

Git 与版本管理

1

版本管理

多人协作项目代码怎么管理

- 用 U 盘或者网盘保持同步?
- 用 SVN?
- 你听说过 Git 吗?

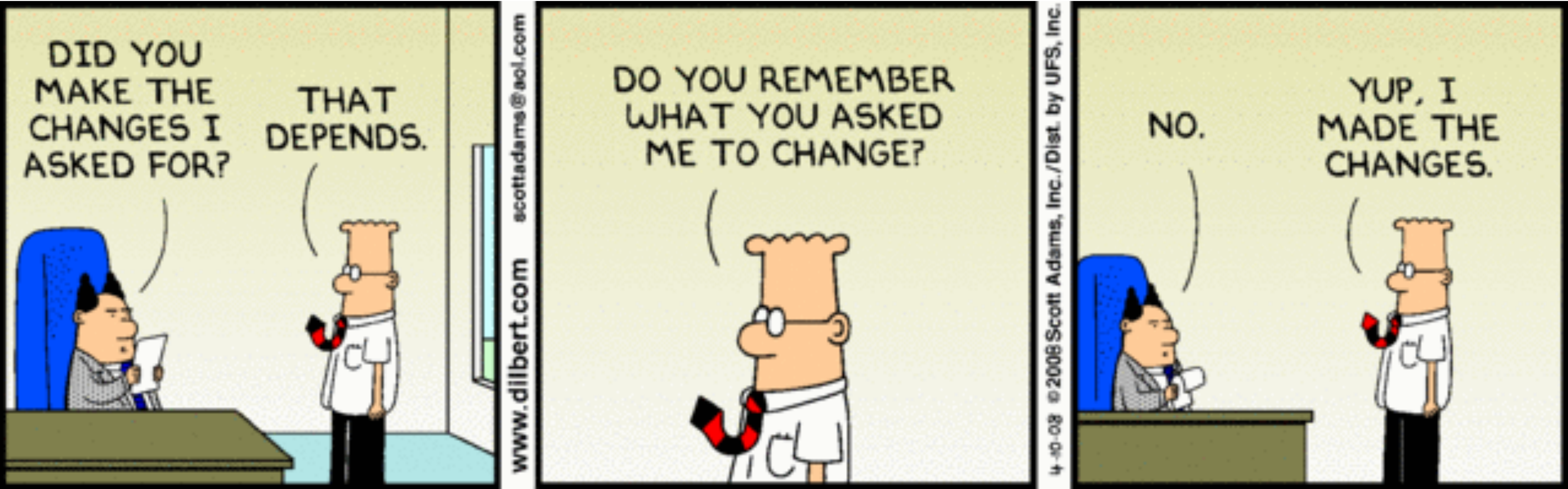


据说你们的学长曾经这么做

- 毕业论文
- 毕业论文1
- 毕业论文2
- 毕业论文改
- 毕业论文改1
- 毕业论文完成版
- 毕业论文完成版1
- 毕业论文最终版
- 毕业论文最终版1
- 毕业论文最终版2
- 毕业论文最终绝不改版
- 毕业论文最终绝不改版1
- 毕业论文最最最最最最最终版
- 毕业论文最最最最最最最终版1

系统工程第一定则

“No Matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle”



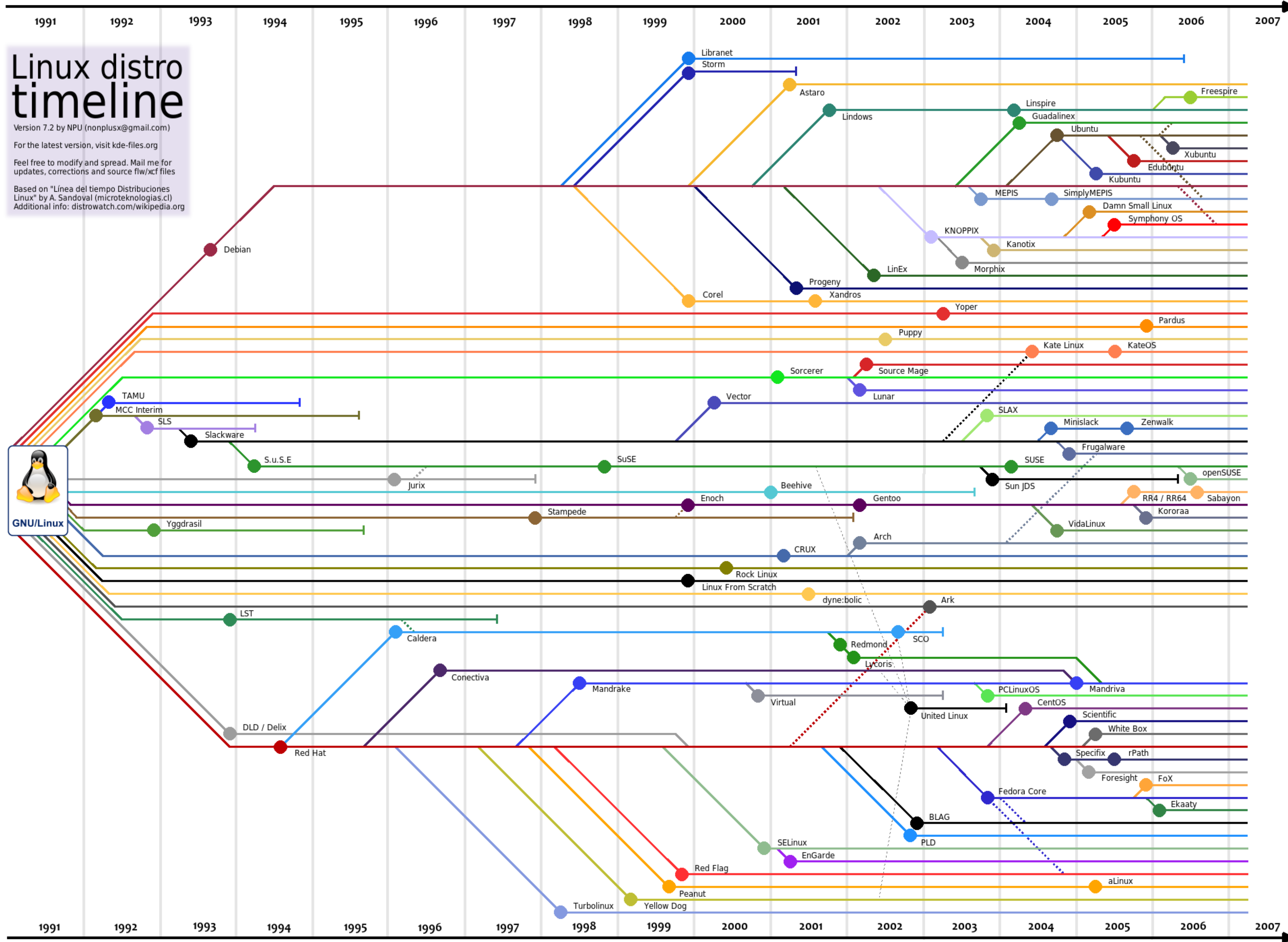
Linux distro timeline

Version 7.2 by NPU (nonplusx@gmail.com)

For the latest version, visit kde-files.org

Feel free to modify and spread. Mail me for updates, corrections and source flw/xcf files

Based on "Línea del tiempo Distribuciones Linux" by A. Sandoval (microtecnologias.cl)
Additional info: distrowatch.com/wikipedia.org



我们面对的问题

- 01 | 多个人需要共同参与一个软件开发
- 02 | 一个软件可能有多个要支持的版本
- 03 | 一个软件可能会用多种不同的运行环境

术语定义：版本（Version）

- 最初发布或再发布的“代码及其附属品”的组合，它应该是可被完整编译或被认定为完整可用的。
- 不同的版本表现出不同的功能特性。

术语定义：基准（Baseline）

- 根据质量管理所需确定的阶段性规格说明，这个说明应是被正式评审认可的。
- 之后的开发过程都应该要遵循“基准”的要求进行。
- 对于基准的修改是要极为慎重的，要通过严格的变更流程。

例子：

Baseline A：所有接口应被完备的定义，各方法的内容为空。

Baseline B：所有的数据访问方法应该被实现并测试。

Baseline C：GUI 被完成

术语定义：基准（Baseline）

- 基准的命名有一个非常常用的三点命名法。

A.B.C
如 9.3.1

- A：从用户（消费者）角度看到的发布出的标记数（Release）
- B：从开发者角度看到的关键的版本（Version）号
- C：从开发者角度关注的修改版本（Revision）号

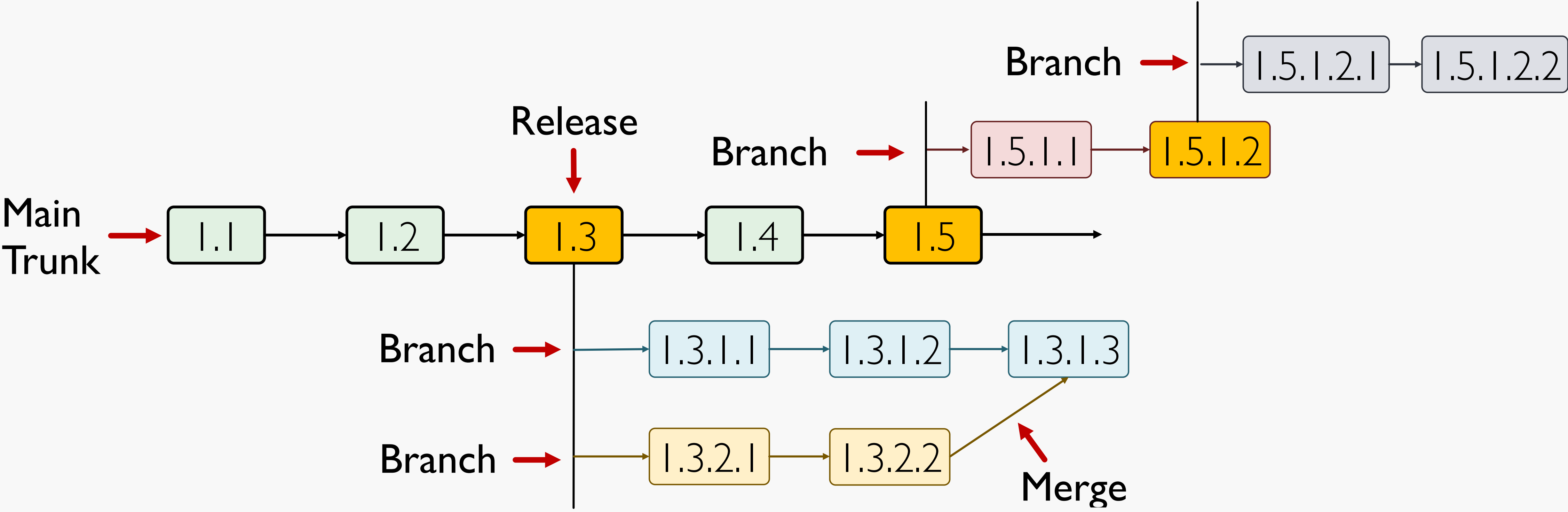
术语定义：发布（Release） / 版本（Version） / 修改（Revision）

