产品

主页

工作

博客

开源

HubSpot产品和工程博客

Git和GitHub入门指南(教程)

2015年10月1日/梅根·纳尔逊

分享

今年8月，我们在HubSpot举办了一次女性代码会议，并为初学者举办了一次关于使用git和GitHub的研讨会。我首先浏览了一个关于git的基础和背景的幻灯片演示，然后我们分成小组来学习我创建的一个教程，以模拟一个大型的协作项目。活动结束后，我们得到了反馈，认为这是一个有用的动手介绍。因此，如果您也是git的新手，请按照下面的步骤对代码库进行修改，打开一个拉请求，并将代码合并到主分支中。任何重要的git和GitHub术语都以粗体显示，链接到官方git参考资料。

步骤0:安装git并创建一个GitHub帐户

你要做的前两件事是安装git和创建一个免费的GitHub帐户。

请按照此处的说明安装git(如果尚未安装)。请注意，对于本教程，我们将只在命令行上使用git。虽然有一些很棒的git图形用户界面，但是我认为首先使用git特定的命令来学习git，然后在您对该命令更熟悉的时候尝试git图形用户界面会更容易。

完成后，在这里创建一个GitHub帐户。(公共存储库的帐户是免费的，而私有存储库是收费的。)

步骤1:创建一个本地git存储库

使用git在本地机器上创建新项目时，首先要创建一个新的存储库(或者通常简称为“repo”)。

为了使用git，我们将使用终端。如果您对终端和基本命令没有太多经验，请查看本教程(尤其是“浏览文件系统”和“四处移动”部分)。

首先，打开一个终端，并使用cd(更改目录)命令移动到您希望在本地计算机上放置项目的位置。例如，如果您的桌面上有一个“项目”文件夹，您会执行以下操作:

步骤2:向回购中添加一个新的文件

使用您喜欢的任何文本编辑器或运行触摸命令，继续向项目添加新文件。

一旦您在包含git repo的文件夹中添加或修改了文件，git将会注意到repo中已经进行了更改。但是，git不会正式跟踪该文件(也就是说，将它放入提交中——我们接下来将详细讨论提交)，除非您明确告诉它。

创建新文件后，可以使用git status命令查看git知道存在的文件。

在分支机构主管

初次提交

未跟踪的文件：

（使用“ git add <文件> ...”来包含将要提交的内容）

没有添加任何内容提交但存在未跟踪的文件（使用“ git add”进行跟踪）

查看GitHub托管的raw gitstatus.md

这基本上是说：“嘿，我们注意到您创建了一个名为mnelson.txt的新文件，但是除非您使用'git add'命令，否则我们将不会对其进行任何处理。”

插曲：过渡环境，提交和您

初学git时最容易混淆的部分之一是暂存环境的概念及其与提交的关系。

提交是自上次提交以来您更改了哪些文件的记录。本质上，您可以对存储库进行更改（例如，添加文件或修改文件），然后告诉git将这些文件放入提交中。

提交构成了项目的本质，并允许您随时回到项目。

那么，如何告诉git提交哪些文件呢？这就是暂存环境或索引的引入。如步骤2所示，当您对存储库进行更改时，git会注意到文件已更改，但不会对其进行任何操作（例如在提交中添加文件）。

要将文件添加到提交中，首先需要将其添加到登录环境中。为此，您可以使用git add <filename>命令（请参见下面的步骤3）

使用git add命令将所需的所有文件添加到暂存环境后，您可以使用git commit命令告诉git将它们打包到提交中。

注意：暂存环境，也称为“暂存”，是此的新的首选术语，但您也可以将其称为“索引”。

步骤3：将文件添加到暂存环境

使用git add命令将文件添加到暂存环境。

如果重新运行git status命令，则会看到git已将文件添加到暂存环境（请注意“要提交的更改”）。

mnelson：myproject mnelson $ git状态

在分支机构主管

初次提交

所做更改：

（使用“ git rm --cached <文件> ...”取消登台）

查看GitHub托管的原始addtostaging.md

重申一下，该文件尚未添加到提交中，但即将添加到提交中。

步骤4:创建提交

是时候创建您的第一个承诺了！

运行命令“关于提交的消息”

“这是我第一次提交！”

[Master(根提交)b345d9a]这是我第一次提交！

1个文件已更改，1个插入(+)

创建模式100644 mnelson.txt

查看GitHub在托管的原始提交. md

提交结束时的消息应该与提交包含的内容相关——也许是一个新特性，也许是一个bug修复，也许只是一个打字错误。不要放像“asdfadsf”或“foobar”这样的信息。这让其他看到你的承诺的人很难过。非常非常悲伤。

