**The Hello World Project 官方教程文档**

The **Hello World** project 是一个在编程界历史悠久的传统. 当你学习一个新的编程知识的时候，它是带你入门的一个很好的练习。我们一起开始学习GitHub!

**在这里你会学到：**

* 如何创建和使用一个代码版本库
* 如何启动和管理一个新的代码分支
* 如何修改一个文件并把它们提交到GitHub 上
* 如何打开一个代码合并请求

**什么是 GitHub?**

GitHub 是一个代码版本控制和协作的平台。它能够使你和他人无论在什么地方都能够一起去开发一个项目。这个辅助教程教你一些例如代码库、分支、托管、处理请求等。你将会创建你自己的’Hello World’ 代码库，并且学习GitHub’s 处理请求的工作流，它是一个非常流行的创建和查看代码的工具。

**不需要编写代码**

要完成这个教程，你只需要能够上网以及一个GitHub账号，你无需知道如何编程、不需要知道命令行代码，或者安装Git （GitHub建立在这个版本控制软件之上).

**Step 1. 建立代码库(Repository)**

一个代码库只能用于组织管理一个项目。代码库可以包含文件夹和文件、图片、视频、电子表格、数据集——你项目需要的任何东西。在你创建新的代码库的时候你就可以把这些东西添加进去，GitHub 是这一切变得特别容易。 *它也提供了其他公共的选项，比如说项目协议文件。*

你的 hello-world 代码库可以存放你的创意、资源，甚至能够与他人分享讨论一些事情。

**建立一个新的代码库**

1. 在你主页界面的右上角，点击’+’ ，选择’New repository’。
2. 把你的代码库命名为 hello-world。
3. 写一个简单的描述。
4. 选择“Initialize this repository with a README”（代码库初始化说明）



点击 **Create repository**. :tada:

**Step 2. 创建一个分支(Branching)**

**Branching**(分支)是同一个代码库在同一时间操作不同版本的工作方式。

你的版本库会默认有一个最终版本，这个最终版本名称叫做 master 在提交给 master之前，我们利用这个分支来实验和编辑

当你创建一个master 的分支的时候，你是对 master 的一个时点 进行的一个复制或快照。如果当你在修改你的分支的时候，别人也在 master 的一个分支上修改，你可以获取到他们的修改内容。

**如图所示：**

* Master分支
* 名叫 feature 的分支(因为我们正在在这个主分支上做‘重要的功能工作’)
* feature分支在最终被合并到master分支的过程！



你是否把同一个文件内容保存到几个版本中？ 像以下这些：

* story.txt
* story-joe-edit.txt
* story-joe-edit-reviewed.txt

GitHub 托管代码库要实现的目标就类似这些。

在GitHub 上，我们的开发工程师、编辑、设计师在master分支（生产系统）以外使用分支来修复漏洞或新功能开发。当一个漏洞或者新功能测试没有问题后就会把分支合并到 master.分支中。

**建立一个新的分支**

1. 前往你的代码库 hello-world。
2. 点击右边中间“**branch: master**”下拉菜单。
3. 在分支输入框中输入 readme-edits。
4. 填写完成后选择Create branch **Create branch** ,然后回车。