- 版本（Version）：最初发布或再发布的“代码及其附属品”的组合，它应该是可被完整编译或被认定为完整可用的。不同的版本表现出不同的功能特性。
- 修改（Revision）：对于一个版本的修改，只做了代码设计的错误修正，对于已经“附属品”中文档所描述的功能特性没有任何改动。
- 发布（Release）：被批准的面向用户进行分发的版本。

版本管理系统

- 用于追踪和管理对于一系列文件和资源修改的软件系统
- 帮助团队在代码项目上进行协作
 - 所有成员可以访问的代码
 - 将当期版本呈现在文件系统中，将过往版本备份在不直接被看到的地方
 - 可以看见过去发生了什么
 - 当不同人修改了同一部分内容时可以解决冲突

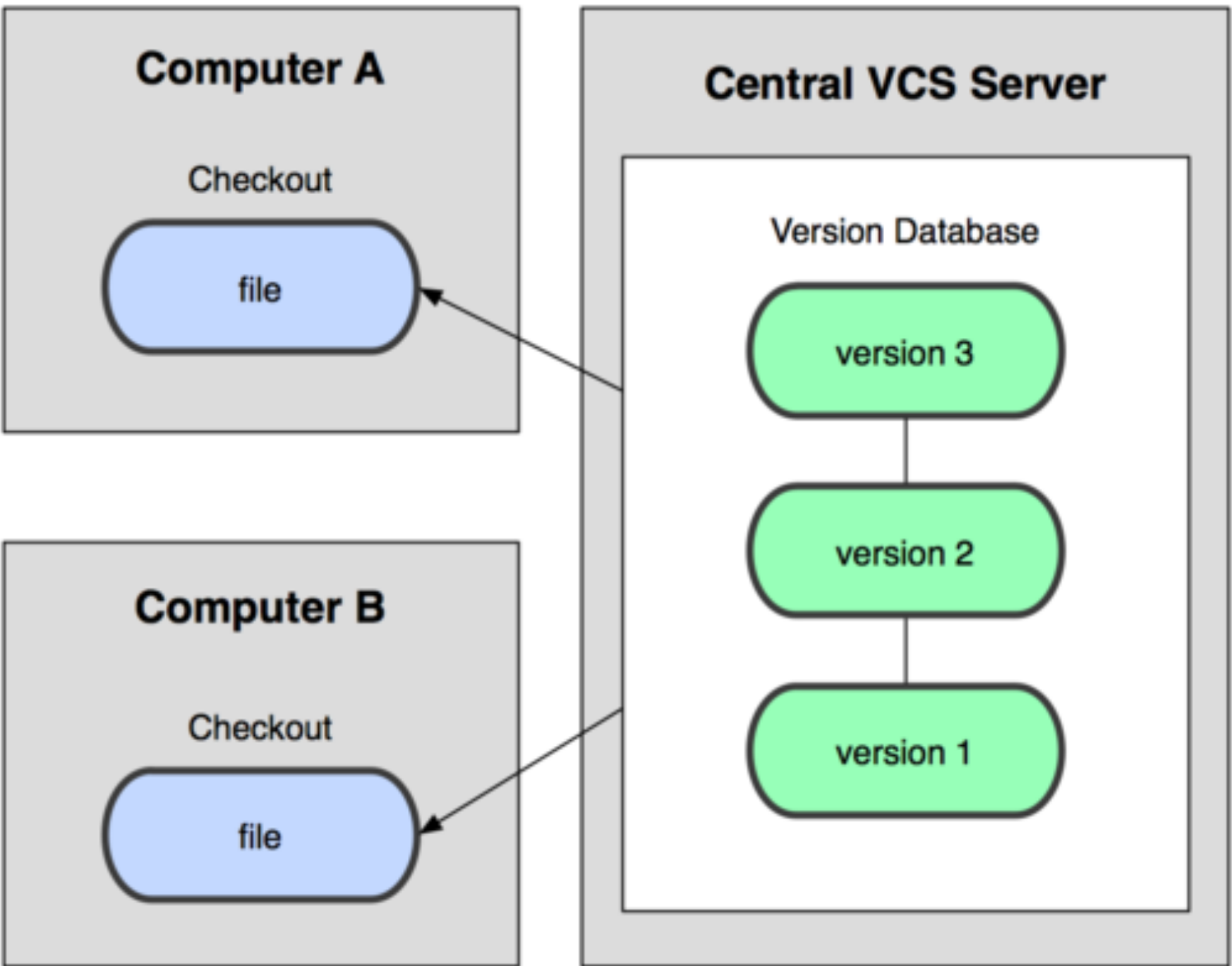
备注：也可以用于非代码的文本版本管理（如论文）



中心式 vs 分布式版本管理

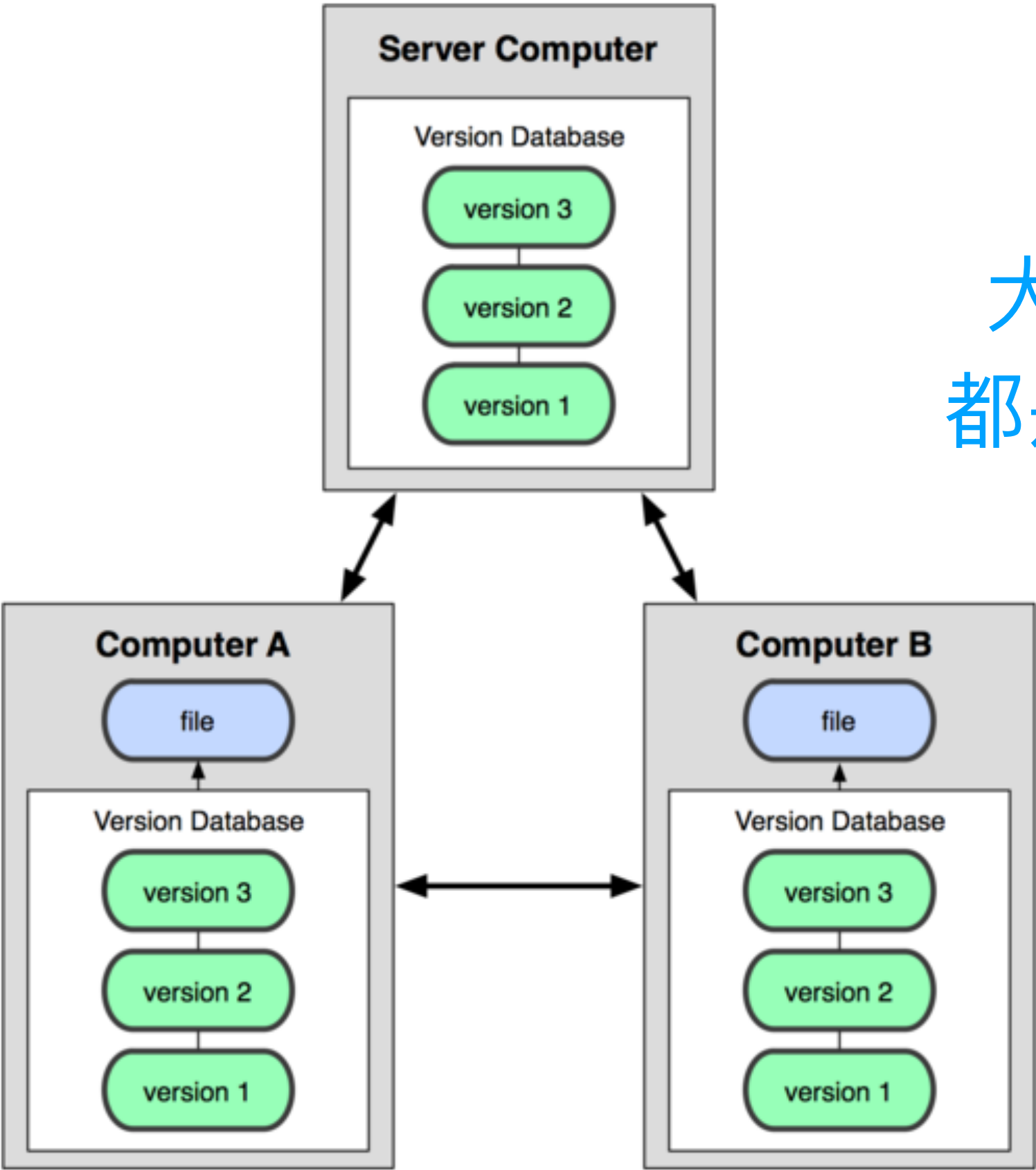
中心式模型

(CVS, Subversion, Perforce)



分布式模型

(Git, Mercurial)



大量的操作
都是本地化的

2

Git

- 01 | Linux 社区中，为管理 Linux 操作系统内核迭代产生
- 02 | 2005 年 Linus Torvalds 作为初始创作者
- 03 | 为了替代闭源的 BitKeeper 方案而产生

