02讲VSCode的Why、How和What

从这一篇文章开始，我就要切入正题聊VS Code了，VS Code的全称是Visual Studio Code，但这全名实在是太长了，我和很多用户一样，喜欢叫它VS Code。说起VS Code，官方定义它是一个免费的、开源的跨平台编辑器。之所以强调“编辑器”，我想是因为 VS Code 并无意成为一个全尺寸的集成开发环境，也就是IDE。

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。

如果你有兴趣，可以打开自己喜欢的编辑器官网看看它是怎么样的定位。总体来说，近几年流行风向是轻量的编辑器，这也是大势所趋。

要理解VS Code代码编辑器的设计思路，就需要先看看VS Code的发展轨迹。从我的角度看，不管你是学习编程语言，还是框架、编辑器，都应该先去看看它的来龙去脉，了解它们是怎么发展而来的，曾经遇到了什么问题，又是怎么解决的，这些信息都便于你从大局上提高对事情本质的认识。

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，也就是这个专栏的主角 Visual Studio Code，即 VS Code。

VS Code 继承了 Monaco Editor 的设计原则，其核心是做一个高性能的轻量级编辑器；个性化的功能，则交给插件系统来完成。这一点可以说是师承 Eclipse，但同时又吸取了 Eclipse 的教训，把插件系统运行在主进程之外，高度可定制但同时又是可控的。

与此同时，VS Code 也有自己的使命，那就是让开发者在编辑器里拥有 IDE 那样的开发体验，比如对源代码有智能的理解、图形化的调试工具、版本管理等等。

不难发现，VS Code 希望在编辑器和 IDE 之间找到一个平衡。在这样的设计思路下，你打开编辑器，不需要创建任何的项目工程文件就可以开始使用，并高效便捷地操作文本；同时在编程语言插件的支持下能够得到语法检查、智能提示；你还可以借助丰富的插件 API 拓展 VS Code 以满足自己的需求。

要达成这样的目标，难度可以说是非常大的，但 VS Code 取得了不错的成果。究其原因，在我看来就是微软打造了一个开放的平台。虽然有“马后炮”的嫌疑，但让我们一起来看看这样的一个开放平台是怎么助力 VS Code 的吧。

开源与开放的平台

首先，VS Code 的源代码以 MIT 协议开源。这不仅意味着大家能够免费获取到 VS Code 的核心代码，更意味着社区能够基于 VS Code 的代码，开发自己的产品。

业界现在比较知名的基于 VS Code 的项目有 SourceGraph、StackBlitz、CodeSandbox 等，这些产品可以提供非常接近VS Code 的开发体验，而VS Code 也经常从它们身上吸取技术和产品层面的宝贵经验。

其次，开发过程和反馈渠道的开放。 VS Code 源代码托管在 GitHub 上，同时使用 GitHub 管理项目的开发计划和测试，每个用户都可以在 GitHub 上了解到 VS Code 的开发进度。与此同时，GitHub 也是 VS Code 唯一的反馈渠道，开发团队根据反馈的影响程度进行统筹安排。作为用户，你可以近乎实时地跟开发团队进行交流，了解产品的发展情况。

再次，接口的开放。VS Code 自带了 TypeScript 和 Node.js 的支持，用户下载 VS Code 后，立刻就能够在书写 JavaScript 和 TypeScript 时获得智能提示，而且无需任何配置即可立即调试 Node.js代码。VS Code 核心团队有 Node.js 高手，TypeScript 也是微软官方出品的，VS Code 能把对这两个语言的支持做好，似乎并不是什么值得惊讶的事情。但是 VS Code 团队不可能精通所有语言，对于他们不熟悉的语言，VS Code 该怎么支持呢?

最好的办法莫过于**把专业的事情交给专业的人来做**。为此，VS Code 为编程语言工作者提供了统一的 API ，即Language Server Protocol 和 Code Debugging Protocol，每种语言都能够通过实现两个 API 在 VS Code 上得到类似 IDE 的开发和调试体验。

而且 VS Code 也并没有因为 TypeScript 是微软嫡出就给开小灶，而是对大家都一视同仁，TypeScript 能够得到的支持，其他语言一个也不落下。比如 Rust 的语言支持，就是由 Rust 官方团队开发和维护的，他们可以说是这个世界上最懂怎么给 Rust 做语法支持的一群人了。

在这样的平台上，编辑器开发者、编程语言工作者和社区，各自做自己最擅长的事情，把份内事做到极致。同时，从开发到测试，再到用户反馈都是公开透明的，每个人都能参与其中，把产品往自己希望的方向推进。VS Code 的技术实践和成果，最后也以开源的形式回馈给社区，让大家都能够借助 VS Code 去打造自己的产品，一起成功。

VS Code 学习指南

简短地了解了 VS Code 的历史后，如果你也认同它的设计哲学和使命，你肯定还想知道该如何把 VS Code 的这一套转化为自己的内力。我在第一讲 “学编辑器，到底应该‘学’什么？” 里讲过编辑器学习的通用办法，在 VS Code 身上也是适用的。你

