16讲怎么在编辑器里做好版本管理

软件版本管理（Version Control）的重要性，恐怕已经是一个老生常谈的问题。无论是团队开发，还是个人项目，都能够从版本管理中获利。其中，团队可以通过版本管理有效地处理团队成员各自代码之间的冲突，管理代码历史和发布；个人则可以借助版本管理实现代码的备份，通过版本变化历史记录掌握自己的工作进展等等。

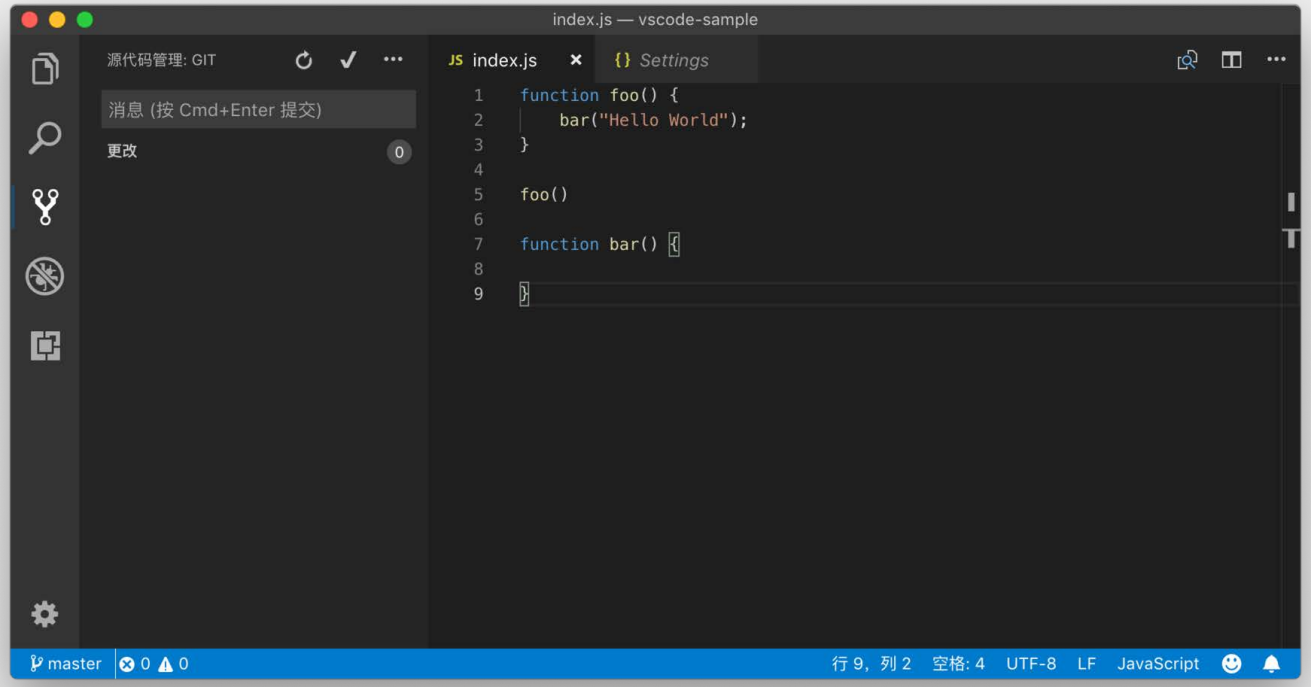
Git 以及 GitHub、GitLab 等在线代码托管服务，更是将分布式的开发协作和版本管理推到了极致。假如你还没有使用过版本管理，那么请先花点时间学习研究一下某个相对主流的版本管理软件。比如 Git、Subversion、Mercurial、CVS、Perforce，它们各有各的历史使命和特点。如果一定要我做个推荐的话，我觉得Git 或者 Subversion 都值得尝试。

不过，版本管理本身不是我们今天学习的重点，我们要研究的是如何在 VS Code 中使用版本管理，以及如何借助 VS Code 的 UI 让版本管理更便捷。今天的文章里的例子，会使用 Git 和 Subversion 来进行讲解，但 VS Code 里给版本管理的提供的 UI 都是一致的，所以如果你使用的是别的版本管理软件，也不必担心。

如果你使用的是 Git ，那你无需做任何设置，VS Code 自带了 Git 的版本支持。如果你是使用 SVN，则需要先下载安装插件。

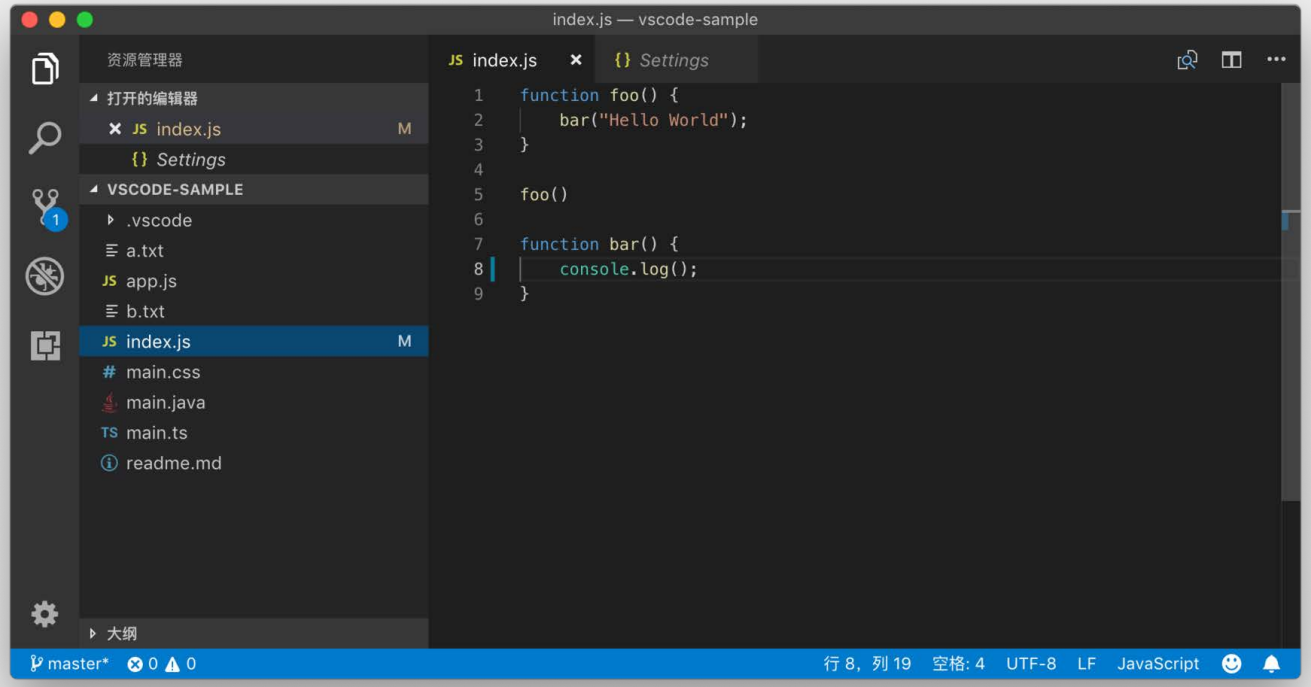
版本管理视图

首先，在工作台的左侧活动栏上，我们能看到一个版本管理的图标。



左侧活动栏，版本管理图标

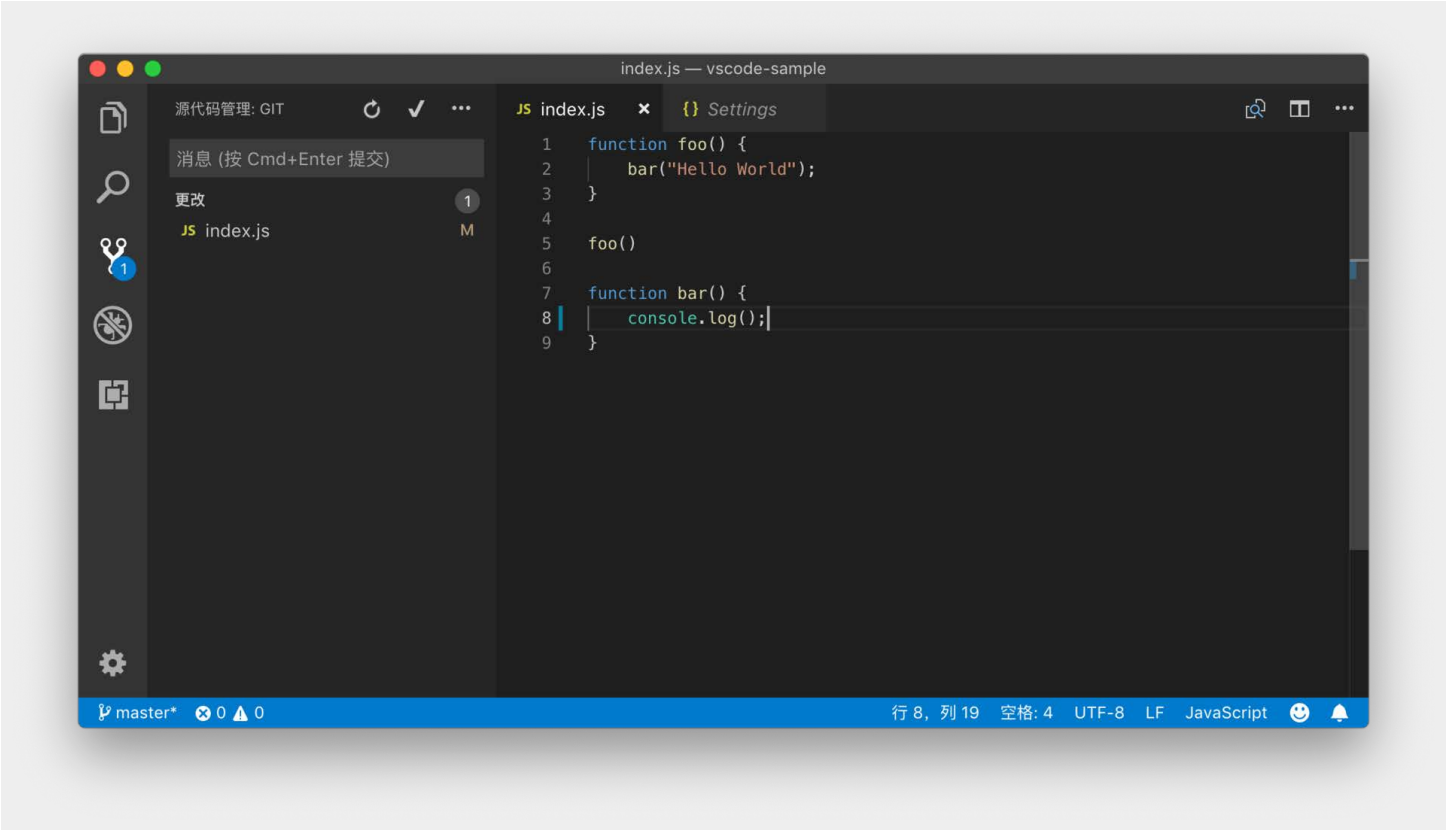
如果在当前的文件夹下有尚未被提交的改动，我们能够在这个图标上面看到一个数字。数字代表了有多少个文件被修改了，只是虽改动但还未被提交。



一个文件被修改了

当我们点击这个图标，或者是按下 Ctrl + Shift + G 快捷键，就能够打开版本管理这个视图。这个视图的最上方是一个多行输入框，用于输入代码提交时候的描述内容。在这个输入框的下面，我们则能够看到各种不同状态下的代码改动，按照它们的不同状态被归类到不同的列表中。

