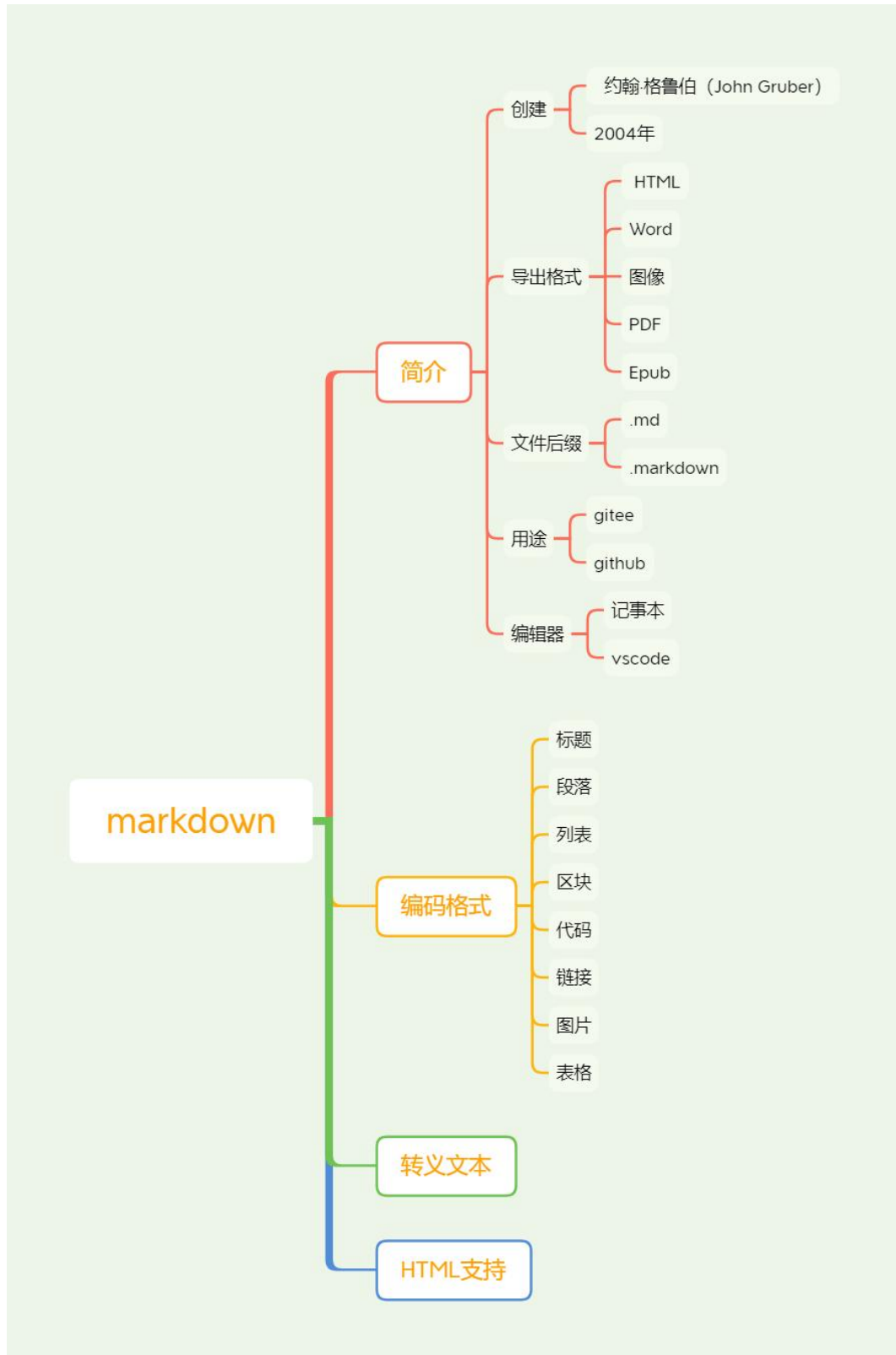


# Markdown



Markdown 是一种轻量级标记语言，它允许人们使用易读易写的纯文本格式编写文档。

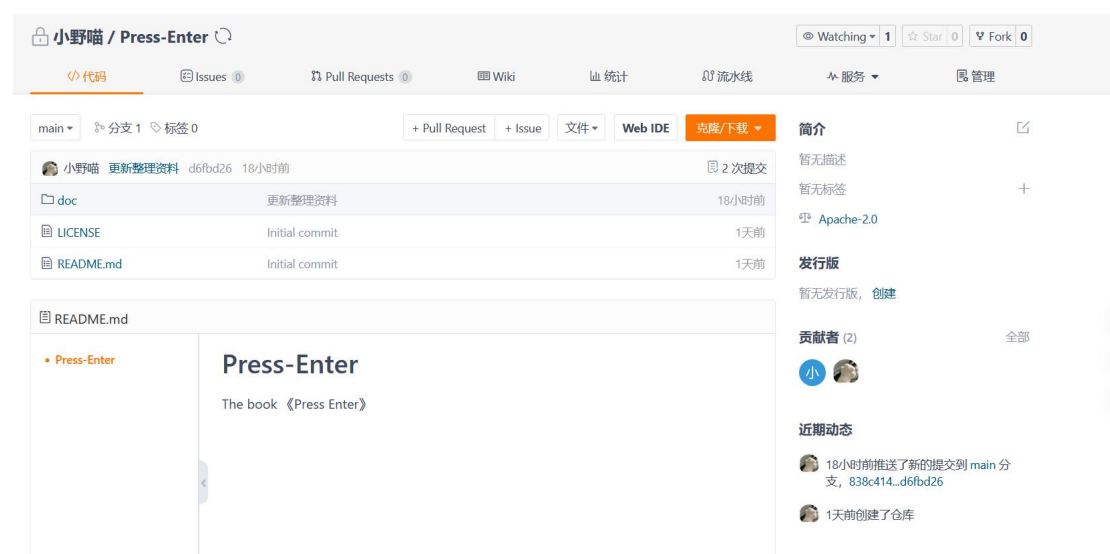
Markdown 语言在 2004 年由约翰·格鲁伯（英语：John Gruber）创建。

Markdown 编写的文档可以导出 HTML、Word、图像、PDF、Epub 等多种格式的文档。

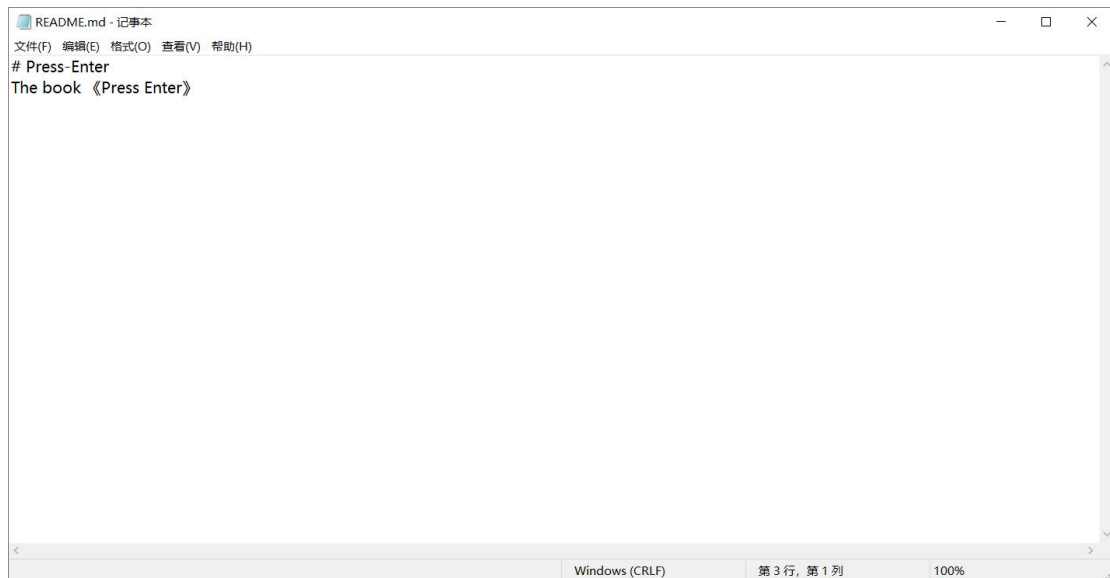
Markdown 编写的文档后缀为 .md, .markdown。

Markdown 的编写工具可以是任何文本编辑器，包括记事本和 vscode。

作为一名合格的程序员，我们总可以在国内最大的代码储存分享网站 Gitee 和世界最大的代码分享网站 github 中看到 markdown 的身影，在每个仓库的根目录下又一个叫做 README.md 结尾的文件，这是我们仓库的文档文件，包含了我们对仓库的介绍和对代码的使用方法和说明等。可以说，README.md 文件是我们代码仓库不可缺少的一部分。我们可以看到这个文件的文件名以.md 结尾，使用我们判断这就是一个 markdown 文件。



