



SWEN90016

软件流程 & 项目管理

配置
和
版本控制



熟悉
持续集成

&

通用电气

<https://github.com/microsoft>



版本控制系统持续集成 (CI)

(不要与CI-配置项-讲座混淆)

- 自动将多个贡献者的代码更改集成到一个
 单个软件项目。
- DevOps 最佳实践, 开发人员将代码更改合并到一个中央存储库中
 构建和测试然后运行。
- 自动化工具用于在集成之前断言新代码的正确性。

- 版本控制系统谁，什么，哪里，
什么时候的代码

- 由 Linus Torvalds 制作!

著名大师 - 两次改造技术为 Linux 操作系统、Android 和
Chrome 操作系统创建内核

创建了 Git，源代码管理系统



遇见小企鹅，稍微超重的企鹅



- 什么：在版本控制下识别人工制品
- WHERE：存储在 REPO 中的项目：版本控制存储库
- 谁：用您的名字标记项目（集合保持一致）
- WHEN：用日期标记项目（存在版本）
- 为什么：高质量代码共享、备份、回滚、并行分支

“为什么” 记录在您对提交的评论中！

- 分布式（每个人都有自己的本地代码库！）
- 开源（人人都喜欢开源代码）
- 防弹

GitHub hit by Massive DDoS Attack From China

Friday, March 27, 2015 Mohit Kumar

[Tweet](#) [G+ Share](#) [Share](#) 29 [in Share](#) [f Share](#) [Share](#)





SWEN90016

1.git初始化

初始化 本地存储库调用 **掌握**

1.git 添加 文档名称

追踪 对此文件内容所做的更改

1.git commit -m “信息”

