

# 在VS Code中使用GitHub

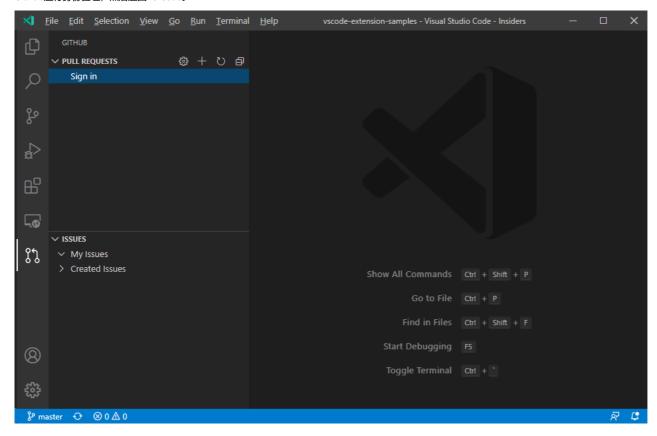
(https://github.com/Microsoft/vscode-docs/blob/master/docs/editor/github.md)

将GitHub (https://github.com)与Visual Studio Code结合使用,可以共享源代码并与他人协作。通过GitHub Pull Requests and Issues (https://marketplace.visualstudio.com/items?itemName=GitHub.vscode-pull-request-github)扩展提供GitHub集成。要使用VS Code开始使用GitHub,您需要创建一个帐户 (https://help.github.com/github/getting-started-with-github/signing-up-for-a-new-github-account)并安装GitHub Pull Requests and Issues (https://marketplace.visualstudio.com/items?itemName=GitHub.vscode-pull-request-github)扩展。在本主题中,我们将演示如何在不离开VS Code的情况下使用GitHub中一些您喜欢的部分。

如果您不熟悉源代码控制并希望从此处开始,则可以了解VS Code的源代码控制集成 (/docs/editor/versioncontrol)。

### GitHub Pull请求和问题入门

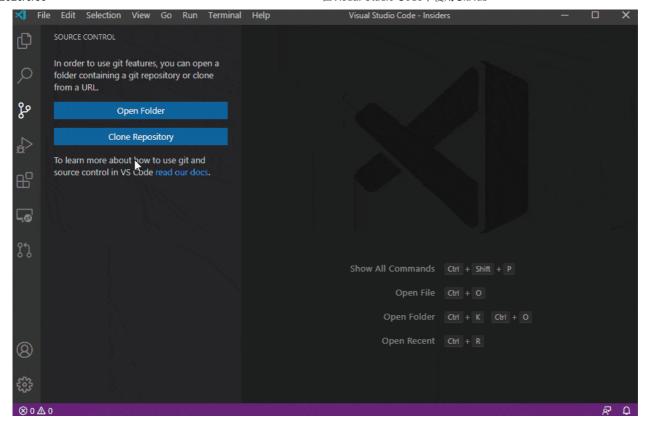
安装GitHub Pull Requests and Issues (https://marketplace.visualstudio.com/items?itemName=GitHub.vscode-pull-request-github)扩展后,您需要登录。按照提示对GitHub进行身份验证,然后返回VS Code。



## 建立资料库

### 克隆存储库

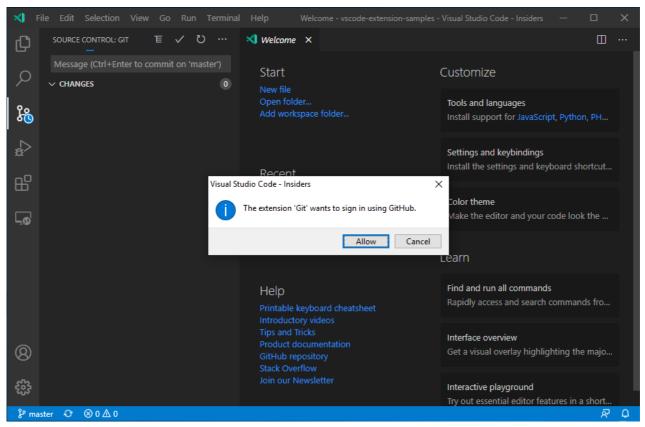
您可以使用Command Palette中的**Git: Clone**命令(Ctrl + Shift + P)或使用Source Control视图中的**Clone Repository**按钮(在没有打开文件夹的情况下可用)从GitHub搜索和克隆存储库。



#### 使用现有存储库进行身份验证

当您在需要GitHub身份验证的VS Code中运行任何Git操作时,就会通过GitHub启用身份验证,例如,推送到您所属的资源库或克隆私有资源库。您无需安装任何特殊的扩展即可进行身份验证;它内置于VS Code中,因此您可以有效地管理存储库。

