[GitHub实战系列~3.提交github的时候过滤某些文件 2015-12-10](http://www.cnblogs.com/dunitian/p/5036028.html)

**GitHub实战系列汇总：**[http://www.cnblogs.com/dunitian/p/5038719.html](http://www.cnblogs.com/dunitian/p/5038719.html" \t "_blank)

**——————————————————————————————————————————————————————**

**很多人问，明明有git gui 和 github可以直接图形化操作的吗？全部指令干啥？？？**

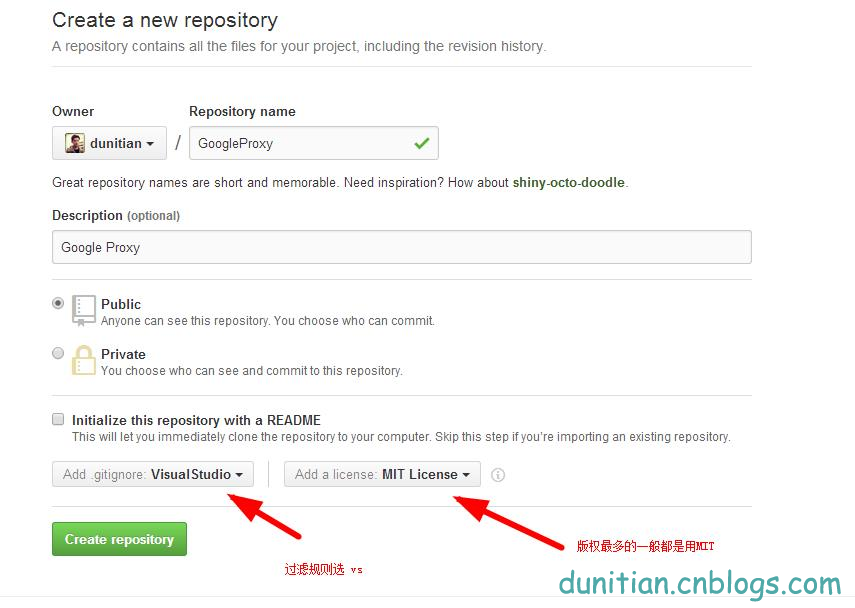
**呃(⊙o⊙)…呃(⊙o⊙)… ===> 装逼~**

**O(∩\_∩)O~，开玩笑的，其实就是为了通用和熟悉git，linux里面照样这样用，多熟悉点基础指令很有用的，**

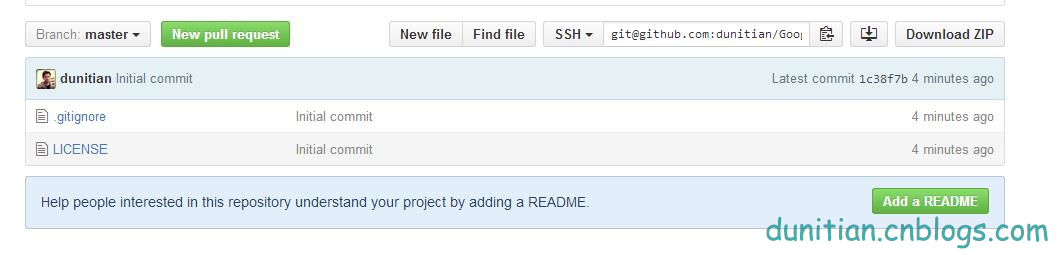
**如果觉得顿时不开心了、无爱了==>推荐你快速入门：**http://www.imooc.com/learn/390