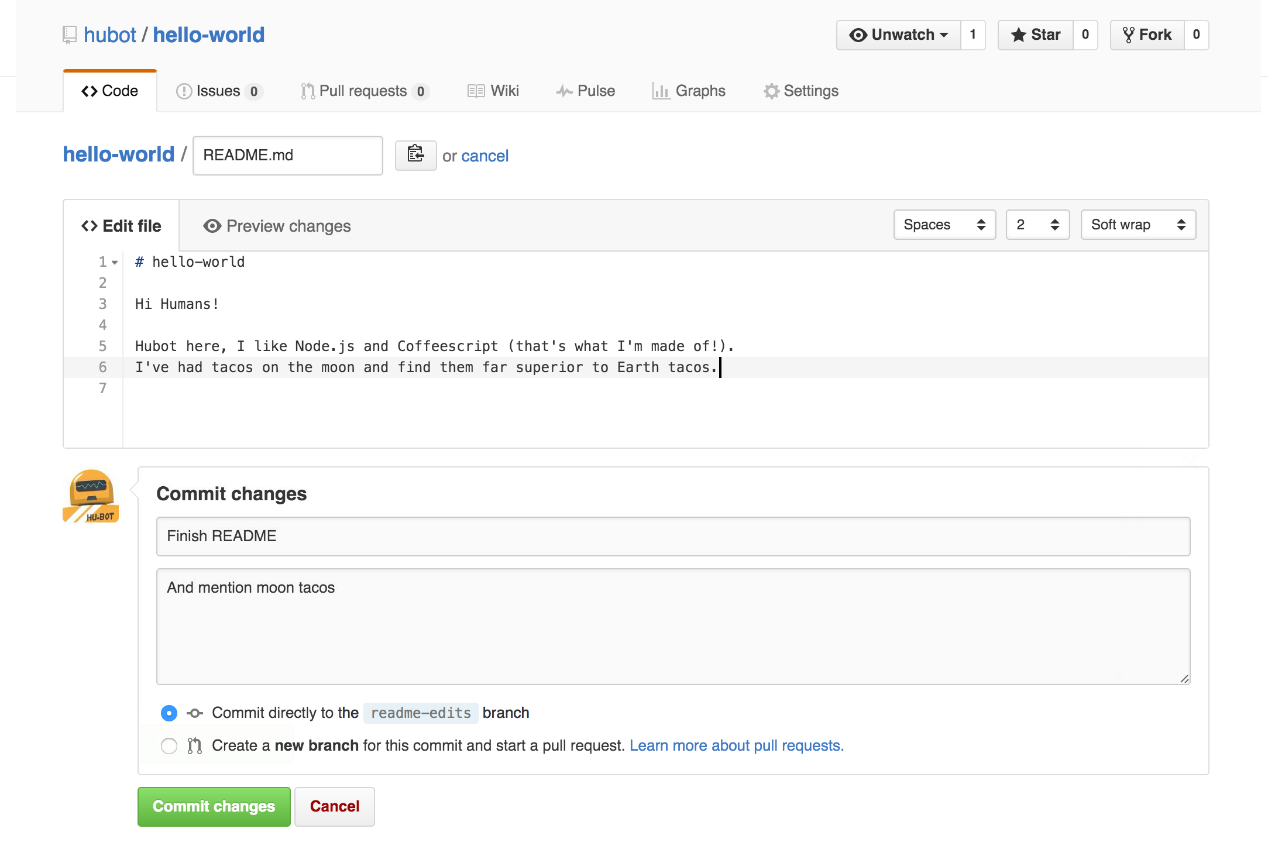
现在你有两个分支，master 和readme-edits. 它们现在看起来完全一样，分支readme-edits修改后就不一样了。接下来我们会修改分支readme-edits。

**Step 3. 修改代码并提交**

好极了!现在，你是在你创建的 readme-edits 分支的代码页面上，它是master的一个分支，让我们做一些修改。

在GitHub上，保存修改叫做提交。每个提交都会有一个关联的提交信息（由提交者和版本控制系统共同完成），这个关联的提交信息用于解释这一块代码为什么要修改。这个关联的提交信息能够捕捉到当前版本和历史版本之间的变化，这样其它的代码贡献者才能够知道和理解你修改了什么以及为什么要这样修改。

**修改分支代码库并提交**

1. 点击 README.md 文件.
2. 点击 ‘README.md’ 文件右上面的铅笔图标进行编辑
3. 在编辑界面，写一些关于你自己的一些信息和这个项目的一些信息。
4. d.写一些你修改的地方以及为什么修改。
5. e.点击提交修改按钮。

以上修改只是修改了readme-edits 分支中的 ‘README’ 文件，现在你修改后的这个分支就于master 分支有所不同了。

**Step 4.发起修改提交合并请求**

修改的很好！ 现在你已经在master的一个分支上做好了修改，你可以针对你的修改向项目内的所有人发起修改提交合并请求。

Pull Requests 是 GitHub 上各个贡献者协作的核心。当你打开一个Pull Request的时候，你是希望别人能够针对你的修改作出回复，同步你修改或新增的代码，最好能够把这些代码合并到他们的分支中。Pull requests 能够展示不同分支代码之间的区别。修改和新增的代码用绿色标出，删除的代码用红色标出。

一旦你提交了代码，你就相当于打开了一个Pull Request 供大家讨论，直到代码结束（合并到master中，或者最终没有被采纳）。

通过使用 GitHub’s 的@metion system 通知栏，你可以向特定的人或团队发出反馈请求，无论他们就在你的大厅下面或者十个街区以外。

你也可以在你自己的代码库中打开Pull Requests 并把一些修改的代码合并到你的代码中。 在开发稍微大的项目之前，这不失为一个了解GitHub 工作流的好方法。

