

## 前端开发工具--vs code 使用教程

### 摘 要

Visual Studio Code 是一款轻巧但功能强大的源代码编辑器，适用于 Windows、macOS 和 Linux。它内置了对 JavaScript、TypeScript 和 Node.js 的支持，并且具有丰富的其他语言（如 C++、C#、Java、Python、PHP、Go）和运行时（如 .NET 和 Unity）的扩展生态系统。由于 vscode 是由 Typescript 开发，因此对前端开发的支持最好，且近些年 vscode 在前端编辑器界越来越火，因此，学会使用 vscode 这个充满无限可能的编辑器是非常重要的。而本文则将详细介绍如何使用 vscode 进行前端开发。

**关键词：**vscode；vscode 扩展插件；前端开发；Git；远程开发

## 目 录

1	vscode 的下载及安装 .....	1
1.1	软件下载 .....	1
1.2	安装 vscode.....	1
2	Vscode 界面及基本功能介绍.....	3
2.1	Vscode 用户界面简介.....	3
2.2	vscode 基本功能介绍.....	4
2.2.1	基本编辑 .....	4
2.2.2	常用快捷键 .....	4
2.2.3	主题设置 .....	7
2.2.4	用户及工作区设置 .....	8
2.2.5	扩展商店 .....	8
3	使用 vscode 进行前端开发 .....	13
3.1	使用 Emmet 快速编写 HTML 代码.....	13
3.1.1	生成元素 .....	14
3.1.2	生成文本+元素 .....	14
3.1.3	生成带属性的元素 .....	14
3.1.4	生成嵌套元素 .....	15
3.1.5	用乘法同时生成多个元素 .....	15
3.2	使用 preview on web server 插件调试代码.....	16
4	在 vscode 中使用 Git .....	17
4.1	设置本地项目目录 .....	17
4.2	git 基础配置 .....	17
4.3	拉取远程仓库中的代码 .....	18
4.4	使用 vscode 对代码进行编辑.....	18
4.5	保存更改，填写修改注释 .....	19
4.6	推送到远程仓库 .....	19
5	使用 vscode 进行远程开发 .....	20

5.1 配置免密登陆 .....	20
5.2 配置 vscode 远程开发插件.....	20

## 1 vscode 的下载及安装

### 1.1 软件下载

直接前往 vscode 官网下载即可：<https://code.visualstudio.com/>

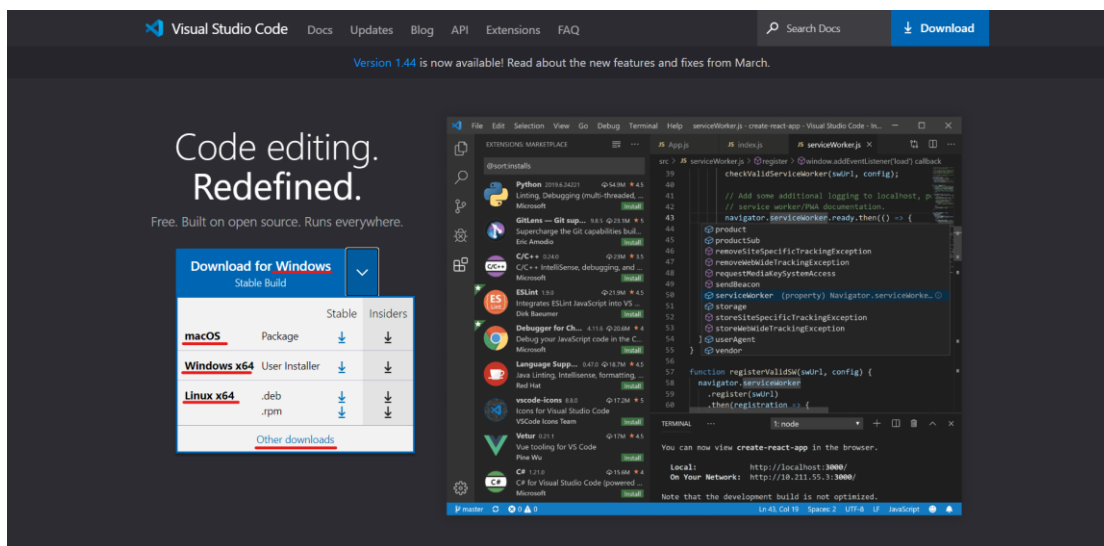


图 1.1.1 vscode 官网下载页面

根据自己的平台下载不同的安装包，下载完成后双击打开

### 1.2 安装 vscode

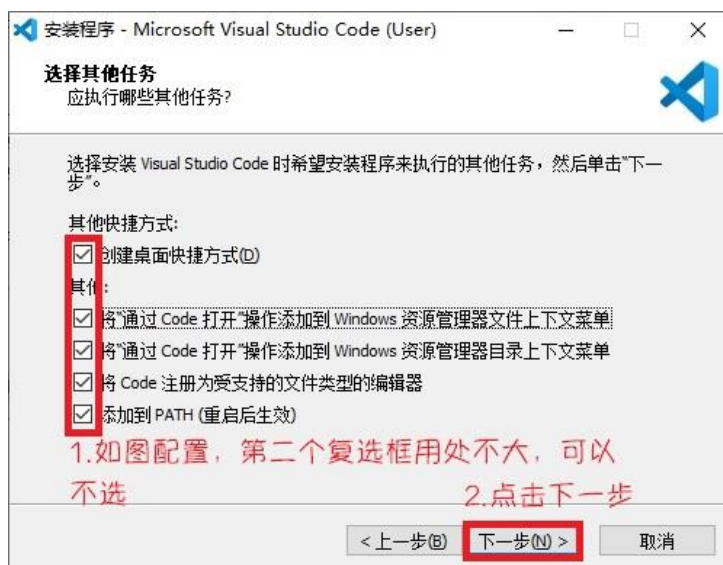


图 1.2.1 安装界面

安装完成后通过桌面图标打开，此时 **vscode** 是默认的英文界面，我们接下来安装中文扩展包。

具体方法是：单击左侧在线扩展商店（即第五个图标），在搜索框内键入 **Chinese**，安装第一个扩展（中文简体）。完成后重启 **vscode**，界面就会变成中文。