本地 Git 仓库的三个分区

01

工作目录（修改/没修过的文件）

`working directory`

02

暂存区（暂存的文件）

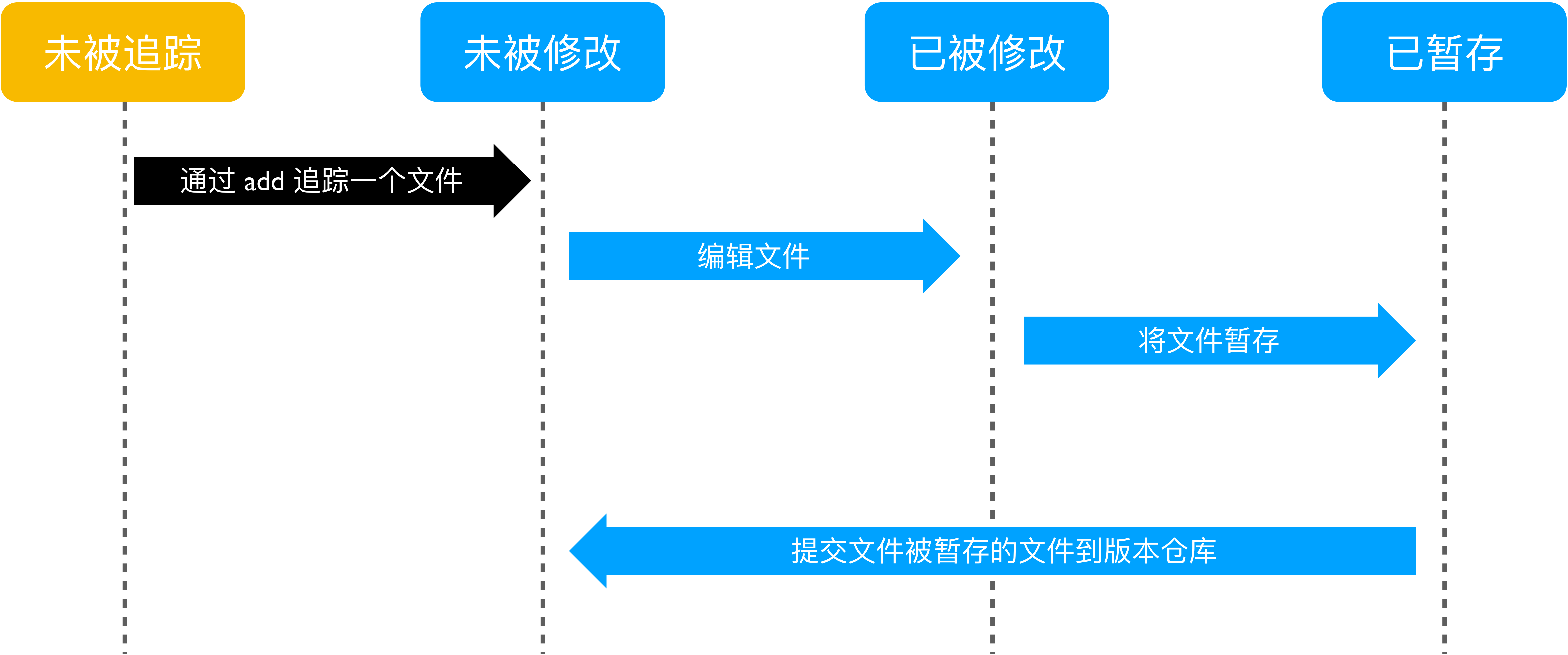
`staging area`

03

Git 仓库（提交的文件）

`repo`

Git 中文件生命周期



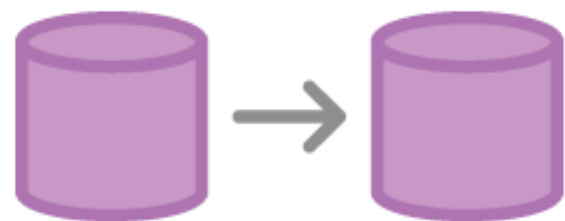
基本的 Git 使用

配置一个 Git 仓库



Git init

创建 Git 仓库，让当前目录处于 Git 管理之下



Git clone

将一个已经存在的 Git 仓库克隆到本地



Git config

对 Git 的基本配置进行设置（如邮箱、姓名）

初次使用时

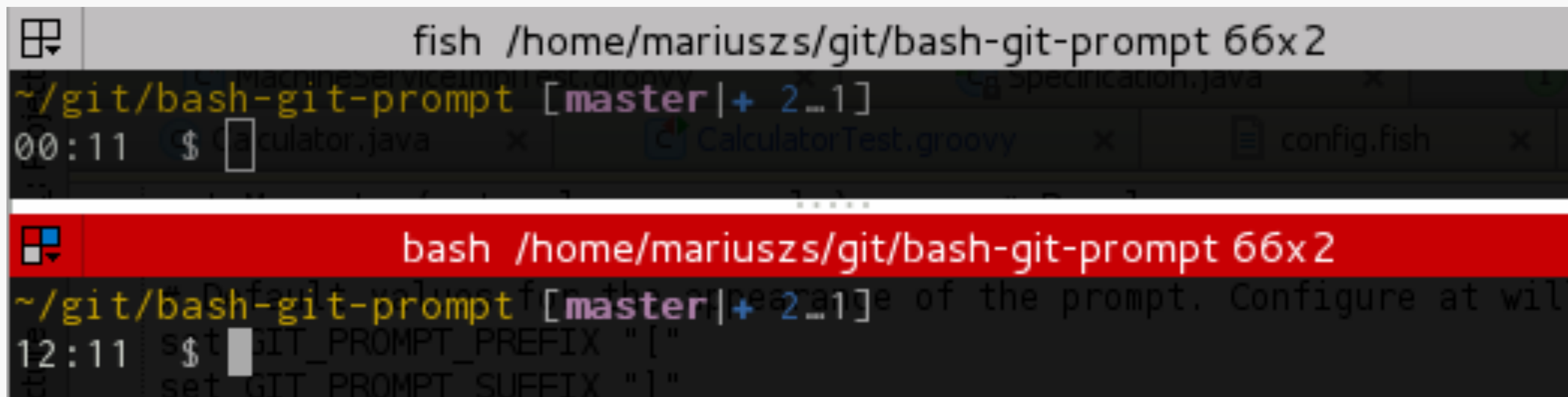
```
git config --global user.name "your name"  
git config --global user.email "your email"
```

令配置被写入 ~/.gitconfig

初次使用时

```
git config --global user.name "your name"  
git config --global user.email "your email"
```

令配置被写入 ~/.gitconfig



```
fish /home/mariuszs/git/bash-git-prompt 66x2  
~/git/bash-git-prompt [master|+ 2_1]  
00:11 $  
...  
bash /home/mariuszs/git/bash-git-prompt 66x2  
~/git/bash-git-prompt [master|+ 2_1]  
12:11 $ set GIT_PROMPT_PREFIX "["  
set GIT_PROMPT_SUFFIX "]"
```

创建仓库

01

在当前目录下创建

```
git init
```

02

在指定的目录创建

```
git init <指定的目录路径>
```

03

创建一个 bare git 仓库

```
git init -bare myrepo.git
```

01

克隆本地仓库

```
git clone /path/to/repo.git
```

02

克隆远端仓库

```
git clone http://gitlab.inspurcloud.cn/WuHanU/slides.git
```

03

通过 ssh 协议完成克隆

```
git clone git@gitlab.inspurcloud.cn:WuHanU/slides.git
```

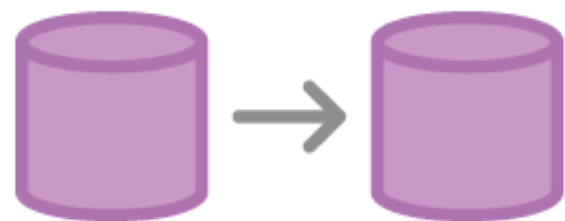
基本的 Git 使用

配置一个 Git 仓库



Git init

创建 Git 仓库，让当前目录处于 Git 管理之下



Git clone

将一个已经存在的 Git 仓库克隆到本地



Git config

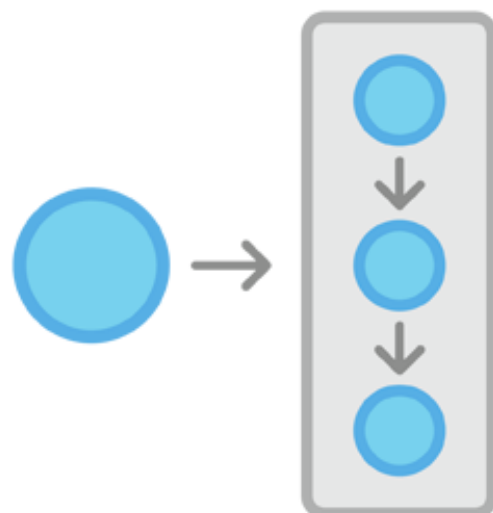
对 Git 的基本配置进行设置（如邮箱、姓名）

进行修改的记录



Git add

将修改从工作目录添加到暂存区。创建一个待提交的“快照”。



Git commit

将暂存区的“快照”提交到 Git 仓库的历史中。


```
echo "Hello" > README
```

```
git add README
```

```
git commit -m "Add README"
```

```
echo "Hello again" >> README
```

```
git commit -am "Modify README"
```

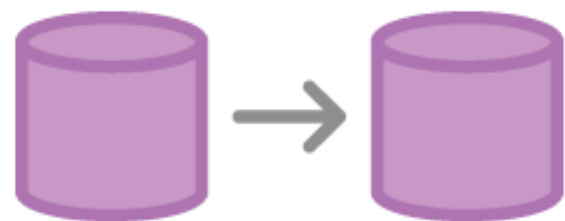
基本的 Git 使用

配置一个 Git 仓库



Git init

创建 Git 仓库，让当前目录处于 Git 管理之下



Git clone

将一个已经存在的 Git 仓库克隆到本地



Git config

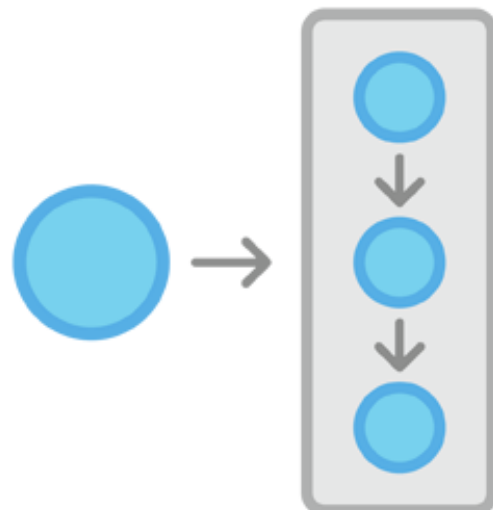
对 Git 的基本配置进行设置（如邮箱、姓名）

进行修改的记录



Git add

将修改从工作目录添加到暂存区。创建一个待提交的“快照”。



Git commit

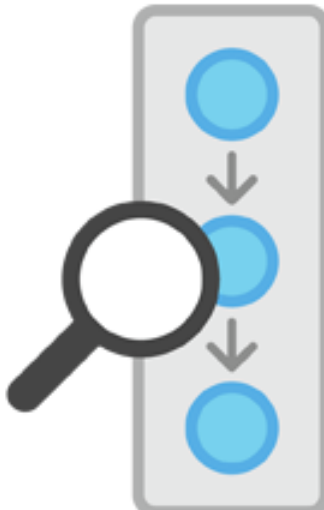
将暂存区的“快照”提交到 Git 仓库的历史中。

查看 Git 仓库的信息



Git status

显示当前目录的修改情况以及暂存区的“快照变化”