比如在下图中，我们只有一个更改：

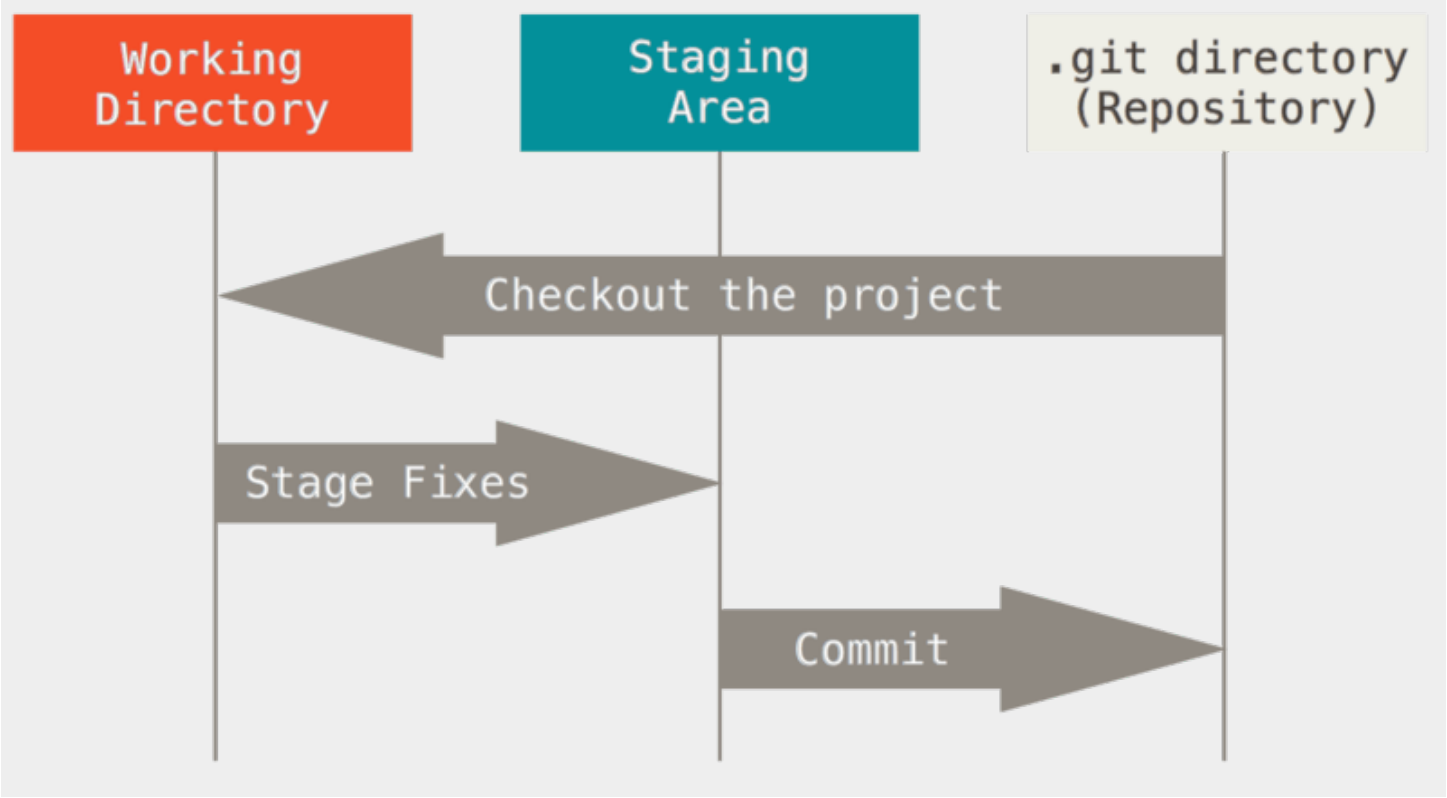


代码改动

不同的版本管理有不同的状态管理方式，不过 VS Code 的版本管理界面为它们提供了统一的界面，并且相同状态下的文件会被归类在一起。

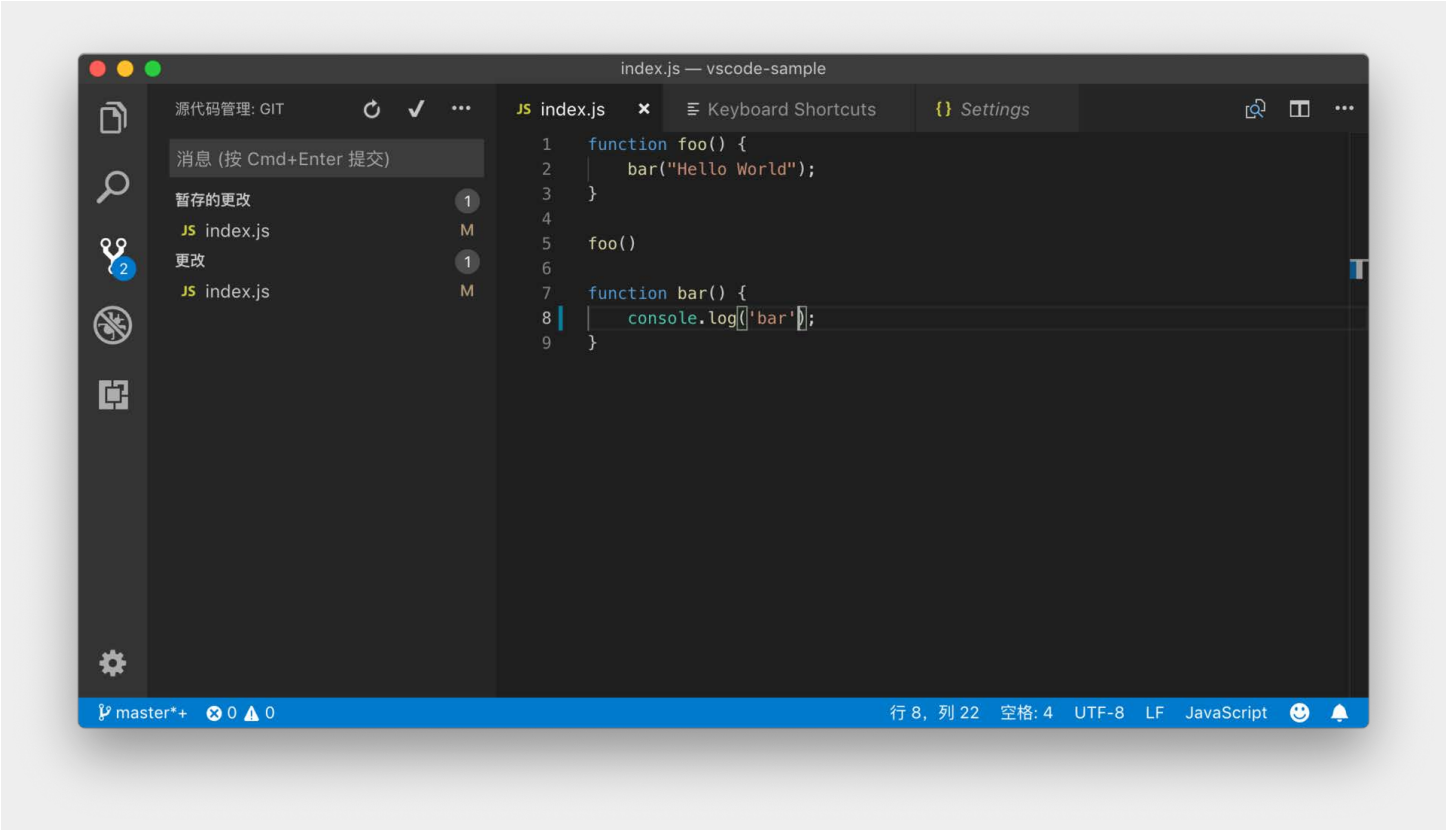
我们先拿 Git 来举例。比如说 Git 中有三种主要的文件状态：已提交（Committed）、修改（ModiÞed）和暂存（Staged）。这里借用 Git-SCM 文档 Git - Git Basics 里的一张图来解释，一个文件或者修改，能够在这三个状态直接切换。

当我们修改了一个文件，它会变成修改状态（ModiÞed）；然后我们可以通过脚本 “git add ${Þlename}” 把这个文件的状态改为暂存（Staged），被标记为暂存状态的文件，才有机会被提交。最后我们可以通过 “git commit” 来提交所有在暂存状态里的文件。



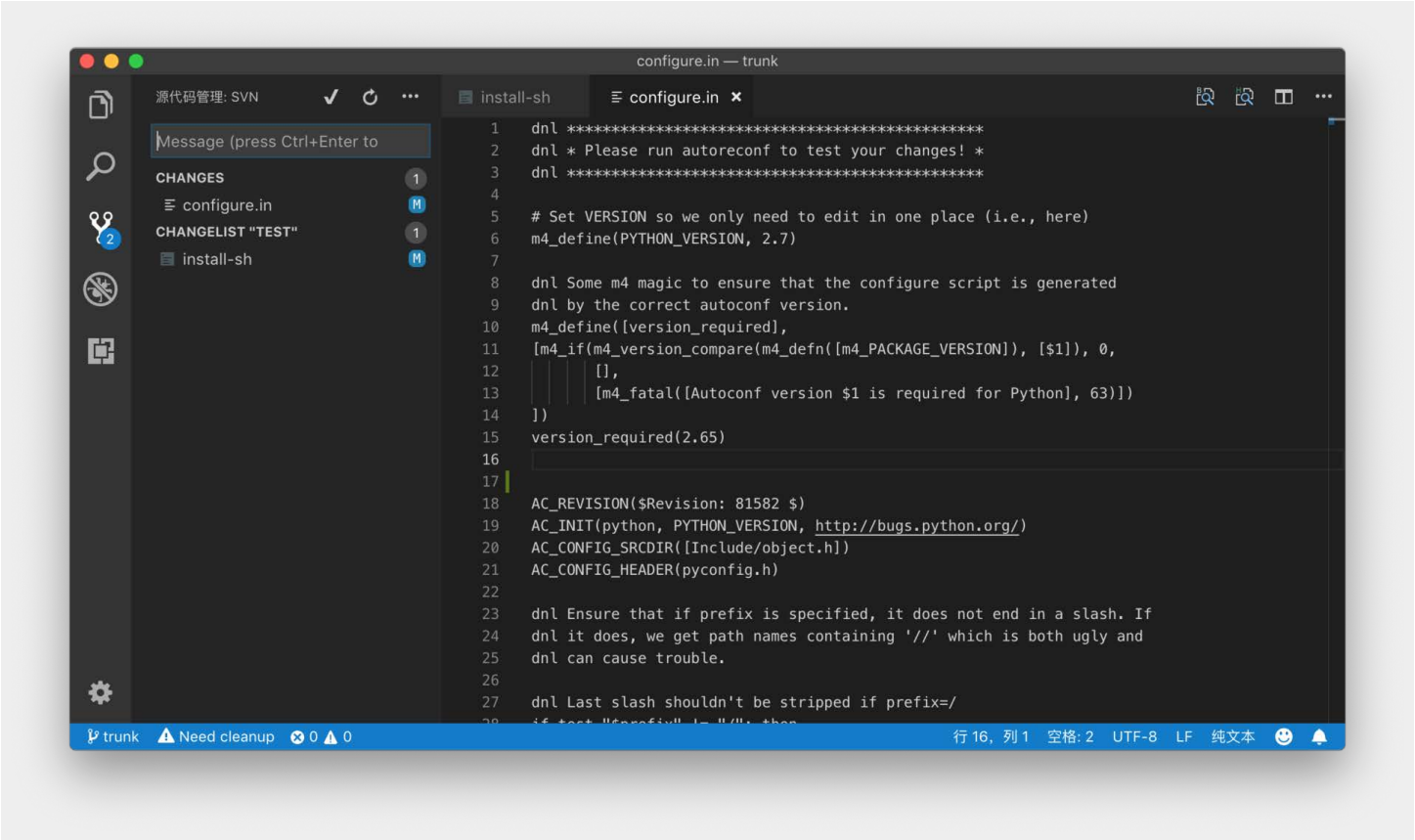
Git 里状态变化

回到 VS Code 的版本管理界面，在下图里，我们能够看到两种不同的状态。第一个是 “暂存的修改”，也就是所有处于暂存状态的修改，它们都有机会被提交；第二种则是普通的修改。此时侧边活动栏上的版本管理图标已经显示了数字 2，它提示我们当前项目里有两个不同的修改。