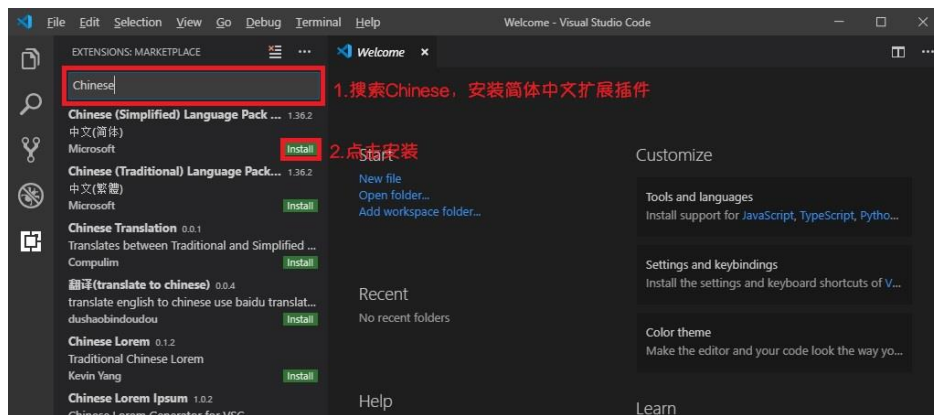


图 1.2.2 安装中文语言包

至此，**vscode** 的安装完成

## 2 Vscode 界面及基本功能介绍

### 2.1 Vscode 用户界面简介

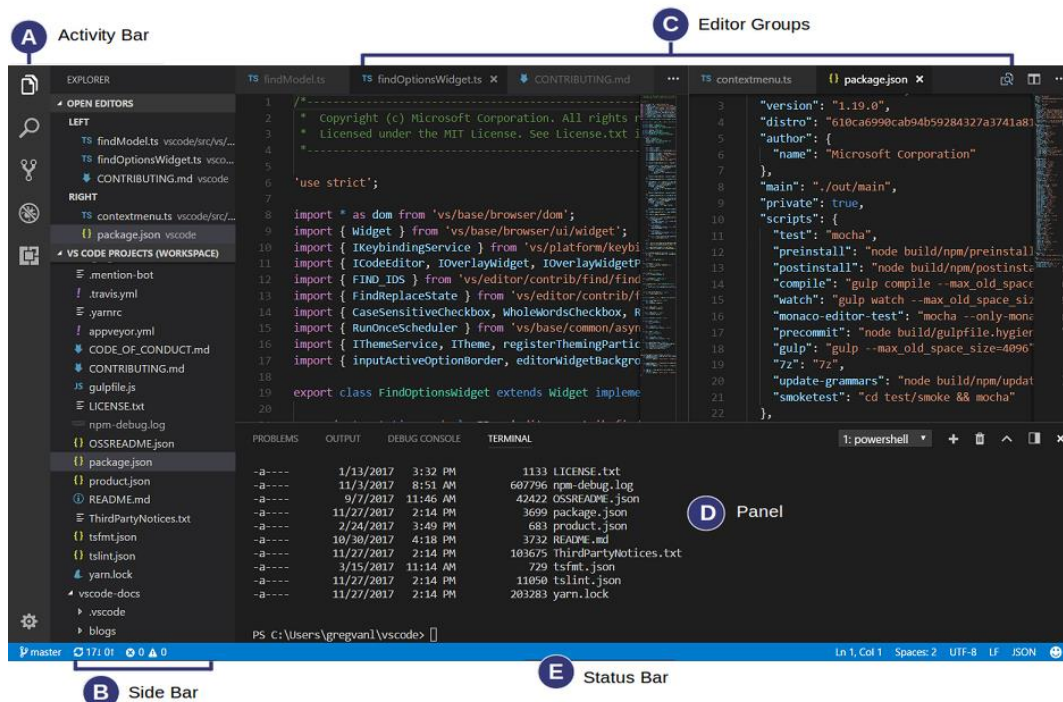


图 2.1.1 vscode 界面划分

vscode 的 UI 分为 5 个部分：

**A. Active bar**：活动栏。位于最左侧，允许我们在视图之间切换，并提供其他特定于上下文的指示器，例如启用 **Git** 时传出更改的数量。

**B. Side Bar**：侧边栏。包含如资源管理器等的视图以协助我们更高效的写代码。

**C. Editor Groups**：编辑器组。包含了已打开的文件，可以拆分成多行或多列

**D: Panel**：面板。显示输出、调试信息、错误警告或集成终端。面板也可以向右移动，以争取更大的垂直空间。

**E: Status Bar**：状态栏。展示了当前文档或工程的一些信息。

## 2.2 vscode 基本功能介绍

### 2.2.1 基本编辑

#### (1) 多光标

vscode 支持多个光标同时编辑，可同时更改多个地方。方法：按住 **alt** 并用鼠标单击添加多个光标

#### (2) 收缩/展开选择

快速收缩或展开当前选择代码块。方法：**alt+shift+右/左**

#### (3) 列选择

用于选择特定的列。方法：将光标放到一个地方，按住 **alt+shift** 时拖动至另一个地方。这样便选中了两个光标之间的列。

#### (4) 查找和替换

通过快捷键 **ctrl+f** 可以在当前编辑器打开查找小工具，默认查找范围是当前文档。当我们先选中一部分代码后，再按 **ctrl+f**，此时的查找范围则为选中的代码块。侧边栏搜索按钮则提供了全局搜索和替换功能，它的搜索范围是当前工程或文件夹中所有文件的内容。

#### (5) 同时更改所有相同项

在 **vscode** 中，当我们选中一段文字时，当前文本中所有与这段文本相同的文本背景都会半高亮，此时按下 **ctrl+f2**，每个文字段尾部都会出现一个光标，此时我们就可以同时更改这些内容了

#### (6) 快速注释当前所选代码：**ctrl+/\*\***

#### (7) 滚动速度倍增：**alt+滚轮**

### 2.2.2 常用快捷键

#### (1) 编辑器与窗口管理

➤ 打开一个新窗口：**Ctrl+Shift+N**

- 关闭窗口: `Ctrl+Shift+W`
- 同时打开多个编辑器 (查看多个文件)
- 新建文件 `Ctrl+N`
- 文件之间切换 `Ctrl+Tab`
- 切出一个新的编辑器 (最多 3 个) `Ctrl+\`, 也可以按住 `Ctrl` 鼠标点击 `Explorer` 里的文件名
- 左中右 3 个编辑器的快捷键 `Ctrl+1` `Ctrl+2` `Ctrl+3`
- 3 个编辑器之间循环切换 `Ctrl+`
- 编辑器换位置, `Ctrl+k` 然后按 `Left` 或 `Right`

## (2) 代码格式调整

- 代码行缩进 `Ctrl+[` 、 `Ctrl+]`
- `Ctrl+C` 、 `Ctrl+V` 复制或剪切当前行/当前选中内容
- 代码格式化: `Shift+Alt+F`, 或 `Ctrl+Shift+P` 后输入 `format code`
- 上下移动一行: `Alt+Up` 或 `Alt+Down`
- 向上向下复制一行: `Shift+Alt+Up` 或 `Shift+Alt+Down`
- 在当前行下边插入一行 `Ctrl+Enter`
- 在当前行上方插入一行 `Ctrl+Shift+Enter`

## (3) 光标相关

- 移动到行首: `Home`
- 移动到行尾: `End`
- 移动到文件结尾: `Ctrl+End`
- 移动到文件开头: `Ctrl+Home`



- 移动到定义处: **F12**
- 定义处缩略图: 只看一眼而不跳转过去 **Alt+F12**
- 移动到后半括号: **Ctrl+Shift+]**
- 选择从光标到行尾: **Shift+End**
- 选择从行首到光标处: **Shift+Home**
- 删除光标右侧的所有字: **Ctrl+Delete**
- 扩展/缩小选取范围: **Shift+Alt+Left** 和 **Shift+Alt+Right**
- 多行编辑(列编辑): **Alt+Shift+鼠标左键**, **Ctrl+Alt+Down/Up**
- 同时选中所有匹配: **Ctrl+Shift+L**
- **Ctrl+D** 下一个匹配的也被选中 (在 **sublime** 中是删除当前行, 后面自定义快捷键中, 设置与 **Ctrl+Shift+K** 互换了)
- 回退上一个光标操作: **Ctrl+U**