节省 此文件在本地存储库中

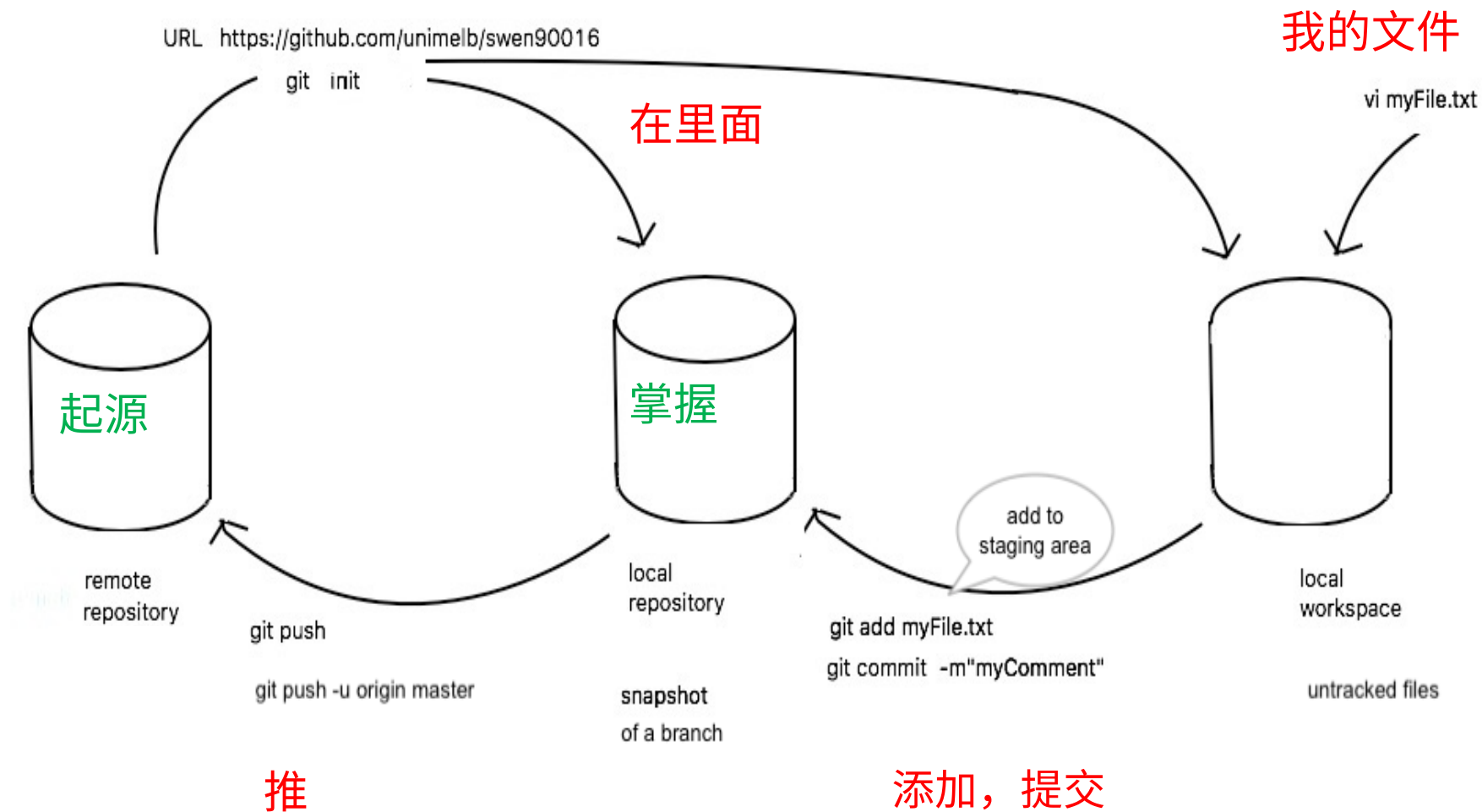
1.git推送

节省 这个集合

- 调用远程存储库 **起源**
- 从本地存储库调用 **掌握**



REPOURNS





无需编码

要完成本教程，您需要一个<https://github.com/microsoft> 和互联网接入。您不需要知道如何编码、使用命令行或安装 Git（版本控制软件 GitHub 是基于它构建的）。

提示： 在单独的浏览器窗口（或选项卡）中打开本指南，以便您在完成教程中的步骤时可以看到它。

步骤 1. 创建存储库

一种 **存储库** 通常用于组织单个项目。存储库可以包含文件夹和文件、图像、视频、电子表格和数据集——您的项目需要的任何内容。

GitHub 可以让您在创建新存储库的同时轻松添加一个。它还提供其他常用选项，例如许可证文件。

您的 hello-world 存储库可以是您存储想法、资源甚至与他人共享和讨论事物的地方。



创建新存储库

在右上角，您的头像或标识旁边，单击 +，然后选择 **新仓库**. 命名您的存储库 *你好，世界*。写一个简短的描述。

选择 **使用 README** 初始化此存储库。

Owner: hubot / Repository name: hello-world

PUBLIC

Great repository names are short and memorable. Need inspiration? How about [petulant-shame](#).

Description (optional): Just another repository

☒ Public
Anyone can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

☒ Initialize this repository with a README
This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: None Add a license: None