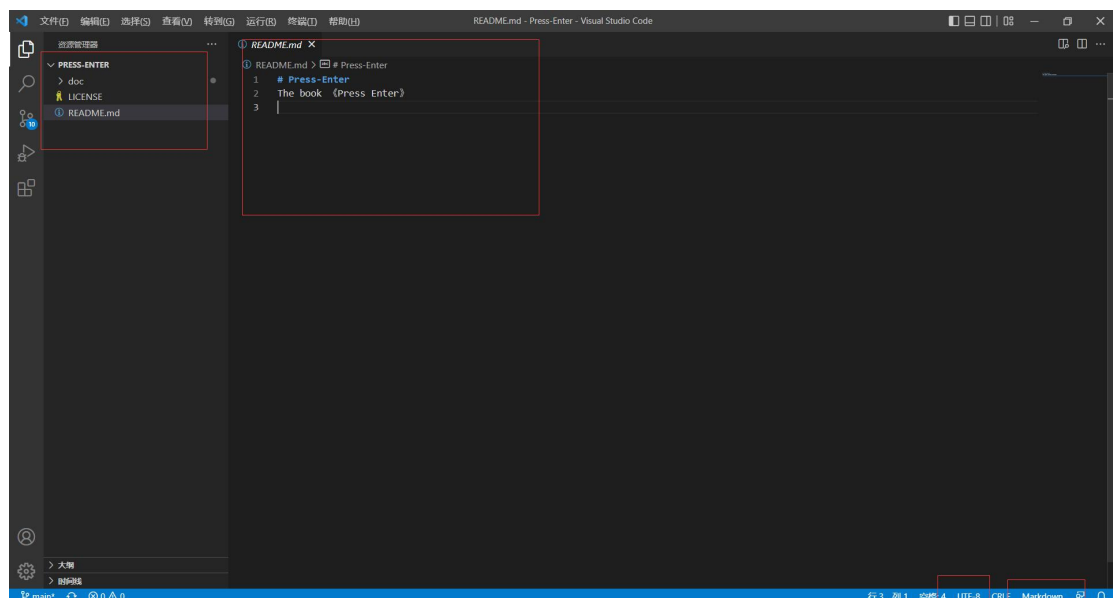
在 Gitee 上面，我们看到的 markdown 文件都是以可视化网页的形式展示的，其实，gitee 上面的 markdown 可以可视化展示出形式不一的文本，是因为 gitee 对我们的 markdown 文件进行解析然后渲染的网页页面。我们可以用记事本打开 README.md 文件来查看该文件的源码。



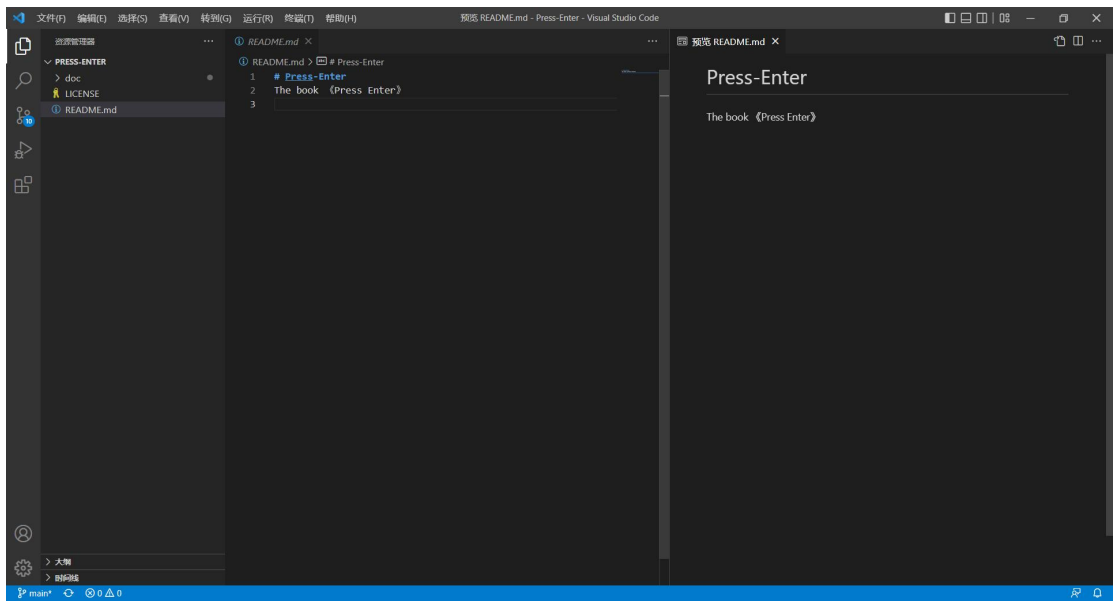
我们现在明白了我们看到的形形色色的文本是 Gitee 解析的结果，既然可以解析 markdown 文件并且渲染到页面，那么我们可以思考 markdown 肯定有一种标准的编码格式，不然 Gitee 也解析不出来，下面我们将详细介绍 markdown 的语法格式。

在下面的演示中，我们将使用 **vscode** 这个轻量级扩展性强的编辑器来操作。在 markdown 编码过程中，我们不希望我们编写的 markdown 文件的内容必须要上传到 Gitee 上面解析并且渲染我们才能看到效果，所以在前辈的不断改进下，我们现在的大部分编辑器已经实现了“所见即所得”的编辑模式，非常方便。在 **vscode** 中也同样支持“所见即所得”！