当您执行需要GitHub身份验证的操作时,将看到登录提示:



请按照以下步骤登录GitHub并返回VS Code。

# 编辑器整合

### 盘旋

当您打开存储库并且@-提到了用户时,可以将鼠标悬停在该用户名上,然后看到GitHub风格的悬停。

```
🔀 File Edit Selection View Go Run Terminal Help
                                                          vscode.proposed.d.ts - vscode - Visual Studio Code - Insiders
                                                                                             ប្ ↔ ⊹ ⊹ Ռ 🏻
      TS vscode.proposed.d.ts X
      src > vs > TS vscode.proposed.d.ts > {} 'vscode'
                        export const onDidChangeTunnels: Event<void>;
scheme:
Q
                        authori
                                   Isidor Nikolic isidorn
                        formatt
                                 VSCode
You, a few secon 👂 Zurich
                   export inte of Committed to this repository on Apr 29, 2020
                        label: 

Member of Microsoft
        251
                        // eslint-disable-next-line vscode-dts-literal-or-types
separator: '/' | '\\' | '';
Ï
                        tildify?: boolean;
                        normalizeDriveLetter?: boolean;
                        workspaceSuffix?: string;
                        authorityPrefix?: string;
(Q)
                   isidor, a year ago | 3 authors (Alex Dima and others)
                   export namespace workspace {
                        export function registerRemoteAuthorityResolver(authorityPrefix: string, re
                        export function registerResourceLabelFormatter(formatter: ResourceLabelForm
                 ♦ Johannes Rieken, a year ago Ln 265, Col 17 CRLF TypeScript 3.9.0-dev.20200427
```

#-提及的问题编号,完整的GitHub问题URL和存储库指定的问题也有类似的悬停。

```
X File Edit Selection View Go Run Terminal Help
                                                              diskFileService.test.ts - vscode - Visual Studio Code - Insiders
      TS diskFileService.test.ts X
                                                                                                   ប្ ↔ ⊹ ⊹ Ռ 🏻
ф
       src > vs > platform > files > test > electron-browser > 78 diskFileService.test.ts > 🔗 suite('Disk File Service') callback > 😚 testFilesNotIntermingled
                         const content = await service.readF microsoft/vscode on Nov 14, 2017
                                                                        Files are intermingled #38331
                          assert.equal((await streamToBuffer()
                                                                     open 'checker.ts' in the left-hand-side editor open user settings in
                                                                     the right-hand-side editor close and start again vscode The contents
                                                                     are intermingled:
                     test('readFile - Files are intermingled
Q
                          return testFilesNotIntermingled();
test('readFile - Files are intermingled #38331 - buffered', async () ⇒ {
    setCapabilities(fileProvider, FileSystemProviderCapabilities.FileOpenReadW)
留
                          return testFilesNotIntermingled();
ſΊ
                     test('readFile - Files are intermingled #38331 - unbuffered', async () \Rightarrow {
                          setCapabilities(fileProvider, FileSystemProviderCapabilities.FileReadWrite)
                          return testFilesNotIntermingled();
                     test('readFile - Files are intermingled #38331 - streamed', async () \Rightarrow {
(Q)
                          setCapabilities(fileProvider, FileSystemProviderCapabilities.FileReadStream
                          return testFilesNotIntermingled();
                  Ln 1218, Col 48 (45 selected) CRLF TypeScript 3.9.0-dev.20200427
```

#### 意见建议

用户建议由"@"字符触发,发布建议由"#"字符触发。在编辑器和"源代码管理"视图的输入框中可以找到建议。

```
📢 File Edit Selection View Go Run Terminal Help
                                                           abstractTaskService.ts - vscode - Visual Studio Code - Insiders
      TS abstractTaskService.ts X
                                Settings
                                                                                               ព្ ↔ ⊹ ⊹ Ռ 🏻 …
       src > vs > workbench > contrib > tasks > browser > TS abstractTaskService.ts > 😭 AbstractTaskService
                                  nls.localize('TaskService.noWorkspace', "Tasks are only available (
R
                                  [{
                                       label: nls.localize('TaskService.learnMore', "Learn More"),
                                       run: () ⇒ this.openerService.open(URI.parse('https://code.visu
                                  }]
Q
                        return true:
船
                   private showDetail(): boolean {
                        return this.configurationService.getValue<boolean>(QUICKOPEN_DETAIL_CONFIG
IJ.
                    private async createTaskQuickPickEntries(tasks: Task[], group: boolean = false,
                        let count: { [key: string]: number; } = {};
if (tasks == undefined || tasks == null || tasks.length == 0) {
                             return [];
                        const TaskQuickPickEntry = (task: Task): TaskQuickPickEntry ⇒ {
   let entryLabel = task._label;
                             if (count[task._id]) {
                                  entryLabel = entryLabel + ' (' + count[task._id].toString() + ')';
                                  count[task._id]++;
                 ⊗ o A o
                                                                         Ln 2138, Col 32 CRLF TypeScript 3.9.0-dev.20200427
```