Create repository



步骤 2. 创建一个分支

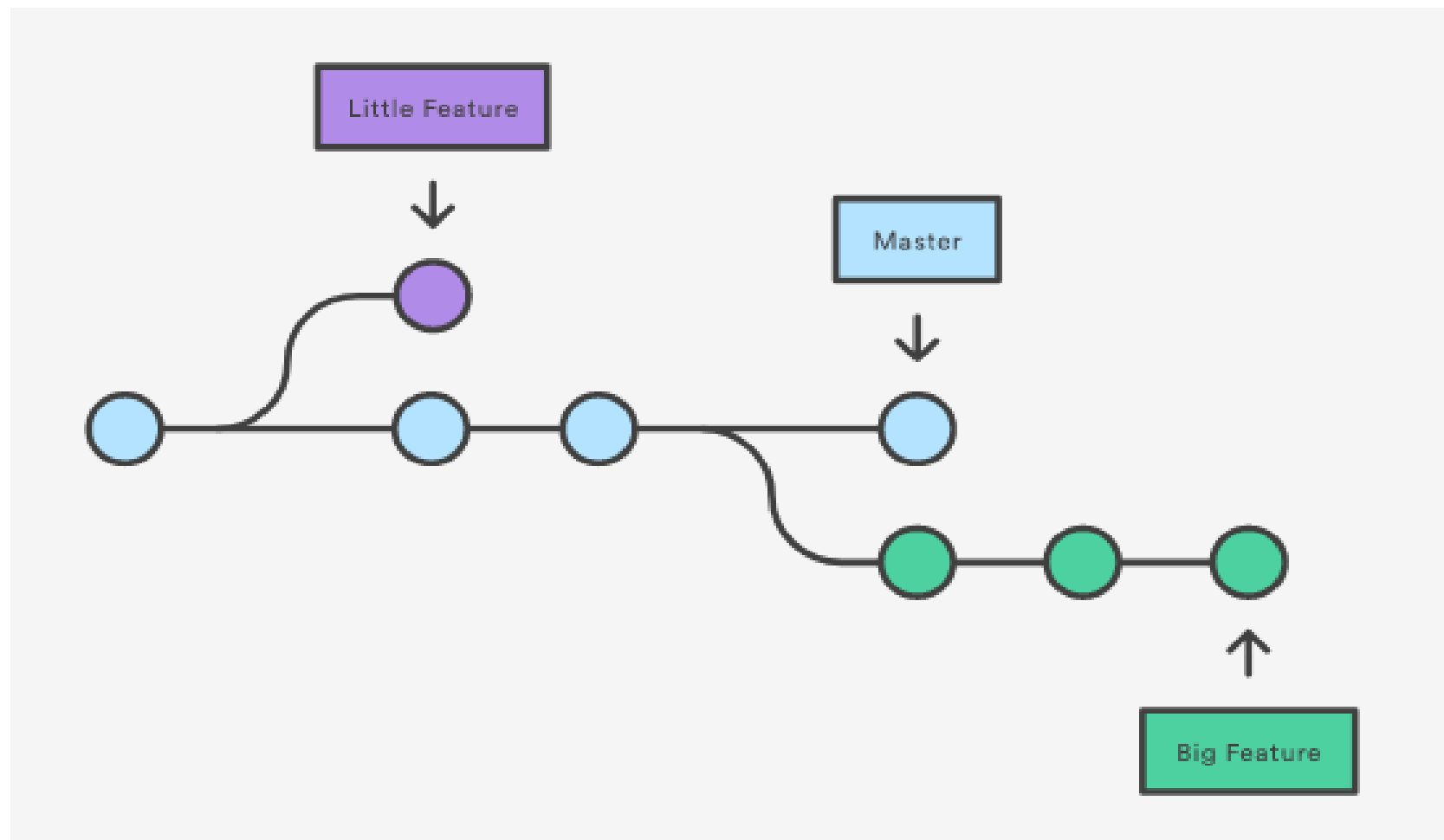
分枝 是一种同时处理不同版本存储库的方式。

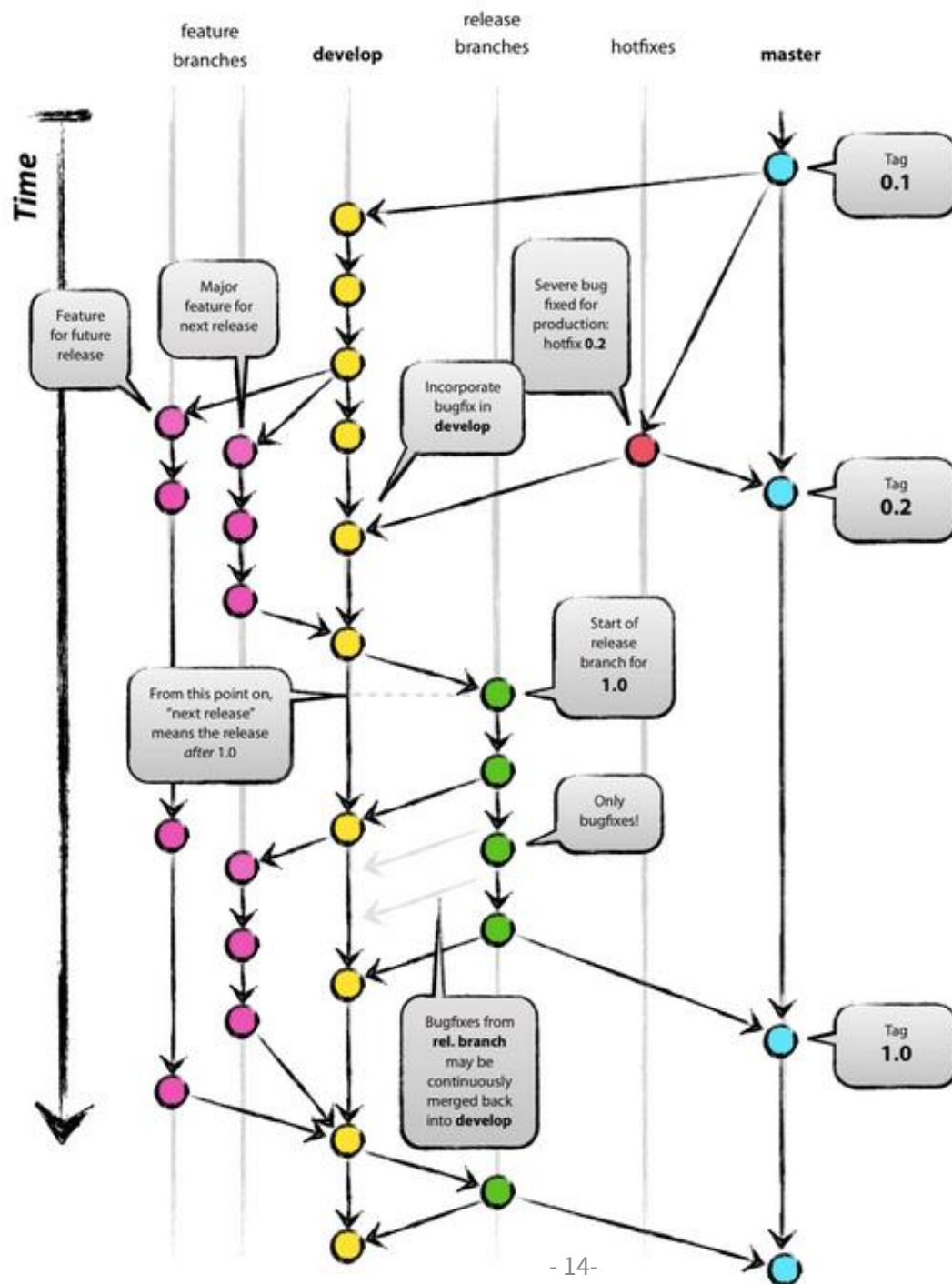
默认情况下，您的存储库有一个名为 master 的分支，它被认为是最终分支。在将它们提交给 master 之前，我们使用分支进行试验和编辑。