Git log

查看 Git 的各个版本历史情况。

git status

On branch master

Changes to be committed:

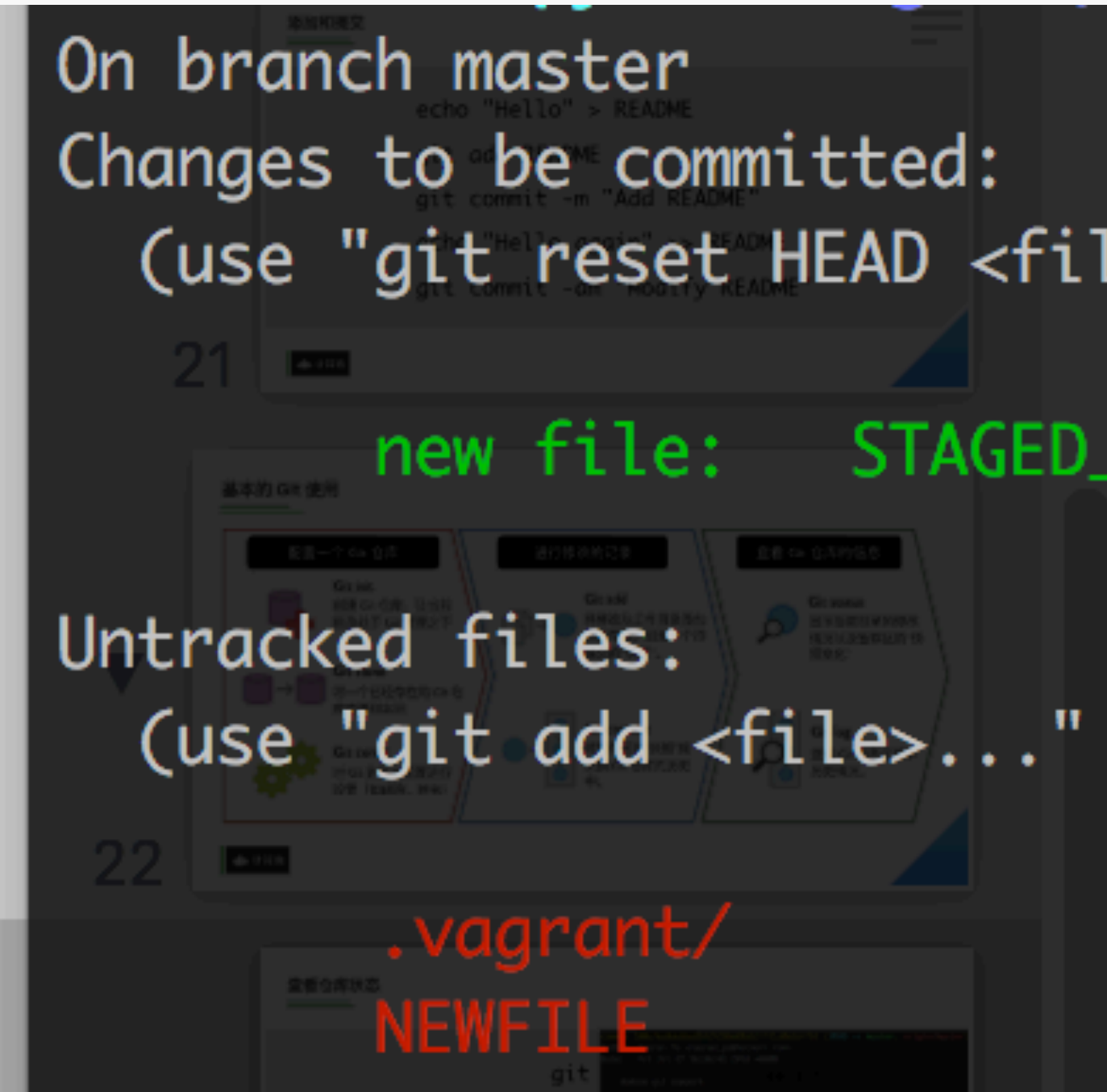
(use "git reset HEAD <file>..." to unstage)

new file: STAGED_FILE

Untracked files:


(use "git add <file>..." to include in what will be committed)

.vagrant/
NEWFILE



21

22



git log

```
commit 180a76d8de5ba82d24780a08ab2f731d0e6ef7bf (HEAD -> master, origin/master)
Author: Haoran Yu <haoran_yu@hotmail.com>
Date:   Fri Jul 27 16:56:42 2018 +0800

    Update git support

commit 40f83f599e801b1d5d298ce499c0712cdb086891
Author: Haoran Yu <haoran_yu@hotmail.com>
Date:   Fri Jul 27 13:48:32 2018 +0800

    Update Readme to guide students better

commit bb47cc3d0cdb82b276228e6dfc928c70c14e2c52
Author: Haoran Yu <haoran_yu@hotmail.com>
Date:   Fri Jul 27 13:34:53 2018 +0800

    Roll back to fedora 27 and support ipython, bpython and pip18

commit b29a7c86397728732b765ca33ddaeba082353d3b
Author: Haoran Yu <haoran_yu@hotmail.com>
Date:   Fri Jul 27 10:43:14 2018 +0800

    Initialize fedora28-python37 environment
```


查看仓库提交历史（最近3次详情）

git log -3 -p

```
commit 180a76d8de5ba82d24780a08ab2f731d0e6ef7bf (HEAD -> master, origin/master)
Author: Haoran Yu <haoran_yu@hotmail.com>
Date:   Fri Jul 27 16:56:42 2018 +0800

    Update git support

diff --git a/README.md b/README.md
index f921a65..be67e04 100755
--- a/README.md
+++ b/README.md
@@ -119,4 +119,15 @@ Type in `pip --version` or `pip3 --version` you should get
    pip 18.0 from /home/vagrant/.local/lib/python3.7/site-packages/pip (python 3.7)
    ...

+#### Extra preparation
+
+Type in `cd ~` and then
+
+```
+git clone https://github.com/django/django.git
+
+
+We would use it later during our lectures.
+
+
+*If any test above cannot pass on your machine, contact Haoran Yu (俞昊然) on IM or though Email
+yuhaoran@jisuanke.com as soon as possible.*
diff --git a/Vagrantfile b/Vagrantfile
index b664b1c..0318673 100755
--- a/Vagrantfile
+++ b/Vagrantfile
@@ -15,10 +15,10 @@ Vagrant.configure(2) do |config|
    end

    ...skipping...
commit 180a76d8de5ba82d24780a08ab2f731d0e6ef7bf (HEAD -> master, origin/master)
Author: Haoran Yu <haoran_yu@hotmail.com>
Date:   Fri Jul 27 16:56:42 2018 +0800

    Update git support

diff --git a/README.md b/README.md
index f921a65..be67e04 100755
```


Git Pretty Log

```
sergio@soviet-russia < b1.2.4 > : ~/projects/external/rubinius
% git log
commit 107632cd03c00f48c6f90dbbaca8fdaf4ee3a6b6
Author: Evan Phoenix <evan@fallingsnow.net>
Date: Tue Jul 5 18:33:08 2011 -0700
```

Update website for 1.2.4

```
commit 88d9f687c1672662acad9b3ec04cef3cd73b30f5
Author: Evan Phoenix <evan@fallingsnow.net>
Date: Tue Jul 5 17:51:31 2011 -0700
```

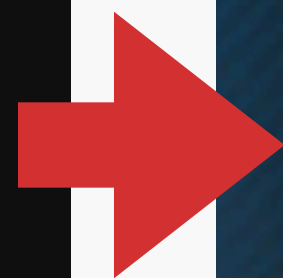
Bump version number

```
commit b7df3fd98aabbfce9a33cc5cda3f933d2c9806e0
Author: Shane Becker <veganstraightedge@gmail.com>
Date: Tue Jul 5 18:16:13 2011 -0700
```

regenned site for new blog post about status board

```
commit f76e532b185720b8eba663e325d94f331531918b
Author: Shane Becker <veganstraightedge@gmail.com>
Date: Tue Jul 5 18:13:37 2011 -0700
```