我们在 **vscode** 中打开文件，可以看到整个文件的布局中，右下角显示我们目前的编码语言为 markdown 语言，编码格式为 UTF-8。在左边是我们的目录的层次结构展示，中间为我们可以编码的部分。



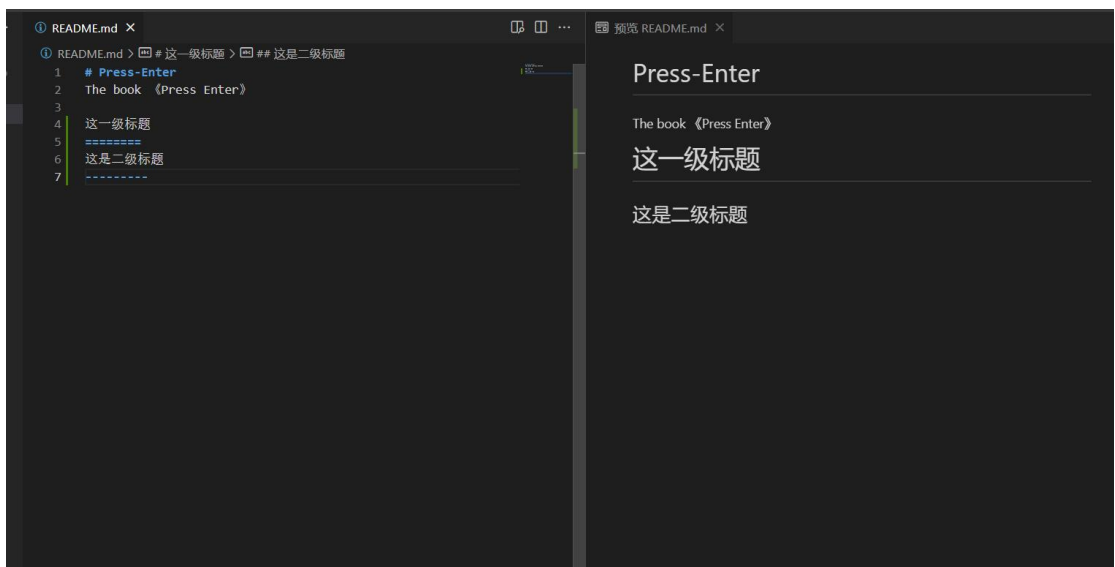
我们可以点击右上角的第一个按钮打开侧边栏预览，体验“所见即所得”的编辑快感。



下面我们演示 markdown 的基础语法。

首先是我们可以很清晰的看到的标题栏，在展示时也将展示 markdown 中最大的文本字体。

第一种展示标题的方法，在文本的下方使用 = 和 - 标记一级和二级标题。