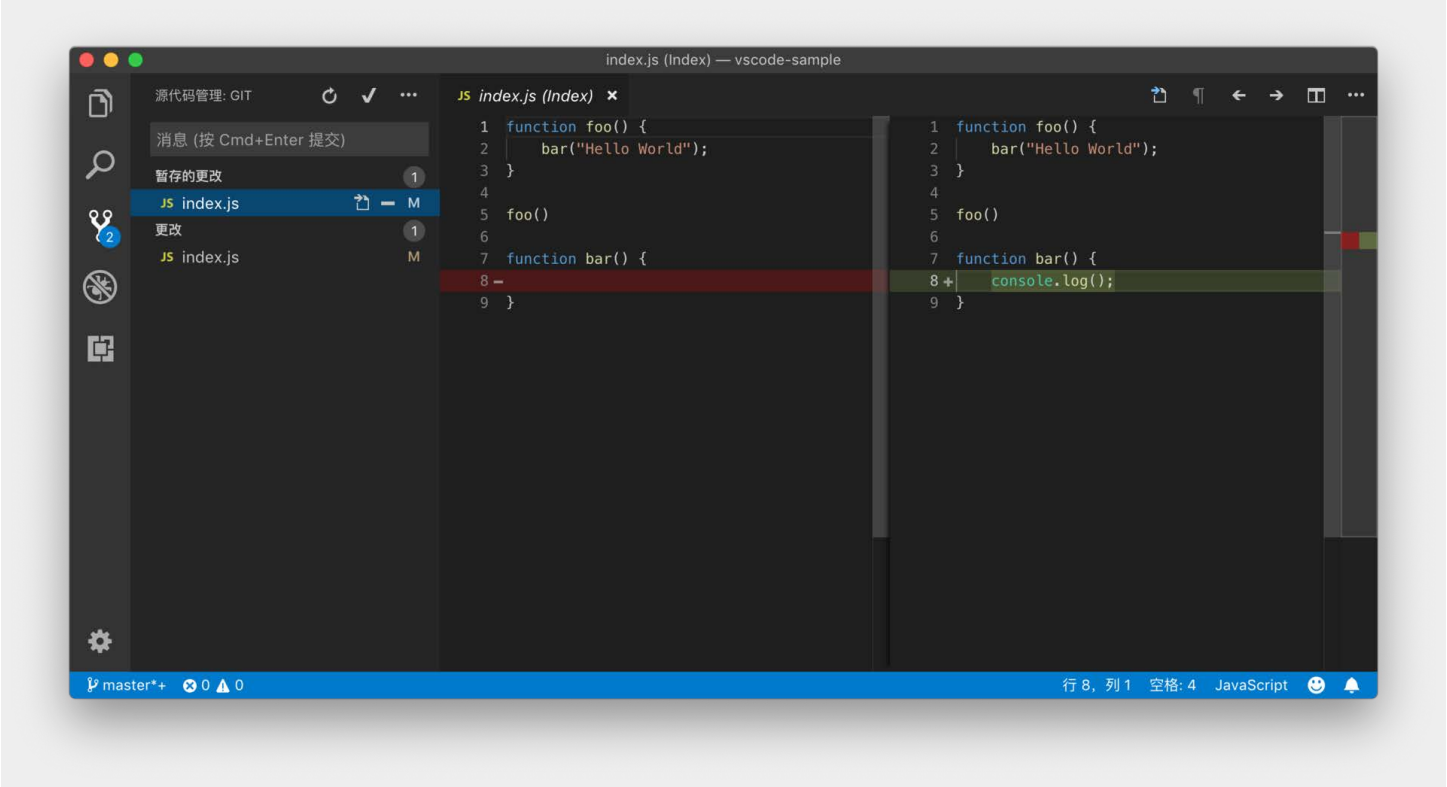
SCM 状态

在 SVN 中，则有一个变更列表（changelist 的概念），即可以使用变更列表来将不同任务里的代码变更统一到一起。在下图里，我们能够看到一个“两个分组”，一个是尚未被添加到变更列表里的代码变动，另一个是名叫 TEST 的变更列表。



SVN 变更列表

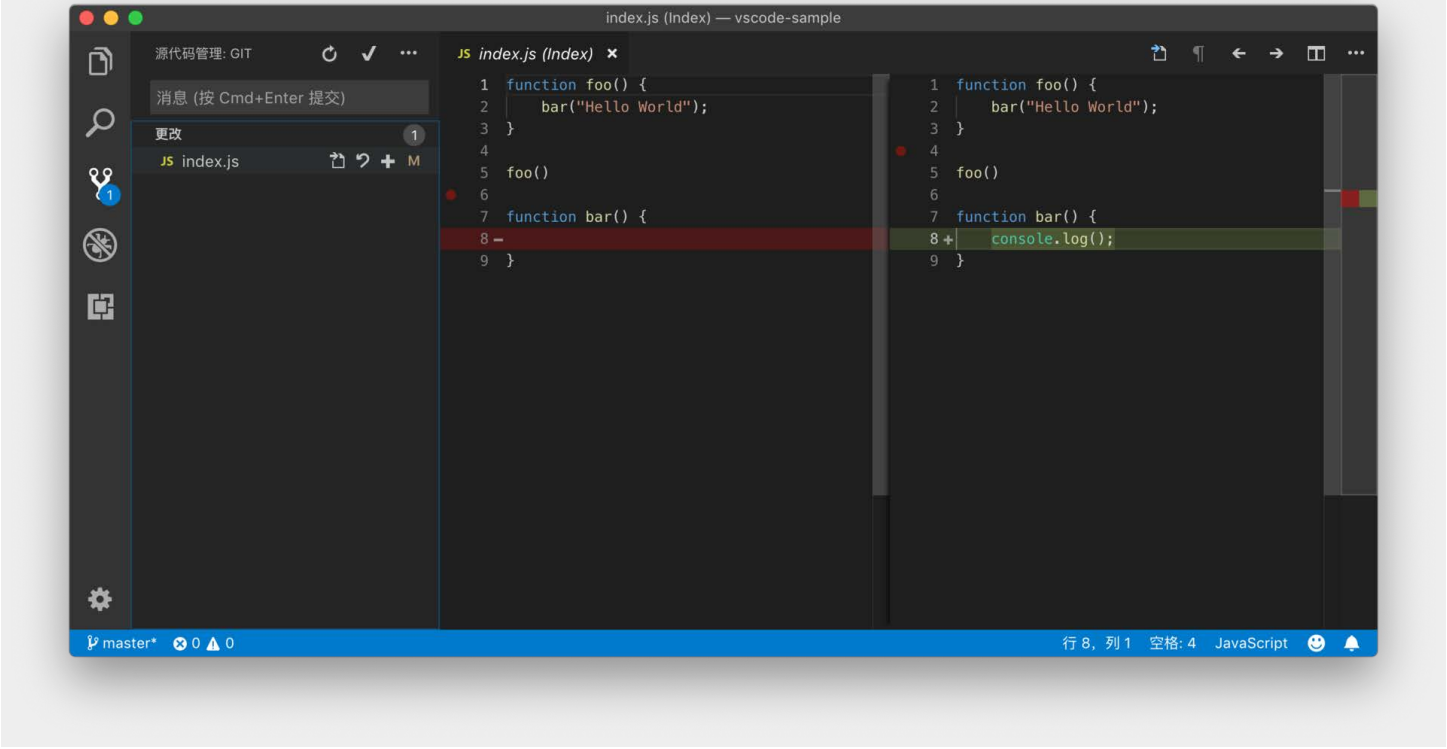
当我们点击某个代码改动项时，VS Code 则会打开一个差异编辑器（diff editor ）。这个编辑器显示了这次代码改动之前和之后对应的文件内容。这样我们就能够很清楚地分辨出对哪些代码做了改动。



差异编辑器

通常我在提交代码之前，都会依次打开这次改动的所有文件，并在差异编辑器中重新审查一下每个改动，进行最后一次的查漏补缺。

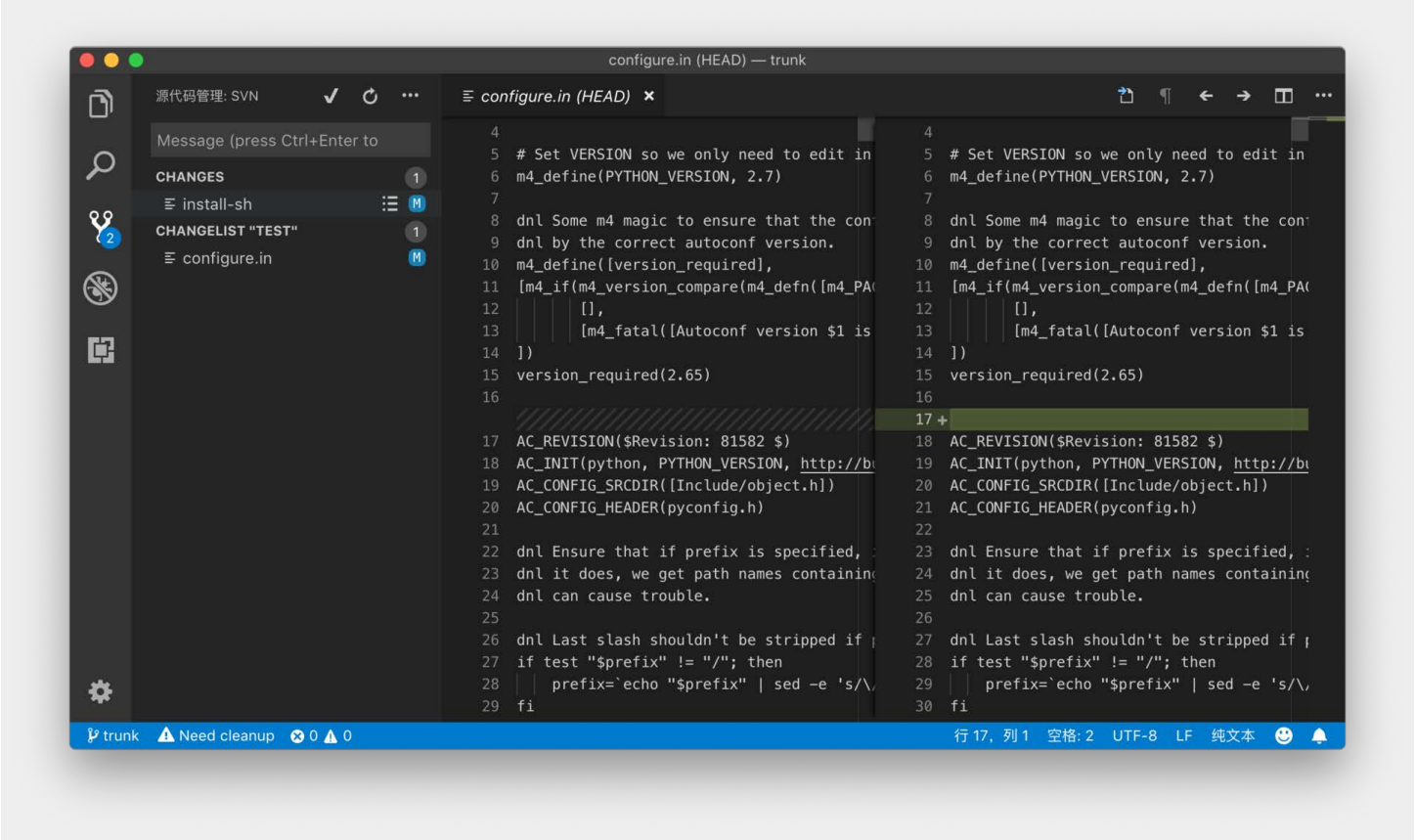
上面我们提到，在 Git 中提交代码前，我们需要先将代码改动变成暂存状态。这个操作的对应的命令行是 `git add ${fileName}`。我们同样也可以通过版本管理的视图来完成这样的操作：只需将鼠标移动到我们想要提交的文件上，就能够看到三个图标。