new blog post: rubinius status board

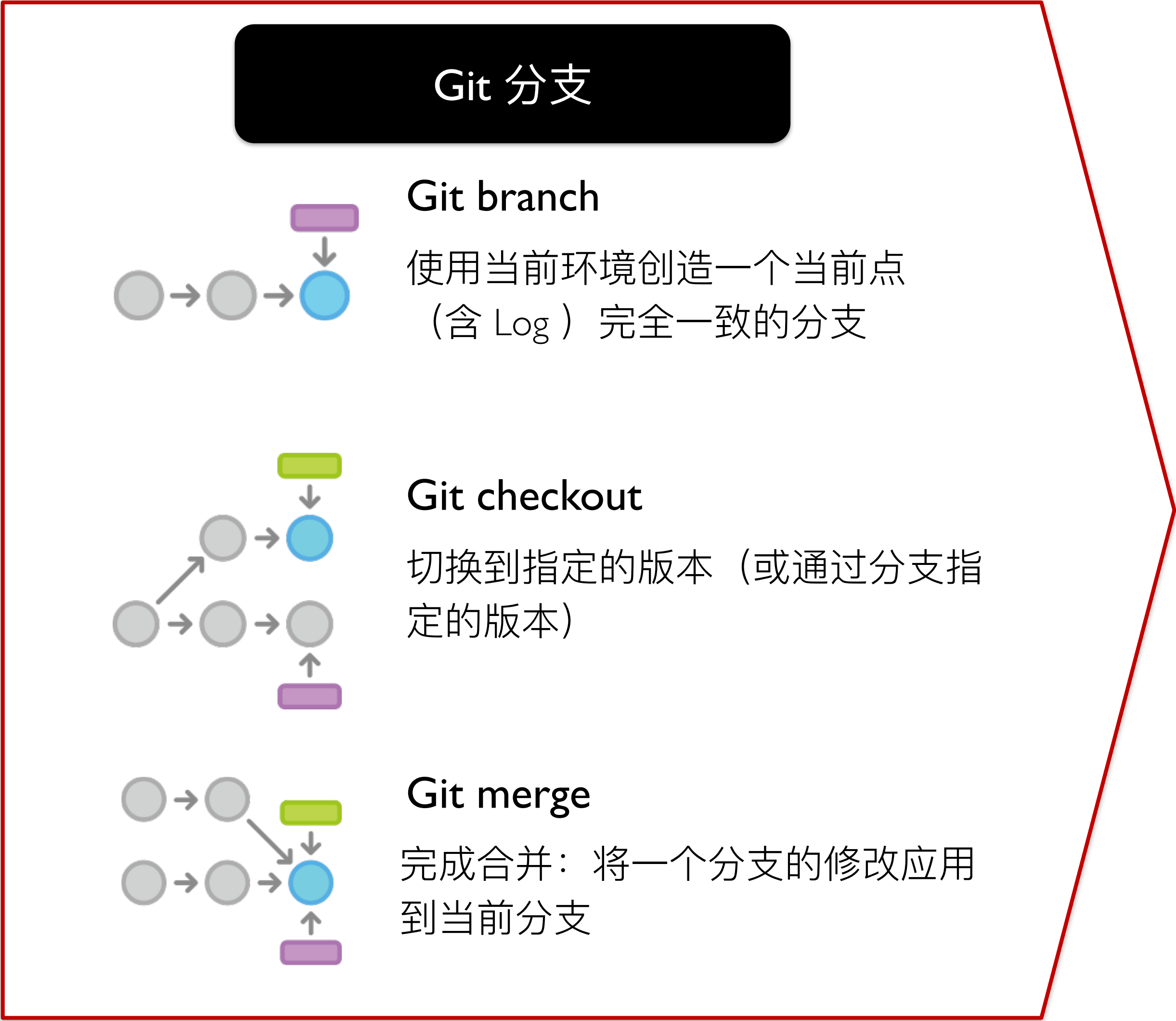


```
sergio@soviet-russia < b1.2.4 > : ~/projects/external/rubinius
% git log --pretty=format:'%Cred%H%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr)%Creset' --abbrev=40
107632c - (HEAD, release-1.2.4, b1.2.4) Update website for 1.2.4 (1 year, 4 months ago)
88d9f68 - Bump version number (1 year, 4 months ago)
b7df3fd - regenned site for new blog post about status board (1 year, 4 months ago)
f76e532 - new blog post: rubinius status board (1 year, 4 months ago)
42f7c72 - added capitalize to String case benchmarks (1 year, 4 months ago)
bddf636 - yet another way of removing the first elements from an array (1 year, 4 months ago)
6e4ed98 - new bench for Array#slice (1 year, 4 months ago)
049bace - Remove tags for now passing specs (1 year, 4 months ago)
44c3886 - Socket needs it's own shutdown (1 year, 4 months ago)
8374734 - regenned site for new blog post (map pins) (1 year, 4 months ago)
f90da99 - new blog post: rubinius around the world map and pins of shirts/tshirts (1 year, 4 months ago)
cf13e6b - Add a few more errno's based on OS X and Linux (1 year, 4 months ago)
0b8b477 - Add a bunch of errno's from FreeBSD (1 year, 4 months ago)
4b34345 - Load correct digest file, fixes broken Rubygems (1 year, 4 months ago)
e2be2d5 - Remove unused rubinius::guards (1 year, 4 months ago)
23e97d5 - Remove used flag and file it was defined in (1 year, 4 months ago)
cff4ee2 - Remove unused CallFrameList and some maps (1 year, 4 months ago)
dd8f2b1 - Removed unused async message and mailbox code (1 year, 4 months ago)
c4b54ba - Remove unused code (1 year, 4 months ago)
744e9f0 - Fix tiny typo's (1 year, 4 months ago)
912d530 - Cleanup last remnants of dynamic interpreter (1 year, 4 months ago)
6b29b21 - Remove unused IndirectLiterals (1 year, 4 months ago)
83db68a - Fixed Digest requires in const missing. (1 year, 4 months ago)
```


- <https://try.github.io/>
- <https://git-scm.com/book/en/v2>
- <https://speakerdeck.com/mattketmo/understanding-git>