**为修改的README打开一个Pull Request**

*Click on the image for a larger version*

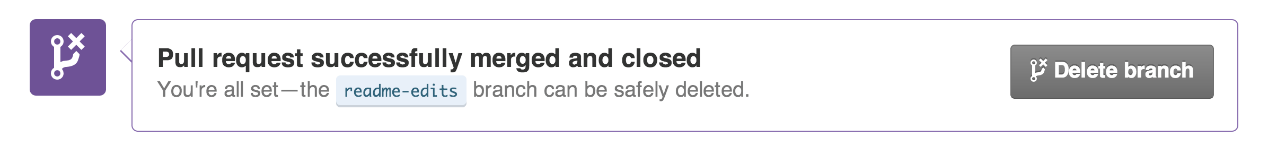
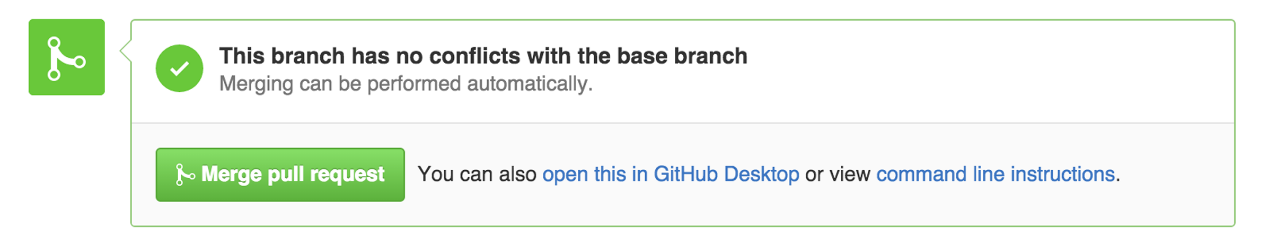
| **Step** | **Screenshot** |
| --- | --- |
| 点击 **Pull Request** 菜单，进入**Pull Request** 页面，单击**New pull request** 按钮 | [pr-tab](https://guides.github.com/activities/hello-world/pr-tab.gif) |
| 在**Example Comparisons** 列表中选择你修改的分支，并与原始的master 项目进行比较 | [branch](https://guides.github.com/activities/hello-world/pick-branch.png) |
| 在比较界面仔细检查你修改的地方与**master** 的不同之处，确保那些修改的地方就是你想提交的 | [diff](https://guides.github.com/activities/hello-world/diff.png) | |
| 当你对你想提交的代码感觉很满意的时候，单击那个绿色的**Create Pull Request** 大按钮 | [create-pull](https://guides.github.com/activities/hello-world/create-pr.png) | |
| 给你的 ‘pull request’ 一个标题和简短描述，用于说明你修改的内容 | [pr-form](https://guides.github.com/activities/hello-world/pr-form.png) | |

当这一切填好以后，单击**Create pull request**!

**小贴士:你可以把emoji 表情，也可以把图片或者gifs 动画图片拖放到评论的地方和Pull Requests上面.**

**Step 5.合并你的Pull Request**

最后，要把你修改的代码即 readme-edits 分支和 master 进行合并。

1. 点击 绿色的**Merge pull request** 按钮把你修改的代码合并到主分支 master中
2. 点击**Confirm merge**.
3. 去前一页删除分支，如果你修改的代码已经合并到主分支中，**Delete branch**按钮会在紫色框中显示

**恭喜!**

看完这个教程后，你就学会如何在GitHub上创建一个项目，如何提出合并请求。 :tada: :octocat: :zap:

以下是你在这个教程中完成的:

* 创建一个开源的资源库
* 开始管理一个新的版本分支
* 修改一些代码文件并把它们提交到GitHub 上
* 打开和合并一个Pull Request

浏览一下你的GitHub 文件，你会看到你的巨大贡献。

如果你想学习更多关于 Pull Requests 的伟大力量，我们建议你阅读一下GitHub 的工作流程指导。你也可以探索一下 GitHub 的开源项目列表，并参与到其中一个开源项目中 :octocat:

**小贴士：去我们另一个关于GitHub 的指南 和 YouTube 频道 ,进一步了解GitHub。**

Last updated April 7, 2016