# Git 基本操作

Git 的工作就是创建和保存你项目的快照及与之后的快照进行对比。本章将对有关创建与提交你的项目快照的命令作介绍。

## 获取与创建项目命令

### **git init**

用 git init 在目录中创建新的 Git 仓库。 你可以在任何时候、任何目录中这么做，完全是本地化的。

在目录中执行 git init，就可以创建一个 Git 仓库了。比如我们创建 runoob 项目：

$ mkdir runoob\_x000D\_$ cd runoob/\_x000D\_$ git initInitialized empty Git repository in /Users/tianqixin/www/runoob/.git/# 在 /www/runoob/.git/ 目录初始化空 Git 仓库完毕。

现在你可以看到在你的项目中生成了 .git 这个子目录。 这就是你的 Git 仓库了，所有有关你的此项目的快照数据都存放在这里。

ls -a. .. .git

### **git clone**

使用 git clone 拷贝一个 Git 仓库到本地，让自己能够查看该项目，或者进行修改。

如果你需要与他人合作一个项目，或者想要复制一个项目，看看代码，你就可以克隆那个项目。 执行命令：

git clone [url]

[url] 为你想要复制的项目，就可以了。

例如我们克隆 Github 上的项目：

$ git clone git@github.com:schacon/simplegit.gitCloning into 'simplegit'...\_x000D\_remote: Counting objects: 13, done.\_x000D\_remote: Total 13 (delta 0), reused 0 (delta 0), pack-reused 13Receiving objects: 100% (13/13), done.Resolving deltas: 100% (2/2), done.Checking connectivity... done.

克隆完成后，在当前目录下会生成一个 simplegit 目录：

$ cd simplegit/ $ ls README Rakefile lib

上述操作将复制该项目的全部记录。

$ ls -a. .. .git README Rakefile lib\_x000D\_$ cd .git\_x000D\_$ ls\_x000D\_HEAD description info packed-refs\_x000D\_branches hooks logs refs\_x000D\_config index objects

默认情况下，Git 会按照你提供的 URL 所指示的项目的名称创建你的本地项目目录。 通常就是该 URL 最后一个 / 之后的项目名称。如果你想要一个不一样的名字， 你可以在该命令后加上你想要的名称。

## 基本快照

Git 的工作就是创建和保存你的项目的快照及与之后的快照进行对比。本章将对有关创建与提交你的项目的快照的命令作介绍。

### **git add**

git add 命令可将该文件添加到缓存，如我们添加以下两个文件：

$ touch README\_x000D\_$ touch hello.php\_x000D\_$ ls\_x000D\_README hello.php\_x000D\_$ git status -s?? README?? hello.php\_x000D\_$

git status 命令用于查看项目的当前状态。

接下来我们执行 git add 命令来添加文件：

$ git add README hello.php

现在我们再执行 git status，就可以看到这两个文件已经加上去了。

$ git status -s\_x000D\_A README\_x000D\_A hello.php\_x000D\_$

新项目中，添加所有文件很普遍，我们可以使用 **git add .** 命令来添加当前项目的所有文件。

现在我们修改 README 文件：

$ vim README<pre><p>在 README 添加以下内容：<b># Runoob Git 测试</b>，然后保存退出。</p><p>再执行一下 git status：</p>\_x000D\_$ git status -s\_x000D\_AM README\_x000D\_A hello.php

"AM" 状态的意思是，这个文件在我们将它添加到缓存之后又有改动。改动后我们在执行 git add 命令将其添加到缓存中：

$ git add .\_x000D\_$ git status -s\_x000D\_A README\_x000D\_A hello.php

当你要将你的修改包含在即将提交的快照里的时候，需要执行 git add。

### **git status**

git status 以查看在你上次提交之后是否有修改。

我演示该命令的时候加了 -s 参数，以获得简短的结果输出。如果没加该参数会详细输出内容：

$ git statusOn branch master\_x000D\_Initial commit\_x000D\_Changes to be committed:\_x000D\_ (use "git rm --cached <file>..." to unstage)\_x000D\_\_x000D\_ new file: README\_x000D\_ new file: hello.php

### **git diff**

执行 git diff 来查看执行 git status 的结果的详细信息。

git diff 命令显示已写入缓存与已修改但尚未写入缓存的改动的区别。git diff 有两个主要的应用场景。

* 尚未缓存的改动：**git diff**
* 查看已缓存的改动： **git diff --cached**
* 查看已缓存的与未缓存的所有改动：**git diff HEAD**
* 显示摘要而非整个 diff：**git diff --stat**

在 hello.php 文件中输入以下内容：

<?php\_x000D\_echo '菜鸟教程：www.runoob.com';?>

$ git status -s\_x000D\_A README\_x000D\_AM hello.php\_x000D\_$ git diff\_x000D\_diff --git a/hello.php b/hello.php\_x000D\_index e69de29..69b5711 100644--- a/hello.php+++ b/hello.php@@ -0,0 +1,3 @@+<?php+echo '菜鸟教程：www.runoob.com';+?>

git status 显示你上次提交更新后的更改或者写入缓存的改动， 而 git diff 一行一行地显示这些改动具体是啥。

接下来我们来查看下 git diff --cached 的执行效果：

$ git add hello.php \_x000D\_$ git status -s\_x000D\_A README\_x000D\_A hello.php\_x000D\_$ git diff --cached\_x000D\_diff --git a/README b/READMEnew file mode 100644\_x000D\_index 0000000..8f87495--- /dev/null+++ b/README@@ -0,0 +1 @@+# Runoob Git 测试\_x000D\_diff --git a/hello.php b/hello.phpnew file mode 100644\_x000D\_index 0000000..69b5711--- /dev/null+++ b/hello.php@@ -0,0 +1,3 @@+<?php+echo '菜鸟教程：www.runoob.com';+?>

### **git commit**

使用 git add 命令将想要快照的内容写入缓存区， 而执行 git commit 将缓存区内容添加到仓库中。

Git 为你的每一个提交都记录你的名字与电子邮箱地址，所以第一步需要配置用户名和邮箱地址。

$ git config --global user.name 'runoob'\_x000D\_$ git config --global user.email test@runoob.com

接下来我们写入缓存，并提交对 hello.php 的所有改动。在首个例子中，我们使用 -m 选项以在命令行中提供提交注释。

$ git add hello.php\_x000D\_$ git status -s\_x000D\_A README\_x000D\_A hello.php\_x000D\_$ $ git commit -m '第一次版本提交'[master (root-commit) d32cf1f] 第一次版本提交\_x000D\_ 2 files changed, 4 insertions(+)\_x000D\_ create mode 100644 README\_x000D\_ create mode 100644 hello.php

现在我们已经记录了快照。如果我们再执行 git status:

$ git status# On branch master\_x000D\_nothing to commit (working directory clean)