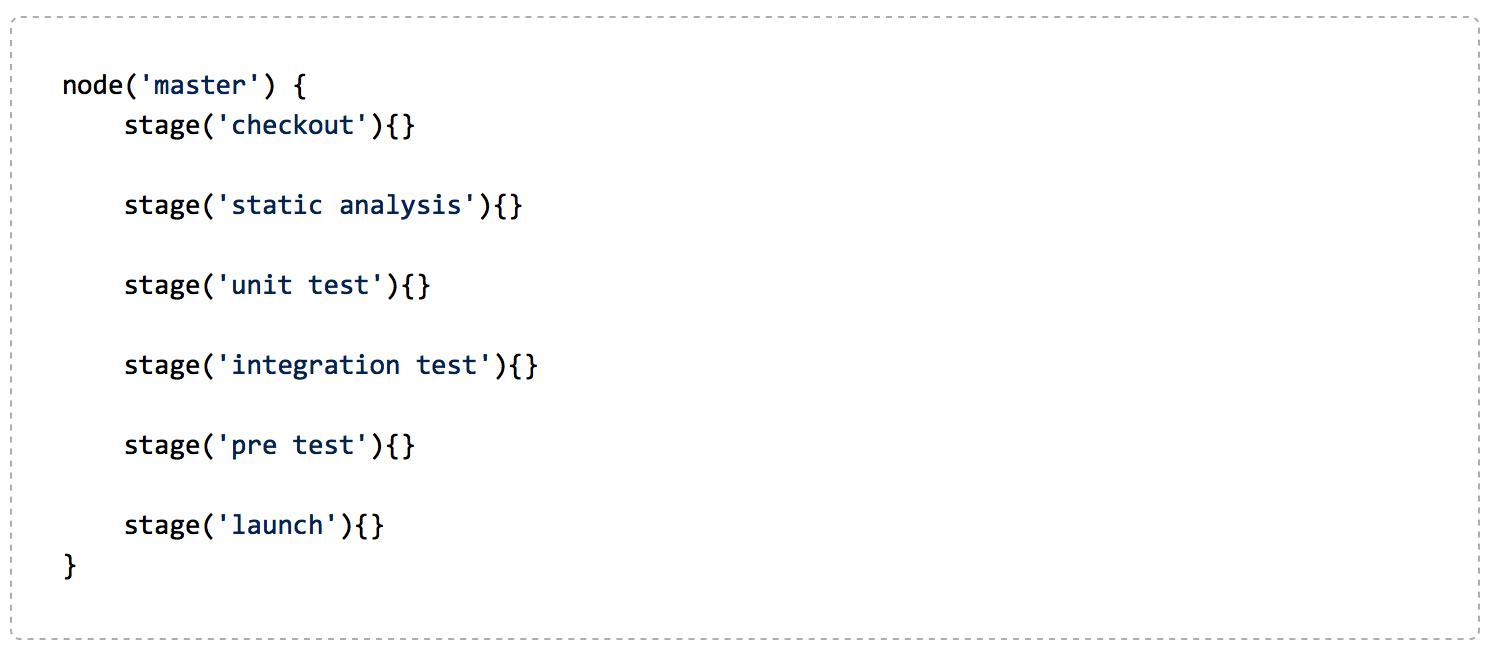
除了比较复杂的checkout操作，其他类型的step并不都是这么复杂，比如：



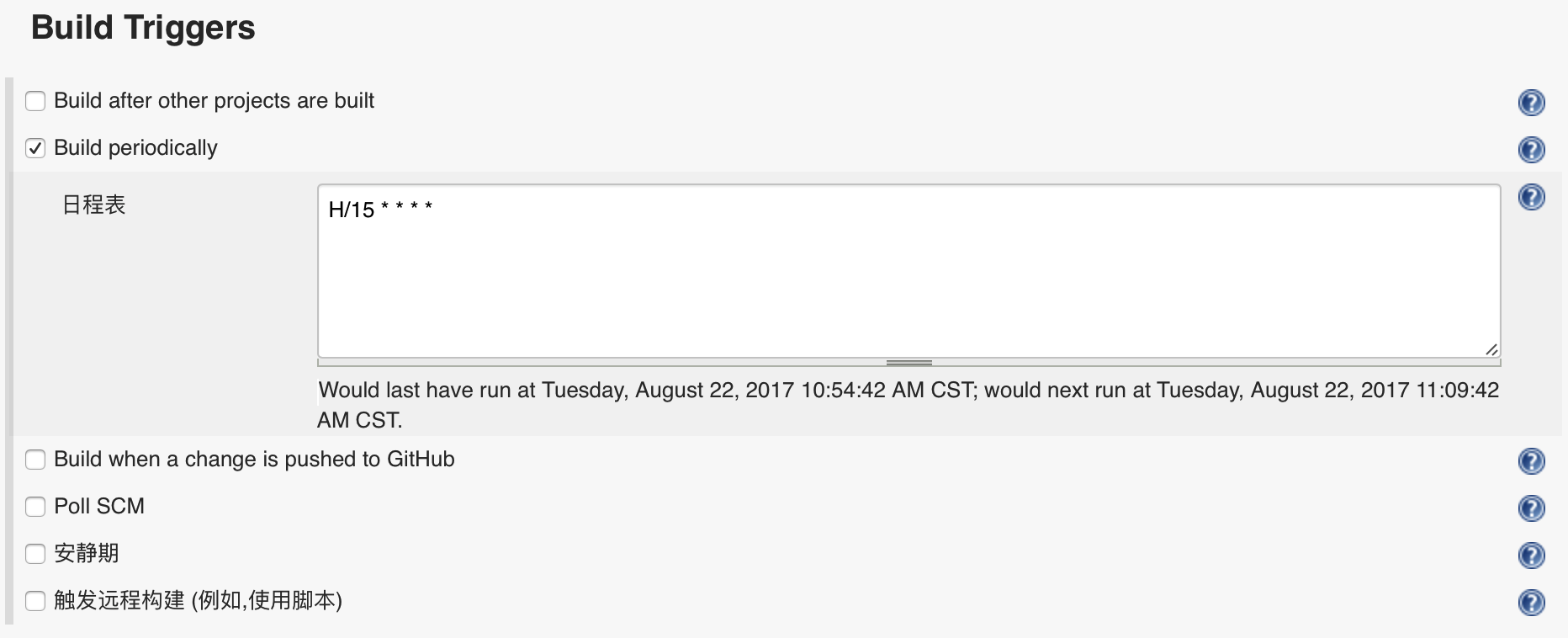
同理，其他step也可以通过Generator页面生成。

**四、应用Pipeline模型完成持续交付**

首先，我们通过stage来定义出完整的持续交付流程。根据之前的持续交付事件图，我们可以定义如下的stage：



我们通过Jenkins配置页面的Build Triggers配合checkout的stage来实现”代码提交”监控的功能：



这里要求每隔15分钟执行一次任务，也可以根据实际需求定义其他trigger。

接下来，我们需要编写相应的step，填入对应的stage中。在我们的项目中，我们使用JUnit、Groovy等语言作为测试代码，使用mvn作为项目生命周期管理工具。

对于mvn相关的命令，只需要改写成sh ‘mvn test -Dtest=my\_test’即可。通过echo，sleep，retry等关键字，可以用来辅助mvn命令的执行。

在具体的实践中，我们碰到了一些问题：

1) 由于开发代码和测试代码不在同个git repo下，所以需要在checkout的stage下载两份代码。这个时候，我们只需要写两个checkout命令。



2) 在静态代码扫描中，我们使用了PMD作为扫描工具，此时需要配置PMD结果解析的step。幸运的是，Snippet Generator提供了PMD结果文件解析step的写法。