#### (4) 重构代码

- 找到所有的引用: **Shift+F12**
- 同时修改本文件中所有匹配的: **Ctrl+F12**
- 重命名: 比如要修改一个方法名, 可以选中后按 **F2**, 输入新的名字, 回车, 会发现所有的文件都修改了
- 跳转到下一个 **Error** 或 **Warning**: 当有多个错误时可以按 **F8** 逐个跳转
- 查看 **diff**: 在 **explorer** 里选择文件右键 **Set file to compare**, 然后需要对比的文件上右键选择 **Compare with file\_name\_you\_chose**

#### (5) 查找替换

- 查找 **Ctrl+F**
- 查找替换 **Ctrl+H**

- 整个文件夹中查找 `Ctrl+Shift+F`

## (6) 显示相关

- 全屏: `F11`
- `zoomIn/zoomOut`: `Ctrl +/-`
- 侧边栏显/隐: `Ctrl+B`
- 显示资源管理器 `Ctrl+Shift+E`
- 显示搜索 `Ctrl+Shift+F`
- 显示 Git `Ctrl+Shift+G`
- 显示 Debug `Ctrl+Shift+D`
- 显示 Output `Ctrl+Shift+U`

### 2.2.3 主题设置

主题设置包括颜色主题和图标主题,包括用户界面的颜色及代码块的高亮颜色和背景颜色。在 `vscode` 中打开文件>首选项>颜色主题,上下键切换主题预览, `enter` 键确认切换。如果对于自带的主题不感兴趣,还可以从扩展商店自行下载主题使用,十分方便。

