目录

[创建 1](#_Toc168665657)

[建造 2](#_Toc168665658)

[测试 2](#_Toc168665659)

[部署 3](#_Toc168665660)

[部署三阶段 4](#_Toc168665661)

[使用环境变量 4](#_Toc168665662)

[设置环境变量 5](#_Toc168665663)

### 创建



Step 一个简单的执行步骤，比如执行一个sh脚本

Stage 将你的命令组织成一个更高一层的逻辑单元

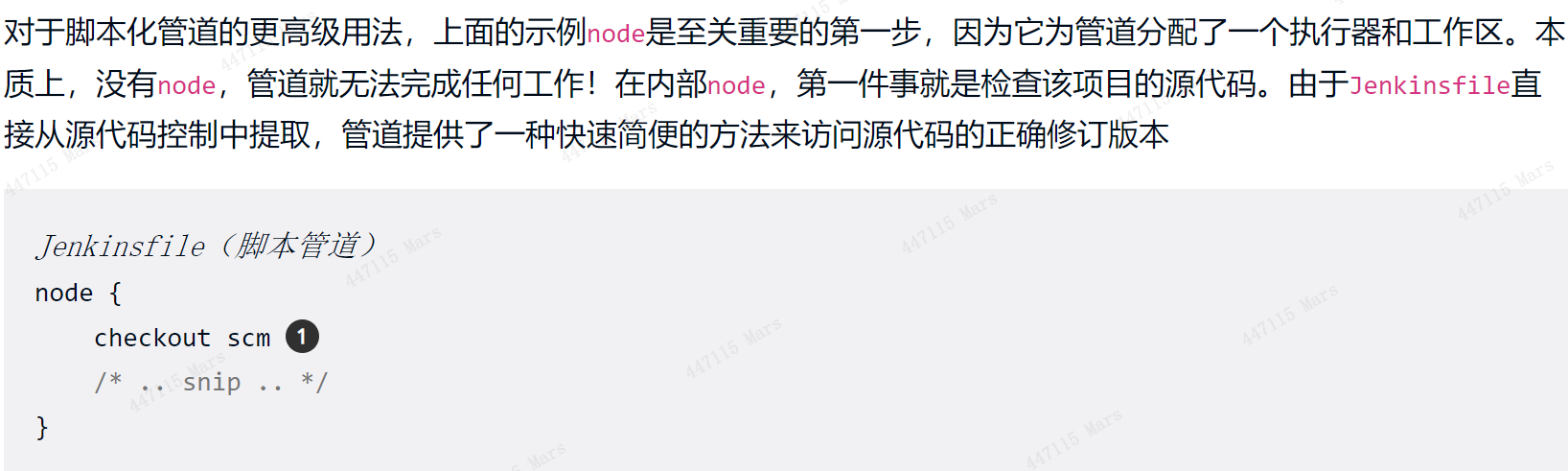
Node 指定这些任务在哪执行

Stage和Step可以放到一个Node下面执行，不指定就默认在master节点上面执行。 另外Node和Step也能组合成一个Stage。

声明式：agent指示 Jenkins 为整个管道分配一个执行器（在 Jenkins 环境中的任何可用代理/节点上）和工作区。

脚本式：node实际上与（上面）作用相同agent。

echo在控制台输出中写入简单的字符串。



该checkout步骤将从源代码控制中检出代码；scm是一个特殊变量，它指示该checkout步骤克隆触发此管道运行的特定修订版。

### 建造

对于许多项目来说，管道中“工作”的开始是“构建”阶段。通常，管道的这个阶段是源代码被组装、编译或打包的地方。它并不是Jenkinsfile现有构建工具（如 GNU/Make、Maven、Gradle 等）的替代品，而是可以看作是将项目开发生命周期的多个阶段（构建、测试、部署等）绑定在一起的粘合层。

Jenkins 有许多插件可用于调用几乎所有常用的构建工具，但此示例将简单地make从 shell 步骤 ( sh) 调用。此sh步骤假定系统是基于 Unix/Linux 的，对于基于 Windows 的系统，bat可以使用 。