当您从 master 分支创建一个分支时，您正在制作 master 的副本或快照，就像当时的情况一样。如果其他人在您的分支上工作时对主分支进行了更改，您可以拉入这些更新。



WELBOURNE





步骤 2. 创建一个分支

此图显示：

主分支

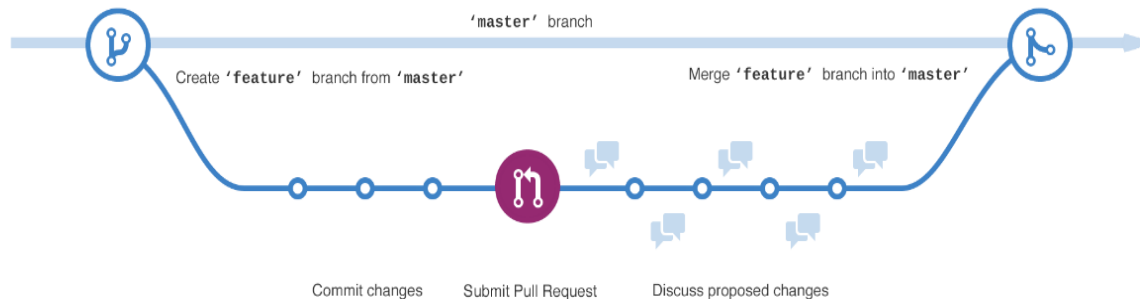
一个名为 feature 的新分支（因为我们正在这个分支上做“功能工作”）

该功能在合并到 master 之前所经历的旅程

您是否曾经保存过不同版本的文件？就像是：

故事.txt

故事乔编辑.txt

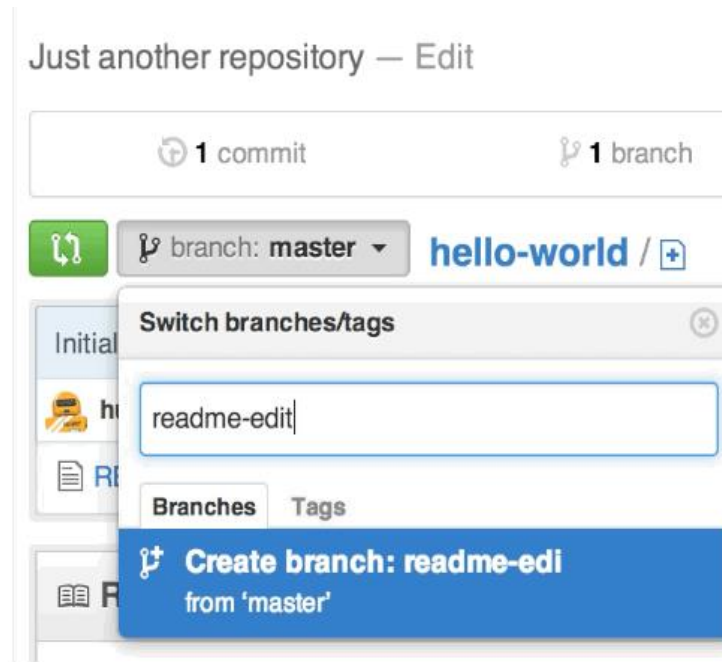




创建新分支

1. 转到您的新存储库 hello-world。
2. 单击文件列表顶部的下拉菜单 **分支：主**。
3. 在新分支文本框中键入分支名称 readme-edits。
4. 选择蓝色 **创建分支** 框或按键盘上的 “Enter” 。
5. 现在你有两个分支，master 和 readme-edits。它们看起来完全一样，但不会持续太久！接下来，我们将更改添加到新分支。

story-joe-edit-reviewed.txt



Now you have two branches, `master` and `readme-edits`. They look exactly the same, but not for long! Next we'll add our changes to the new branch.

步骤 3. 进行并提交更改

太棒了！现在，您在代码视图中 *自述文件编辑* 分支，这是一个副本 *掌握*。让我们做一些编辑。

在 GitHub 上，保存的更改被称为 *提交*。每个提交都有一个关联的 *提交消息*，这是解释为什么进行特定更改的描述。提交消息会记录您更改的历史记录，因此其他贡献者可以了解您所做的事情以及原因。