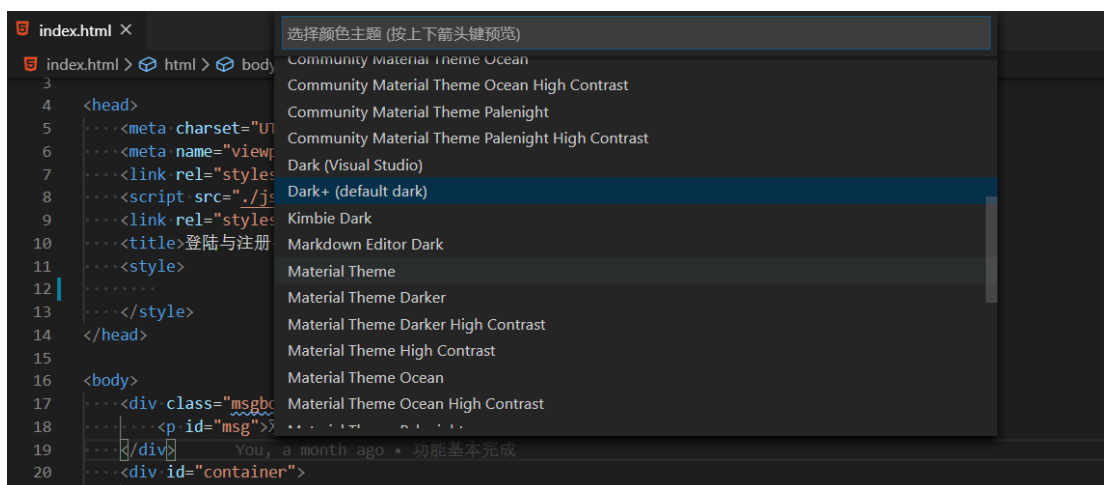


图 2.2.3.1 更改颜色主题

同样的，在 **vscode** 中打开文件>首选项>文件图标主题，即可设置文件或文件夹图标的样式了。如图是三种不同的图标主题：

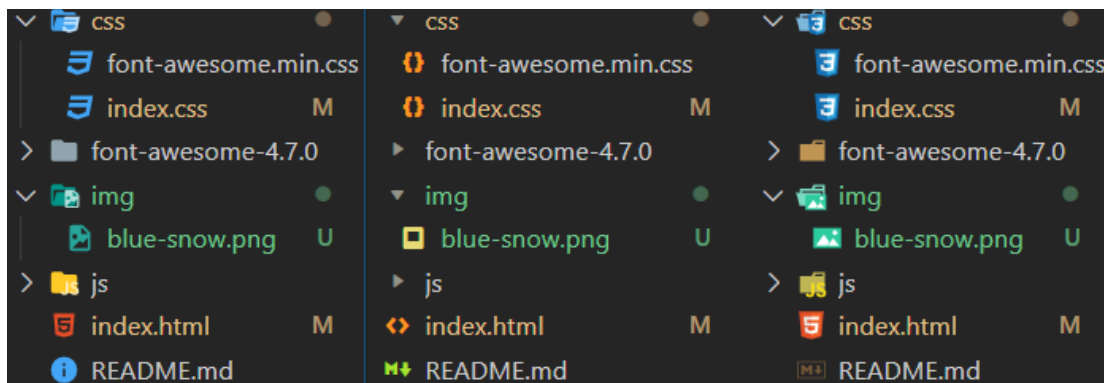


图 2.2.3.2 三种图标主题

## 2.2.4 用户及工作区设置

依次点击文件>首选项>设置或者使用快捷键 **ctrl+,** 即可为当前用户或这个工作区配置 **vscode** 设置。对于前端开发来说，建议使用如下设置

设置一个制表符等于两个空格：

文本编辑器>Tab Size，其值改为 2

开启自动猜测文件编码：

文本编辑器>文件>auto guessing encoding，打上对勾，可有效解决文档乱码

打开自动保存：

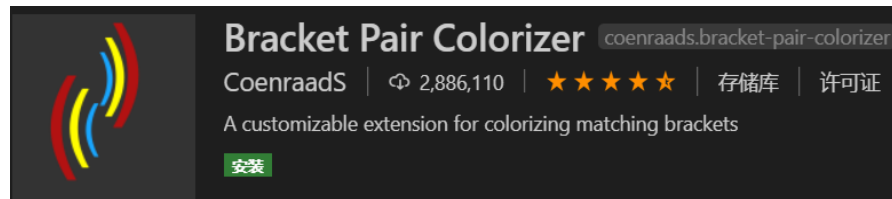
文本编辑器>文件>auto save 防止文档断电丢失

## 2.2.5 扩展商店

工欲善其事必先利其器，为了使 **vscode** 成为专业的 web 前端开发工具，下面这些高效率插件是很有必要去装的。

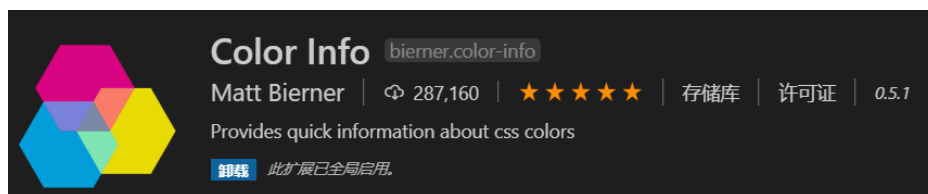
### (1) Bracket Pair Colorizer

友好的给括号加上不同的颜色，便于区分不同的区块，使用者还可以定义括号颜色和类型，尤其是 **dart** 语言开发，尤为重要



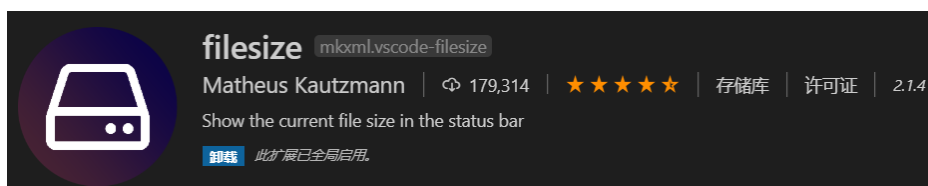
## (2) Color Info

提供你在 CSS 中使用颜色的相关信息，悬停光标，就可以预览色块中色彩模型的 HEX、RGB、HSL 和 CMYK。



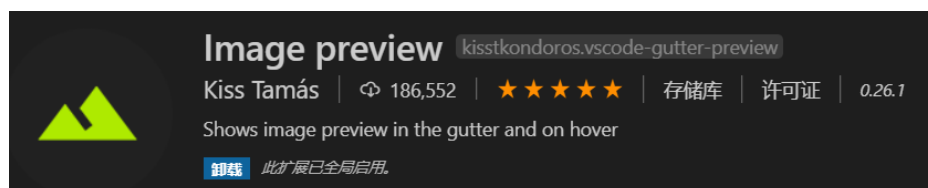
## (3) Filesize

在底部状态栏显示当前文件大小，点击后还可以看到详细创建、修改时间



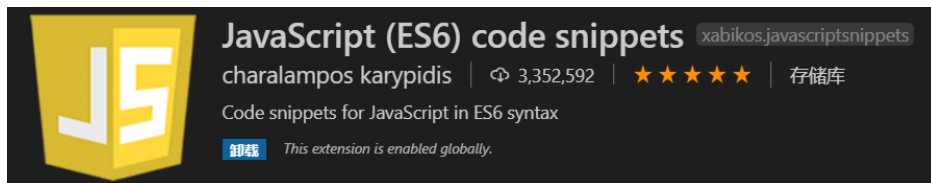
## (4) Image Preview

当鼠标悬浮在链接或者装订线（gutter）左边可以预览到图片



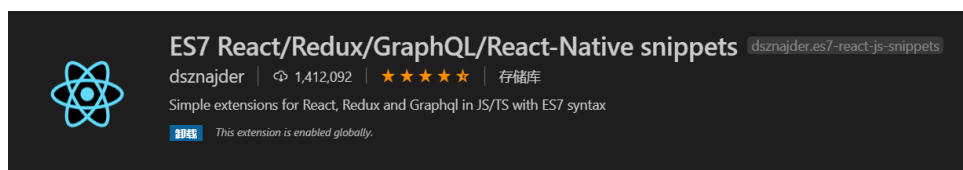
## (5) JavaScript (ES6) code snippets

ES6 语法智能提示及快速输入，不仅仅支持 .js，还支持 .ts .jsx .tsx .html .vue



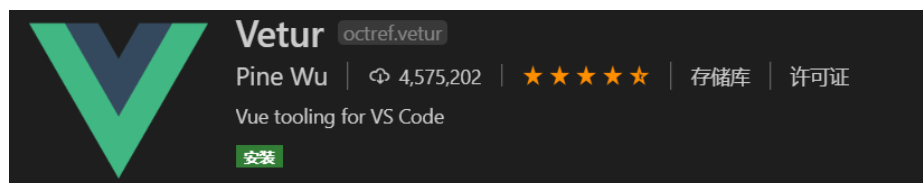
#### (6) ES7 React/Redux/React-Native/JS snippets

React/Redux/React-Native/react-router 语法智能提示,react 开发者必备。需要注意的是,如果你装这个插件的话,那么就不必再同时装 JavaScript (ES6) code snippets 了,因为这两款插件功能相似,而且本插件功能更全



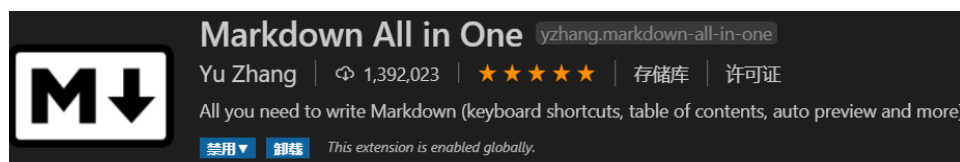
#### (7) Vetur

Vue 多功能集成插件。包括: 语法高亮, 智能提示, emmet, 错误提示, 格式化, 自动补全, debugger。Vscode 官方推荐的 Vue 插件, Vue 开发者必备。



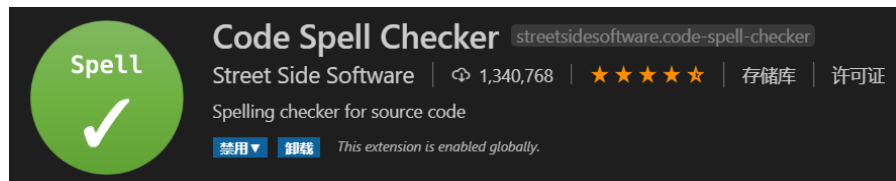
#### (8) Markdown All in One

Markdown 使用者必备, 支持目录, 书写时预览(Ctrl + Shift + V or Ctrl + K V), 可导出为 HTML 或 PDF, 可格式化表格(Alt + Shift + F) 以及 Task list (Alt + C), 支持特殊数学符号渲染, 可配合 markdownlint 使用, 十分强大。