Git SCM 中文件项上的按钮

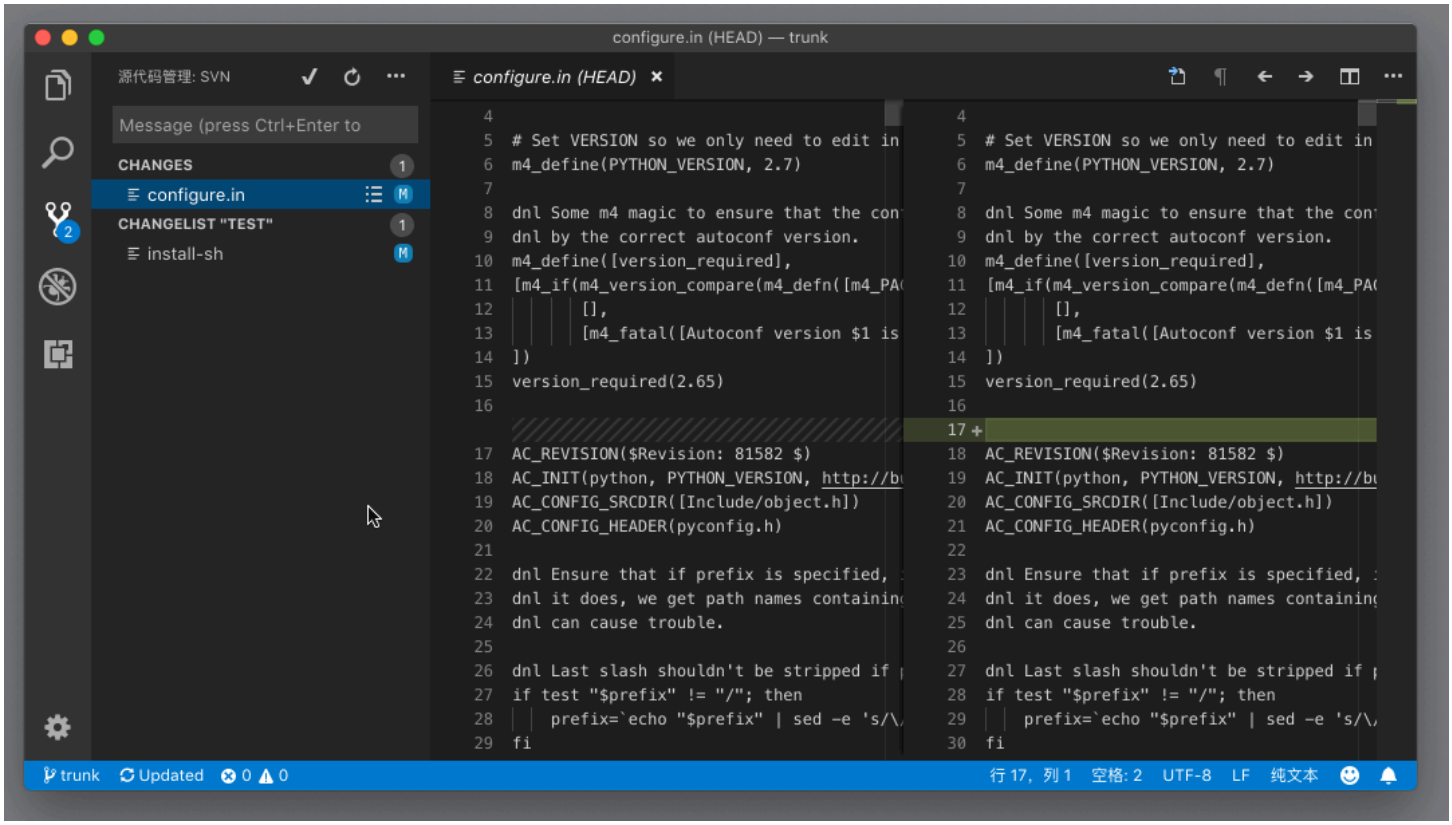
它们分别是：**打开文件**、**放弃更改**和**暂存更改**。我们只需点击 “暂存更改” 这个按钮，就能完成跟上面`git add` 一样的操作。

而如果我们使用的是 SVN，鼠标移动到文件改动上时，只能看到一个按钮 “Set Changelist”。



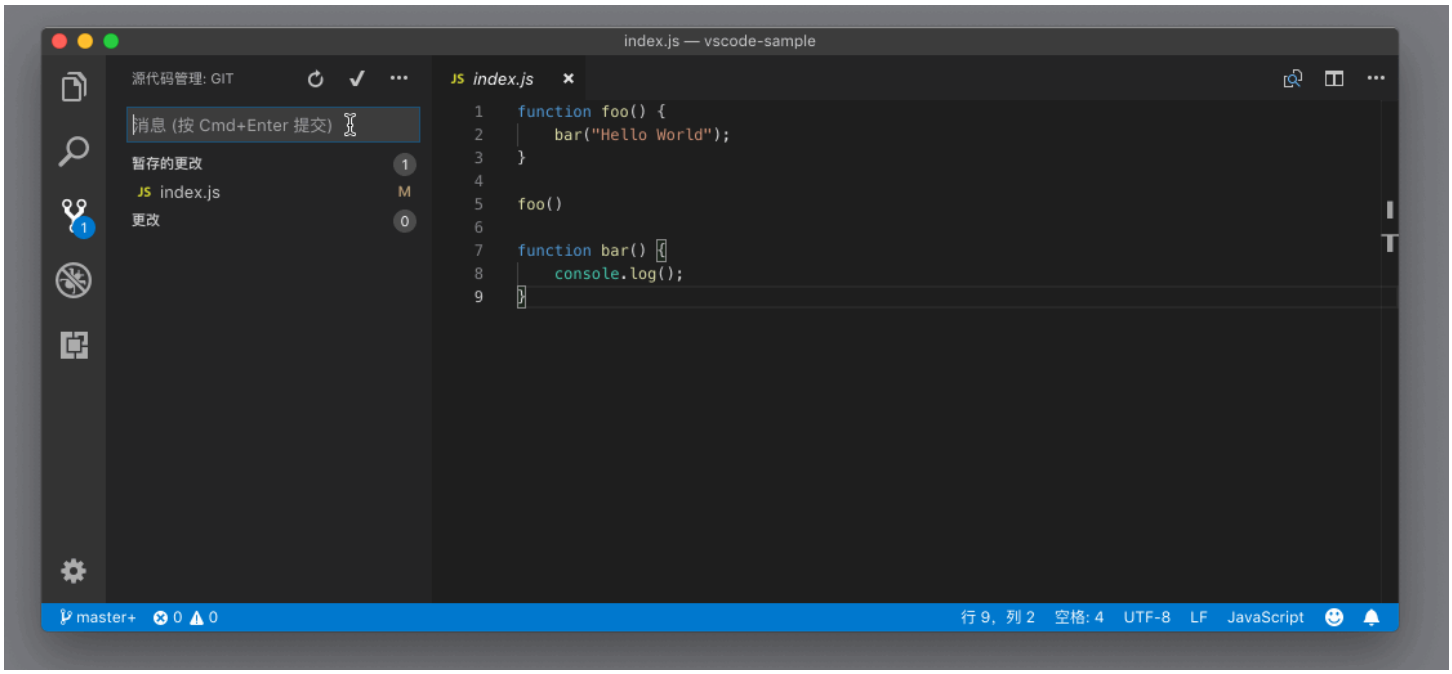
SVN SCM 中文件项上的按钮

按下这个按钮后，我们就能看到一个列表，通过这个列表，可以选择创建新的变更列表（changelist），或者选择将这个改动增加到现有的某个变更列表中。



将代码改动添加到变更列表中

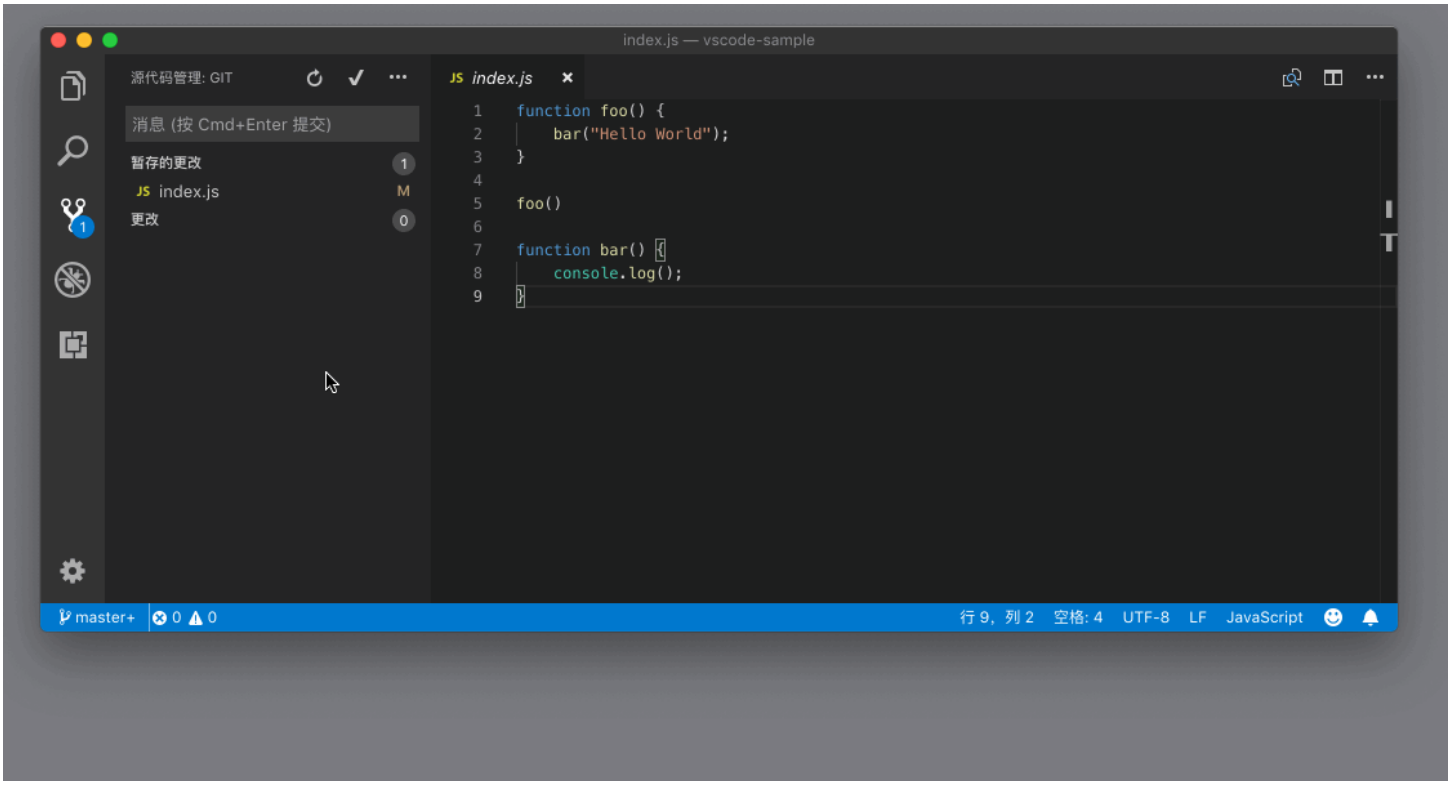
在选择完哪些代码变更要提交之后，下一步就是填写描述信息并提交了。在版本管理视图的上方，有一个输入框，我们把描述信息填入到这个输入框后，按下 Cmd + Enter 就能提交了（Subversion 是 Ctrl + Enter），也可以通过输入框上方的对号按钮来提交。



提交变更

查看代码变化、改变文件状态和代码提交，这就是我们日常最常使用的几个操作了。而其他的操作，比如更新、回退、发布、储藏代码等等，我们可以通过点击输入框上方最右侧的三个点的图标，打开一个