步骤 3. 进行并提交更改

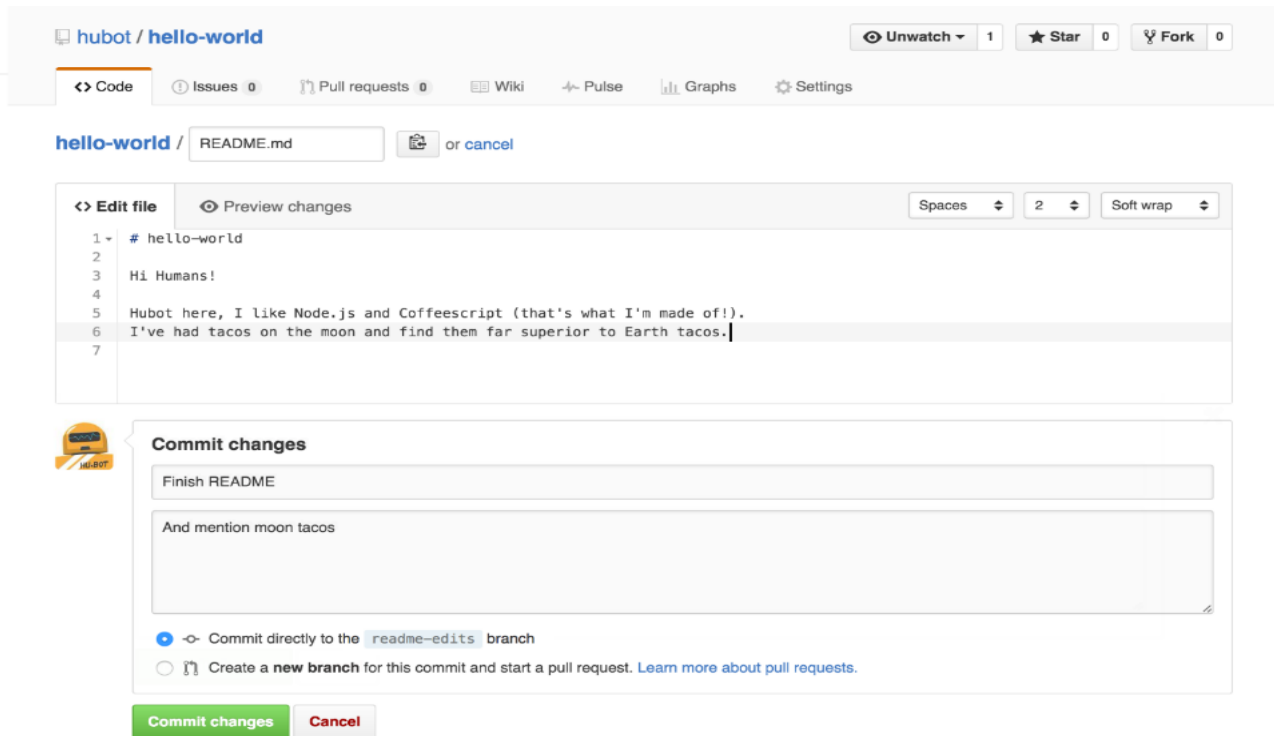
单击 README.md 文件。

单击文件视图右上角的铅笔图标进行编辑。

在编辑器中，写一些关于你自己的信息。

编写描述更改的提交消息。点击**提交更改** 按钮。

这些更改将仅对您的 README 文件进行 *自述文件编辑* 分支，
所以现在这个分支包含的内容不同于 *掌握*。



hubot / hello-world


Unwatch 1 Star 0 Fork 0

<> Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settings

hello-world / README.md or cancel

<> Edit file Preview changes Spaces 2 Soft wrap

```
1 # hello-world
2
3 Hi Humans!
4
5 Hubot here, I like Node.js and Coffeescript (that's what I'm made of!).
6 I've had tacos on the moon and find them far superior to Earth tacos.
7
```

 **Commit changes**

Finish README

And mention moon tacos

☒ Commit directly to the `readme-edits` branch

☐ Create a **new branch** for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes Cancel

步骤 4. 打开拉取请求

拉取请求是 GitHub 上协作的核心。当你打开一个 *拉取请求*，您正在提议您的更改并要求某人审查并提取您的贡献并将它们合并到他们的分支中。拉取请求显示 *差异* 或来自两个分支的内容的差异。变化、增加和减少以绿色和红色显示。