可以按照以下三个步骤来逐步掌握 VS Code。

1. 核心编辑器的使用。VS Code 有一套自己的快捷键，你可以通过快捷键的学习了解核心编辑器所支持的功能。同时， VS Code 允许自定义快捷键的映射，如果你有自己熟悉的一套快捷键操作，也可以无缝地在 VS Code 上使用。除了快捷键，VS Code 对鼠标操作、多光标、搜索都有完备的支持；在编程语言的支持上面，VS Code 也向 IDE 看齐，自动补全、代码片段等一应俱全。掌握了核心编辑器，VS Code 就能够胜任你的日常通用编辑器。

2. 工作台、工作区的使用。VS Code 中除了编辑器区域，还有很多其他的功能，像是资源管理器、跨文件搜索、插件管理等，它们一起组成了统一的界面，我们称之为工作台。这个工作台的设计，代表了 VS Code 对工作流的选择。内置的软件版本管理，终端模拟器，调试器等，掌握这些 VS Code “钦定”的工具，进一步提升工作效率。

3. VS Code 定制和插件开发。作为一个百万级别用户量的工具，很多功能的默认设置不可能满足每个人或者每个工作场景，你可以学习如何定制 VS Code 的各个部件，不能永远按部就班；对于 VS Code 没有实现的功能，还可以学习一下如何使用 JavaScript 书写插件，把自己的想法，变成工具的一部分。

通过这三个步骤，你在使用 VS Code 时就能够“随心所欲”了。除此之外，我也建议你关注 VS Code 每月的发布更新日志，官方团队会详细讲解每个版本新增的功能。VS Code 的官方博客也非常值得订阅，团队成员会经常分享开发过程的心得感悟，算得上是最前沿的技术分享。

另外，我想你还会问我VS Code支持哪些编程语言，我这里就不列举了，你可以去官网看看，除了JavaScript，其实Java、Go、Python、PHP等后端编程语言也都在VS Code的考虑范围之内，并且我个人认为做得也足够专业。

好了，关于VS Code的why、how和what就聊到这里，我再简单总结下：VS Code的定位是轻量级的代码编辑器，它综合了Eclipse等很多优秀开发工具的优势，同时，也解决了它们的痛点问题，在性能、语言支持、开源社区方面都做得不错，所以一经发布也就受到了社区的喜欢，如果你经常上Reddit，也能看到它经常霸占Reddit头条。

其实我今天聊到的why、how和what也是一个经典的问题思考模型，外界称之为黄金圆环法则，如果你不知道，可以去谷歌查查，这里就不再赘述。

留个思考题，你之前都用过哪些编辑器或者IDE呢？你理解它的why、how和what么？欢迎在留言区与我分享。

参考链接：

戳此查看VS Code 官方博客

戳此查看VS Code 更新日志

戳此查看Erich Gamma 在 Goto Conference 上对 VS Code 的介绍

精选留言

谢小二

用clion写cpp服务端代码，基本上总不带调试功能，只是当编辑器用，感受就是太卡了，我i7，16g的电脑，经常都会转菊花，因此才来投奔vscode。但发现居然不支持查找所有引用

2018-09-15 09:03

scott

emeditor，世界上最快的文本编辑器，快捷键与vs很接近，所以一用就喜欢了，甚至很多快捷键是他现有然后vs才有的。

他的宏可以用vbs或js写，无帮助就能自行上手。

正则也是用他学的，筛选功能很强大

现在他已经是我思维的一部分了。

2018-09-15 09:11

作者回复

赞　VSCode 还没有支持宏

2018-09-18 11:08

李博越

作为个Java程序员，今天试了下vsc，简直太难用了，各种编译问题都要配置，网上资料还不全，现在连个main都没跑起来

2018-09-15 20:06

作者回复

可以试试 https://github.com/redhat-developer/vscode-java/

2018-09-18 11:05

QVoid

说vscode不支持查找所有引用的那位，试试安装gtags

2018-09-18 18:32

作者回复

赞

2018-09-20 10:14

linhaixiaohuo

是否可以做一个快捷键功能及相关代码实现的福利视频。加深对快捷键的理解。我相信从代码中我们能够更好的吸收为什么开发者对快捷键设定的思考方式。

2018-09-17 10:03

鹏

正在研究vsc如何保存以后，自动上传文件到服务器。。

2018-11-22 22:19

王相博

pyLint 是个什么。一直报错的用不了

a = input('hahha') 不知道味很

2018-09-20 18:10

曾祥荣

请问一下老师，在VS Code编辑文件后保存能直接标识这个文件已修改嘛（类似atom）？现在还得切到git栏目刷新一下，如果可以请问在哪设置?

2018-09-17 22:30

作者回复

默认情况下，在左侧资源管理里，能看到被修改的文件被高亮了

2018-09-18 11:15

Harry

今天算是正式了解了 VS Code 的历史。

2018-09-17 12:47

我叉巨魔

追剧

2018-09-15 11:22