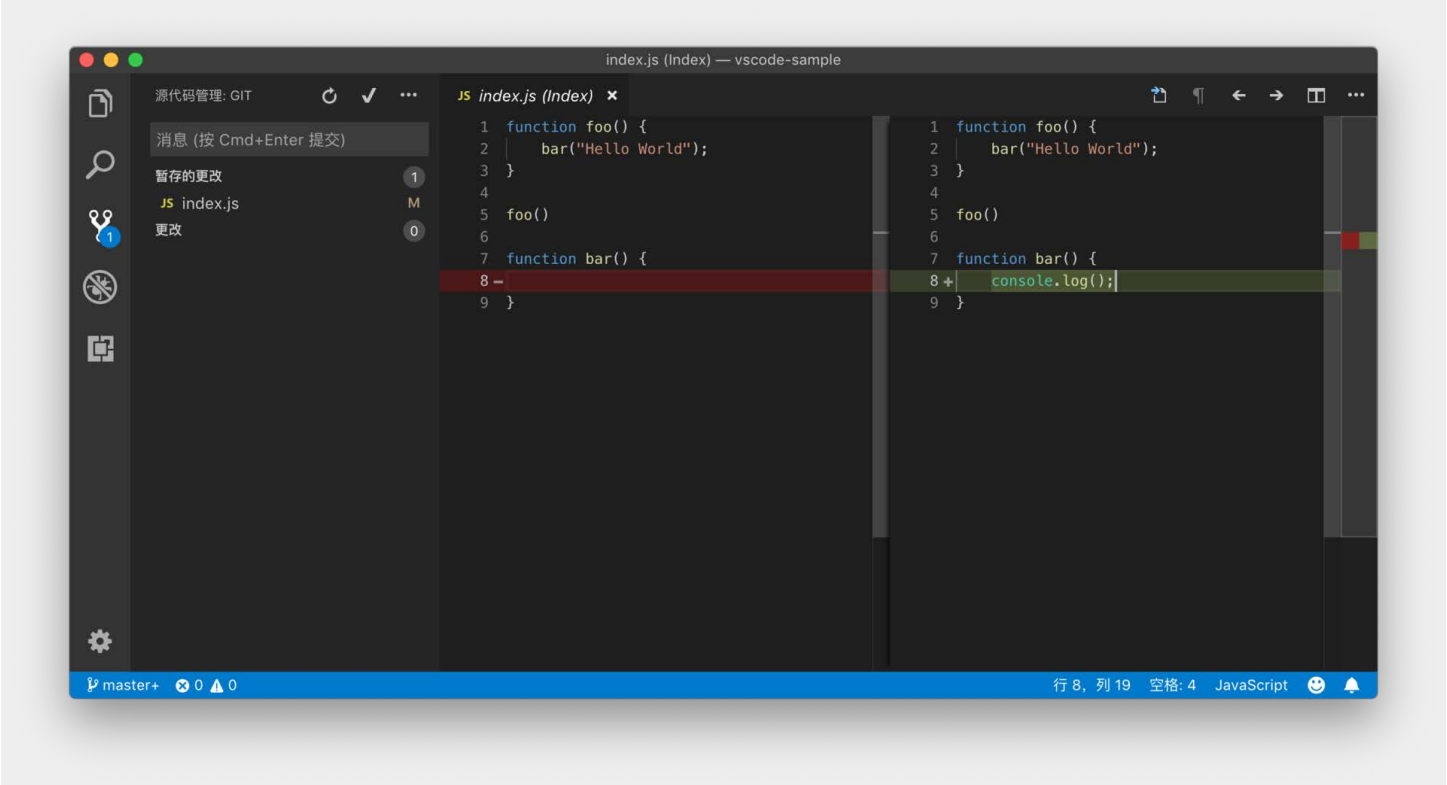
下拉菜单，选择我们想要的功能并执行。



SCM 的其他操作

差异编辑器

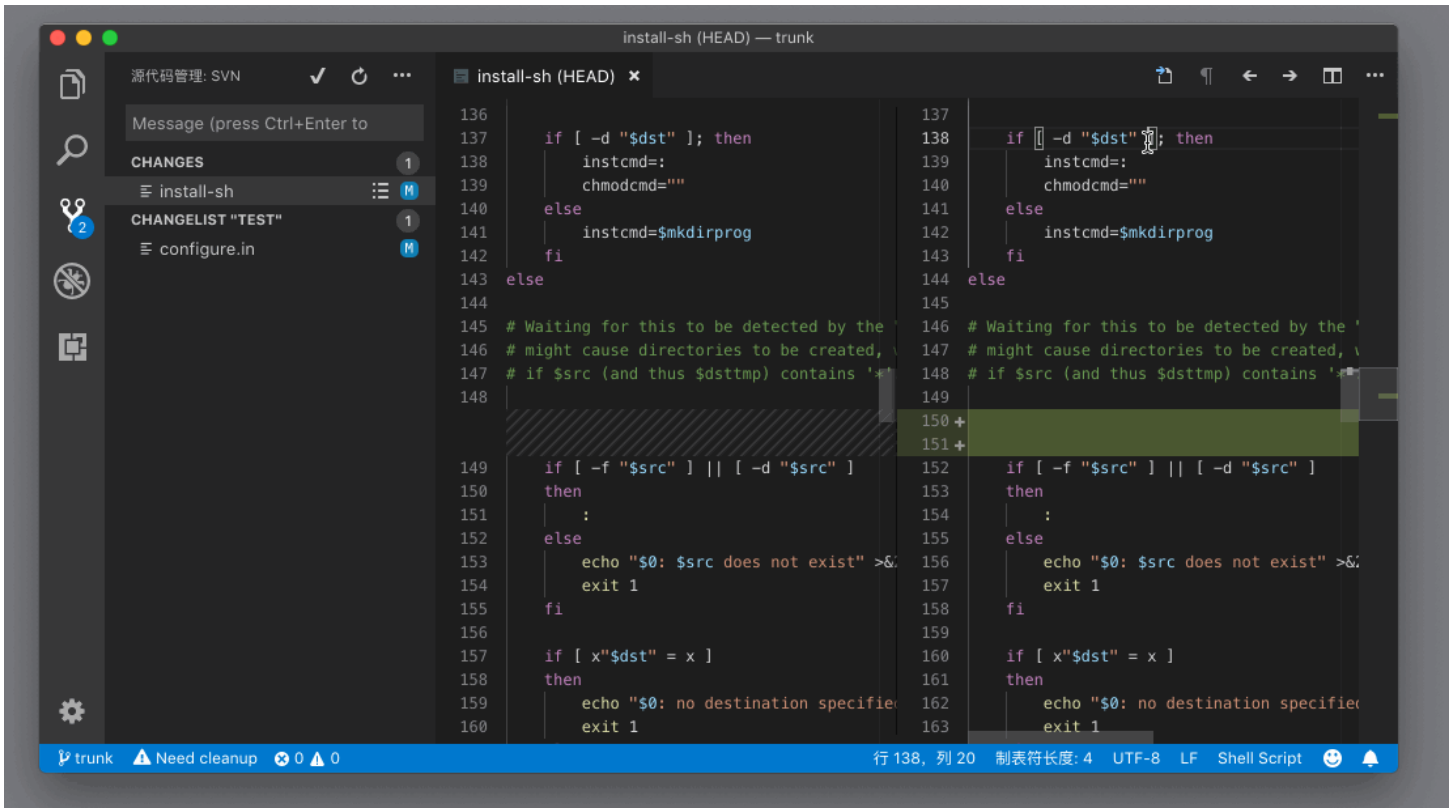
下面我们来说一下差异编辑器。在差异编辑器的右上角，我们能够看到一排按钮。



差异编辑器的功能按钮

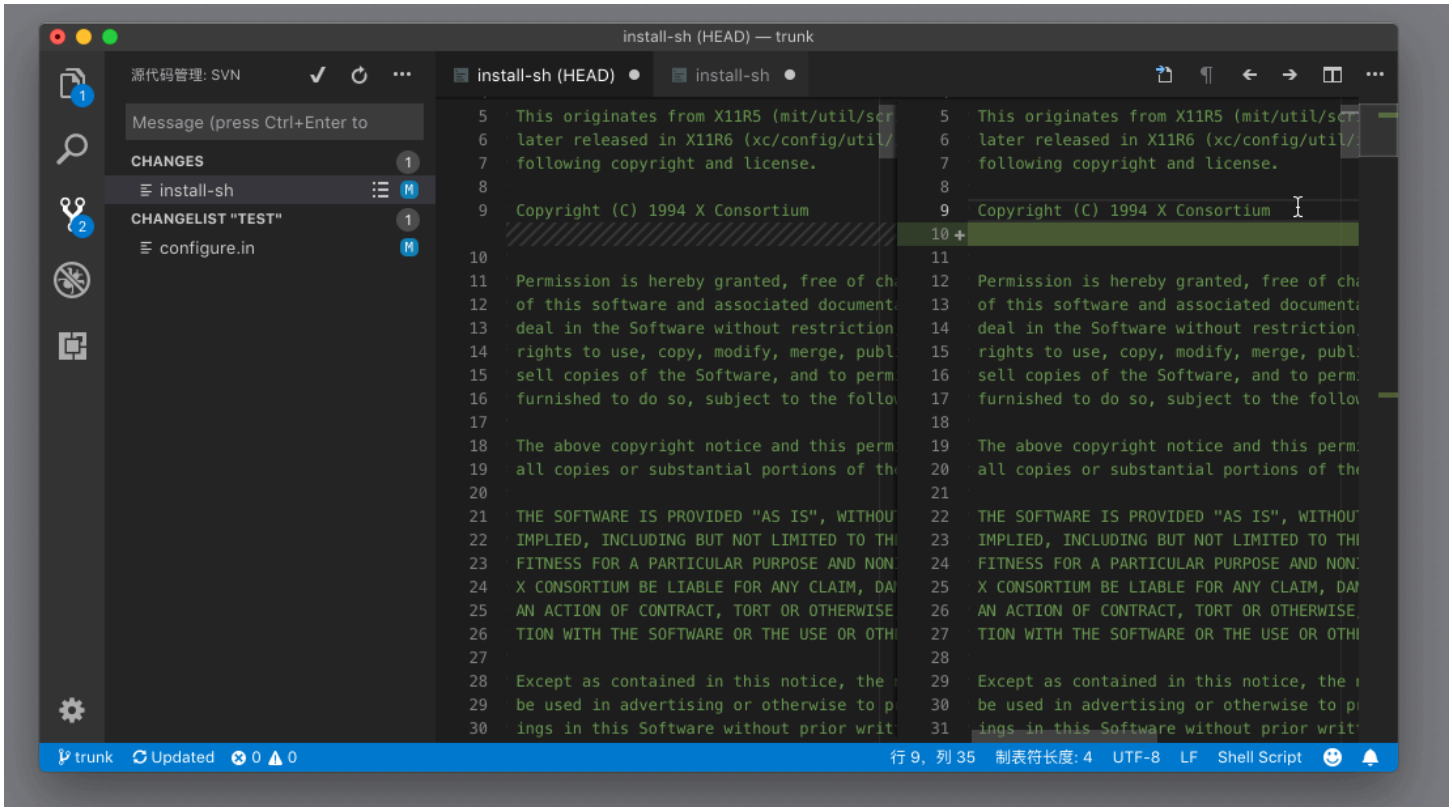
前四个按钮

第一个按钮的功能是从差异编辑器跳转到一个普通的编辑器，并且打开这个文件。这样我们能够无干扰地进行编辑操作了。



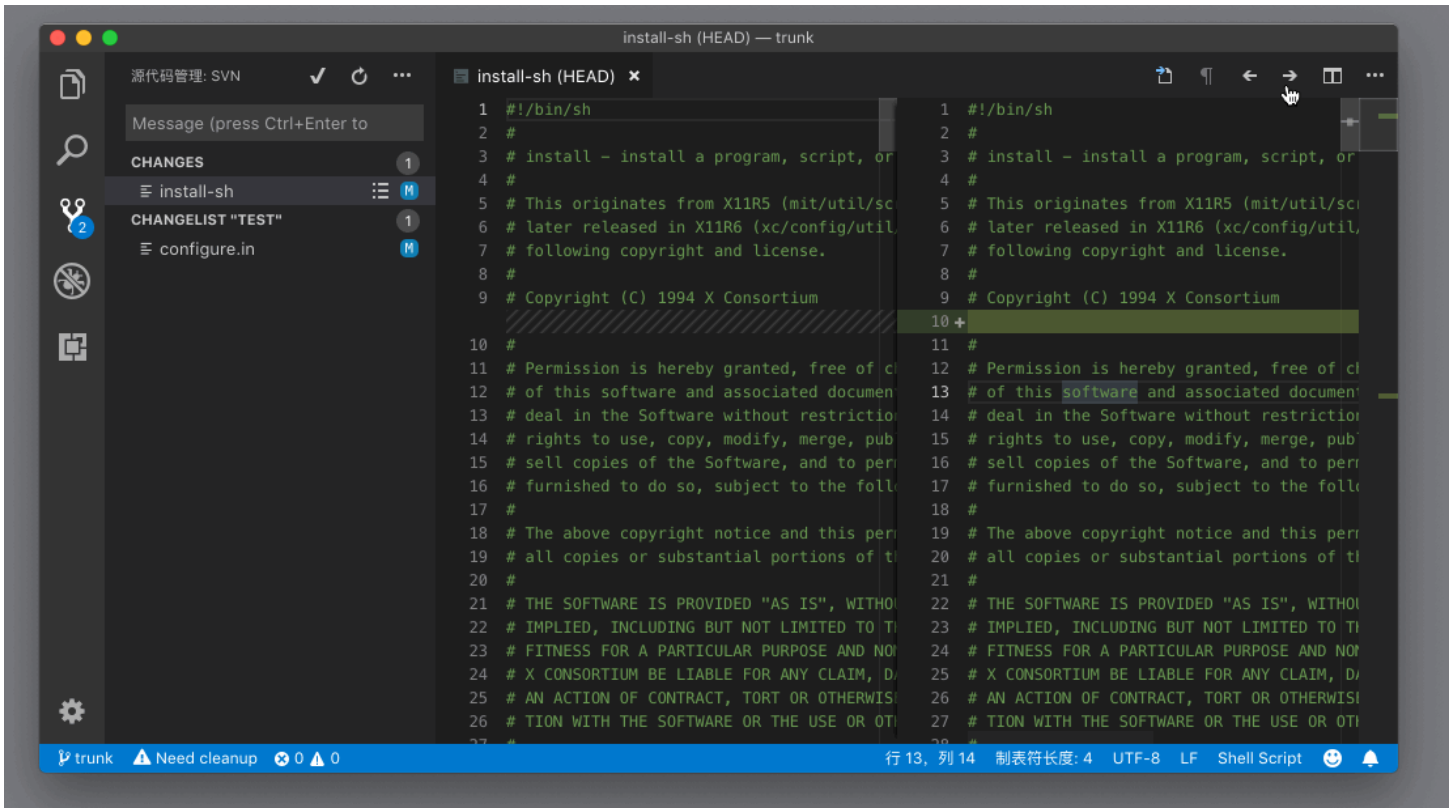
差异编辑器跳转到普通编辑器

第二个按钮控制的是：是否要在差异编辑器里显示代码里行末的空格符的变化。比如说你不小心在行末添加了几个空格，默认情况下，VS Code 觉得这几个空格不影响代码，就不会在差异编辑器里显示。但我建议把它打开，这样你就可以确保能够看到所有的代码改动。



差异编辑器是否检测行末空格的变化

接下来的两个箭头按钮，就是用于在当前文件里的多个变动之间进行跳转了。



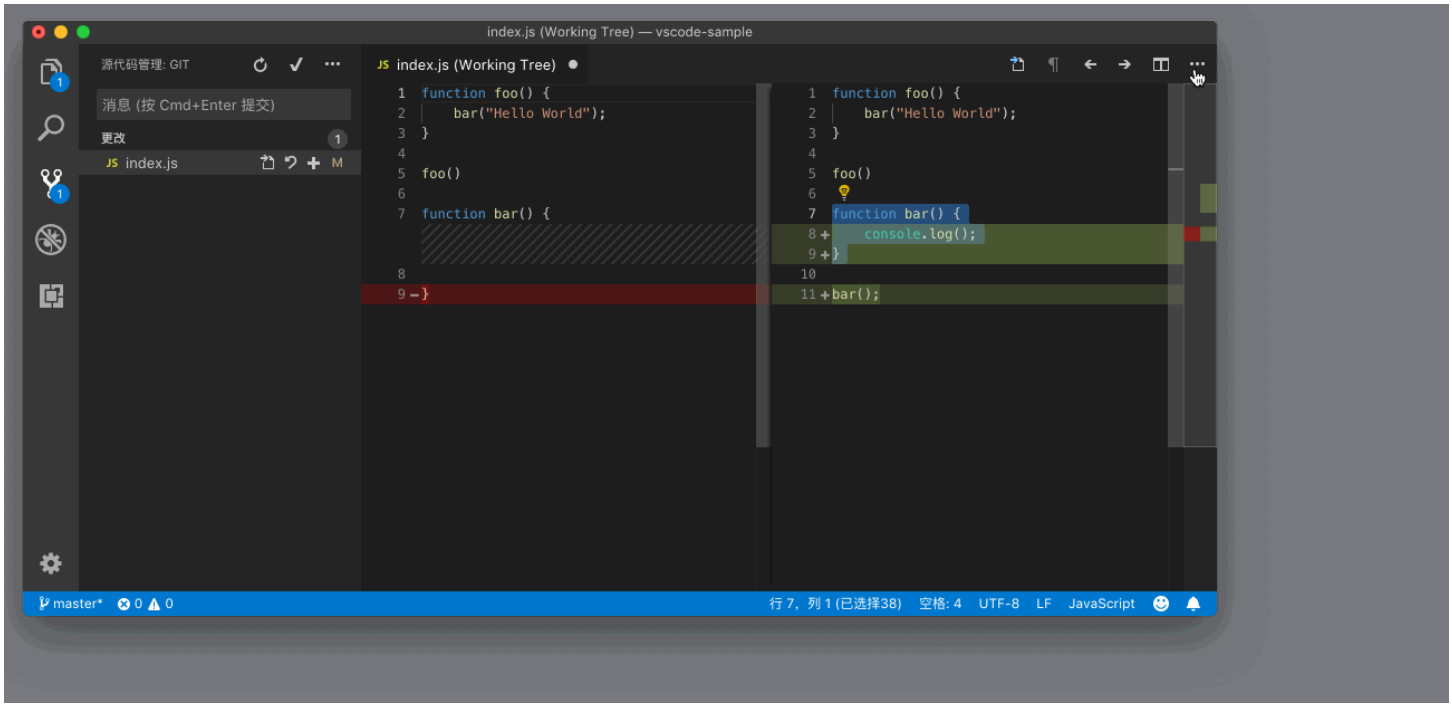
差异编辑器，在改动之间跳转

下拉菜单