提交后，即使在代码完成之前，您也可以打开拉取请求并开始讨论。

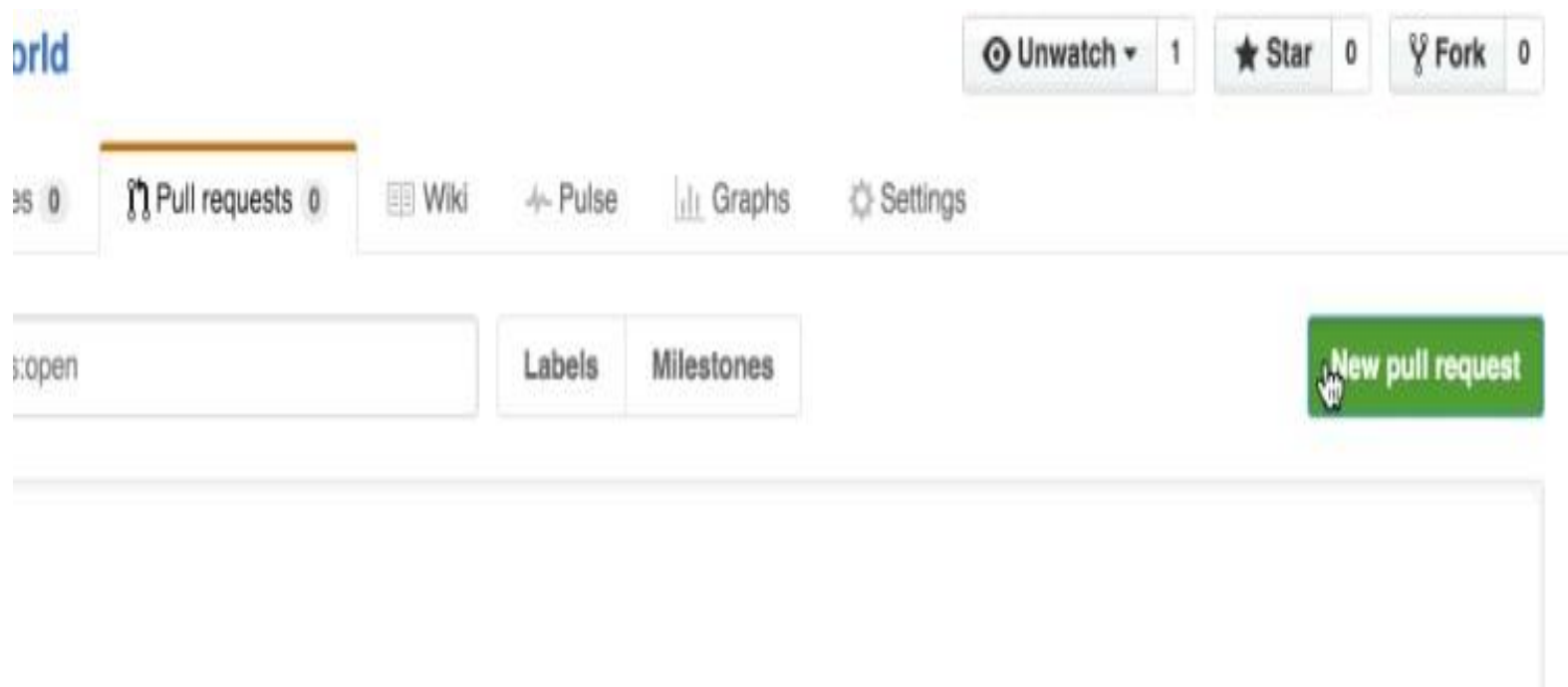
通过使用 GitHub 的 [@提及系统](#) 在您的拉取请求消息中，您可以询问特定人员或团队的反馈，无论他们是在大厅内还是在 10 个时区之外。