步骤5:创建一个新分支

既然您已经做出了新的承诺，让我们尝试一些更高级的东西。

假设您想要创建一个新功能，但是在开发该功能时担心对主项目进行更改。这就是git分支的作用。

分支允许您在项目的“状态”之间来回移动。例如，如果你想给你的网站添加一个新的页面，你可以只为这个页面创建一个新的分支，而不影响项目的主要部分。完成页面后，您可以将分支中的更改合并到主分支中。当您创建一个新的分支时，Git会跟踪哪个提交了您的分支，因此它知道所有文件背后的历史。

假设您在主分支上，想要创建一个新的分支来开发您的网页。下面是你要做的:运行git checkout -b &lt;我的分支机构名git。这个命令将自动创建一个新的分支，然后在它上面“检查你”，这意味着git将把你从主分支移到那个分支。

运行上述命令后，您可以使用git branch命令来确认您的分支已创建:git分支

掌握

\*我的新分支

查看GitHub与host的原始gitbranch.md

旁边带星号的分支名称表示您在给定时间指向哪个分支。

现在，如果您切换回主分支并进行更多提交，您的新分支将不会看到任何这些更改，直到您将这些更改合并到新分支。

步骤6:在GitHub上创建一个新的存储库

如果你只想在本地跟踪你的代码，你不需要使用GitHub。但是如果你想和一个团队合作，你可以使用GitHub来协作修改项目的代码。

要在GitHub上创建新报告，请登录并转到GitHub主页。您应该会看到绿色的“+新建存储库”按钮:

点击该按钮后，GitHub将要求您命名您的回购并提供简短描述:

填写完信息后，按“创建存储库”按钮进行新的回购。

GitHub会询问您是否要从头开始创建新的存储库，或者是否要添加在本地创建的存储库。 在这种情况下，我们已经在本地创建了一个新的存储库，因此希望将其推送到GitHub上，请遵循“....或从命令行推送现有存储库”部分：

计数对象：3，完成。

写入对象：100%(3/3)，263字节，0字节/s，已完成。

总计3（增量0），重复使用0（增量0）

设置分支母版以跟踪来自原点的远程分支母版。

（您需要将第一个命令行中的URL更改为本节中GitHub列出的内容，因为GitHub用户名和存储库名称不同。）

步骤7：将分支推送到Gihub

现在，我们将分支中的提交推送到新的GitHub存储库。 这使其他人可以看到您所做的更改。 如果它们被存储库所有者批准，则可以将更改合并到master分支中。

要将更改推送到GitHub上的新分支上，您需要运行git push 追溯您的分支名称。 GitHub将在远程存储库上自动为您创建分支：

计数对象：3，完成。

增量压缩最多使用8个线程。

压缩对象：100%(2/2)，完成。

写入对象：100%(3/3)，313字节，0字节/s，已完成。

总计3（增量0），重复使用0（增量0）

您可能想知道上面命令中“origin”一词的含义。当您将远程存储库克隆到本地计算机时，git为您创建了一个别名，此别名都称为“origin”，它实质上是远程存储库URL的简写。 因此，要将更改推送到远程存储库，可以使用以下命令之一：

（如果您第一次在本地使用GitHub，则会提示您使用GitHub用户名和密码登录。）

刷新GitHub页面，您会看到说明，说您的名字的分支刚刚被推送到存储库中。您也可以单击“分支”链接以查看此处列出的分支。

单击上方屏幕快照中的绿色按钮，拉取请求！

步骤8：创建拉取请求（PR）

拉取请求（或PR）是一种提醒存储库所有者要更改其代码的方式。它允许他们检查代码，并确保代码看起来不错，然后再将更改放入master分支。

这是提交PR页之前的样子：

这是您提交PR请求后的样子：

您可能会在底部看到一个绿色的大按钮，上面显示“合并拉取请求”。单击这意味着您将把更改合并到master分支中。

请注意，此按钮并非总是绿色。在某些情况下，它将是灰色的，这意味着您面临合并冲突。这是当一个文件中的更改与另一个文件中的更改冲突并且git无法确定要使用哪个版本时。您必须手动输入并告诉git使用哪个版本。

有时，您将成为回购的共同所有者或唯一所有者，在这种情况下，您可能无需创建PR来合并您所作的更改。但是，创建一个分支仍然是一个好主意，这样您可以保留更完整的更新历史记录，并确保在进行更改时始终创建一个新分支。