该sh步骤调用make命令，并且仅当命令返回零退出代码时才会继续。任何非零退出代码都将导致管道失败。

archiveArtifacts捕获与包含模式 () 匹配的文件\*\*/target/\*.jar并将其保存到 Jenkins 控制器以供稍后检索。

### 测试

运行自动化测试是任何成功的持续交付流程的关键组成部分。因此，Jenkins 拥有许多 插件提供的大量测试记录、报告和可视化功能。从根本上讲，当测试失败时，让 Jenkins 记录失败以便在 Web UI 中进行报告和可视化非常有用。下面的示例使用了JUnit 插件junit提供的步骤 。

在下面的示例中，如果测试失败，则管道将被标记为“不稳定”，如 Web UI 中的黄色球所示。根据记录的测试报告，Jenkins 还可以提供历史趋势分析和可视化。



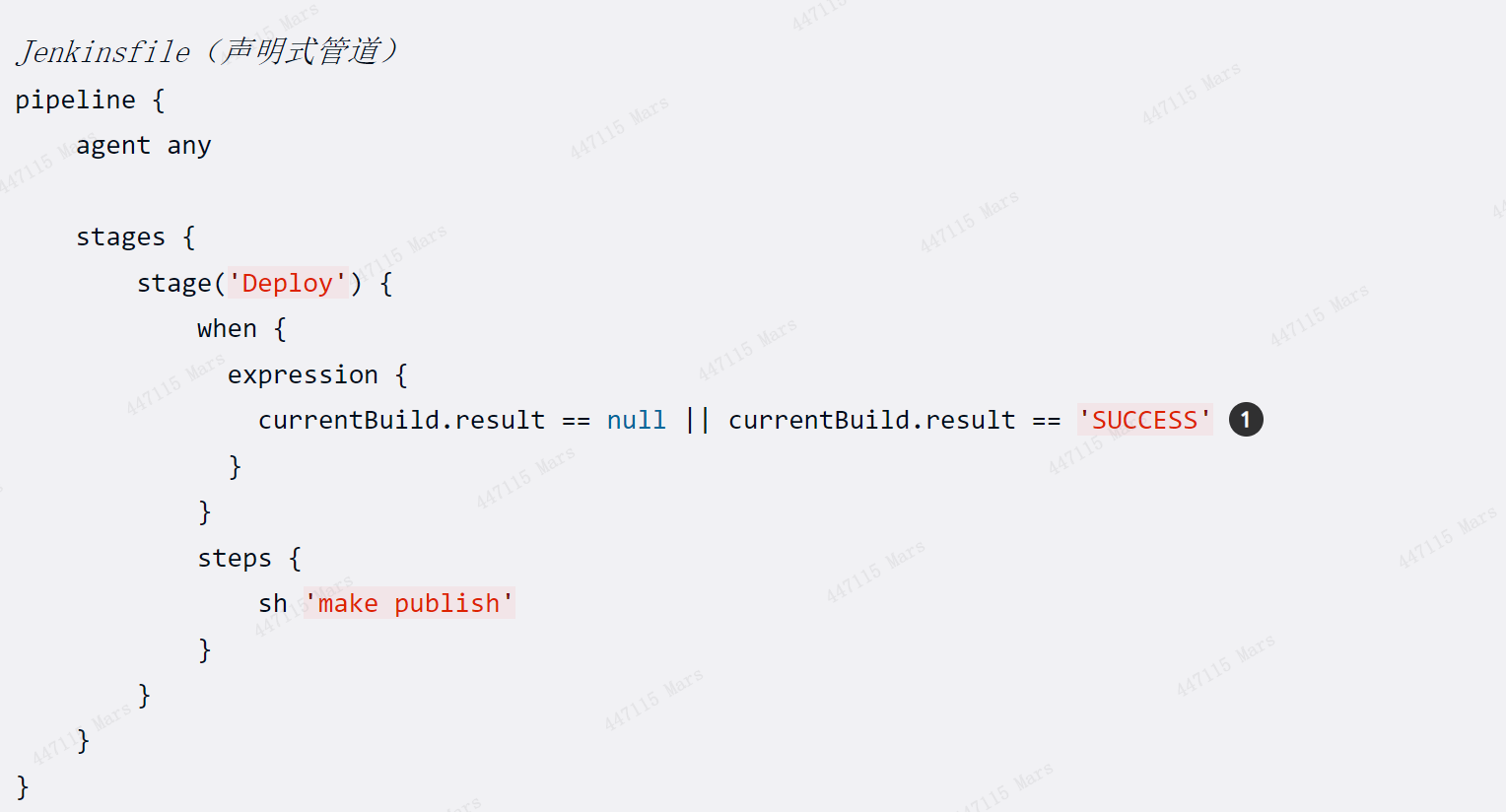
使用内联 shell 条件 ( sh 'make check || true') 可确保 sh步骤始终看到零退出代码，从而使junit步骤有机会捕获和处理测试报告。下文的“处理失败”部分将更详细地介绍替代方法。

