VS Code的正确打开方式

寒假的第一天，我准备学习VS Code的正确打开方式。我感觉我现在对整个VS Code比较nice的快捷键还认识不是很到位，只会比较基础的调试等功能，于是，就有了这份笔记。同时还有一些比较nice的东西，但可以不必在这里花费太多学习成本，到时候参考即可。

Part 1

三步走，但可能后期有的放矢：

- 最开始的时候，我特别关心快捷键和语言支持，在这上面花了很多时间，这个过程就像是打怪升级，不断更新自己的装备库；
- 再往后，我就会开始挑剔编辑器的其他组件，但凡是跟自己的工作习惯或者工作流不匹配的，就会想办法换掉它，这是个做减法的过程；
- 最后一步，就是自己学习写插件了，编辑器本身的功能和社区不能够完全满足自己的需求，可以但没必要？

Part 2 Historia

很多人都把编辑器等同于IDE，其实从专业角度来讲并非这样。IDE 更为关注开箱即用的编程体验、对代码往往有很好的智能理解，同时侧重于工程项目，为代码调试、测试、工作流等都有图形化界面的支持，因此相对笨重，Java程序员常用的Eclipse定位就是IDE；而编辑器则相对更轻量，侧重于文件或者文件夹，语言和工作流的支持更丰富和自由，VS Code 把自己定位在编辑器这个方向上，但又不完全局限于此。

2011 年底，微软从 IBM 请来了 Erich Gamma。Erich Gamma 是《设计模式》一书的作者之一，曾和肯特·贝克（Kent Beck） 一起发明了 JUnit，并且在 IBM 领导 Java 开发工具的开发工作。微软把他请过来，就是希望他能够打造一款在线的开发工具，让开发者们能够在浏览器里获得 IDE 般的开发体验，这也就是之后为人所知的 Monaco Editor。

Erich Gamma 见证了 Eclipse 从崛起到逐渐臃肿，再逐渐式微的整个历程，他深刻认识到 Eclipse 成功的一部分原因是极度的可定制化特性，任何功能在Eclipse中都可以用插件来实现；但是由于 Eclipse 的插件跟核心代码运行在同一个进程内，随着插件的增多，核心功能经常会被插件拖累，也就更加让人觉得笨重。

因此，在打造 Monaco Editor 时，开发团队非常注重核心功能的性能，尽可能地保持轻量，而对资源和性能消耗较大的功能，则运行在其他的进程之中。2015 年，Erich Gamma 带领团队把 Monaco Editor 移植到桌面平台上，也就有了**VS Code**。

Part 3 TERMINAL

- CTRL shift P to open command panel
- VS Code 强调无鼠标操作 【Terminal】
- code --help
- code (-r) [Directory name]
- 你也可以使用参数 -g <file:line[:character]> 打开文件，然后滚动到文件中某个特定的行和列，比如输入 code -r -g package.json:128命令，你就可以打开 package.json 这个文件，然后自动跳转到 128 行。这个命令告诉我们某个文件的某一行出现了错误，我们就能够快速定位了。
- VS Code 也可以用来比较两个文件的内容，你只需使用 -d参数，并传入两个文件路径，比如输入 code -r -d a.txt b.txt命令，就可以比较a.txt和b.txt两个文件的内容了。有了这个命令，你就可以既使用命令行

运行脚本，也可以借助 VS Code 的图形化界面进行文件内容的对比了。
- 你可以把当前目录下所有的文件名都展示在编辑器里，此时只需使用ls | code -命令。
- 另外 VS Code 的命令行的各个参数，其实能够定制 VS Code 是怎样运行的，比如--disable-extensions、--max-memory，它们都有特殊的应用场景。

Part 4 No mouse?