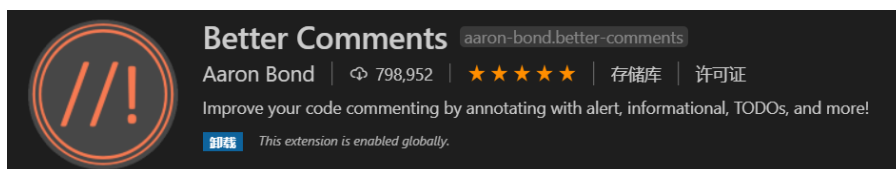
#### (9) Code spell checker

检查你的英文单词是否有拼写错误，如果有库里不存在的单词则会在下方用波浪线提示。



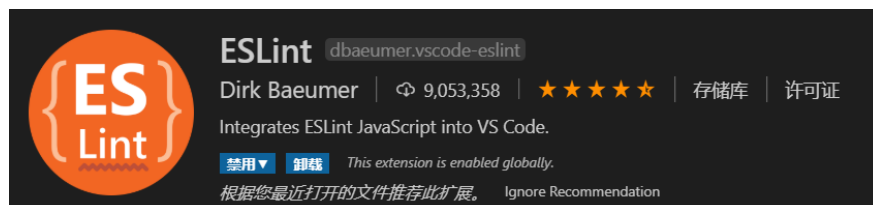
### (10) Better Comments

使注释有人性化的高亮显示，团队中非常适用，突出重点提示。符号有：\*、!、TODO、@param 等。



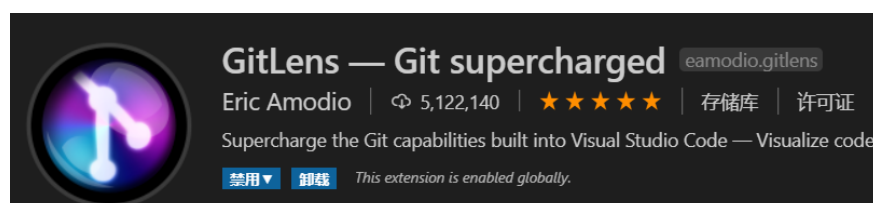
### (11) ESLint

JavaScript 语法纠错，可以自定义配置，也可以参照官网的配置，摆正开发者的代码书写格式。



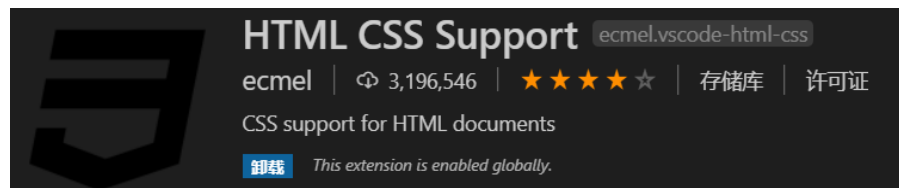
### (12) GitLens

快捷方便查看 Git 日志，高频使用 Git 者必备



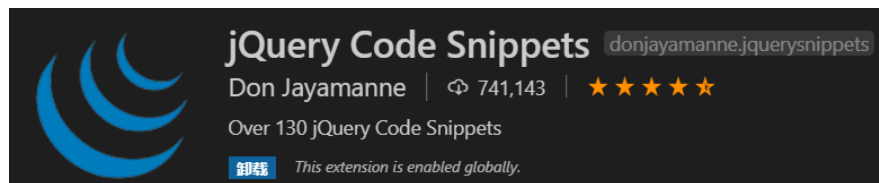
### (13) HTML CSS Support

智能提示 CSS 类名及 id 名，非常高效

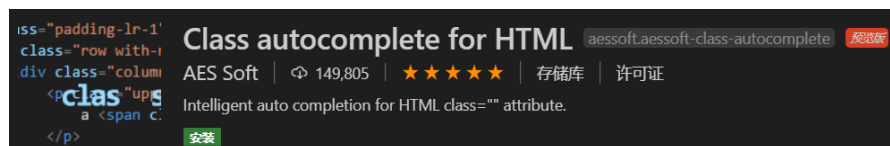


#### (14) jQuery Code Snippets

jQuery 代码智能提示，优雅且方便

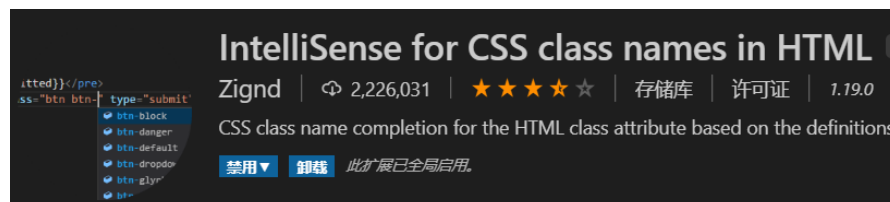


#### (15) Class autocomplete for HTML



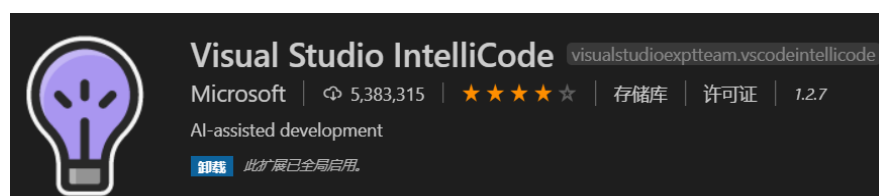
#### (16) IntelliSense for CSS class names in HTML

智能提示 css 的 class 名，无脑录入



#### (17) Visual Studio IntelliCode

给开发者提供最合适的 `intelliSense` 上下文建议功能，除此之外，还有代码格式化和规则推测功能



### 3 使用 vscode 进行前端开发

了解过上述关于 `vscode` 的基础知识并配置好编辑器后，我们就可以把 `vscode` 作为一个强大的前端 `ide` 来使用了。

下面我们以一个登陆注册的项目为例详细介绍。首先，可在磁盘上任意位置新建一个项目文件夹，如”Login”，打开 `vscode`，依次点击文件》打开文件夹，选中你要打开的文件夹。这个项目就正式加载到 `vscode` 中了。在侧边栏的资源管理器中新建如下目录及文件：

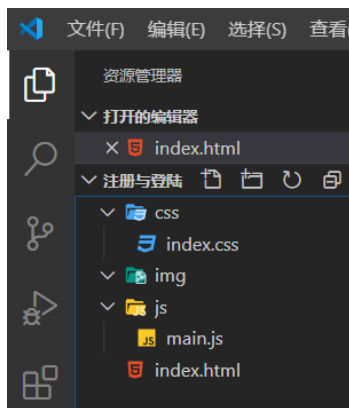


图 3.1 前端工程目录结构

其中，`index.html` 为主页入口文件，`css` 文件夹中存放外部 `css` 样式表，而 `js` 文件夹则存放外部 `JavaScript` 代码。`Img` 文件夹存放媒体图片等。

#### 3.1 使用 Emmet 快速编写 HTML 代码

`Emmet`（前身为 `Zen Coding`）是一个能大幅度提高前端开发效率的一个工具。在前端开发的过程中，一大部分的工作是写 `HTML`、`CSS` 代码。特别是手动编写 `HTML` 代码的时候，效率会特别低下，因为需要敲打很多尖括号，而且很多标签都需要闭合标签等。于是，就有了 `Emmet`，它可以极大的提高代码编写的效率，它提供了一种非常简练的语法规则，然后立刻生成对应的 `HTML` 结构或者 `CSS` 代码，同时还有多种实用的功能帮助进行前端开发。

`VsCode` 内置了 `Emmet` 语法，在后缀为 `.html/.css` 中输入缩写后按 `Tab` 键即会自动生成相应代码，其语法基本规则如下：



E	代表 HTML 标签。
E#id	代表 id 属性。
E.class	代表 class 属性。
E[attr=foo]	代表某一个特定属性。
E{foo}	代表标签包含的内容是 foo。
E>N	代表 N 是 E 的子元素。
E+N	代表 N 是 E 的同级元素。
E^N	代表 N 是 E 的上级元素。

举例如下：

### 3.1.1 生成元素

1. 新建 html 文件，保存之后，输入 “!” 或 “html:5”，按 Tab(或 Enter) 键，自动生成 HTML 结构
2. 标签只要直接输入标签名（不要输入<>），按 Tab(或 Enter) 键自动生成完整的标签。文中用 => 表示 tab（或 Enter）键

```
div => <div> </div>
foo => <foo> </foo>
```

### 3.1.2 生成文本+元素

```
div{这是一段文本}    => <div>这是一段文本</div>
a{点我点我}          => <a href="">点我点我</a>
```

### 3.1.3 生成带属性的元素

```
a:link    => <a href="http://"></a>
link      => <link rel="stylesheet" href="">
script:src => <script src=""></script>
div.test  => <div class="test"></div>
div#pageId => <div id="pageId"></div>
table>.row>.col
=>
<table>
  <tr class="row">
    <td class="col"></td>
  </tr>
</table>
```