您甚至可以在自己的存储库中打开拉取请求并自行合并。这是在处理大型项目之前学习 GitHub 流程的好方法。



打开对自述文件更改的拉取请求

单击**拉取请求** 选项卡，然后从拉取请求页面，单击绿色**新的拉取请求** 按钮。





打开对自述文件更改的拉取请求

在里面 **示例比较** 框中，选择您创建的分支，readme edits，与 master（原始）进行比较。

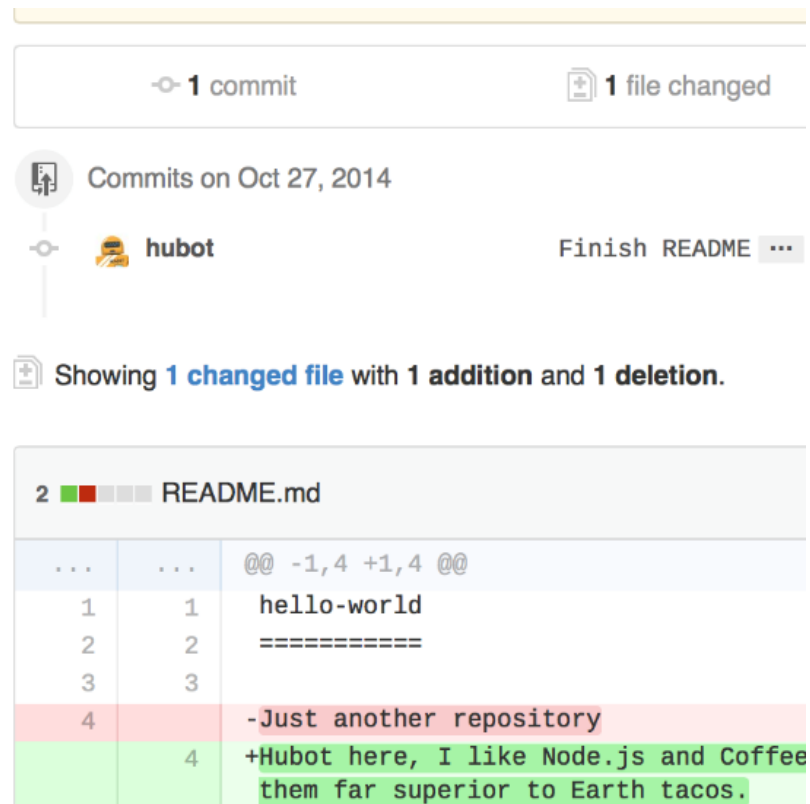
EXAMPLE COMPARISONS

 **readme-edits**

4 minutes ago

打开对自述文件更改的拉取请求

在“比较”页面上查看差异中的更改，确保它们是您要提交的内容。



1 commit 1 file changed

Commits on Oct 27, 2014

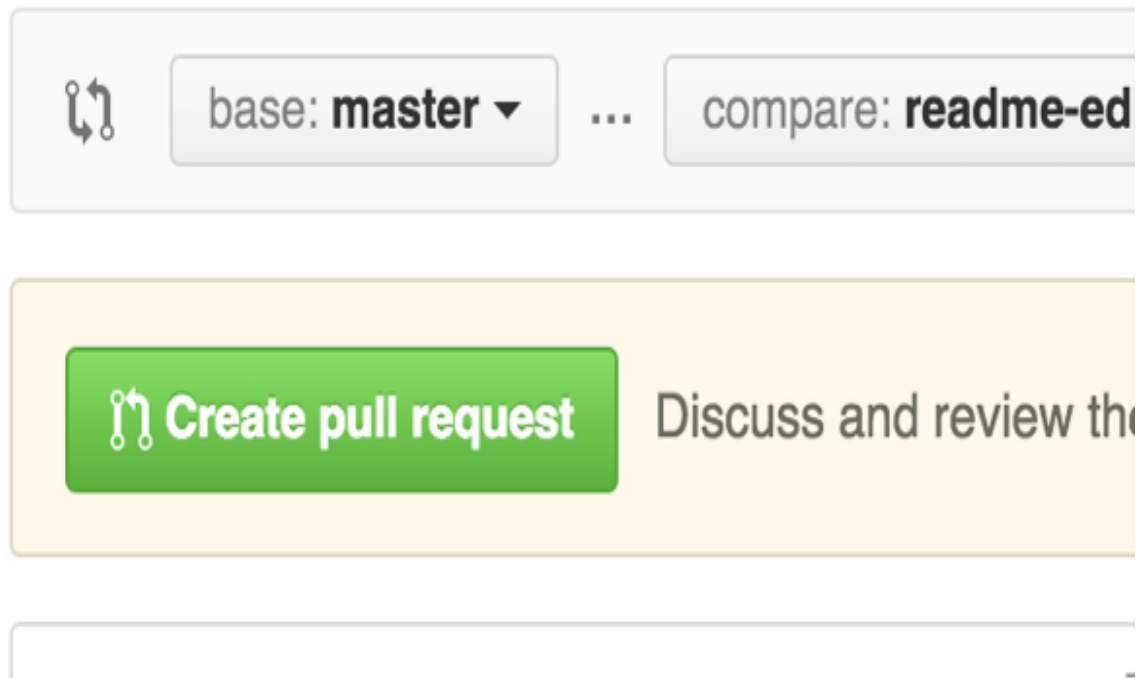
hubot Finish README ...

Showing 1 changed file with 1 addition and 1 deletion.

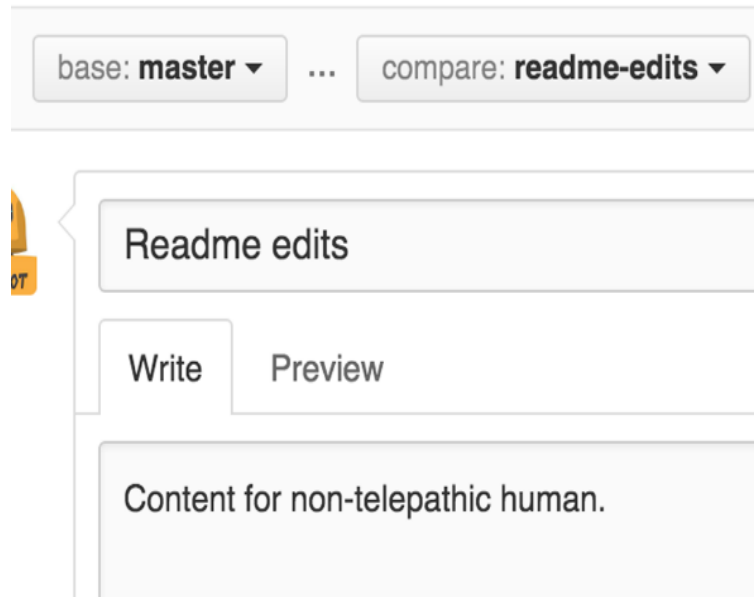
2 README.md		
...	...	@@ -1,4 +1,4 @@
1	1	hello-world
2	2	=====
3	3	
4		-Just another repository
	4	+Hubot here, I like Node.js and Coffee them far superior to Earth tacos.

打开对自述文件更改的拉取请求

当您对这些是您要提交的更改感到满意时，单击大绿色 **创建拉取请求** 按钮。



打开对自述文件更改的拉取请求
为您的拉取请求命名并简要描述您的更改
完成消息后，单击 **创建拉取请求**！



The screenshot shows the GitHub Pull Request interface. At the top, there are two dropdown menus: 'base: master' and 'compare: readme-edits'. Below these, there is a section for the pull request title and description. The title is 'Readme edits'. Below the title, there are two tabs: 'Write' and 'Preview'. The 'Write' tab is active, showing the text 'Content for non-telepathic human.'.