Git 的多人协作 / 远程仓库

基本的 Git 使用



分支

01

查看现有分支

```
git branch
```

02

创建分支

```
git branch <分支名>
```

03

切换分支

```
git checkout <分支名>
```

04

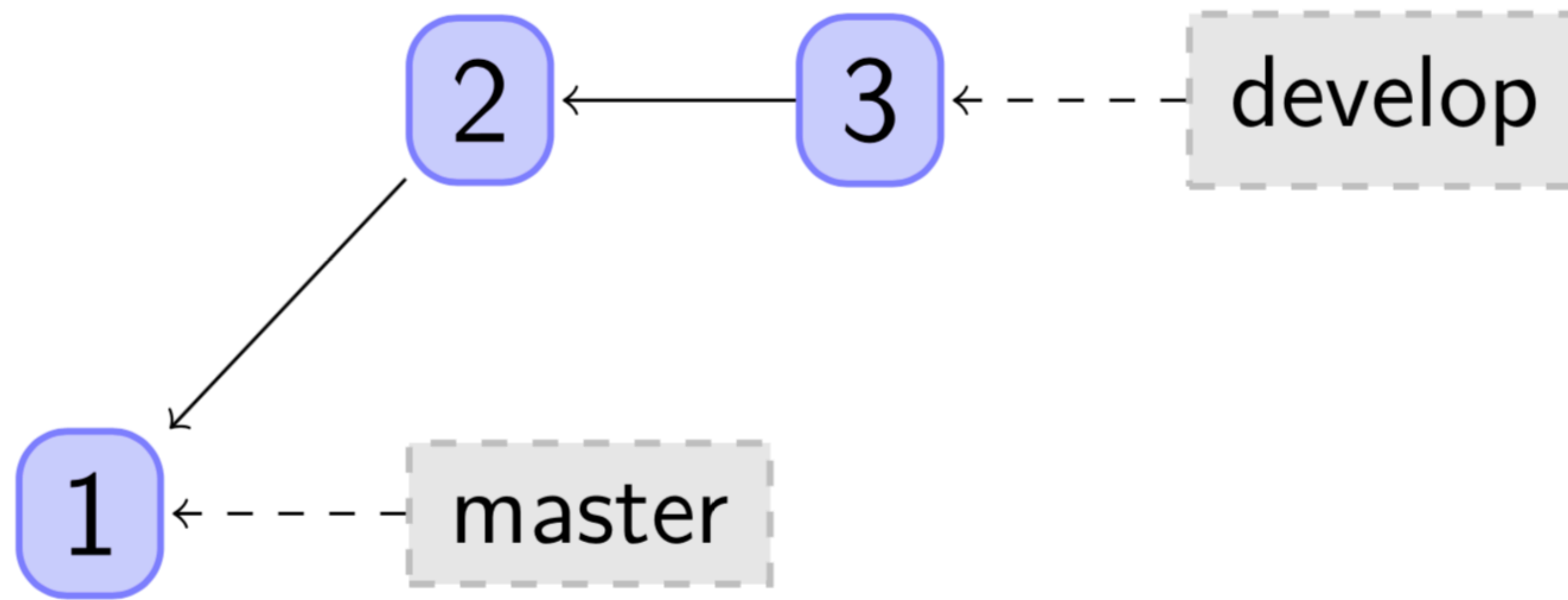
创建并切换到分支

```
git checkout -b <分支名>
```

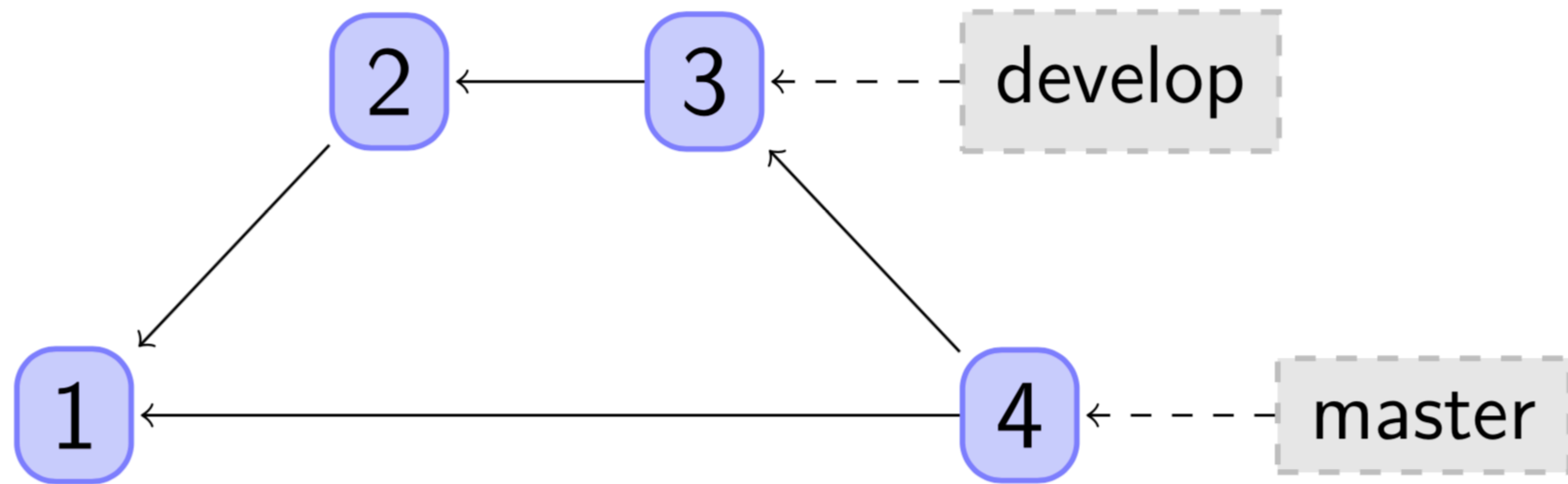
05

删除分支

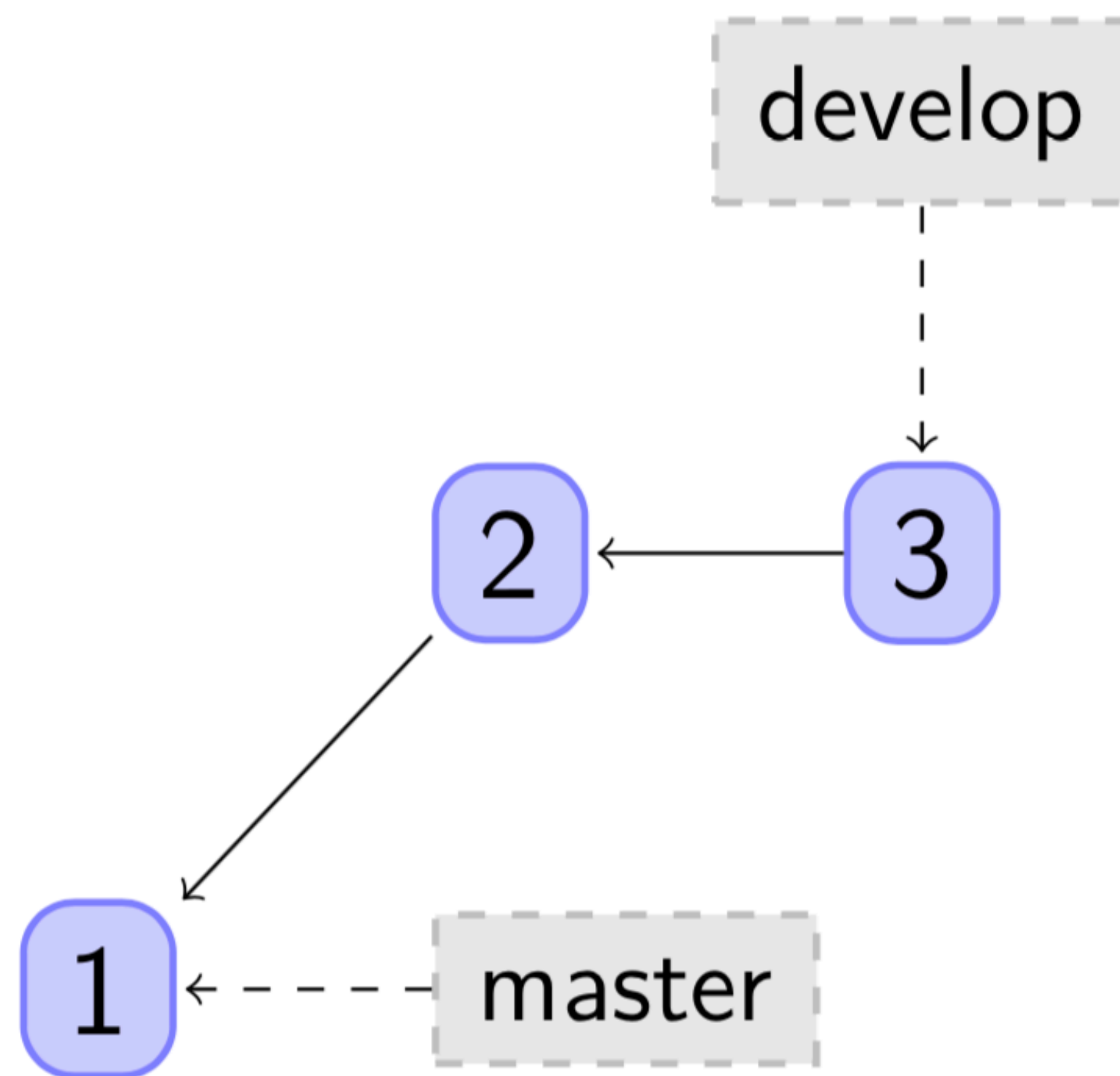
```
git branch -d <分支名>
```



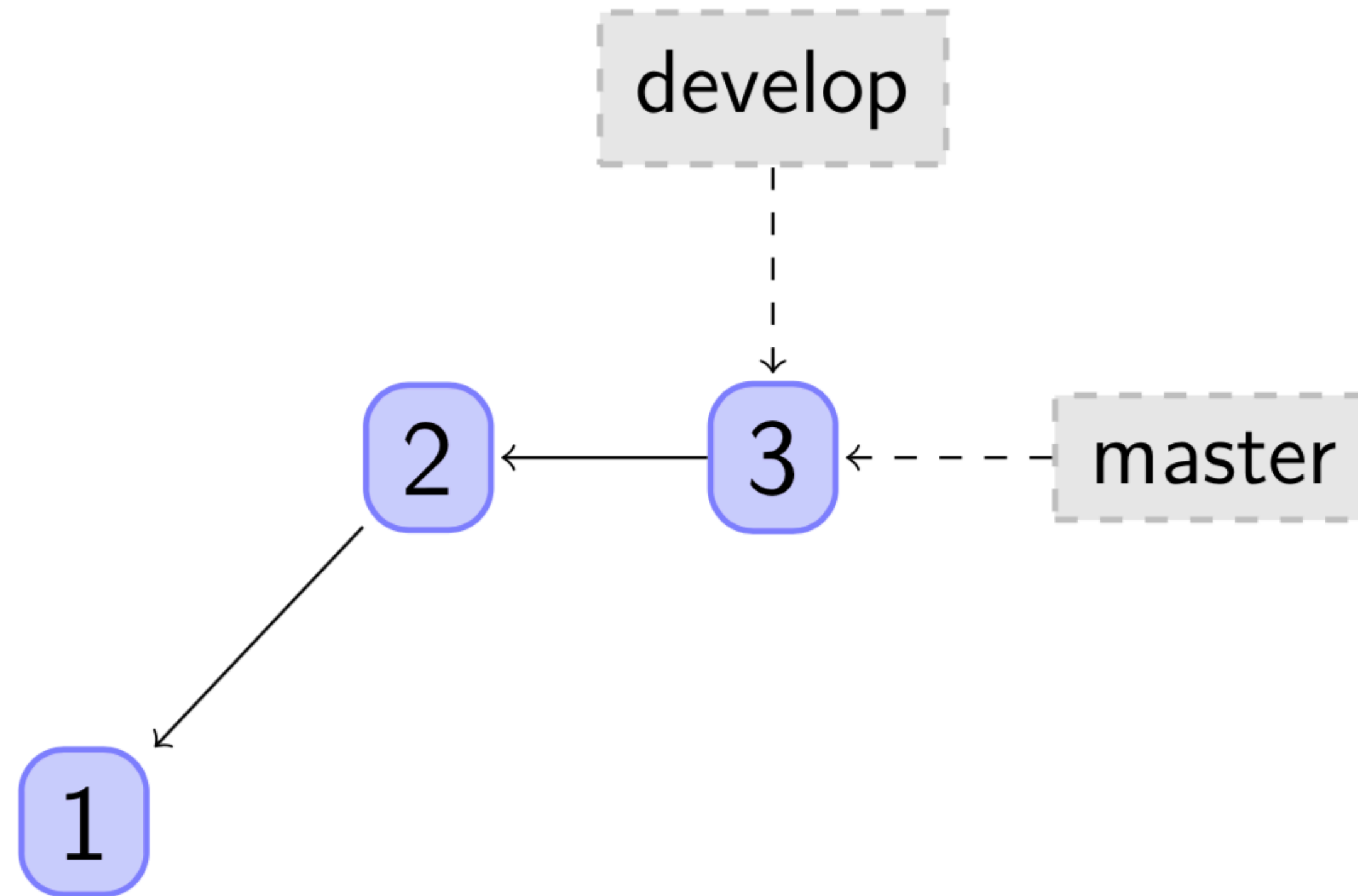
合并前



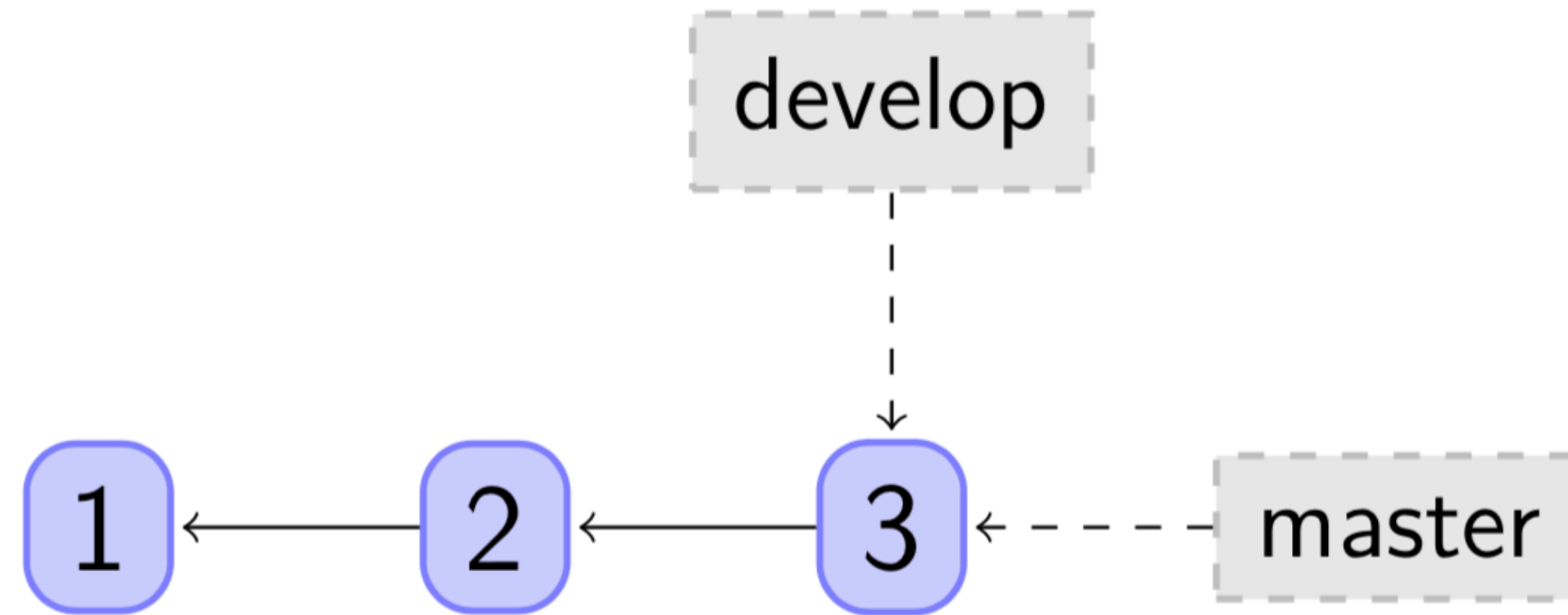
合并后



合并前

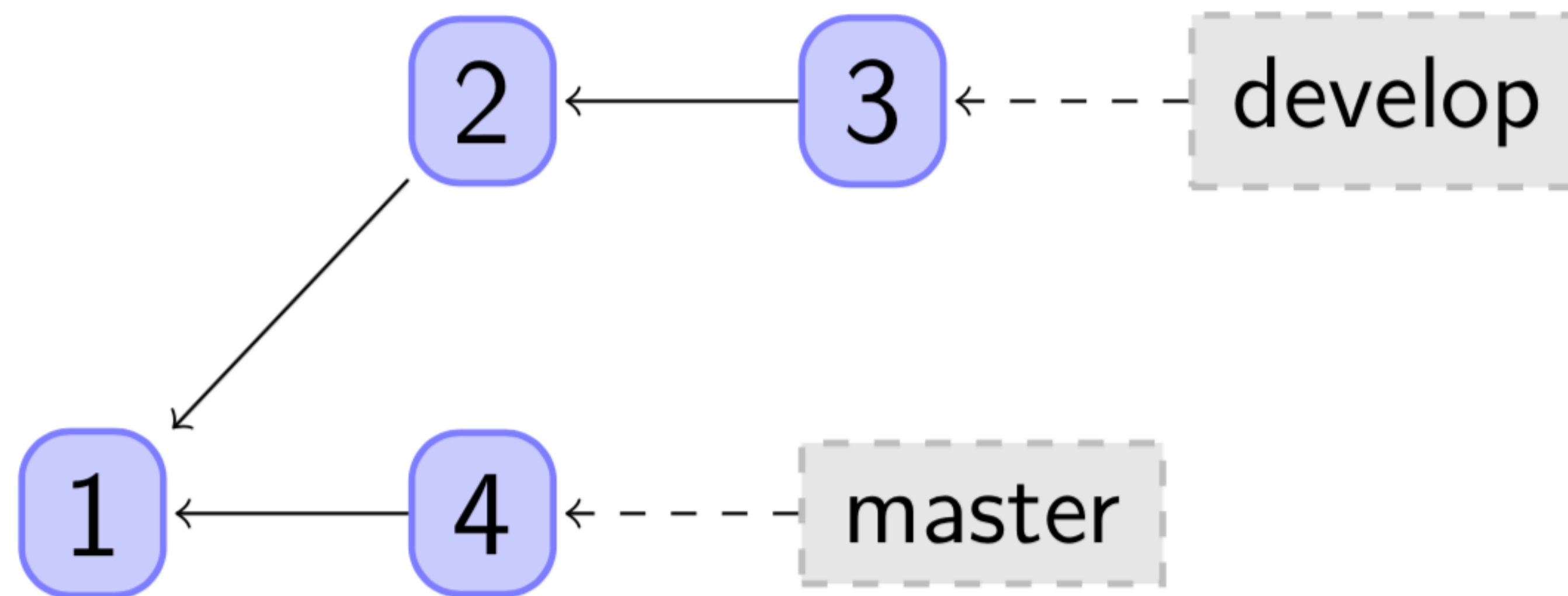


