# [Github使用.gitignore文件忽略不必要上传的文件](http://blog.csdn.net/gjy211/article/details/51607347)

2016-06-07 23:30 999人阅读 [评论](http://blog.csdn.net/gjy211/article/details/51607347" \l "comments)(0) [收藏](http://blog.csdn.net/gjy211/article/details/javascript:void(0);" \o "收藏) [举报](http://blog.csdn.net/gjy211/article/details/51607347" \l "report" \o "举报)

版权声明：本文为博主原创文章，未经博主允许不得转载。

（1）.gitignore介绍

 在工程实现过程中，会生成一些中间文件，或者在项目中的部分文件是不需要进行版本管理的。对于这些文件应该对于Github来讲是透明的。Github提供这种功能，可以自己指定哪些文件可以不被管理。具体方法是在版本管理的根目录下（与.**[Git](http://lib.csdn.net/base/git" \o "Git知识库" \t "http://blog.csdn.net/gjy211/article/details/_blank)**文件夹同级）创建一个  .gitignore（gitignore是隐藏文件，所以前面有个点）{在进行协作开发代码管理的过程中，常常会遇到某些临时文件、配置文件、或者生成文件等，这些文件由于不同的开发端会不一样，如果使用git add . 将所有文件纳入git库中，那么会出现频繁的改动和push，这样会引起开发上的不便。Git可以很方便的帮助我们解决这个问题，那就是建立项目文件过滤规则。Git可以很方便的帮助我们解决这个问题，那就是建立项目文件过滤规则。git中提供两种过滤机制，一种是全局过滤机制，即对所有的git都适用；另一种是针对某个项目使用的过滤规则。个人倾向于第二种。以我的一个项目为例，该项目用.net开发，.config文件、包括生成的bin/Debug, bin/Release文件等，

我希望不加入git管理。

在代码目录下建立.gitignore文件：vim .gitignore ,内容如下：

[plain] view plaincopy

#过滤数据库文件、sln解决方案文件、配置文件

\*.mdb

\*.ldb

\*.sln

\*.config

#过滤文件夹Debug,Release,obj

Debug/

Release/

obj/

然后调用git add. ，执行 git commit即可。

}

方式一：

首先要强调一点，这个文件的完整文件名就是“.gitignore”，注意最前面有个“.”。这样没有扩展名的文件在Windows下不太好创建，这里给出win7的创建方法：

创建一个文件，文件名为：“.gitignore.”，注意前后都有一个点。保存之后系统会自动重命名为“.gitignore”。

一般来说每个Git项目中都需要一个“.gitignore”文件，这个文件的作用就是告诉Git哪些文件不需要添加到版本管理中。

实际项目中，很多文件都是不需要版本管理的，比如**[Python](http://lib.csdn.net/base/python" \o "Python知识库" \t "http://blog.csdn.net/gjy211/article/details/_blank)**的.pyc文件和一些包含密码的配置文件等等。

这个文件的内容是一些规则，Git会根据这些规则来判断是否将文件添加到**[版本控制](http://lib.csdn.net/base/git" \o "Git知识库" \t "http://blog.csdn.net/gjy211/article/details/_blank)**中。

下面我们看看常用的规则：

/mtk/ 过滤整个文件夹

\*.zip 过滤所有.zip文件

/mtk/do.c 过滤某个具体文件

很简单吧，被过滤掉的文件就不会出现在你的GitHub库中了，当然本地库中还有，只是push的时候不会上传。

需要注意的是，gitignore还可以指定要将哪些文件添加到版本管理中：

!\*.zip

!/mtk/one.txt

唯一的区别就是规则开头多了一个感叹号，Git会将满足这类规则的文件添加到版本管理中。

为什么要有两种规则呢？想象一个场景：我们只需要管理/mtk/目录中的one.txt文件，这个目录中的其他文件都不需要管理。那么我们就需要使用：

/mtk/

!/mtk/one.txt

假设我们只有过滤规则没有添加规则，那么我们就需要把/mtk/目录下除了one.txt以外的所有文件都写出来！

最后需要强调的一点是，如果你不慎在创建.gitignore文件之前就push了项目，那么即使你在.gitignore文件中写入新的过滤规则，这些规则也不会起作用，Git仍然会对所有文件进行版本管理。

简单来说，出现这种问题的原因就是Git已经开始管理这些文件了，所以你无法再通过过滤规则过滤它们。

所以大家一定要养成在项目开始就创建.gitignore文件的习惯，否则一旦push，处理起来会非常麻烦。

方式二：

就是新建.gitignore然后再里面声明要忽略的文件夹或者文件就可以了，不过麻烦又来了，因为是点开头，没有文件名，没办法直接在windows目录下直接创建，必须通过右键Git Bash，按照linux的方式来新建.gitignore文件。具体步骤如下：

