MARKDOWN 学习记录

一、文本格式化

1.1 字体格式

1.1.1 GIT 的提交、修改、历史查看、协作

1. 工作区与版本库：*工作区是一个包含.git子目录（内含版本库）的目录，通过`init`在当前目录创建版本库。*

2. 版本提交：*版本提交依次经过工作区->暂存区->版本库，通过`add`命令确定哪些文件纳入下一次提交，以快照的形式放在暂存区，通过`status`命令可以查看下一次提交的状态，通过`commit`命令创建版本提交。*

3. 历史查看：*`log` 命令可用来显示提交历史,通过`--graph`,`-n`,`--oneline`,`--stat`等参数实现对版本提交历史的追溯。`diff`命令可以查看两个版本提交的不同。*

4. 推送和拉回：*`push` 和`pull`命令实现本地和远程版本库间的共享和协作。*

> [1]

二、表格制作

2.1 分支

创建分支	切换分支	记录查看
git branch	checkout命令	reflog命令

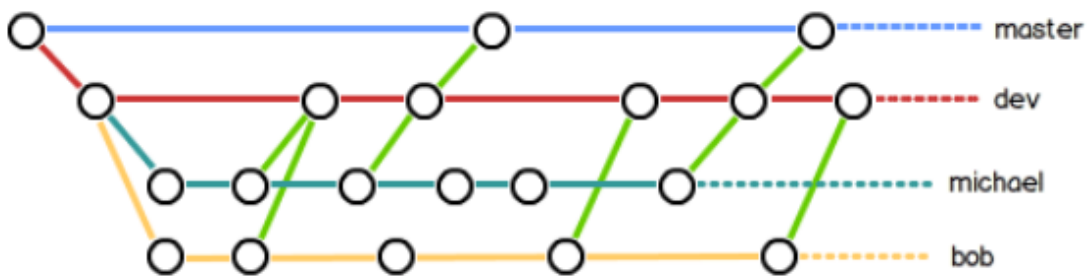
2.2 分支策略

在实际开发中，应该按照几个基本原则进行分支管理：

1. master分支应该是非常稳定的，也就是仅用来发布新版本，平时不能在上面干活；

2. 干活都在dev分支上，dev分支是不稳定的，到某个时候，比如1.0版本发布时，再dev分支合并到master上，在master分支发布1.0版本；
3. 每个人都在dev分支上干活，每个人都有自己的分支，时不时地往dev分支上合并就可以了。

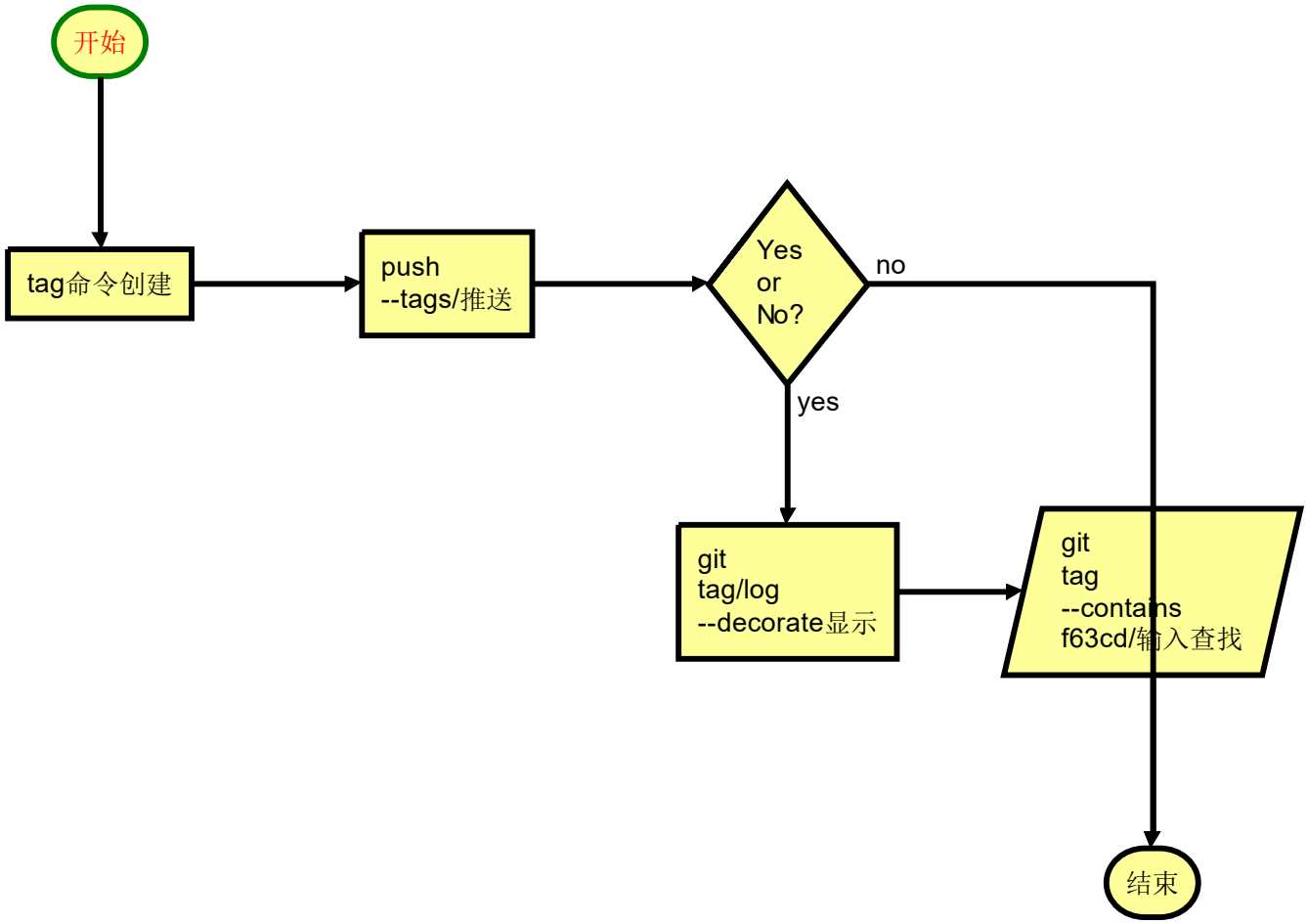
如下图所示：



> [2]

三、流程图制作

3.1 打tag



3.2 其他命令

命令	功能	参数

clone	git clone ssh://...git 用--bare参数创建不带工作区的版本库	bare
stash	将当前修改保存到本地，而去处理其他	
git stash pop	恢复位于栈顶的内容	
rebase	git rebase master 将活动分支上的最新修改纳入某分支 git rebase master --onto release1 活动分支不属于master的提交拷贝到release1	--onto
reset	版本回退	--hard

> [3]

四、序列图制作

4.1 提交过程

4.2 git 分支管理

> [4]

[1]：2020.01.02 周峰

[2]：2020.01.05 周峰

[3]：2020.01.06 周峰

[4]：2020.01.07 周峰