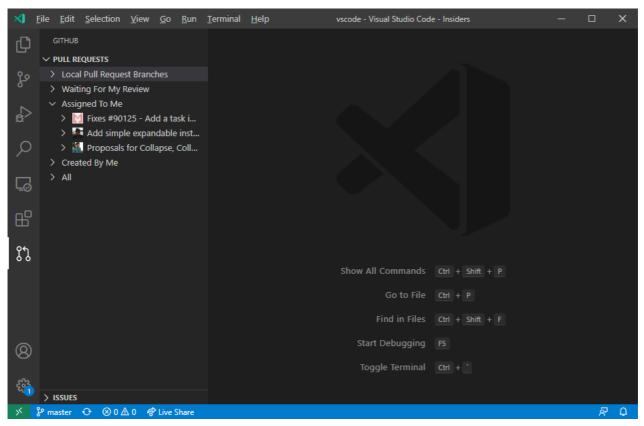
建议中出现的问题可以使用**GitHub Issues: Queries**(githubIssues.queries)设置 (/docs/getstarted/settings)进行配置。查询使用GitHub搜索语法 (https://help.github.com/articles/understanding-the-search-syntax)。

您还可以使用设置**GitHub问题: 忽略完成触发器** (githubIssues.ignoreCompletionTrigger) 和**GitHub问题: 忽略用户完成触发器** (githubIssues.ignoreUserCompletionTrigger)来配置哪些文件显示这些建议。这些设置采用语言标识符 (/docs/languages/identifiers)数组来指定文件类型。

```
// Languages that the '#' character should not be used to trigger issue completion suggestions.
"githubIssues.ignoreCompletionTrigger": [
   "python"
]
```

### 拉取要求

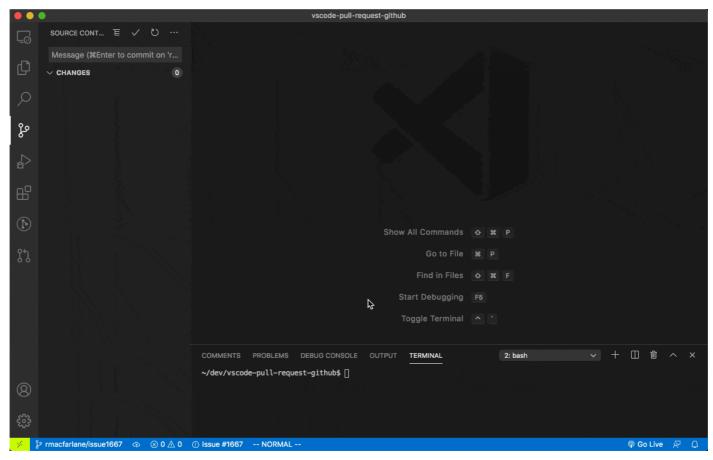
在"拉取请求"视图中,您可以查看,管理和创建拉取请求。



可以使用**GitHub Pull Requests: Queries** (githubPullRequests.queries) 设置配置用于显示拉取请求的**查询**,并使用GitHub搜索语法 (https://help.github.com/articles/understanding-the-search-syntax)。

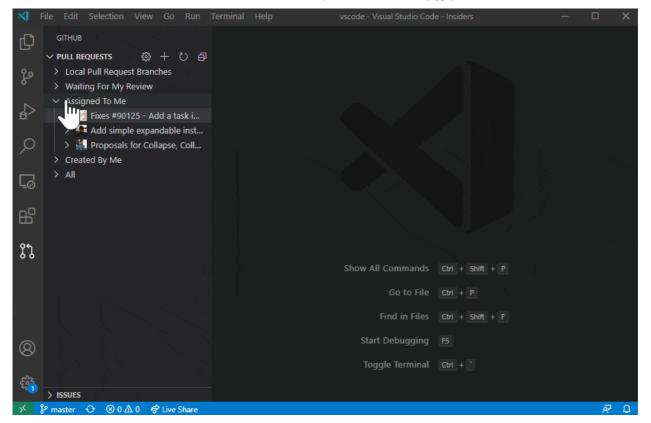
### 创建拉取请求

您可以使用GitHub Pull Requests: Create Pull Request命令或使用Pull Requests视图中的+按钮创建一个Pull Request。如果您尚未将分支机构推送到远程,则扩展程序将为您执行此操作。您可以使用最后的提交消息,分支名称或为拉取请求编写自定义标题。如果您的存储库具有拉取请求模板,则该模板将自动用于描述。