CURSOR

- 同时按住 Option 和方向键，那么光标移动的颗粒度就变成了单词，你就可以在文档中以单词为单位不停地移动光标了
- 第二种方式是把光标移动到行首或者行末，使用**HOME/END**
- 接下来一种是对于代码块的光标移动。很多编程语言都使用花括号将代码块包裹起来，比如 if、for 语句等，你很可能会希望通过一个快捷键，就能实现在代码块的始末快速跳转。Ctrl + Shift + \
- 掌握了上面的快捷键之后，你还可以非常轻松地掌握文本选择的操作。因为对于基于单词、行和整个文档的光标操作，你只需要多按一个 Shift 键，就可以在移动光标的同时选中其中的文本。
  Ctrl+Shift+left/right/up/down **DELETE**
- 命令面板里面delete all left/right 的选项
- 快速删除：Cmd + Shift + K
- 先选中后删除
- 剪切和一般无二 **OTHERS**
- “另启一个新代码段”：ctrl+enter ，而不是enter然后摁方向键
- ALT＋上/下实现上下移动功能+（shift可选复制）
- 代码格式化ALT+shift+f调整代码风格，也可以先选中再格式化
- 代码左右缩进 **ANNOTATION**
- ctrl+/
- 全部注释：先选中后注释 彩蛋
- 第一个是调换字符的位置。你可以按下 “Ctrl + t” （Windows 上未绑定快捷键，可以打开命令面板，搜索 “转置游标处的字符”）来把当前光标前后的字符调换位置。 - 大小写转换，使用命令面板的transform功能
- 光标位于前者，合并代码行，命令面板的joint lines
- 第四个是行排序Sort。无论是你在写代码，还是写 Markdown，你都可以把代码行按照字母序进行重新排序。不过这个命令比较小众，VS Code 并没有给这个命令指定快捷键，你可以调出命令面板，然后搜索 “按升序排列行” 或者 “按降序排列行” 命令执行。

PART 5 多光标

在我们的日常编码过程中，有很多工作，它本身就是具有“重复”属性的。比如你需要把多个单词的第一个字母从小写变成大写，这种跟业务逻辑相关的重复性操作，编辑器很难为它们一个个单独做优化。 而 VS Code 的多光标特性其实就是用来解决这类问题的。当你在一个文本框或者某个输入框里打入字符时，会有一个竖线来显示你将要输入文字的位置，这就是“光标”。顾名思义，多光标其实就是多个输入位置，这里你可以脑补下多个竖线的场景。 **创建多个光标** 以一段CSS代码为例子：

```
.foo {
  padding: 5;
  margin: 5;
  font-size: 5;
}
```

你可以看到，在上面这段 CSS 代码中，所有属性的值都是“5”，但你可能觉得这样的写法不规范，想把它们都改成 “5px”。之前你肯定是吭哧吭哧挨个在5后面加“px”。而现在，有了多光标特性之后，你第一步要做的事情，就是把光标移动到第一个 “5”的前面。接下来就有两种操作方式可以选择。

1. 使用鼠标 第一种添加多光标的方式，就是使用鼠标。在键盘上按住 “Option”（Windows 上是 Alt），然后鼠标点在第二个“5”之前，那么第二个光标就创建好了。现在你可以看到两个光标。
2. 键盘三步走

- Ctrl + Alt + 下方向键，在当前光标的下面创建一个光标
- 移动到行尾
- 调整位置，加上‘px’

3. 重复ctrl+D：首先获取最近的单词，然后匹配到下面最近的同样单词，创建另一个光标
4. VS Code 中还有一个更加便捷的鼠标创建多光标的方式。当然，这首先要求你的鼠标拥有中键。你只需按下鼠标中键，然后对着一段文档拖出一个框，在这个框中的代码就都被选中了，而且每一行被选中的代码，都拥有一个独立的光标。