1. 在需要创建 .gitignore 文件的文件夹, 右键选择

Git Bash 进入命令行，进入项目所在目录。

2. 输入 touch .gitignore 在文件夹就生成了一个“.gitignore”文件。

然后在”.gitignore” 文件里输入你要忽略的文件夹及其文件就可以了。

3.然后用编辑器打开这个文件进行编辑就行了。如果你很牛，熟悉linux下的vi编辑的命令，也可以用命令来声明规则额。在这里就不说了额，

附： vi编辑器的怎么换行呢？

总结：vi保存退出命令x和wq的区别

搜集：不会linux下vi命令，伤不起啊

4.然后就写规则来操作要忽略的文件了。.gitignore文件过滤有两种模式，开放模式和保守模式

4.1开放模式负责设置过滤哪些文件和文件夹

eg：

过滤文件夹设置：

/mtk/ 表示过滤这个文件夹

过滤文件设置

指定过滤某种类型的文件：

\*.zip

\*.rar

\*.via

\*.tmp

\*.err

指定过滤某个文件：

/mtk/do.c

/mtk/if.h

4.2 b保守模式负责设置哪些文件不被过滤，也就是哪些文件要被跟踪。

跟踪某个文件夹

!/plutommi/mmi

跟踪某类文件

!\*.c

!\*.h

跟踪某个指定文件

!/plutommi/mmi/mmi\_features.h

4.3.配置.gitignore 的简易原则

采用共享模式与保守模式结合配置的办法。eg：一个文件夹下有很多文件夹和文件，而我只想跟踪其中的一个文件，这样设置就可以满足这种情况，先用共享模式把整个目录 都设置为不跟踪，然后再用保守模式把这个文件夹中想要跟踪的文件设置为被跟踪，配置很简单，就可以跟踪想要跟踪的文件。

方式三：

.gitignore 配置文件用于配置不需要加入版本管理的文件，配置好该文件可以为我们的版本管理带来很大的便利，以下是个人对于配置 .gitignore 的一些心得。

1、配置语法：

　　以斜杠“/”开头表示目录；

　　以星号“\*”通配多个字符；

　　以问号“?”通配单个字符

　　以方括号“[]”包含单个字符的匹配列表；

　　以叹号“!”表示不忽略(跟踪)匹配到的文件或目录；

　　此外，git 对于 .ignore 配置文件是按行从上到下进行规则匹配的，意味着如果前面的规则匹配的范围更大，则后面的规则将不会生效；

2、示例：

　　（1）规则：fd1/\*  
　　　　  说明：忽略目录 fd1 下的全部内容；注意，不管是根目录下的 /fd1/ 目录，还是某个子目录 /child/fd1/ 目录，都会被忽略；

　　（2）规则：/fd1/\*  
　　　　  说明：忽略根目录下的 /fd1/ 目录的全部内容；

　　（3）规则：

/\*  
!.gitignore  
!/fw/bin/  
!/fw/sf/

说明：忽略全部内容，但是不忽略 .gitignore 文件、根目录下的 /fw/bin/ 和 /fw/sf/ 目录；

方式四：

# cat .gitignore \*.[oa] \*~

      第一行告诉 Git 忽略所有以 .o 或 .a 结尾的文件。一般这类对象文件和存档文件都是编译过程中出现的，我们用不着跟踪它们的版本。第二行告诉 Git 忽略所有以波浪符（~）结尾的文件，许多文本编辑软件（比如 Emacs）都用这样的文件名保存副本。此外，你可能还需要忽略 log，tmp 或者 pid 目录，以及自动生成的文档等等。要养成一开始就设置好 .gitignore 文件的习惯，以免将来误提交这类无用的文件。   
  
文件 .gitignore 的格式规范如下：

* 所有空行或者以注释符号 ＃ 开头的行都会被 Git 忽略。
* 可以使用标准的 glob 模式匹配。
* 匹配模式最后跟反斜杠（/）说明要忽略的是目录。
* 要忽略指定模式以外的文件或目录，可以在模式前加上惊叹号（!）取反。

      所谓的 glob 模式是指 shell 所使用的、简化了的正则表达式。星号（\*）匹配零个或多个任意字符；[abc] 匹配任何一个列在方括号中的字符（这个例子要么匹配一个 a，要么匹配一个 b，要么匹配一个 c）；问号（?）只匹配一个任意字符；如果在方括号中使用短划线分隔两个字符，表示所有在这两个字符范围内的都可以匹配（比如 [0-9] 表示匹配所有 0 到 9 的数字）。  
  
我们再看一个 .gitignore 文件的例子：

# 忽略所有 .a 结尾的文件

\*.a# 但 lib.a 除外

!lib.a# 仅仅忽略项目根目录下的 TODO 文件，不包括 subdir/TODO

/TODO# 忽略 build/ 目录下的所有文件

build/# 会忽略 doc/notes.txt 但不包括 doc/server/arch.txt

doc/\*.txt

其他信息可以直接 man gitignore 查看。