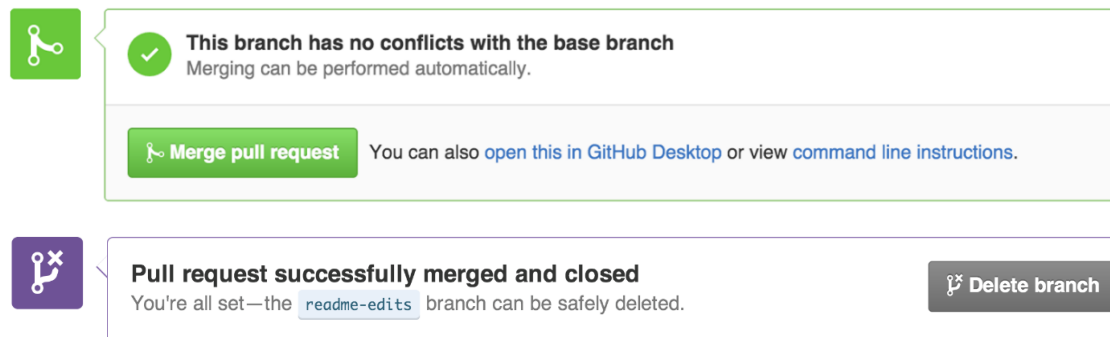
第 5 步。合并您的拉取请求

在这最后一步中，是时候将您的更改整合在一起了——将您的 `readme-edits` 分支合并到 `master` 分支中。

1. 点击绿色 **合并拉取请求** 按钮将更改合并到母版。

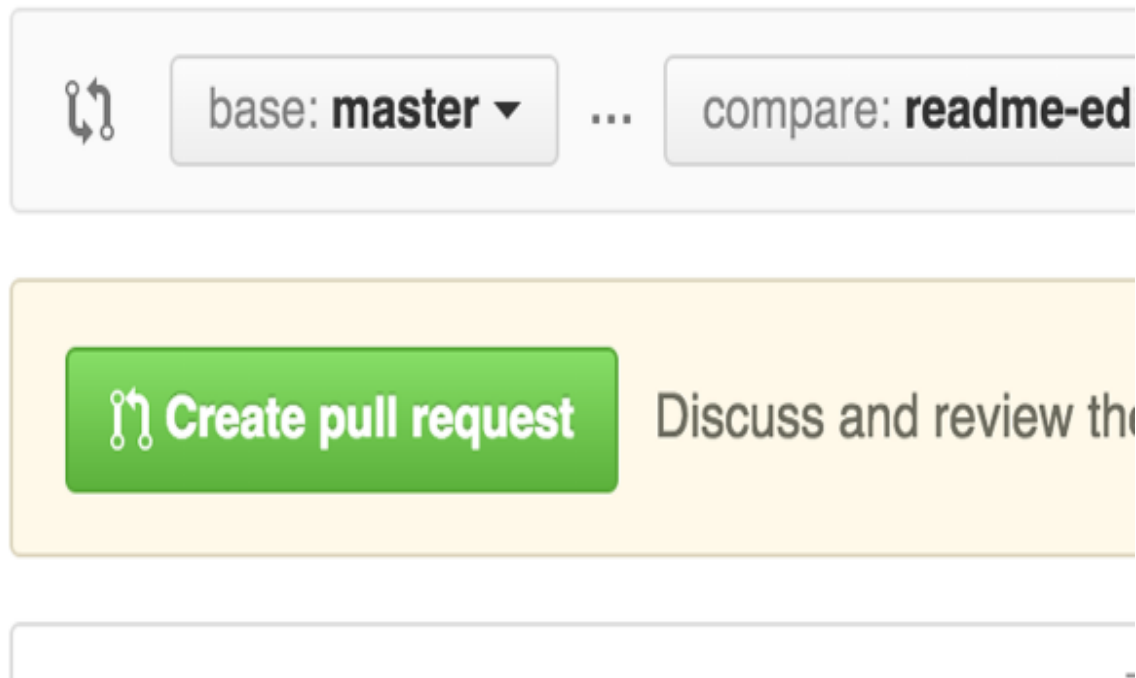
2. 点击 **确认合并**。

3. 继续删除分支，因为它的更改已经合并，使用 **删除分支** 紫色框中的按钮。



打开对自述文件更改的拉取请求

当您对这些是您要提交的更改感到满意时，单击大绿色 **创建拉取请求** 按钮。





在您各自的第 12 周研讨会上到期

期待什么：

- 所有团队成员都需要在车间
- 提名一名团队成员分享他们的屏幕并展示网站
- 您有 5-7 分钟的时间进行演讲
- 展示您的网站/应用

完毕!

参考:

<https://www.atlassian.com/git/教程>

<https://www.atlassian.com/continuous-delivery/continuous>

集成#:~:text=Continuous%20integration%20(CI)%20is%20the,builds%20and%20tests%20then%20run。

<https://git-scm.com/>

http://thehackernews.com/2015/03/github-hit-by-massive-ddos-attack-from_27.html