最后是一个三个点的图标，想必不用点击开你也知道，它意味着里面有更多的功能，点击后即可看到一

个下拉菜单。

这个下拉框里的命令，前三个是和 Git 相关的。在版本管理视图里我们只能够对文件的状态进行操作，但是 Git 同样允许我们修改文件中某一段代码变动的状态，就是通过这三个命令实现的。



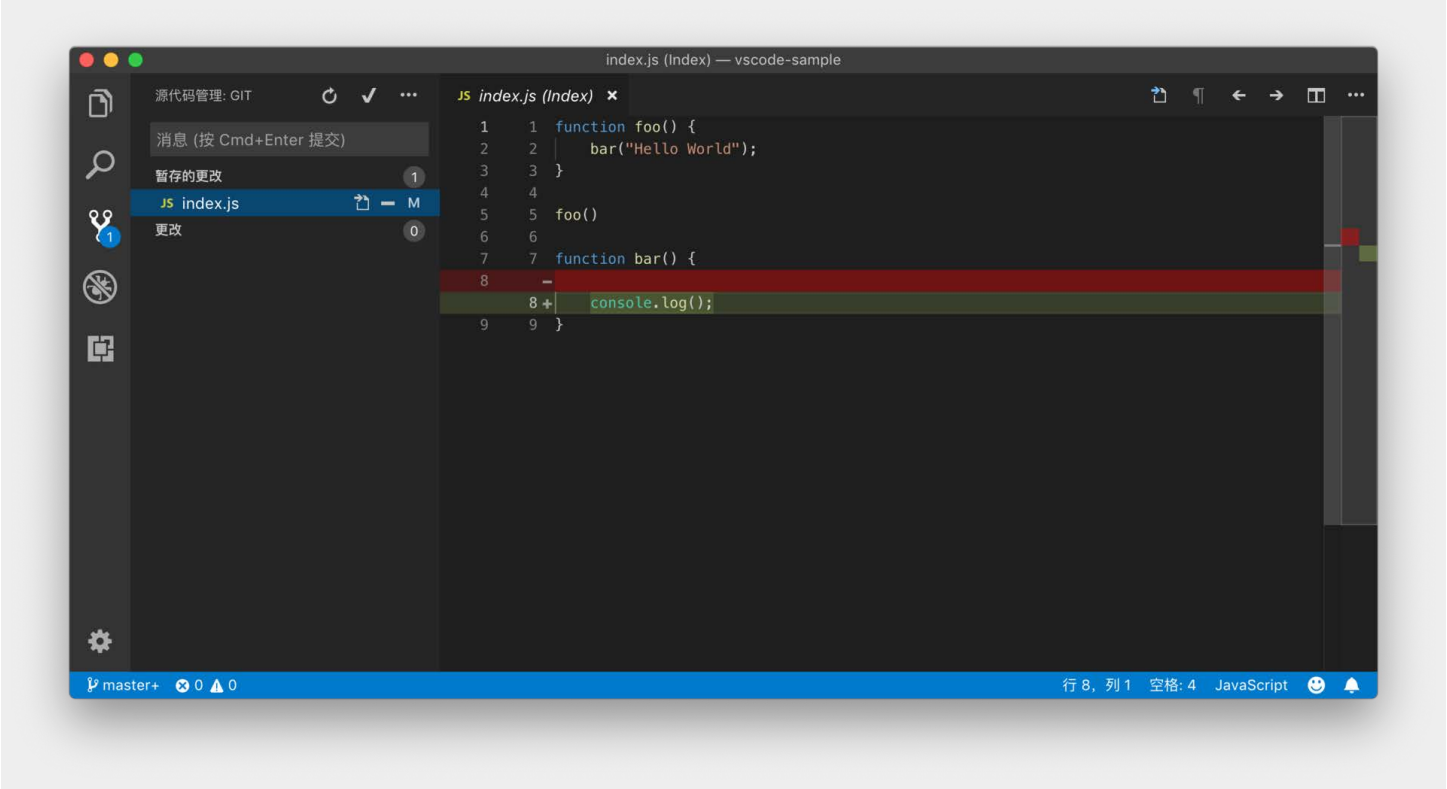
Git 差异编辑器，暂存某段代码变动

比如在上面的动图中，index.js 文件里有两段不同的改动，现在我只想把第一段代码改动变为暂存状态。那接下来我要做的就是选中这段代码，从下拉菜单里选择 “暂存所选范围”运行。这之后，我们在版本管理视图里就能够看到两组代码变动，“暂存的更改”这个组里是我刚才选中的那段代码变动，而“更改”组里则是第二个代码变动。

相信你已经猜想到了这个功能的用途，有的时候我们在修一个 bug 或者完成一个功能时，对代码做了一些临时的变动，但我们既不想提交这个变动，也不想把它清除掉，那通过这几个命令我们就能够真正按需提交代码变动了。

不过遗憾的是，SVN 这个插件并没有支持这个功能。

下拉菜单里的第四个则是和差异编辑器呈现方式相关的。默认情况下，差异编辑器采取并排的方式显示两个编辑器，**左边的编辑器显示改动前的文件内容，右侧则是改动后的文件内容**。其实我们也可以使用“切换内联视图”，将代码改动显示在同一个编辑器里。