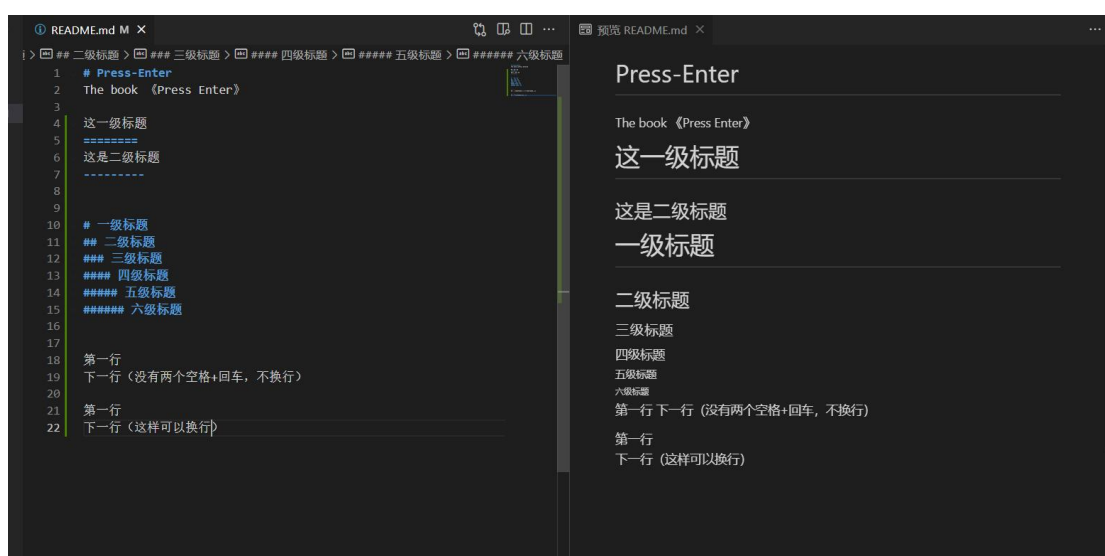


上面的方法展示的标题层次较为有限，我们可以换为使用‘#’的个数来标识标题的层次。

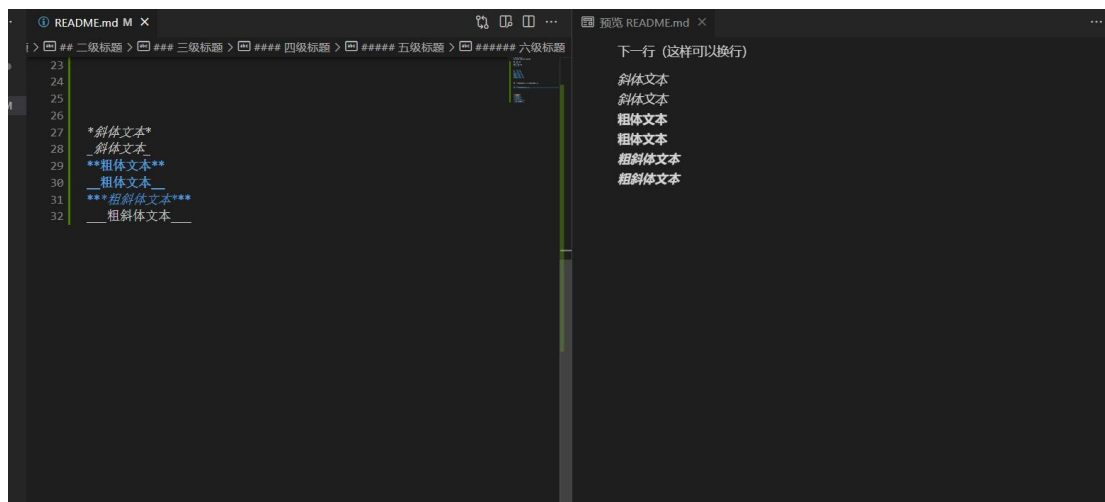


切记：‘#’ 越多，标题层次越小，只有六级标题

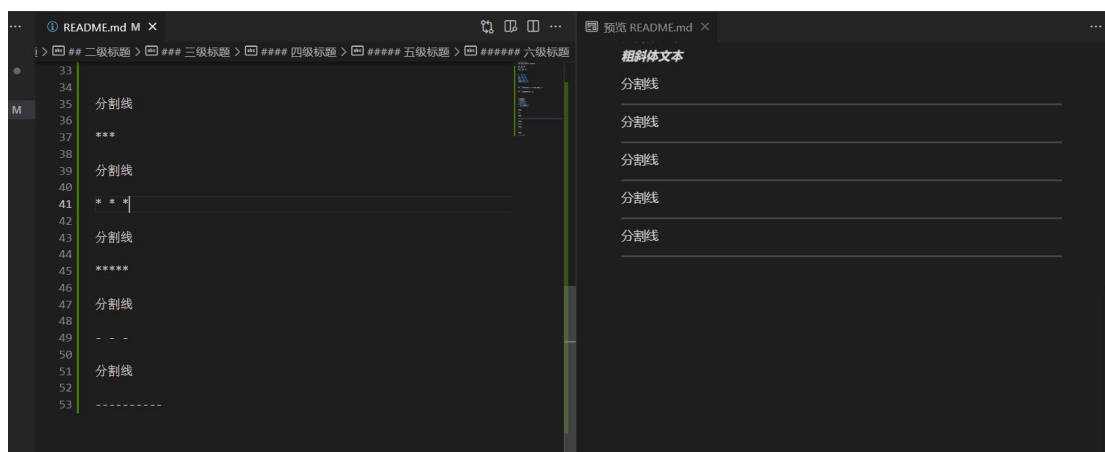
Markdown 的编写的换行与我们程序控制台打印的换行不同，在程序中，我们需要写入 ‘\n’ 来标识换行，但是在 markdown 中不用，我们只需要在源文件中直接敲两个空格回车换行即可。