第9步：合并PR

继续并单击绿色的“合并拉取请求”按钮。这会将您的更改合并到master分支中。

完成后，建议您删除分支（太多分支会变得凌乱），因此也请点击灰色的“删除分支”按钮。

您可以通过单击新存储库首页上的“提交”链接来仔细检查提交是否已合并。

这将显示该分支中所有提交的列表。您可以看到我刚刚合并的那个（合并请求2）。

您还可以在右侧看到提交的哈希码。哈希码是该特定提交的唯一标识符。这对于引用特定的提交以及撤消更改很有用（使用git revert <hash code number>命令回溯）。

步骤10：将GitHub上的更改返回到您的计算机

现在，GitHub上的仓库看起来与本地计算机上的仓库有所不同。例如，您在分支中进行的提交并合并到master分支中的提交在本地计算机上的master分支中不存在。

为了获得您或其他人已在GitHub上合并的最新更改，请使用git pull origin master命令（在master分支上工作时）。

mnelson:我的项目mnelson$ 上传至管理员

远程：计数对象：1，完成。

远程：总计1（增量0），重用0（增量0），重用包0

打开对象：100％（1/1），已完成。来自 https://github.com/cubeton/mynewrepository

\* 分支 管理员 -> FETCH\_HEAD

b345d9a..5381b7c 管理员 -> 源/管理员

通过“递归”策略进行合并.

mnelson.txt | 1 +

更改1个文件，插入1个（+）

查看GitHub管理的原始文件 pulloriginmaster.md这将向您显示所有已更改的文件及其更改方式。现在我们可以再次使用git log命令查看所有新提交。（您可能需要将分支切换回master分支。您可以使用git checkout master命令来执行此操作。）

mnelson:myproject mnelson$ git log

确定 3e270876db0e5ffd3e9bfc5edede89b64b83812c

合并: 4f1cb17 5381b7c

作者: Meghan Nelson <mnelson@hubspot.com>

日期: 9月11日 星期五17:48:11 2015 -0400

合并来自https://github.com/cubeton/mynewrepository的分支与管理员文件添加了一个新文件

确定 3e270876db0e5ffd3e9bfc5edede89b64b83812c

合并: 4f1cb17 5381b7c

作者: Meghan Nelson <mnelson@hubspot.com>

日期: 9月11日 星期五17:43:22 2015 -0400

合并来自cubeton / my-new分支的拉取请求2

在我的文件中添加了更多文本

确定t 1e8dc0830b4db8c93efd80479ea886264768520c

作者: Meghan Nelson <mnelson@hubspot.com>

日期: 9月11日 星期五17:06:05 2015 -0400

在我的文件中添加了更多文本

确定 b345d9a25353037afdeaa9fcaf9f330effd157f1

作者: Meghan Nelson <mnelson@hubspot.com>

日期: 9月10日 星期四 17:42:15 2015 -0400

这是我的第一次确定

查看GitHub托管的原始文件gitlogaftermerge.md

步骤11：沉浸在git的荣耀中您已成功进行PR，并将您的代码合并到master分支中。 恭喜您！ 如果您想更深入了解与使用，请查看此Git101文件夹中的文件，以获取有关使用git和GitHub的更多提示和技巧。我还建议您与您的团队一起模拟一个较小的小组项目，就像我们在这里所做的那样。 让您的团队使用团队名称创建一个新文件夹，并向其中添加一些带有文本的文件。 然后，尝试将这些更改推送到此远程仓库中。 这样，您的团队可以开始对最初没有创建的文件进行更改并使用PR功能进行练习。 然后，使用GitHub上的git blame和git历史记录工具熟悉跟踪文件中进行了哪些更改以及谁进行了这些更改。您使用git的次数越多，您将越感到舒适。 （我无法抗拒。）

### 由Meghan Nelson撰写

订阅更新

窗体顶端



HubSpot Product Blog Email Subscription



HubSpot产品需要您提供给我们的联系信息，以便就您的产品和服务与您联系。 您可以随时取消订阅这些通讯。 有关如何退订以及我们的隐私惯例以及保护您的隐私的承诺的信息，请参阅我们的隐私政策。



窗体底端

主题

* 产品
* 工程
* 设计
* 产品管理
* UX
* 文化
* 职业发展

[什么是 HubSpot](https://www.hubspot.com/what-is-hubspot)|[我们的](https://www.hubspot.com/internet-marketing-company)故事|我们的作品|文化规范

[Facebook](https://www.facebook.com/HubSpotDevelopment?fref=ts)|[Twitter](https://twitter.com/HubSpotDev)|[Instagram](https://instagram.com/hubspotdev/)