**———————————————————————————————————————————————————————**

创建项目的时候在最下面，添加过滤器，选择vs 或者把 .gitignore 拷贝一份放git项目文件夹的根目录

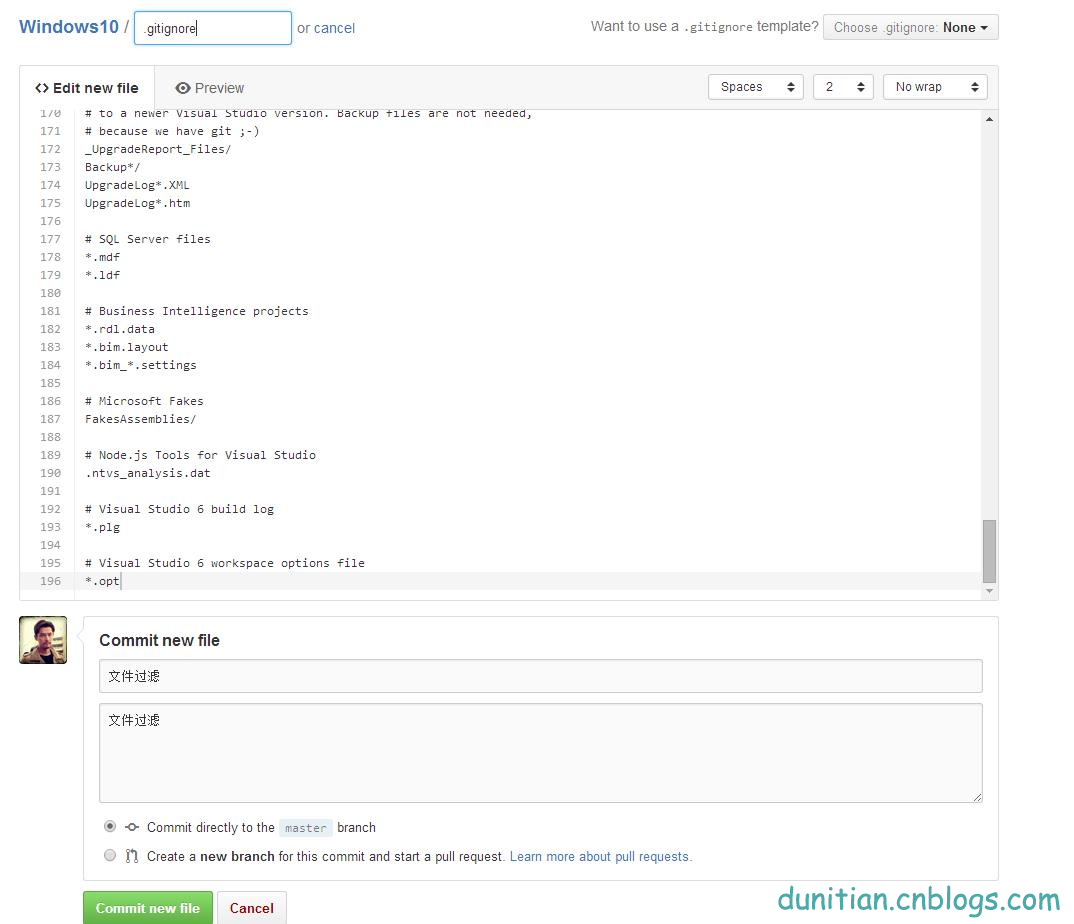


建完就有两个文件了，.gitignore 是过滤文件



补充：在windows里面直接建立这个文件是不行的，windows最后一个.前面必须有名字（linux里面有.号开头的文件），除了拷贝再介绍一个方法：

**进github，单击 New File 然后把最下面的过滤文件加进去**（ssh的左边第一个按钮）



vs的过滤规则贴一下：

|  |
| --- |
| ## Ignore Visual Studio temporary files, build results, and  ## files generated by popular Visual Studio add-ons.  # User-specific files  \*.suo  \*.user  \*.userosscache  \*.sln.docstates  # User-specific files (MonoDevelop/Xamarin Studio)  \*.userprefs  # Build results  [Dd]ebug/  [Dd]ebugPublic/  [Rr]elease/  [Rr]eleases/  x64/  x86/  build/  bld/  [Bb]in/  [Oo]bj/  # Visual Studo 2015 cache/options directory  .vs/  # MSTest test Results  [Tt]est[Rr]esult\*/  [Bb]uild[Ll]og.\*  # NUNIT  \*.VisualState.xml  TestResult.xml  # Build Results of an ATL Project  [Dd]ebugPS/  [Rr]eleasePS/  dlldata.c  \*\_i.c  \*\_p.c  \*\_i.h  \*.ilk  \*.meta  \*.obj  \*.pch  \*.pdb  \*.pgc  \*.pgd  \*.rsp  \*.sbr  \*.tlb  \*.tli  \*.tlh  \*.tmp  \*.tmp\_proj  \*.log  \*.vspscc  \*.vssscc  .builds  \*.pidb  \*.svclog  \*.scc  # Chutzpah Test files  \_Chutzpah\*  # Visual C++ cache files  ipch/  \*.aps  \*.ncb  \*.opensdf  \*.sdf  \*.cachefile  # Visual Studio profiler  \*.psess  \*.vsp  \*.vspx  # TFS 2012 Local Workspace  $tf/  # Guidance Automation Toolkit  \*.gpState  # ReSharper is a .NET coding add-in  \_ReSharper\*/  \*.[Rr]e[Ss]harper  \*.DotSettings.user  # JustCode is a .NET coding addin-in  .JustCode  # TeamCity is a build add-in  \_TeamCity\*  # DotCover is a Code Coverage Tool  \*.dotCover  # NCrunch  \_NCrunch\_\*  .\*crunch\*.local.xml  # MightyMoose  \*.mm.\*  AutoTest.Net/  # Web workbench (sass)  .sass-cache/  # Installshield output folder  [Ee]xpress/  # DocProject is a documentation generator add-in  DocProject/buildhelp/  DocProject/Help/\*.HxT  DocProject/Help/\*.HxC  DocProject/Help/\*.hhc  DocProject/Help/\*.hhk  DocProject/Help/\*.hhp  DocProject/Help/Html2  DocProject/Help/html  # Click-Once directory  publish/  # Publish Web Output  \*.[Pp]ublish.xml  \*.azurePubxml  # TODO: Comment the next line if you want to checkin your web deploy settings  # but database connection strings (with potential passwords) will be unencrypted  \*.pubxml  \*.publishproj  # NuGet Packages  \*.nupkg  # The packages folder can be ignored because of Package Restore  \*\*/packages/\*  # except build/, which is used as an MSBuild target.  !\*\*/packages/build/  # Uncomment if necessary however generally it will be regenerated when needed  #!\*\*/packages/repositories.config  # Windows Azure Build Output  csx/  \*.build.csdef  # Windows Store app package directory  AppPackages/  # Others  \*.[Cc]ache  ClientBin/  [Ss]tyle[Cc]op.\*  ~$\*  \*~  \*.dbmdl  \*.dbproj.schemaview  \*.pfx  \*.publishsettings  node\_modules/  bower\_components/  # RIA/Silverlight projects  Generated\_Code/  # Backup & report files from converting an old project file  # to a newer Visual Studio version. Backup files are not needed,  # because we have git ;-)  \_UpgradeReport\_Files/  Backup\*/  UpgradeLog\*.XML  UpgradeLog\*.htm  # SQL Server files  \*.mdf  \*.ldf  # Business Intelligence projects  \*.rdl.data  \*.bim.layout  \*.bim\_\*.settings  # Microsoft Fakes  FakesAssemblies/  # Node.js Tools for Visual Studio  .ntvs\_analysis.dat  # Visual Studio 6 build log  \*.plg  # Visual Studio 6 workspace options file  \*.opt |