Part 6 跳转

文件跳转

- ctrl+tab，然后松开tab
- ctrl+P，选择最近打开文件
- （如果选中了一个文件还没摁下enter，可以摁下ctrl+enter进行同页面分割） **行跳转** 打开某一个文件之后，你的另外一个需求可能就是要快速跳转到这个文件的某一行。你可能会想，VS Code是不是可以像Vim那样，输入“:13”就能跳转到第13行。是的，VS Code也提供了一种极为简单的方式来支持行跳转，你只需要按下 “Ctrl + g”，紧接着编辑器就会出现一个输入框，输入即可。 **文件行跳转** 如果你想跳转到某个文件的某一行，你只需要先按下 “Cmd + P”，输入文件名，然后在这之后加上 “:”和指定行号即可。 **符号跳转** 文件跳转和行跳转，是代码跳转的基本操作，也是日常编码中的高频操作。不过有的时候，你可能会希望能够立刻跳转到文件里的类定义，或者函数定义的位置。为了支持这种跳转，VS Code 提供了一套 API 给语言服务插件，它们可以分析代码，告诉 VS Code 项目或者文件里有哪些类、哪些函数或者标识符（我们把这些统称为符号）。 如果要在一个文件里的符号之间跳转，你只需按下Ctrl + Shift + O，就能够看到当前文件里的所有符号。 请注意，在按下 “Cmd + Shift +O”后，输入框里有一个 “@”符号，这个符号在这里的意义，现在还不重要。这时，如果你输入 “:”，就可以将当前文件的所有符号，进行分类，这样搜索符号也就更加方便。有些语言除了提供单个文件里的符号，还支持在多个文件里进行符号跳转。比如在 VS Code 里，如果你打开了多个 JavaScript 文件，就可以按下 “Cmd + T” （Windows 上是 Ctrl + T），搜索这些文件里的符号。 **定义与实现跳转**
- CTRL+F12 **引用跳转** 很多时候，除了要知道一个函数或者类的定义和实现以外，你可能还希望知道它们被谁引用了，以及在哪里被引用了。这时你只需要将光标移动到函数或者类上面，然后按下 “Shift + F12”，VS Code 就会打开一个引用列表和一个内嵌的编辑器。在这个引用列表里，你选中某个引用，VS Code 就会把这个引用附近的代码展示在这个内嵌的编辑器里。

Part 7 Mouse

- 在VS Code中，你单击鼠标左键就可以把光标移动到相应的位置。而双击鼠标左键，则会将当前光标下的单词选中。连续三次按下鼠标左键，则会选中当前这一行代码。最后是连续四次按下鼠标左键，则会选中整个文档。到这里你可能会问，如果我想要使用鼠标，选中其中的多行代码该怎么办？VS Code也

考虑到了这个情况，在编辑器的最左边，显示的是每一行的行号。如果你单击行号，就能够直接选中这一行。如果你在某个行号上按下鼠标，然后上下移动，则能够选中多行代码。

- 能不能使用鼠标进行“复制+粘贴”呢？别担心，VS Code 肯定也会考虑到这个情况的，所以答案是：必须能。如果我们在拖拽这段文本的同时，按下 Option 键（Windows 上是 Ctrl 键），鼠标指针上会多一个加号，这时候我们再移动鼠标或虚拟光标至我们想要的位置，然后当我们松开鼠标左键的时候，这段文本将会被复制粘贴到虚拟光标所在的位置，也就是我们既定的目标位置。你看，在移动鼠标的过程中，多按了个 Option 键（Windows 上是 Ctrl 键），操作结果就由原来的“剪切+粘贴”变为“复制+粘贴”了。

Part 8 自动补全等

- 当我们找到了合适的函数后，按下 Tab 键或者回车键就可以将其补全
- 有的时候，当我们看到一个建议列表里的某个函数名，我们可能并不能够立刻想起它的作用是什么，它的参数定义是什么样的。这时候我们可以单击当前这一项建议的最右侧的蓝色图标
- Quickfix
- **重构**：选择函数名，然后F2或者Rename，修改其名字，则调用处也跟着一起改