在 markdown 中，也有字体可以供我们选择和更换，在 markdown 中，我们可以使用以下字体。

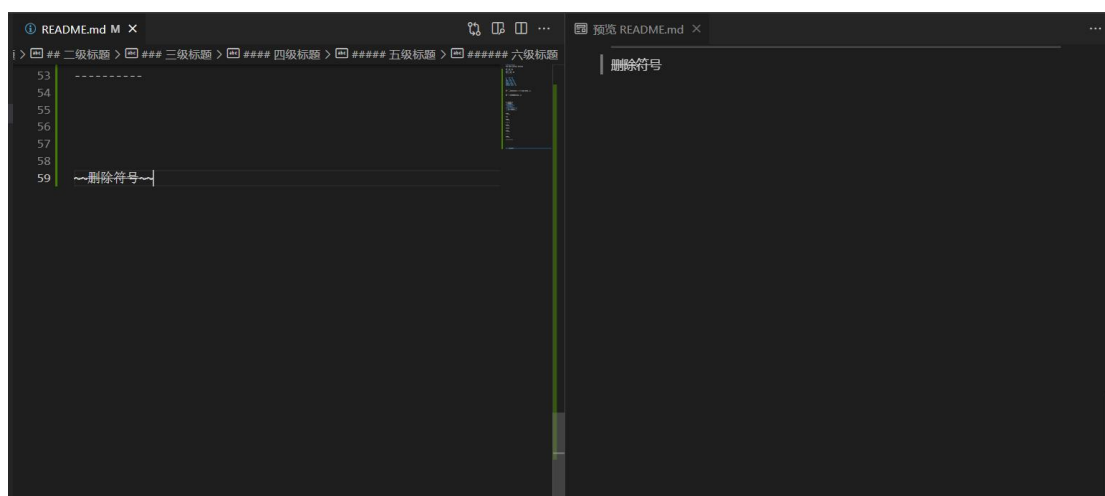


在 markdown 中，也有许多有趣的线段，如分割线，删除线，下划线。



注意：最好在分割线符号与文本之间空一行，负责可能导致分割线符号被识别为标题符号。

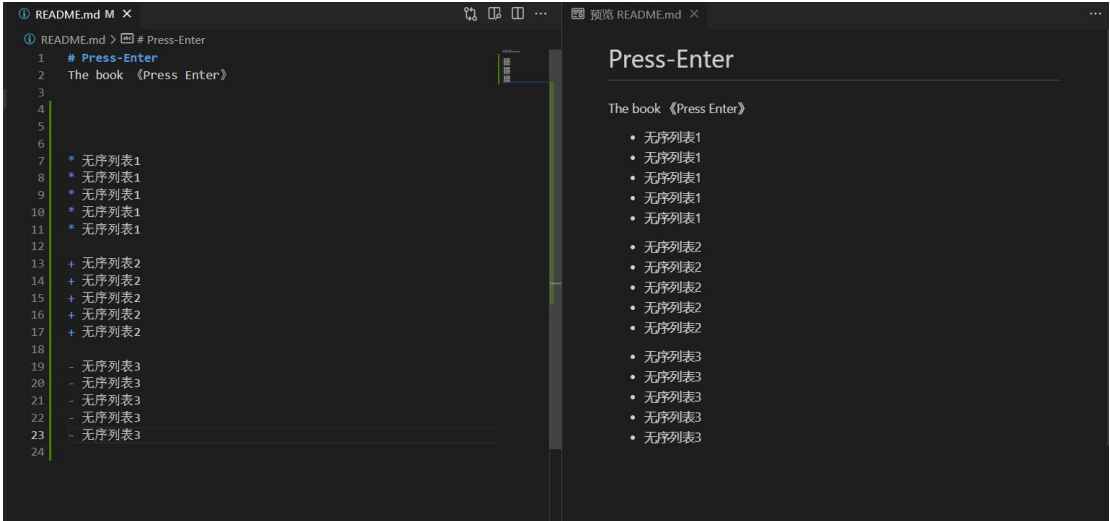
如果段落上的文字要添加删除线，只需要在文字的两端加上两个波浪线 ~~ 即可。



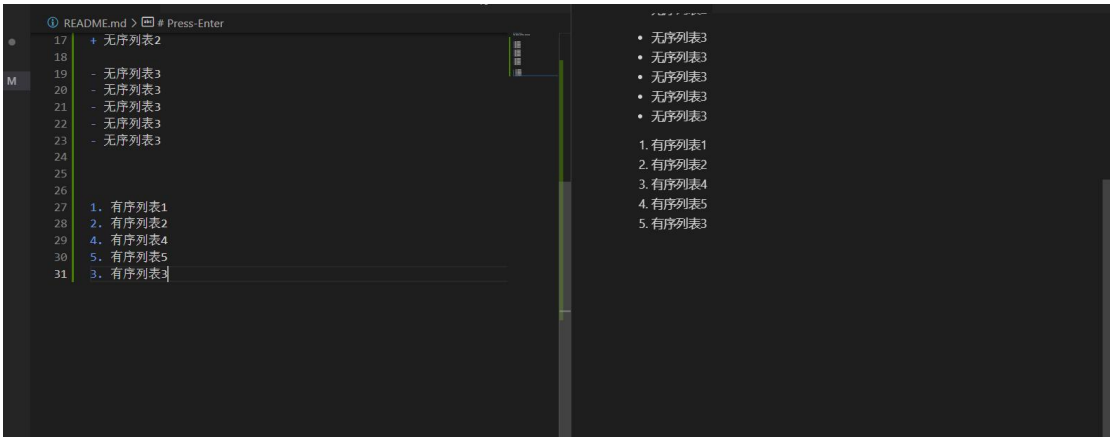
脚注是对文本的补充说明，当鼠标移动到文本上面时，可以看到我们编写的脚注详情。在 markdown 中，添加脚注的格式为： [^要注明的文本]