合并后

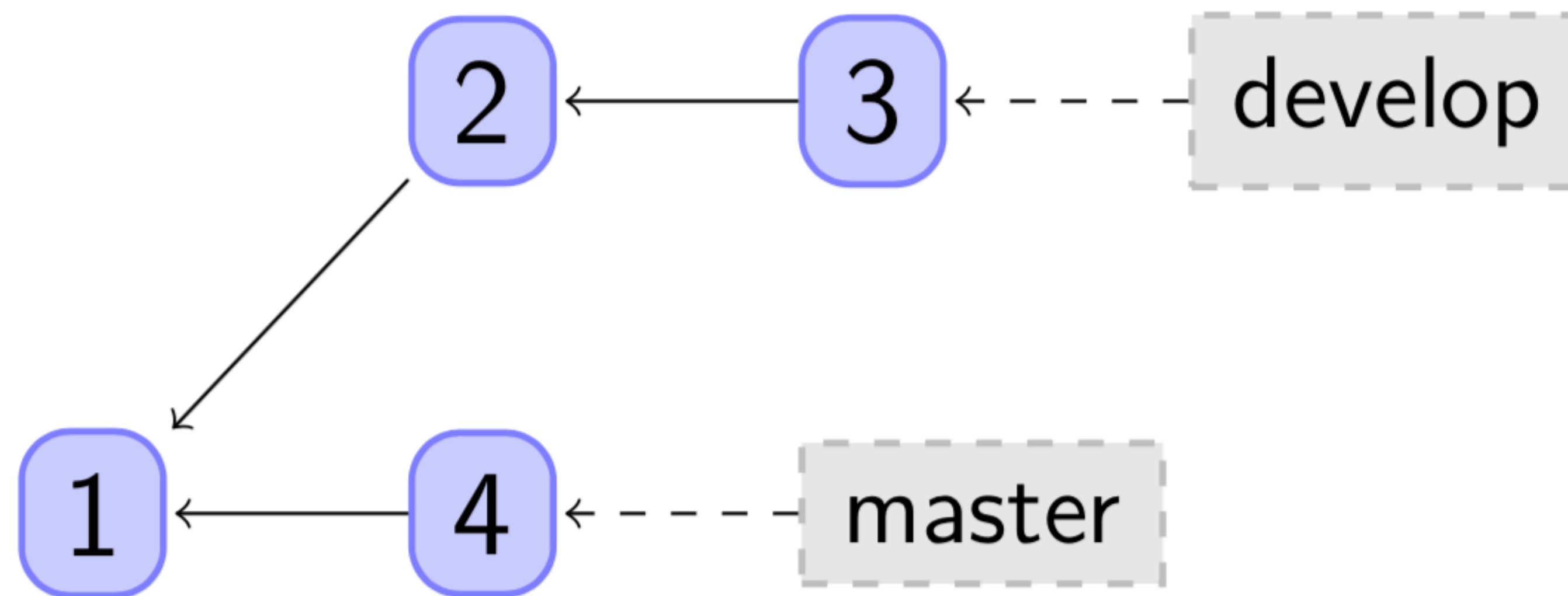


合并后

是否可以 fast-forward 呢?



是否可以 fast-forward 呢?



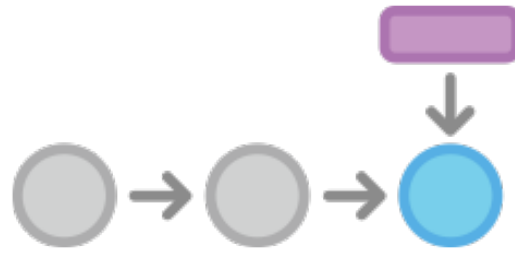
并不可以!

- 只执行 `git merge` 时，如果可以 `fast-forward`，则 `fast-forward`，否则进行 `non fast-forward merge`。
- 在 `non fast-forward merge` 过程中，会尝试合并修改，如果发生不能自动合并的冲突，则需要手动解决冲突。

基本的 Git 使用

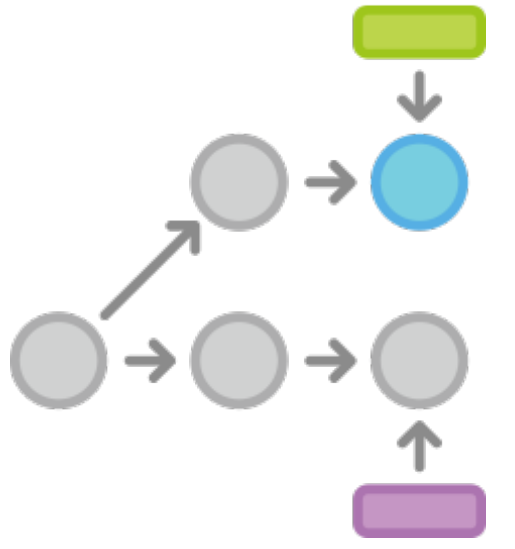
Git 分支

Git branch



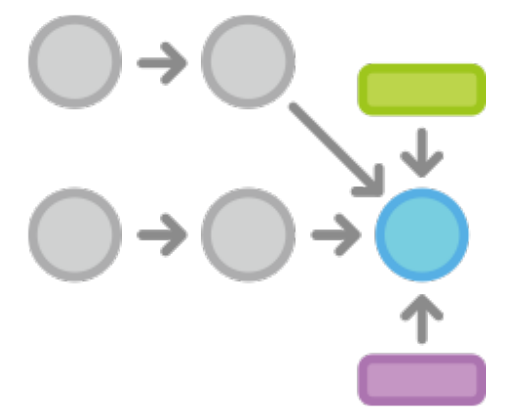
使用当前环境创建一个当前点
(含 Log) 完全一致的分支

Git checkout



切换到指定的版本 (或通过分支指定的版本)