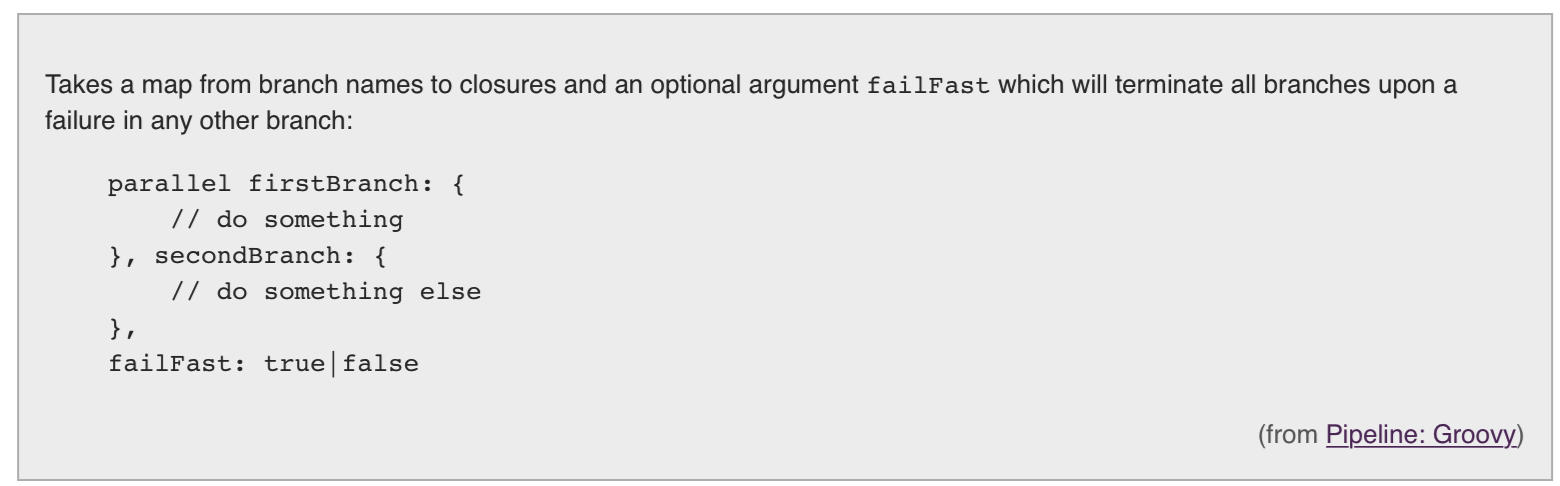
将生成的Groovy语句写到静态代码扫描stage中即可。

同理，对于JUnit结果文件的解析，我们同样适用了junit关键字：



需要注意的是，由于我们有集成测试、预发测试两个stage，且都在同一个代码目录下运行，我们必须保证两次测试的结果文件不同名，否则会因为被覆盖而导致结果不准确。

3) 没有并行执行功能的Pipeline绝不是一个好的Pipeline，Jenkins2.0提供了parallel来实现并行功能。



比如，我们在checkout的stage中下载了两份代码，这两次操作我们可以通过parallel并行执行：



这是属于step的并行。