### 3.1.4 生成嵌套元素

子级:>

通过 > 标识元素可以生成嵌套子级元素，可以配合元素属性进行连写

```
div#pageId>ul>li
=>
<div id="pageId">
  <ul>
    <li></li>
  </ul>
</div>
```

同级:+

+ 字符表示生成兄弟级元素

```
.div#pageId+div.child
=>
<div id="pageId"></div>
<div id="child"></div>
```

父级:^^

^^用于生成父级元素的同级元素，从这个字符所在位置开始，查找左侧最近的元素的父级元素并生成其兄弟级元素 如

```
.div>p.parent>span.child^ul.brother>li
=>
<div class="div">
  <p class="parent"><span class="child"></span></p>
  <ul class="brother">
    <li></li>
  </ul>
</div>
```

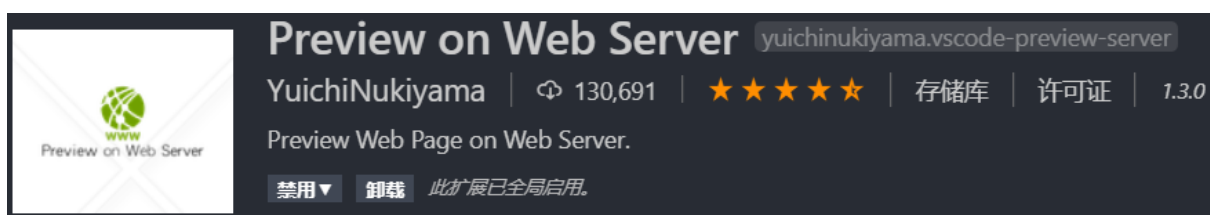
### 3.1.5 用乘法同时生成多个元素

```
li*5
=>
<li></li>
<li></li>
<li></li>
<li></li>
<li></li>
```

### 3.2 使用 preview on web server 插件调试代码

调试前端页面的方法可谓多种多样，可以直接用浏览器打开 html 文件手动刷新进行调试，但这种方法存在跨域问题，也可以写一个本地 http 服务器，把静态页面放进去调试，当然也可以利用 vscode 的插件进行调试，对于使用 vscode 开发的我们，最后一种方法无疑最为简单可靠。

在 vscode 中打开扩展商店，搜索插件 `preview on web server` 并安装。只需这一步，我们就可以开始调试 html 页面了。



打开我们要调试的 html 文件，使用快捷键 `ctrl+shift+L` 或者右键>vscode `preview-server: launch on default browser` 即可在浏览器中打开调试。每当源代码保存时，页面就会自动刷新一次。当然，如果你不使用开发人员工具而只想编写代码边看效果，这时你可以使用 `ctrl+shift+v` 或者右键>vscode `preview-server: preview on side panel` 在 vscode 中拆分出一个编辑器进行预览。

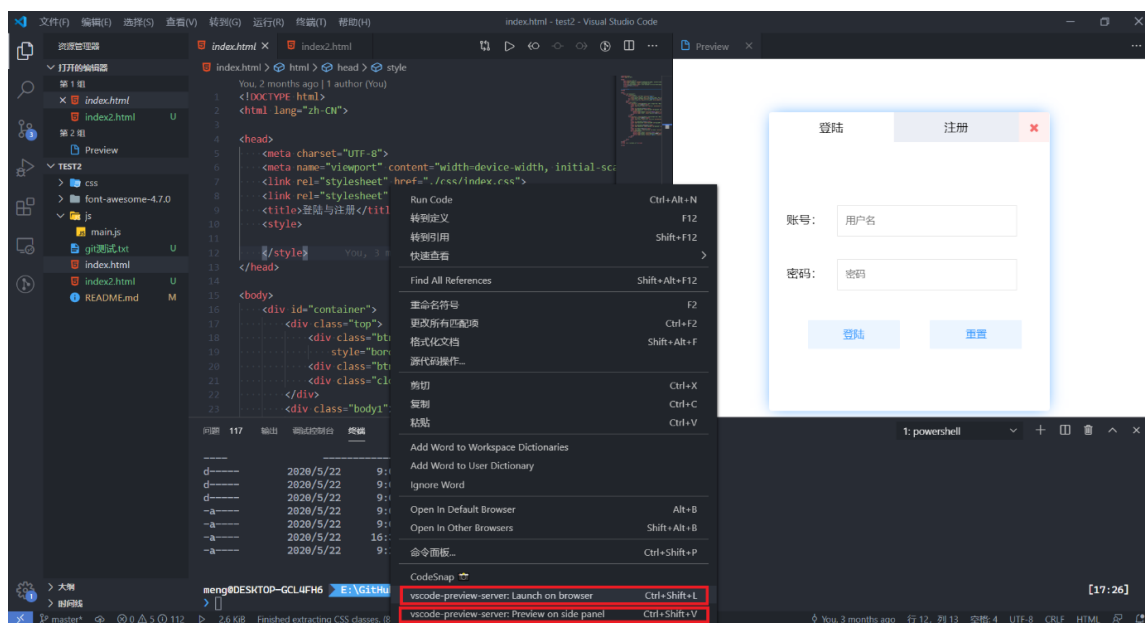



图 3.2.1 窗口内调试预览图

## 4 在 vscode 中使用 Git

在 **vscode** 默认支持 **git** 的所有功能,其侧边栏第三个图标  就是 **git** 的标志。

但使用前提是先去 **git** 官网下载安装 **git**, 这里不做详细介绍。为了使 **git** 更好用, 同时应安装 **Git Lens** 插件。

### 4.1 设置本地项目目录

安装好 **git** 及 **gitLens** 插件后, 在 **vscode** 中打开 **power shell** 终端, 点击文件> 打开或者直接使用 **code** 命令打开项目文件夹。

如: `code E:\GitHub\Login`

### 4.2 git 基础配置

初次使用 **git** 需要在命令行中进行一些简单的设置, 这些设置只需要配置一次, 而且如果你之前在电脑上使用过 **git** 命令行并提交过, 那么第 2、3、4 行代码就不必执行了。

```
git init //初始化仓库

git config user.name "user name" //git config --global 设置全局配置不
带--global 为当前项目配置

git config user.email "user email" //配置当前用户邮箱

git config credential.helper store // 配置记住用户密码

git remote add origin https://github.com/username/Login.git //配置自
己的远程代码仓库地址

//设置错误可以用 git remote remove origin 删除设置重新配置
或 git remote set-url origin 进行修改
```