有时候我们为了方便查看和提炼内容，常常需要展示列表，markdown 也为我们提供了列表展示的标准。

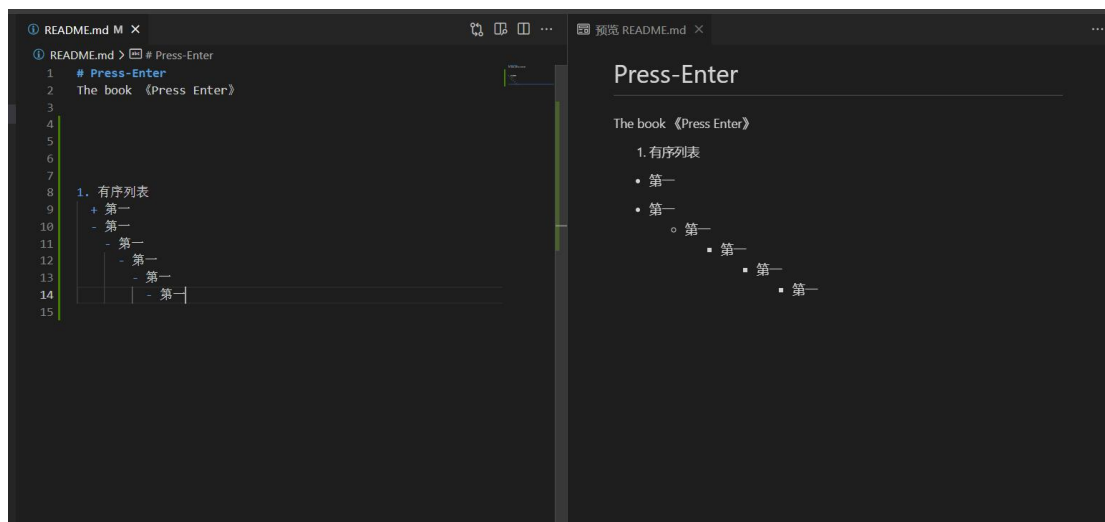
在 markdown 中，我们可以通过星号(\*)、加号(+)或是减号(-)后面加一个空格来标识无序列表。



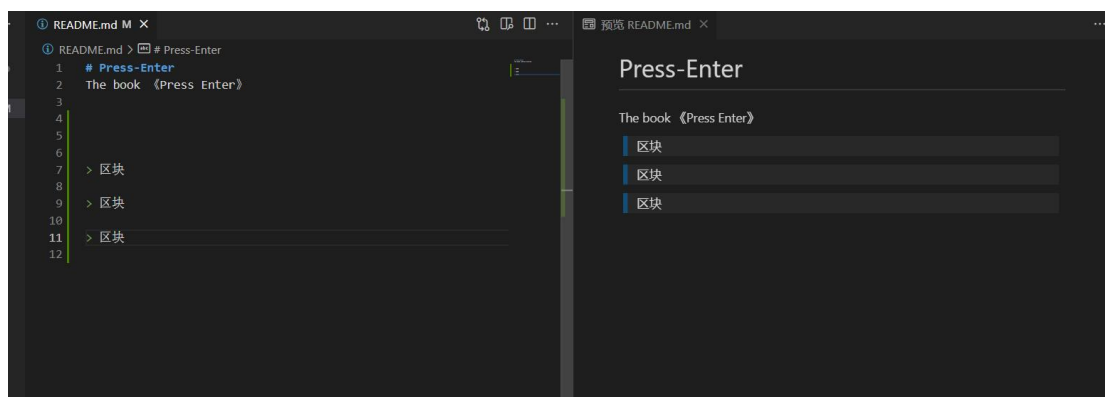
我们还可以标识有序列表，使用阿拉伯数字+ ‘.’ 后面再敲一个空格即可。列表是排列顺序不会因为阿拉伯数字的大小变化而变化！