更进一步，我们可以把stage也并行执行：

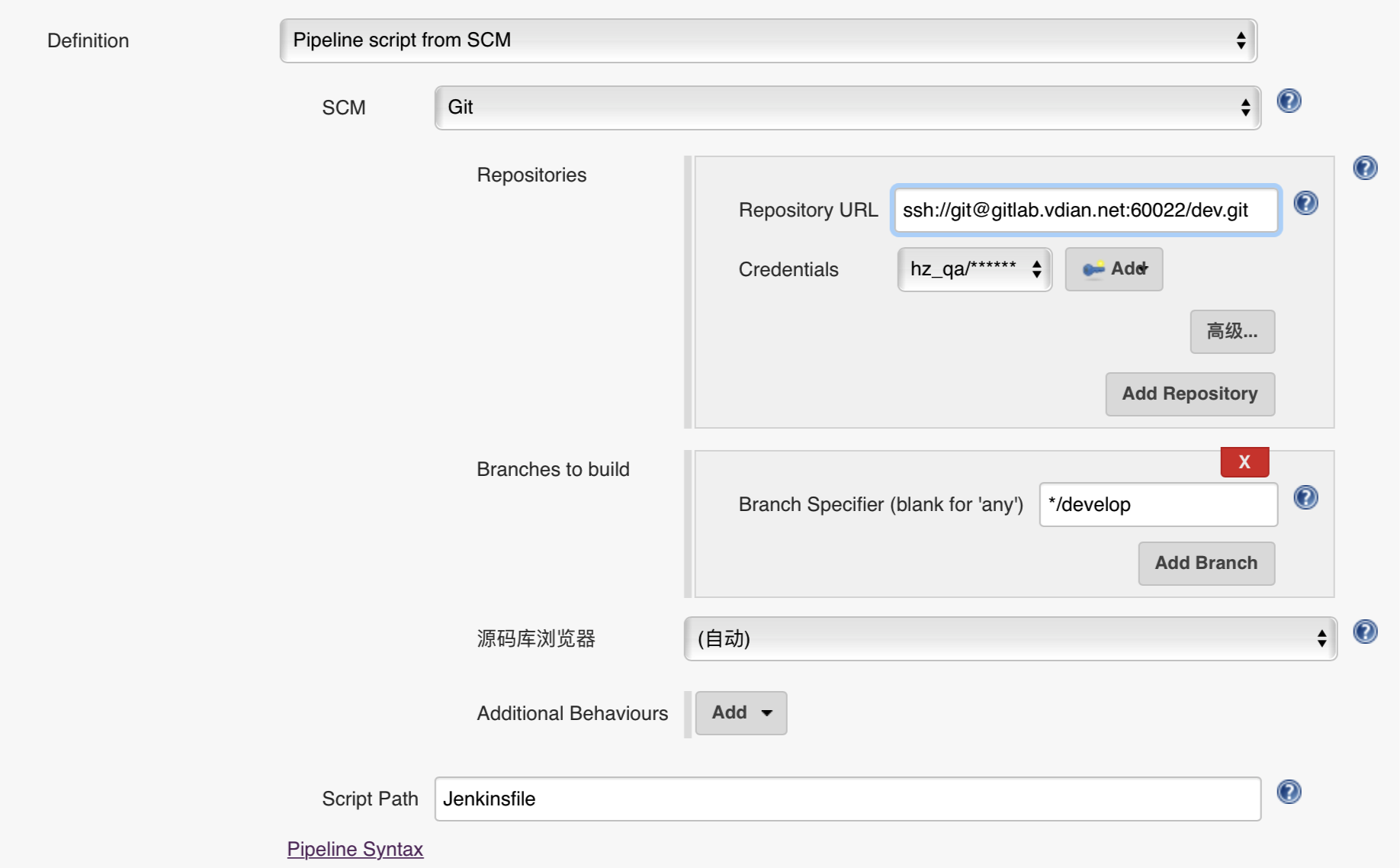


通过将integration test和pre test这两个stage包含在parallel中，实现了两个stage的并行执行，最理想的状况下可以节省50%的时间。

4) 通过Pipeline script from SCM的方式使用Pipeline

我们可以直接在Jenkins配置页面编写Pipeline，也可以以版本控制的形式将Pipeline script直接保存在代码库里。假设我们将刚才在配置页面写入的命令保存为Jenkinsfile，存放在<ssh://git@gitlab.vdian.net:60022/dev.git>的主目录下。