### 4.3 拉取远程仓库中的代码

首先登陆 **GitHub**，进入你要拉取代码的仓库（没有仓库就新建一个），点击图标复制仓库的地址：

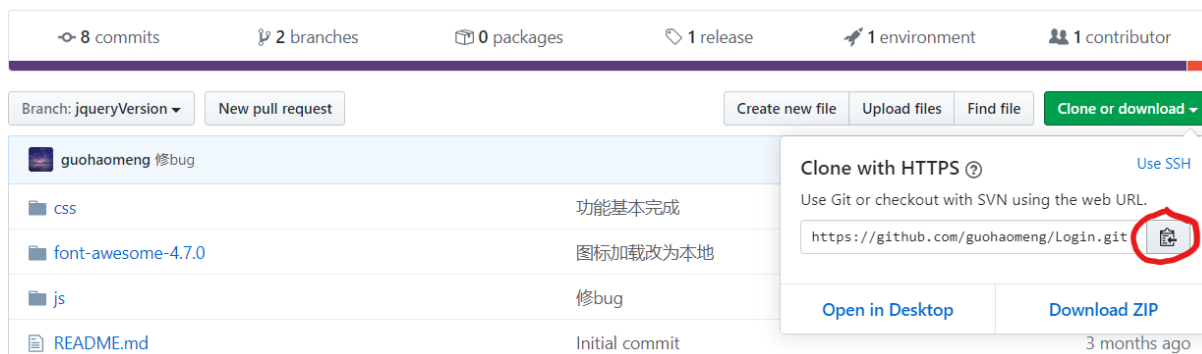


图 4.3.1 GitHub 仓库

之后切换回 **vscode** 终端，输入

```
git clone origin master https://github.com/guohaomeng/Login.git
```

这样，代码就成功克隆到本地了。

### 4.4 使用 **vscode** 对代码进行编辑

这时候你可以对仓库内容做任何更改，包括文件的增删和对代码的编辑，发生更改的文件及代码 **git** 会自动标记出来，其中 **U** 表示未跟踪文件，**M** 表示被修改文件，点击侧边蓝绿条可对比查看更改

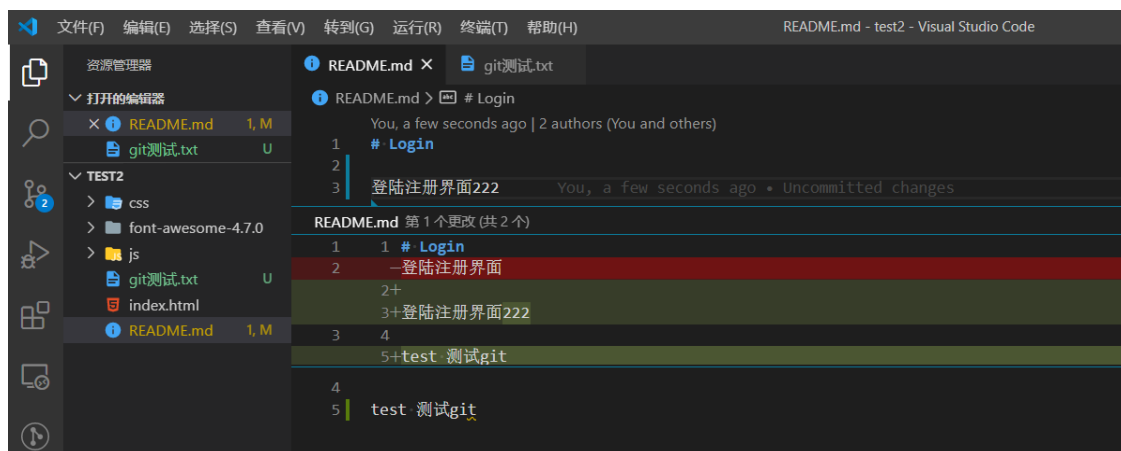



图 4.4.1 查看代码更改

#### 4.5 保存更改，填写修改注释

点击侧边栏源代码管理，点击+号暂存所有更改 ，在消息框中填写修改注释，最后点击 ✓，我们的更改就被提交了。

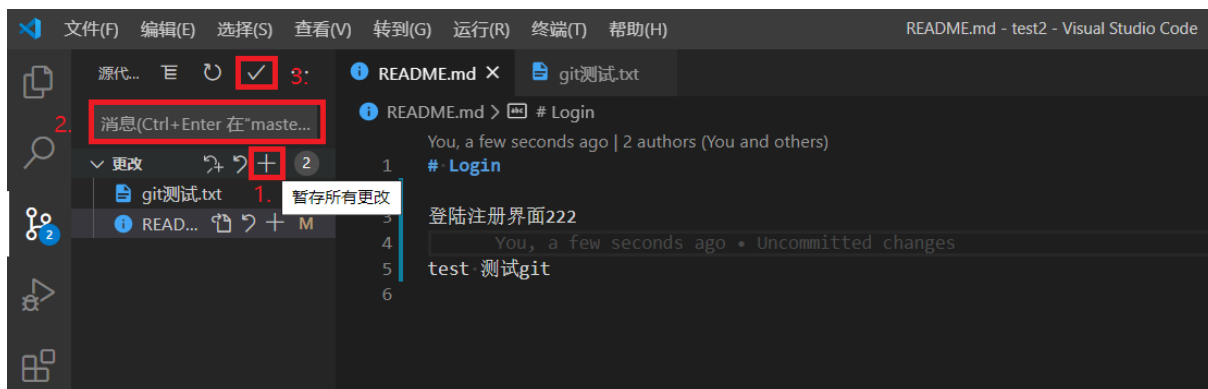


图 4.5.1 用 vscode 提交 git 更改

#### 4.6 推送到远程仓库

点击更多操作>推送到>选择远程仓库地址即可将更改推送到远程仓库。也可以使用命令行操作：

```
git push origin master
```

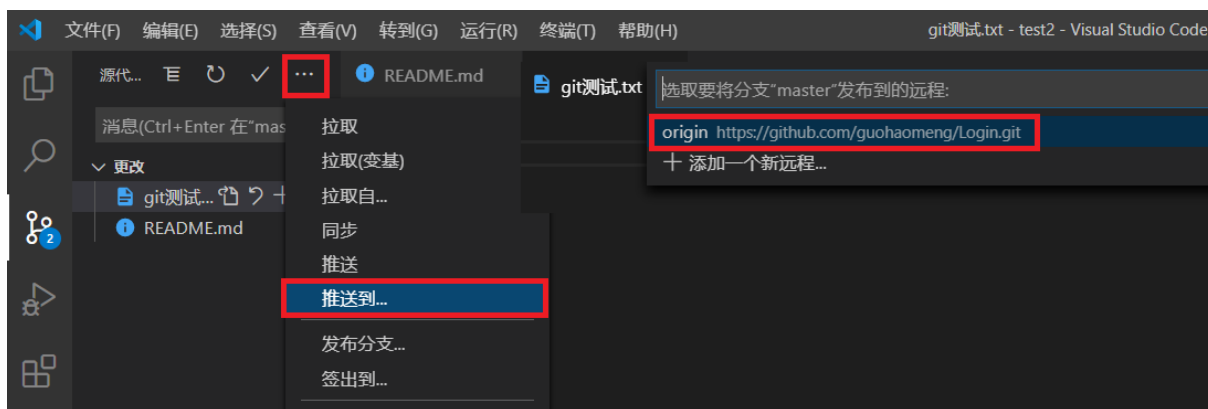


图 4.6.1 用 vscode 推送代码到远程仓库

## 5 使用 vscode 进行远程开发

所谓远程开发，就是通过 VS code 登录服务器后使用服务器的环境进行编码和调试，服务器与本地的 vscode 插件相互独立，VS code 上的所有功能都可以使用，和在本地开发基本无区别。远程开发的好处就是如果你在服务器上有了开发环境，就不用在本地上再搭建一个了，而且调试不占用本地电脑的硬件资源，有网就行。目前 vscode 这个功能支持如下服务器类型：