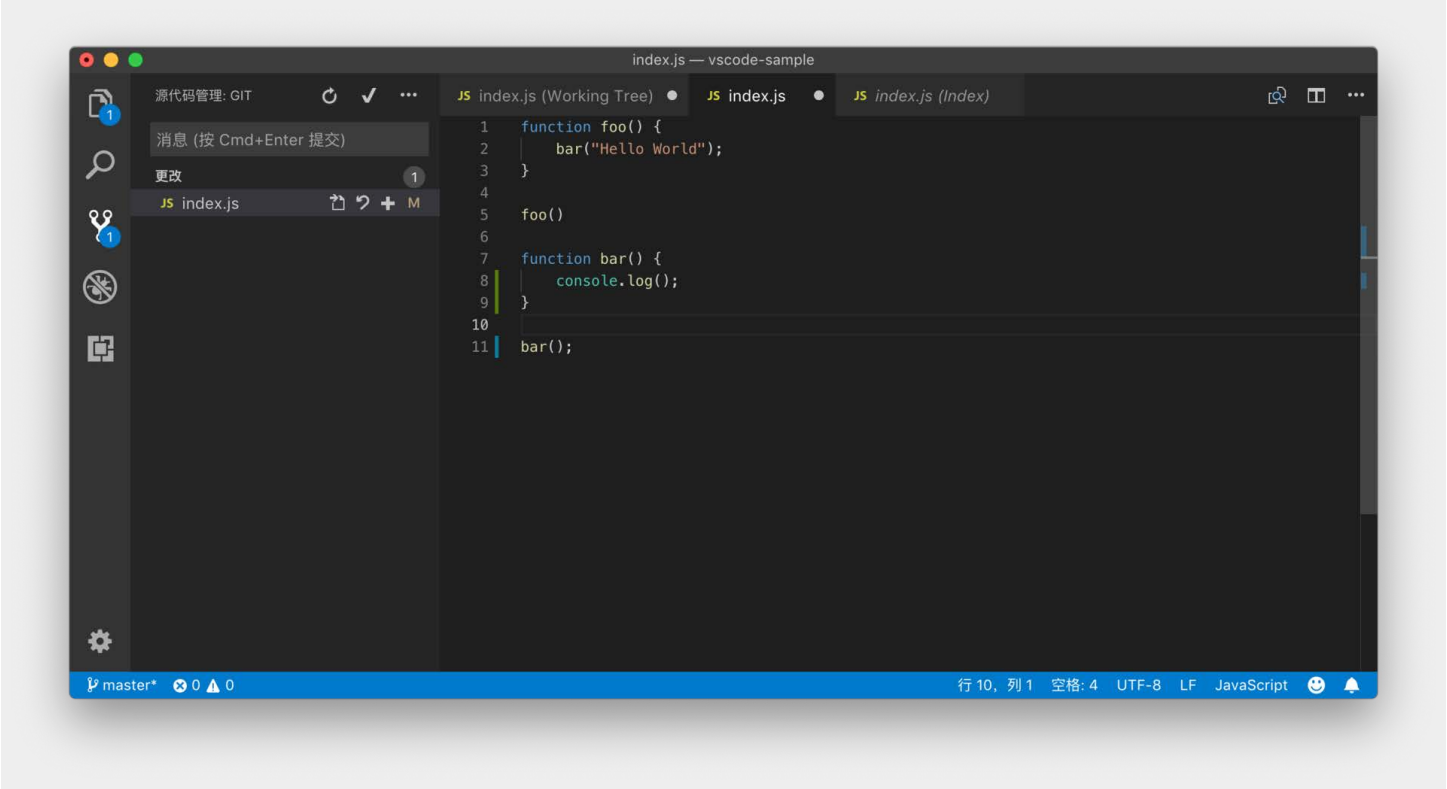
差异编辑器的内联功能

在这个内联的差异编辑器里，我们依然可以看出代码变动的情况。

至于最后的三个则是通用的编辑器命令，打开任意编辑器时，我们都能够使用它们。这里就不多赘述。

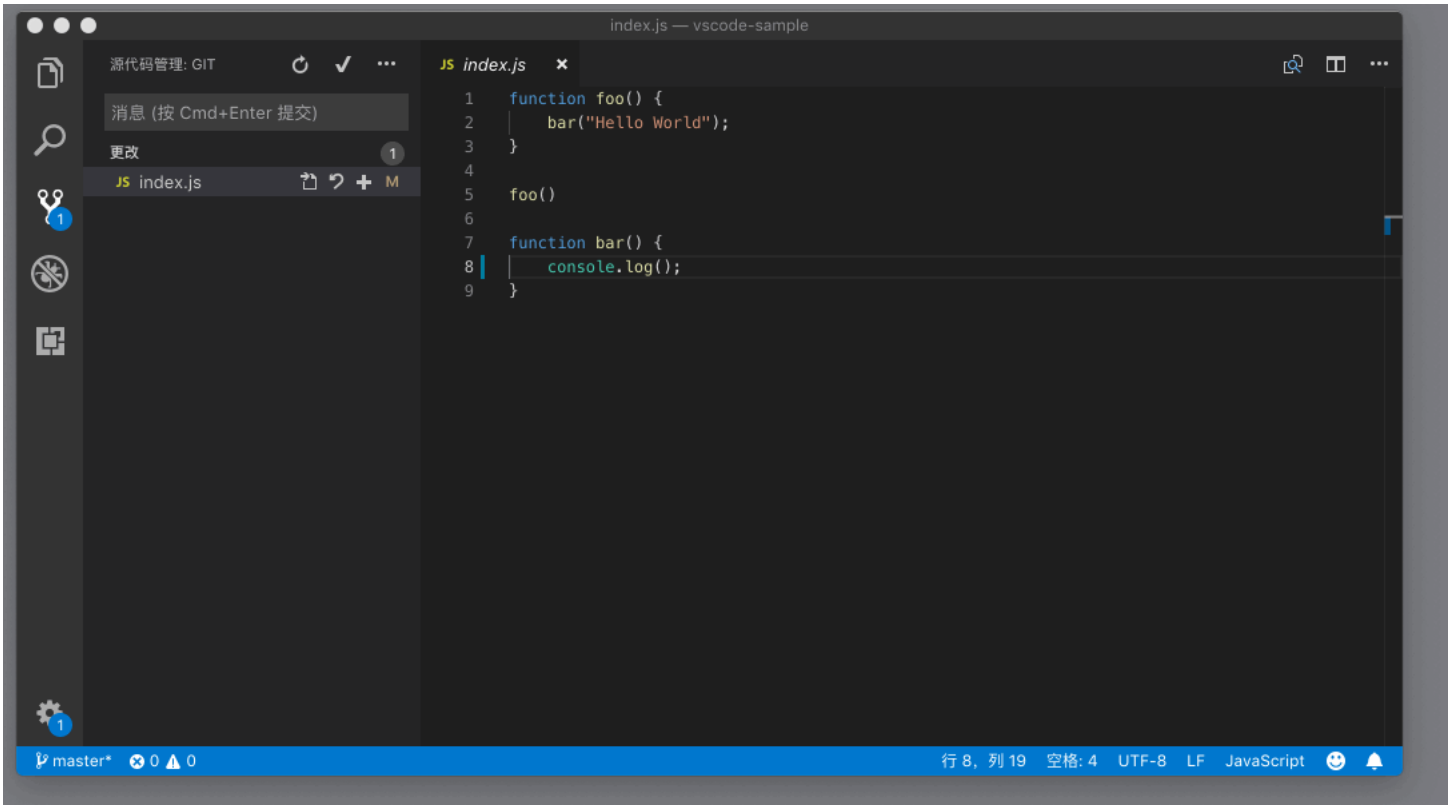
编辑器内置版本管理操作

上面我们一起学习了如何使用版本管理视图和差异编辑器对代码变动进行管理，其实我们同样也能够在普通的编辑器里完成这样的操作。当我们对一个文件作了修改后，我们能够在普通编辑器里，行号的右侧看到不同颜色的竖线。它们代表着不同状态的代码变动，比如代码增加、修改和删除。



编辑器里的 SCM 状态装饰

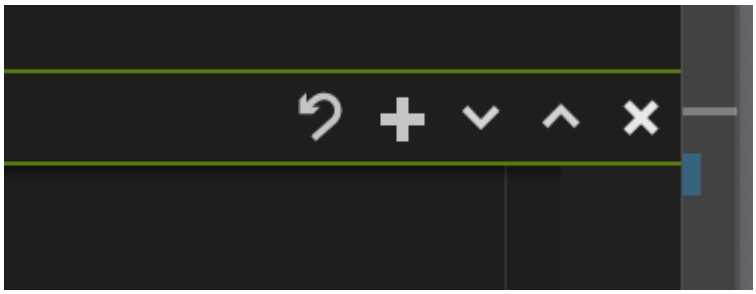
当我们点击这个竖线时，我们就能在当前编辑器里，立刻看到一个内置的差异编辑器。这样我们就不用每次都打开版本管理视图里去查看代码变动了。



在普通编辑器里打开差异编辑器

在这个内置的差异编辑器的内侧，我们能够看到一些按钮。它们的作用跟差异编辑器右上角的那些按钮

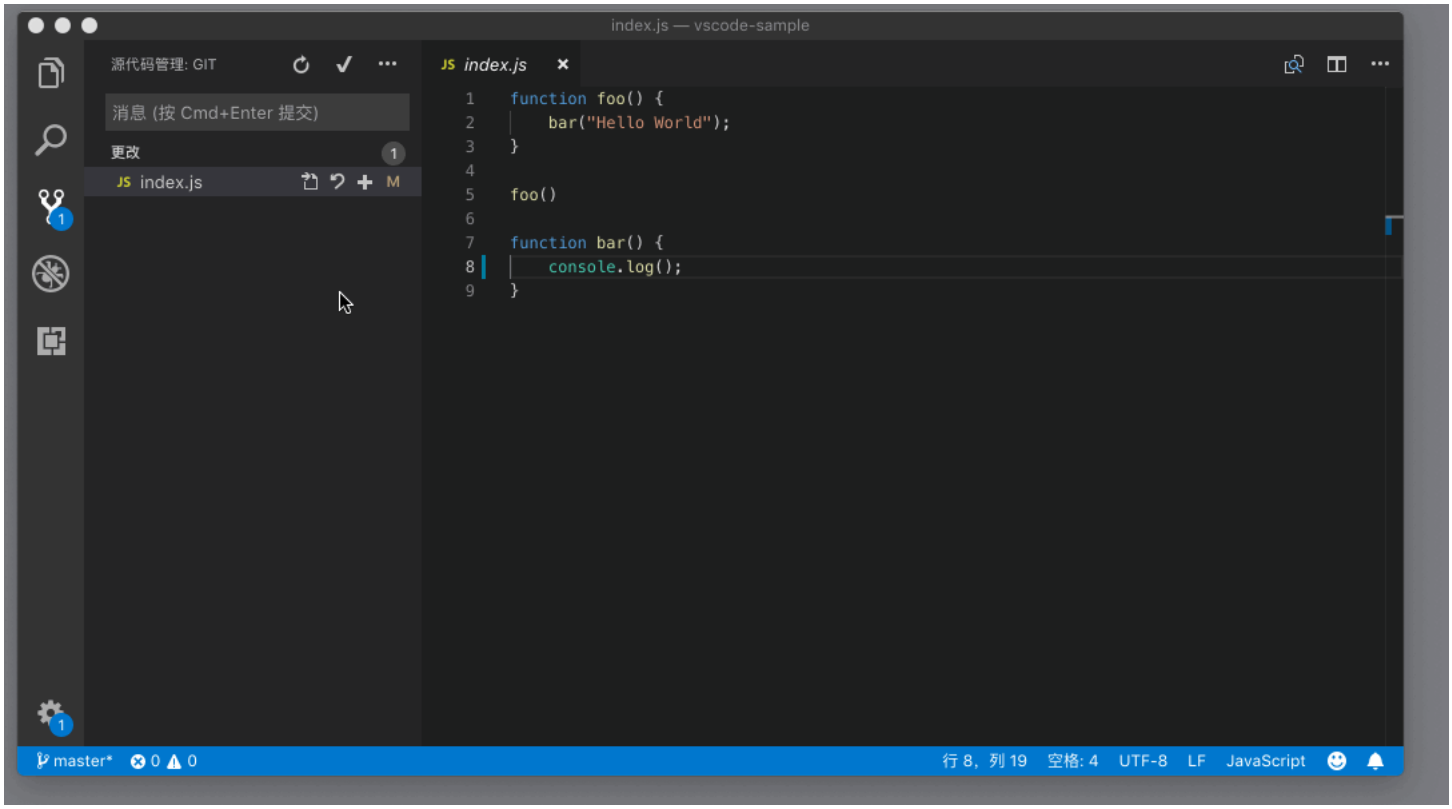
是一样的，你能够通过它们暂存、撤销代码改动，以及在代码改动之间跳转。当我们把鼠标移动上去时，还能够看到它们对应的快捷键。