我们可以用Pipeline script from SCM的方式进行配置：



最终的效果跟在配置页面直接写入命令并没有区别。好处是我们可以通过版本控制的方式将Jenkinsfile与代码绑定。

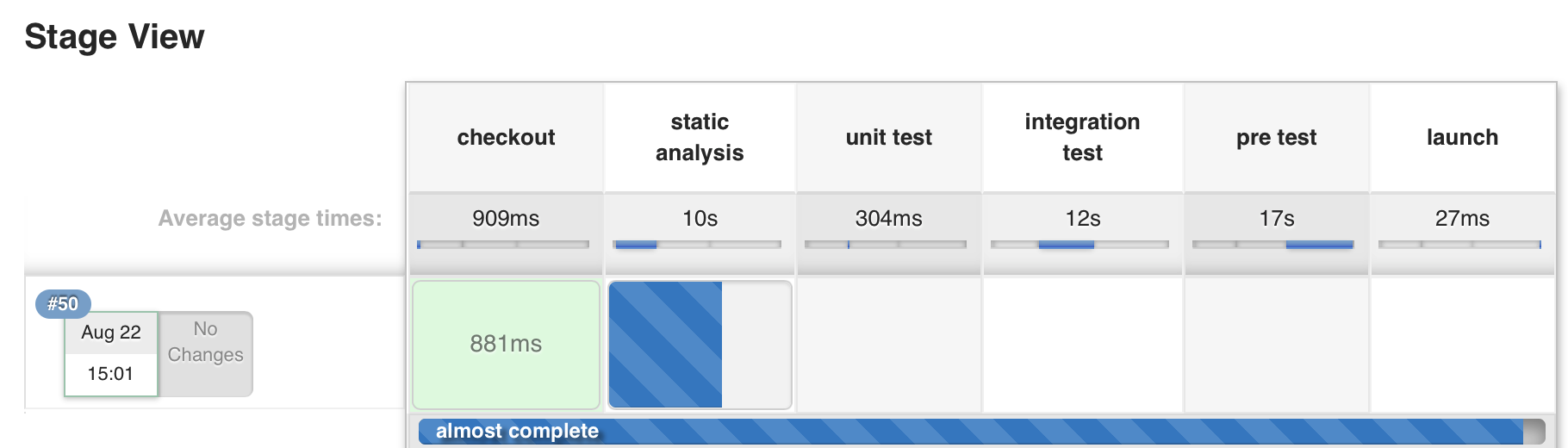
**五、最终效果**

我们将所有Pipeline代码整合到一起：

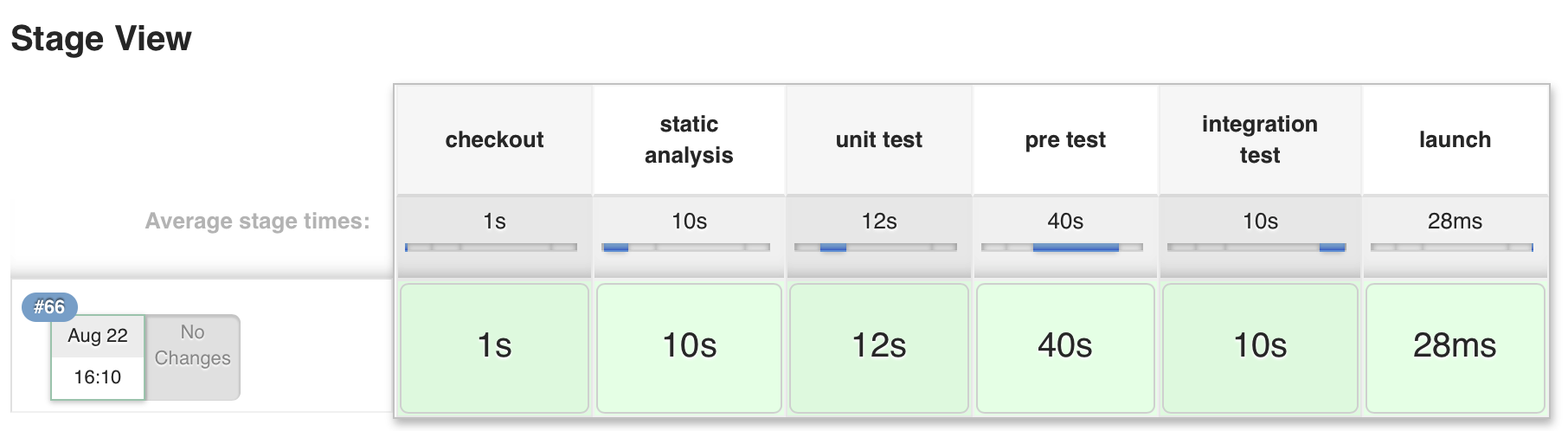


launch的stage我们仅仅用echo做一个示意，真实的持续交付在发布这一环节需要考虑的因素会更复杂。

执行Pipeline任务，任务主页会出现每个stage依次执行的图形界面：



如果所有stage均执行完成，则出现：



这样，一个基础的持续交付Pipeline模型就在Jenkins上搭建起来了。我们可以在此基础上不断补充完善step和stage，继续优化整体流程。

**六、结束语**

Jenkins2.0全新的Pipeline功能相比较之前Jenkins1.0，在持续交付步骤设计、命令编写和界面展示方面有了很大的变化，令人耳目一新，非常值得深入挖掘。

关于作者：金斌，微店技术专家，曾就职于阿里巴巴云测中心，担任阿里集团持续集成平台核心开发，关注领域测试开发、持续交付、DSL设计、小程序产品研发等。

文章署名：金斌

联系地址：浙江省杭州市西湖区蒋村街道 紫霞街西溪谷商务中心g座3楼微店

本人银行账号：6226095710670709

开户行：杭州分行钱塘支行

身份证号码：331082198703060033

Email：bin.jinb@qq.com

电话： 18658863627