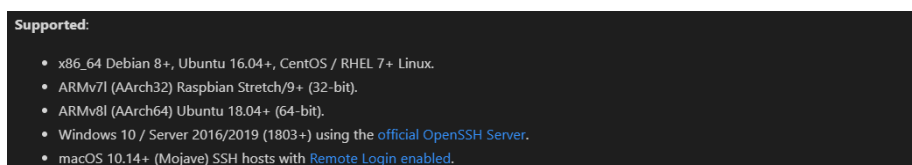


图 5 vscode 远程开发支持硬件及操作系统列表

### 5.1 配置免密登陆

因为是要远程登录，那么需要通过使用 ssh 进行密钥对登录，这样每次登录服务器就可以不用输入密码了。如果本地已经生成请跳过此步骤

打开打开 vscode 内置终端 `power shell`，输入以下命令，之后一路 `next`

```
ssh-keygen
```

然后将本地公钥文件 `id_rsa.pub` 的内容添加到远程主机用户目录下 `.ssh` 文件夹内名为 `authorized_keys` 的文件中。不用去复制粘贴，使用命令 `ssh-copy-id` 来完成

```
ssh-copy-id remote_user@remote_id
```

### 5.2 配置 vscode 远程开发插件

在 vscode 扩展商店中搜索 `Remote Development` 扩展包并安装。安装完成后侧边栏会多出一个远程资源管理器的图标，打开它，点 `+` 号新建远程连接，在弹出的对话框中填写连接命令，格式为

```
ssh -p 端口号 用户名@远程主机名 -A //端口号默认为 22
```

在弹出来的列表里选择第一个 **config** 配置文件，右下角出现 **host added!** 表示配置成功。之后侧边远程资源管理器里就会显示你的远程主机，点它名字右边的文件夹按钮就可以连接啦。

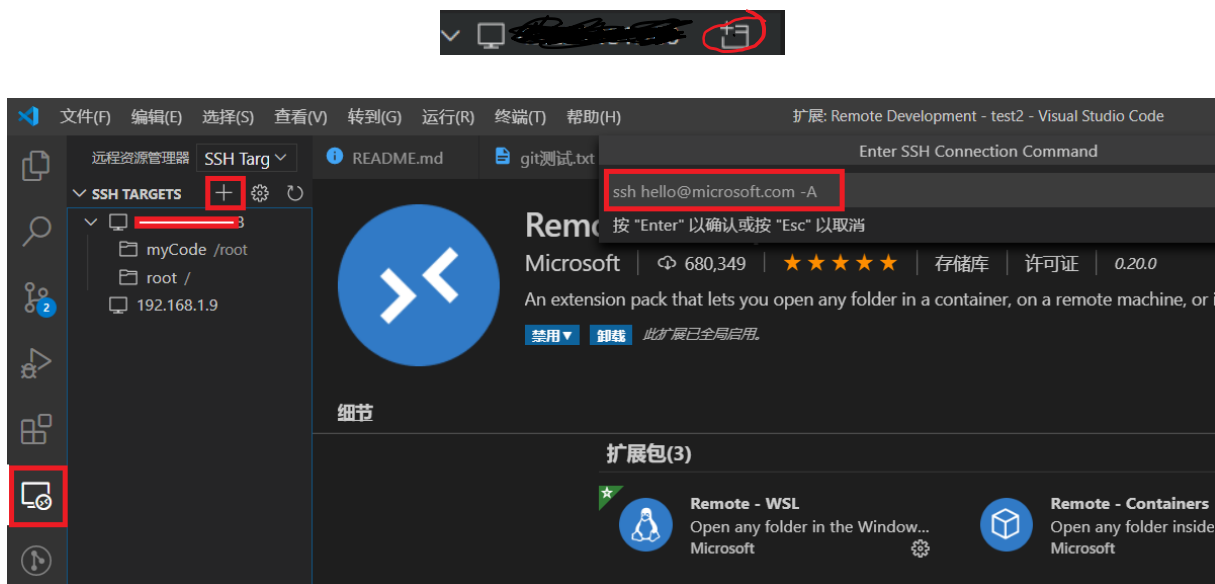


图 5.2.1 vscode 使用 ssh 连接远程服务器

最终连接成功后的界面应如图所示：

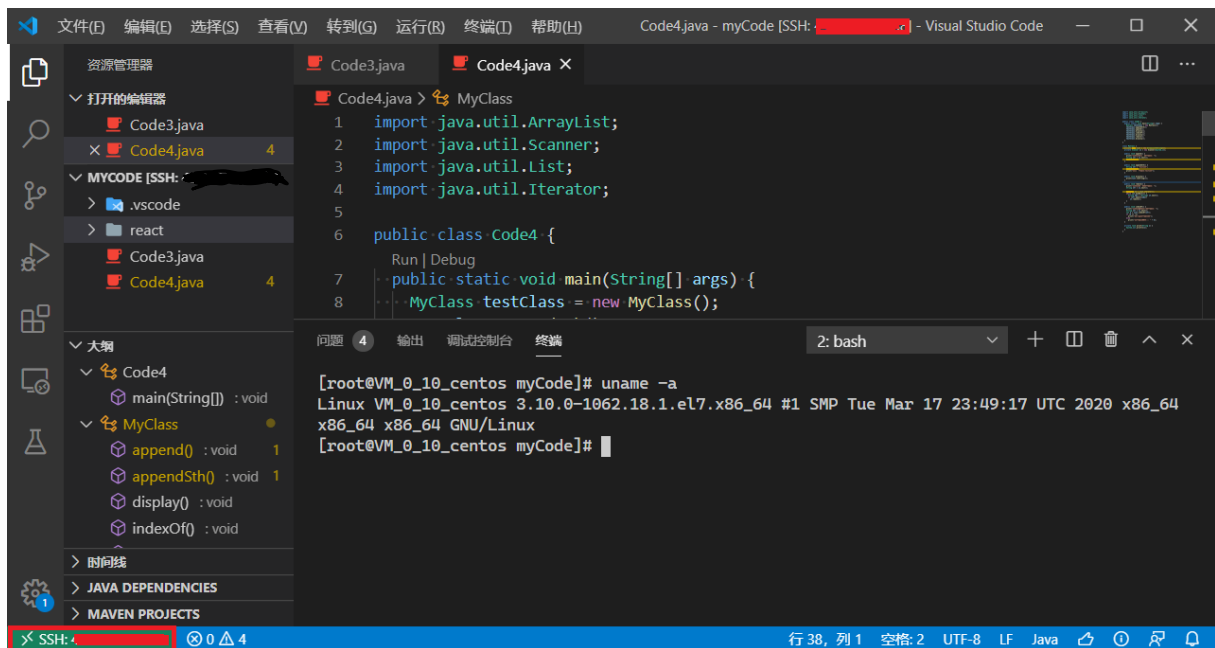


图 5.2.2 连接上远程服务器的 vscode