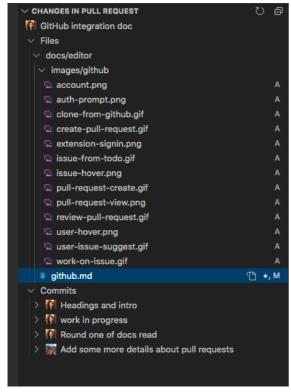


### 审核中

可以从"拉取请求"视图中查看拉取请求。您可以分配请求者和标签,添加评论,批准,关闭和合并所有来自请求请求的描述。



在描述页面上,您还可以使用Checkout按钮在本地轻松签出拉取请求。这将**在"拉取请求**"视图中添加新的**更改,**从中可以查看当前更改以及所有提交以及这些提交中的更改的差异。已注释的文件带有菱形图标。要查看磁盘上的文件,可以使用内联**打开文件**操作。

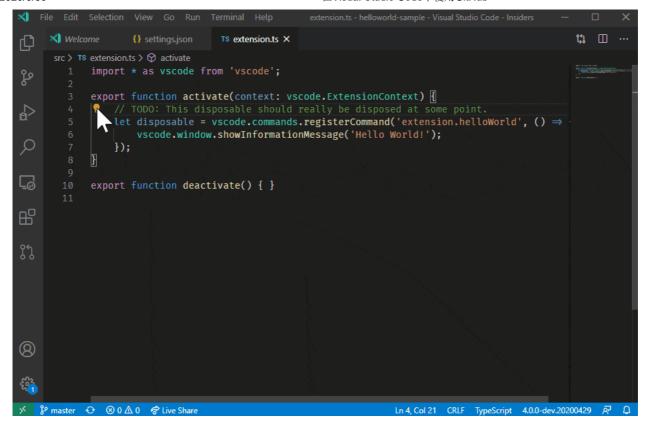


此视图中的差异编辑器使用本地文件,因此文件导航,IntelliSense和编辑正常进行。您可以在编辑器中添加这些差异的注释。支持添加单个评论和创建整体评论。

### 问题

### 创造问题

可以从"**问题**"视图中的+按钮创建问题,也可以使用**GitHub问题:从选择中创建问题**和**GitHub问题:从剪贴板中创建问题**。也可以使用"TODO"注释的代码操作来创建它们。



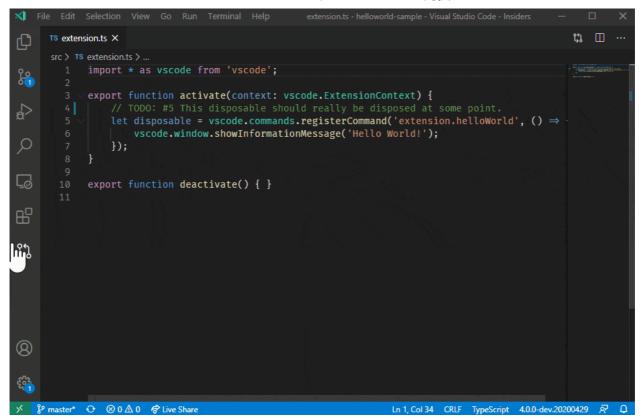
您可以使用GitHub Issues: Create Issue Triggers (githubIssues.createIssueTriggers) 设置为代码操作配置触发器。

#### 默认问题触发器是:

```
"githubIssues.createIssueTriggers": [
  "TODO",
  "todo",
  "BUG",
  "FIXME",
  "ISSUE",
  "HACK"
]
```

### 处理问题 #

在"**问题**"视图中,您可以查看问题并进行处理。默认情况下,当您开始处理问题时,将为您创建一个分支。您可以使用**GitHub Issues:Working Issue Branch** (githubIssues.workingIssueBranch)设置配置分支的名称。**Source Control**视图中的提交消息输入框将填充有一条提交消息,可以使用**GitHub Issues:Working Issue Format SCM** (githubIssues.workingIssueFormatScm )配置该消息。



如果您的工作流不涉及创建分支,或者如果您希望每次都提示您输入分支名称,则可以通过关闭**GitHub Issues:Use Branch For Issues**(githubIssues.useBranchForIssues)设置来跳过该步骤。

### 该文档对您有帮助吗?

是 没有

6/10/2020

你好,西雅图。 按照@code (https://go.microsoft.com/fwlink/?LinkID=533687)