https://www.jianshu.com/p/55b6720264ee

git pull origin <remote_branch>: <local_branch>	将远程分支拉取到指定本地分支
git pull origin <remote_branch>	将指定远程分支同步到当前本地分支
git pull	拉取所有远程分支的新版本"坐标"，并同步当前分支的本地代码(具体根据关联分支而定)

工作中，我们会用到**git pull**来从远程仓库"同步"代码，通常有三种方式；

  git pull origin <remote_branch>：<local_branch>

  git pull origin <remote_branch>

  git pull

这三种用法充分诠释了什么是**简即繁**，**繁即简**；看上去简单的，往往背后蕴藏玄机；

测试环境:

本地分支：master和dev

远程分支：master和dev

```
$ git branch -a
* dev
  master
  remotes/origin/HEAD -> origin/master
  remotes/origin/dev
  remotes/origin/master
```

1.git pull origin <remote_branch>：<local_branch>

这种用法写起来最为繁琐，但最好理解：

  场景：当本地的当前分支不是local_branch；

  作用：将远程分支拉取到指定本地分支；

例如：当前分支是dev，但是你想把远程master”同步”到本地master，但又不想使checkout切换到master分支；

这时你就可以使用git pull origin master：master

```
zhangchangzhi@ZBXXXX /e/02.Workspace-test/gitTest (dev)
$ git pull origin master:master
From https://github.com/jinxintang/gitTest
   a09fdc4..941758f  master     -> master
Already up-to-date.
```

从上述代码可以看到，我当前分支为**dev**,但执行"同步”操作的却是在master分支;

2.git pull origin <remote_branch>

有了上面的例子，这种使用方法的场景和作用就好理解了:

　　场景：在当前分支上进行同步操作;

　　作用：将指定远程分支同步到当前本地分支;

废话不说，上代码:

```
zhangchangzhi@ZBXXX /e/02.Workspace-test/gitTest (dev)
$ git pull origin master
From https://github.com/jinxintang/gitTest
 * branch            master     -> FETCH_HEAD
Already up-to-date.
```

把远程master分支同步到HEAD分支（HEAD分支指向当前位置）；

3.git pull

这种写法最简单，也最常用，但是隐含的知识也是最多的;

　　场景：本地分支已经和想要拉取的分支建立了“关联”关系;

　　作用：拉取所有远程分支的新版本"坐标"，并同步当前分支的本地代码(具体根据关联分支而定)

什么是"关联"分支?

首先我们先使用`git branch -vv`查看一下目前分支的“关联”情况;

```
$ git branch -vv
* dev    1a1b215 [origin/dev] Merge branch 'master' of
https://github.com/jinxintang/gitTest into dev
  master a09fdc4 [origin/master] create pull
```

可以看到我们的本地的dev关联的是远程(origin)的dev，本地的master关联的是远程(origin)的master;

那么这种关联是如何建立、是否可以修改呢;

配置本地分支与远程分支的三种方法:

1.检出时建立关联关系: `git checkout -b dev origin/dev`

当我们检查时，git会自动为我们检出的分支和远程分支建立关联关系;

2.提交时配置关联关系: `git push -u origin <remote_branch>`或`git push --set-upstream origin <remote_branch>`

```
zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest (dev_zcz)
$ git branch -vv
* dev_zcz 3b7001a [origin/dev] cm
  master  a09fdc4 [origin/master] create pull

zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest (dev_zcz)
$ git push -u origin dev_zcz
Everything up-to-date
Branch dev_zcz set up to track remote branch dev_zcz from origin.

zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest (dev_zcz)
$ git branch -vv
* dev_zcz 3b7001a [origin/dev_zcz] cm
  master  a09fdc4 [origin/master] create pull
```

通过上面的例子可以看到push前dev_zcz关联的是origin/dev,执行push -u 后管理分支改为origin/dev_zcz

注：默认配置下，提交时本地分支需和远程分支同名;

3.更改git/config文件: `git branch --set-upstream-to=<remote_branch>`

```
zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest (dev_zcz)
$ git branch --set-upstream-to=origin/zcz
Branch dev_zcz set up to track local branch origin/zcz.

zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest (dev_zcz)
$ git branch -vv
```

```
* dev_zcz    3b7001a [origin/zcz] cm
  master     a09fdc4 [origin/master] create pull
  origin/zcz 3b7001a [dev_zcz] cm
```

无论使用上述那种方法，本地分支和远程分支的“关联”最终都会写到config文件；

```
zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest/.git (GIT_DIR!)
$ cat config
[core]
        repositoryformatversion = 0
        filemode = false
        bare = false
        logallrefupdates = true
        symlinks = false
        ignorecase = true
[remote "origin"]
        url = https://github.com/jinxintang/gitTest.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
        remote = origin
        merge = refs/heads/master
[branch "dev_zcz"]
        remote = .
        merge = refs/heads/origin/zcz
[branch "origin/zcz"]
        remote = .
        merge = refs/heads/dev_zcz
```

注：本项目的配置信息存放目录：项目所在目录/.git/config

看完这三种配置关联分支的方法，想必大家已经对“关联分支”有了一定了解；

关联分支：在git中表现为upstream,无论是使用push -u 或是 git branch --set-upstream-to方法，均会将这种对应关系写入.git/config配置文件，如果一个本地分支没有关联分支，则无法执行 git push 或 git pull指令；

没有"关联"分支的情况下，使用push会先让你设置一个upstream branch.

```
zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest (dev_no_upstream)
$ git branch -vv
* dev_no_upstream 3b7001a cm
  dev_zcz         3b7001a [origin/zcz] cm
  master          a09fdc4 [origin/master] create pull
  origin/zcz      3b7001a [dev_zcz] cm

zhangchangzhi@ZB-PF0SB6DQ MINGW64 /e/02.Workspace-test/gitTest (dev_no_upstream)
$ git push
fatal: The current branch dev_no_upstream has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin dev_no_upstream
```

那么建立了一个关联分支，是否就一定能使用git push呢？请阅读<git 实践(二)push的使用>