junit捕获并关联与包含模式匹配的 JUnit XML 文件 ( \*\*/target/\*.xml)。

### 部署

根据项目或组织的要求，部署可能意味着各种步骤，可能是将构建的工件发布到 Artifactory 服务器，或将代码推送到生产系统。

在示例管道的这个阶段，“构建”和“测试”阶段都已成功执行。实际上，“部署”阶段只会在先前阶段成功完成后才执行，否则管道会提前退出。



访问currentBuild.result变量允许管道确定是否有任何测试失败。在这种情况下，该值将是 UNSTABLE。

假设示例 Jenkins Pipeline 中的所有内容均已成功执行，则每次成功的 Pipeline 运行都会在 Jenkins 中存档相关的构建工件、报告测试结果并显示完整的控制台输出。

### 部署三阶段

一般我们的持续交付都有三个部分：Build、Test、Deploy，典型写法：



### 使用环境变量

Jenkins Pipeline 通过全局变量 公开环境变量env，该变量可从中的任何位置访问Jenkinsfile。Jenkins Pipeline 中可访问的环境变量的完整列表记录在 ${YOUR\_JENKINS\_URL}/pipeline-syntax/globals#env 中

可在文档localhost:8080/pipeline-syntax/globals#env找到， 通过env直接使用它们：

echo "Running ${env.BUILD\_ID} on ${env.JENKINS\_URL}"

包括：

BUILD\_ID

当前构建 ID，与在 Jenkins 1.597+ 版本中创建的构建的 BUILD\_NUMBER 相同

版本号

当前版本号，例如“153”

BUILD\_TAG

jenkins-${JOB\_NAME}-${BUILD\_NUMBER} 字符串。方便放入资源文件、jar 文件等，以便于识别

BUILD\_URL

可以找到此构建结果的 URL（例如 http://buildserver/jenkins/job/MyJobName/17/ ）

EXECUTOR\_NUMBER

标识当前执行此构建的执行器（在同一台机器的执行器中）的唯一编号。这是您在“构建执行器状态”中看到的数字，但该数字从 0 开始，而不是从 1 开始

JAVA\_HOME

如果您的作业配置为使用特定 JDK，则此变量将设置为指定 JDK 的 JAVA\_HOME。设置此变量后，PATH 也会更新为包含 JAVA\_HOME 的 bin 子目录

JENKINS\_URL

Jenkins 的完整 URL，例如 https://example.com:port/jenkins/（注意：仅在“系统配置”中设置了 Jenkins URL 时才可用）

作业名称

此构建的项目名称，例如“foo”或“foo/bar”。

节点名称

当前构建正在运行的节点的名称。对于 Jenkins 主服务器，设置为“master”。

工作区

工作空间的绝对路径

### 设置环境变量

在 Jenkins 管道中设置环境变量的方式取决于使用的是声明式管道还是脚本式管道。

声明式管道支持环境 指令，而脚本式管道的用户必须使用该withEnv步骤。



environment顶层块中使用的指令将pipeline应用于管道内的所有步骤。

environment在内定义的指令将stage仅将给定的环境变量应用于内的步骤stage。

脚本式：