Git merge



完成合并: 将一个分支的修改应用到当前分支

远程仓库交互



Git remote



Git fetch



Git pull



Git push

拉取和推送

01

拉取更新不更新工作目录

```
git fetch
```

02

拉取更新并更新工作目录

```
git pull
```

03

从远端 origin 的指定 master 分支进行拉取

```
git pull origin master
```

04

推动更新

```
git push
```

05

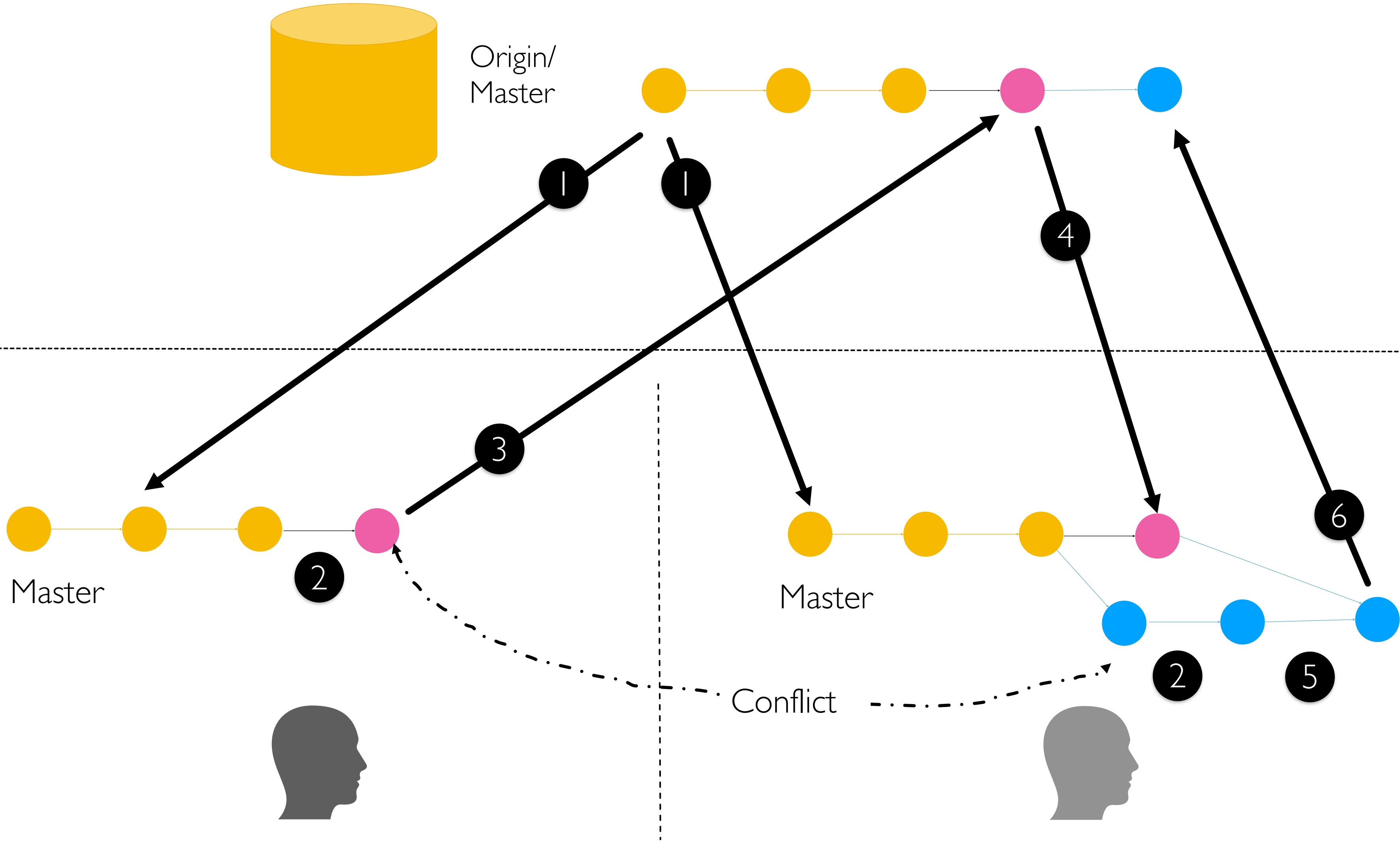
向远端 origin 的指定 branchname 分支进行推送

```
git push origin master
```


场景：中心化工 workflow

中心仓库

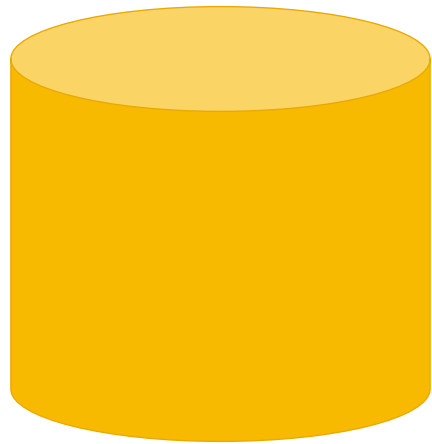
本地仓库



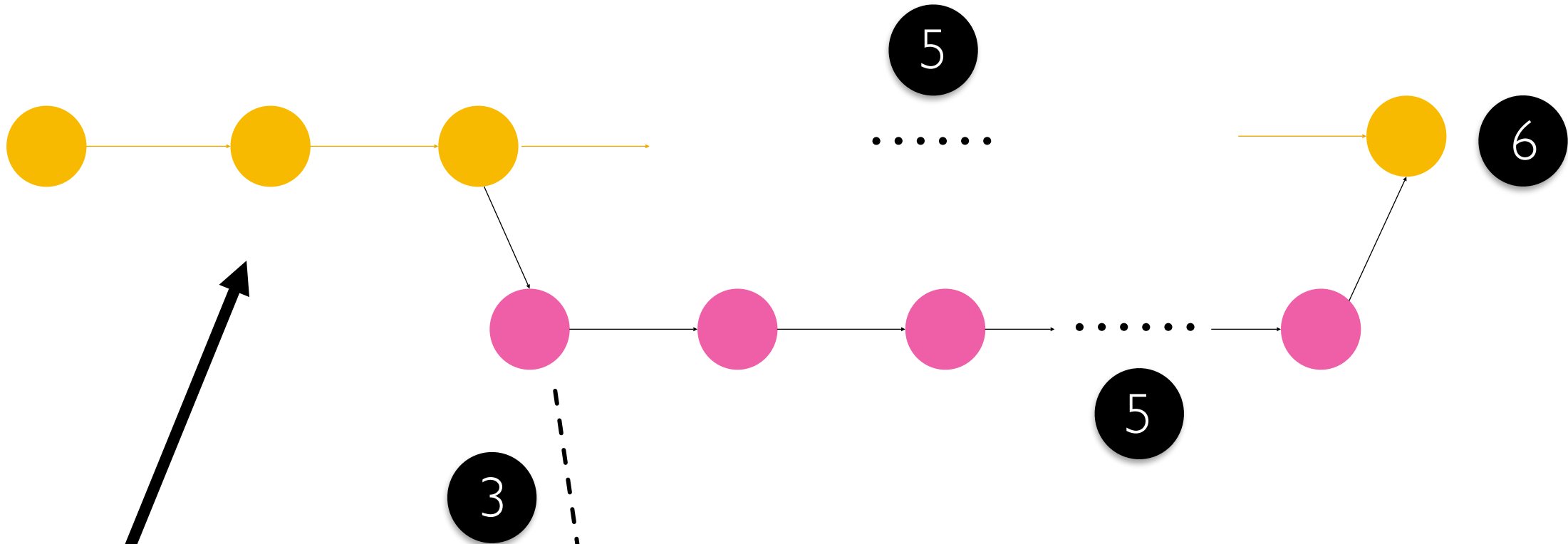
- 1 Git clone
- 2 Git add
- 3 Git commit
- 4 Git pull
- 5 Merge conflicts
- 6 Git push

场景：功能分支 workflow

中心仓库



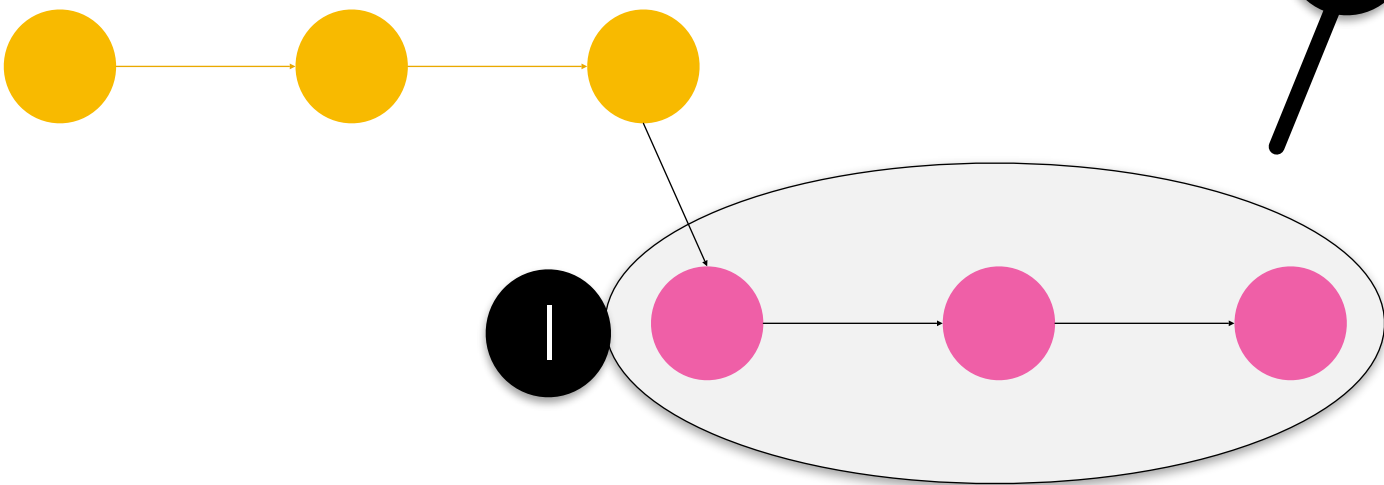
Origin/
Master



- 1 Git checkout
- 2 Git add/commit
- 3 Notify
- 4 Pull
- 5 Revision history
- 6 Merge

本地仓库

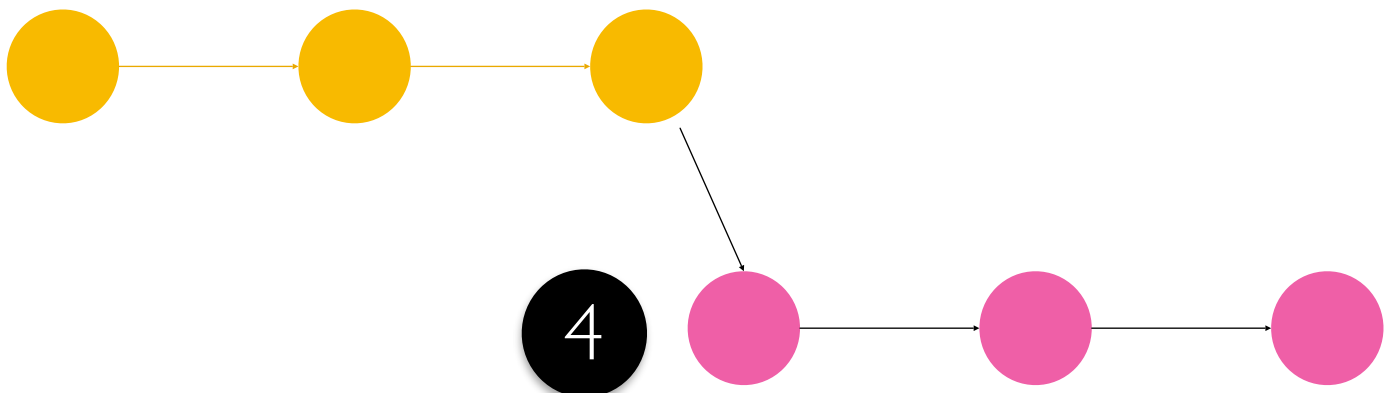
Master



A new branch



Master



- <https://github.com/tj/git-extras>
- <https://speakerdeck.com/halypg/git-remote-branch-comsamples>
- <https://www.atlassian.com/git/tutorials/using-branches>
- <https://git-scm.com/book/en/v2/Git-Branching-Branched-in-a-Nutshell>
- <https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging>
- <https://speakerdeck.com/matthetmo/understanding-git>
- <https://book.douban.com/subject/26107548/>