内联的差异编辑器的功能按钮

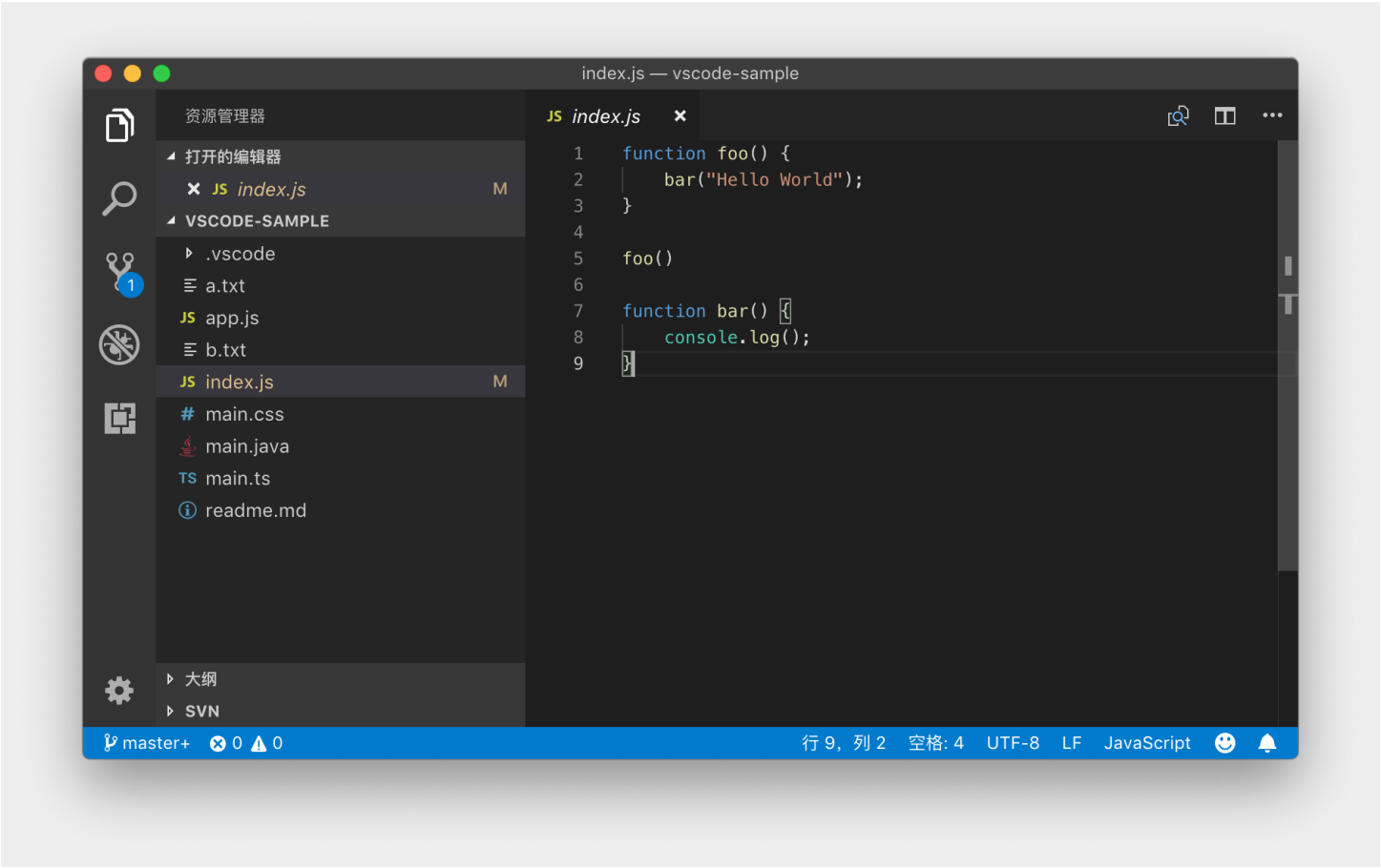
状态栏和资源管理器

除了版本管理视图以外，另外一个能够快速了解项目版本状态的就是状态栏了。无论是使用 Git，还是 SVN 来做版本管理的项目里，我们都能在状态栏的左侧看到当前代码属于哪个分支。不仅如此，我们还能通过点击它来进行分支的创建与切换。



从状态栏创建新分支

此外，当我们在专心于写代码的时候，大部分情况下我们打开的都是资源管理器。为了方便我们在这种情况下快速地了解哪些文件被修改了、有什么类型的改动，**VS Code 会把资源管理器中对应的文件项的颜色改变，同时在这个文件项的最后附上一个简单的字母，用于表明这个文件的状态**。比如说我们在下面的图中看到，index.js 这个文件被修改了，那么在资源管理器里，这个文件项变成了黄色的，同时最后还有一个字母 M，也就是修改 Modified 的首字母。

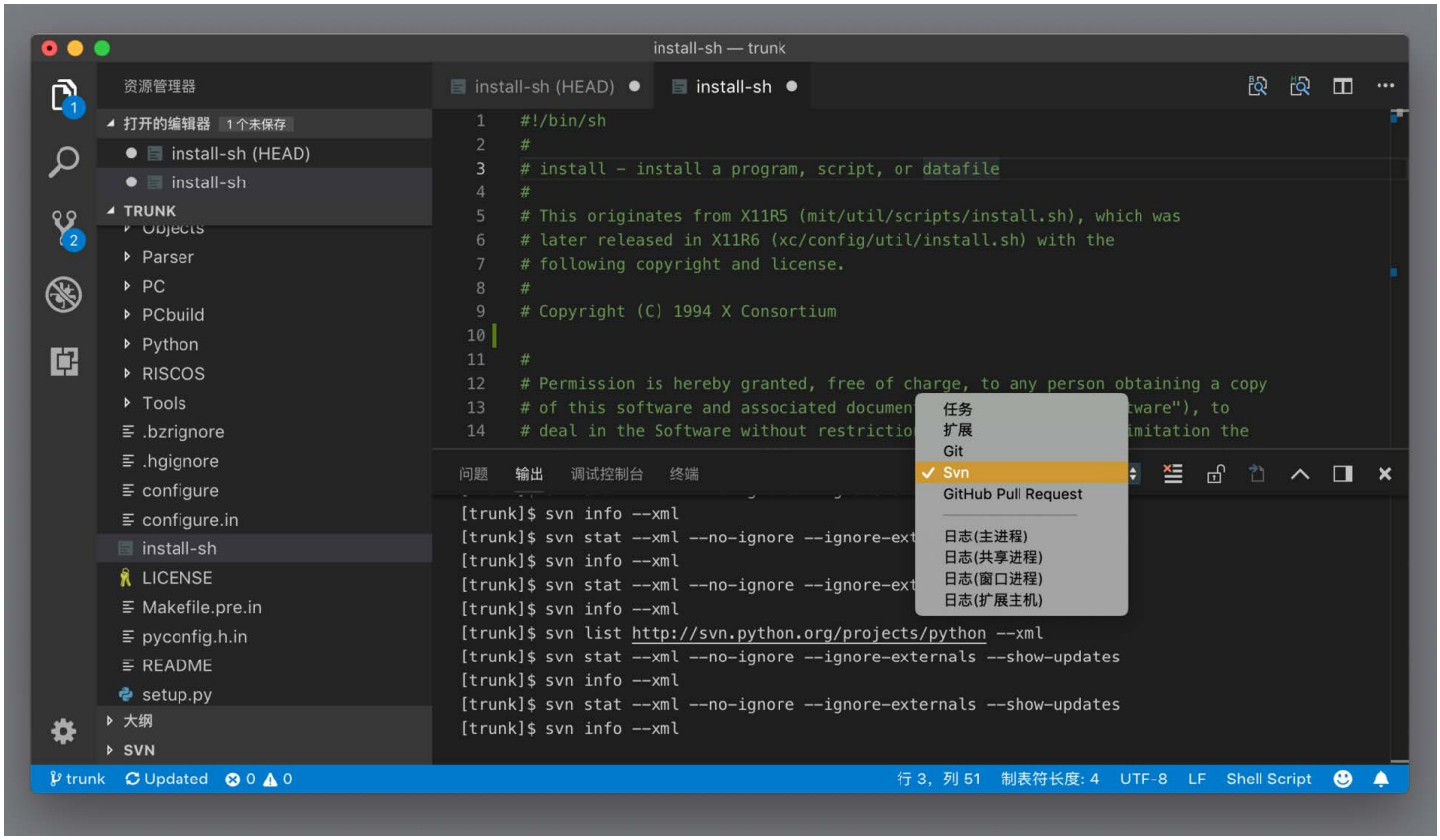


资源管理器上的文件状态装饰

小结

以上，就是我们今天关于 VS Code 内版本管理功能的知识了。

不知道你是否还记得，我们在专栏的最一开始聊 VS Code 的设计哲学的时候，我提到过 VS Code 并不想当一个黑盒，而是尽可能地开放给你。就好比说版本管理，Git 非常流行，但它的学习成本也很高，即便是一个熟练使用它的工程师，也偶尔会遇到奇怪的情况，更不要说刚刚学习它的新手了。为了保证你知道你按下某个按钮后，VS Code 最终转化成了哪个 Git 命令，或者你遇到意外情况的时候 Git 发生了什么，VS Code 会把所有这些信息全部输出到输出面板中。你可以打开输出面板，点击下拉框，然后选择你使用的版本管理工具的 log 进行查看。



输出面板里的 SCM 输出

此外，今天我们的文章，也没有系统介绍版本管理的命令操作，你可以打开命令面板，搜索 “git” 或者 “svn”，看看它们都支持哪些命令，相信你一定会有别样的收获。希望你在 VS Code 里使用版本管理的过程中，能够做到知其然、知其所以然，加油！

精选留言

一步

Vscode 能把查看提交历史树的功能内置进去，就完美了

2018-10-18 10:05

叶小轩

vscode有扩展，去扩展里面可以看到很容易使用的git扩展，可以可视化看分支，我觉得可以提一下。

2018-10-19 12:21

作者回复

你说的是 GitLens ， 后面我们会黑大家推荐的

2018-10-22 11:28

L

老师怎么消除在写js的时候的一些ts的警告啊。。。在写js的时候有时候会给出Declaration expected.这个警告 这个是ts的吧

2018-10-18 21:08

cylim

可以用gitlens插件。

2018-10-18 16:38

万事如意

可不可以再详细的介绍一下 svn 的操作流程，总感觉 vscode 的 git 操作要比 svn 的好很多！

2018-10-18 15:28

悬炫

还是sourceTree 用起来更加直观，并且也更加强大

2018-10-18 12:50

felix

如果能像visual studio那样好用就好了，最不可缺的就是tree view，不然不直观！

2018-10-18 09:11

你看起来很好吃

虽然vscode提供了很方便的可视化版本管理界面，我还是喜欢用命令的方式做这些

2018-10-18 07:37