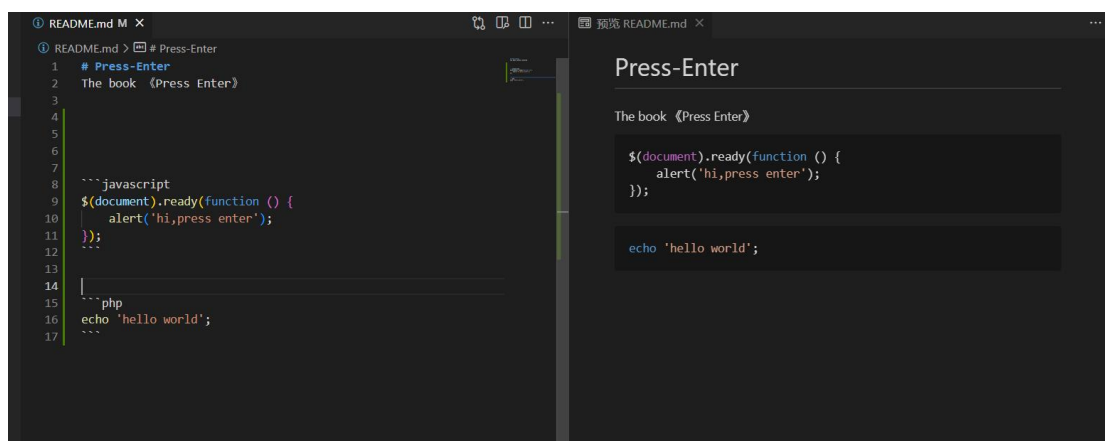
如果列表繁多，且层次更加复杂，我们可以采用列表嵌套的方式来进行表达，列表嵌套只需在子列表中的选项前面添加两个或四个空格即可。按理来说可以无限的嵌套列表，但是禁止套娃。



在 markdown 中，还有区块的概念，可以更好的分区分块，方便识别归属。使用区块，只需要添加 ‘>’ 后面空格就行，并且区块也支持与前面所介绍的列表进行嵌套。

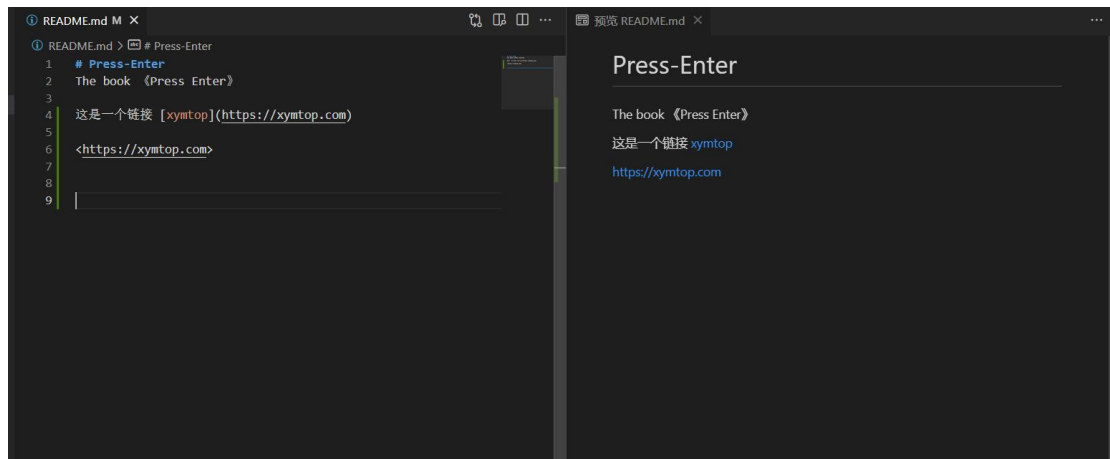


对于程序员来说，展示代码无疑是最核心的，markdown 中也有展示代码块的功能标准，并且根据渲染页面还可以语法高亮展示，语法结构为：“````编程语言 代码 ````”。

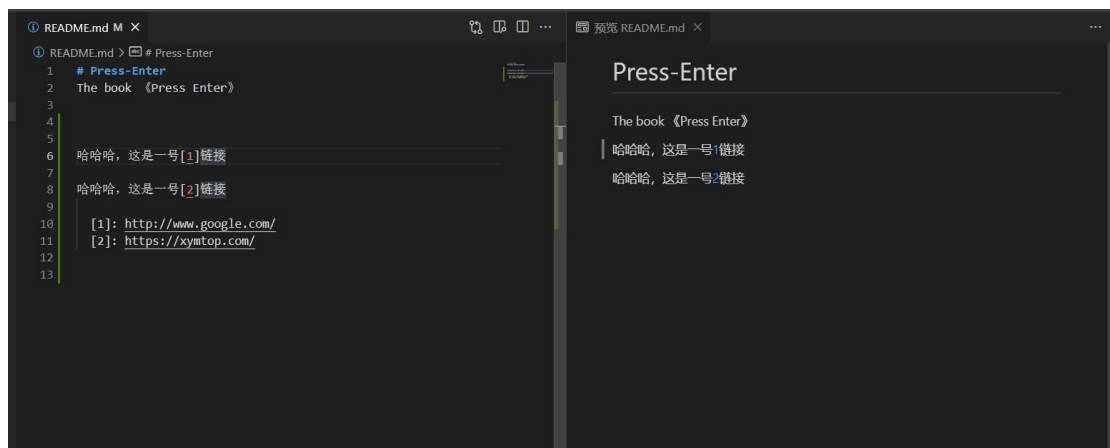




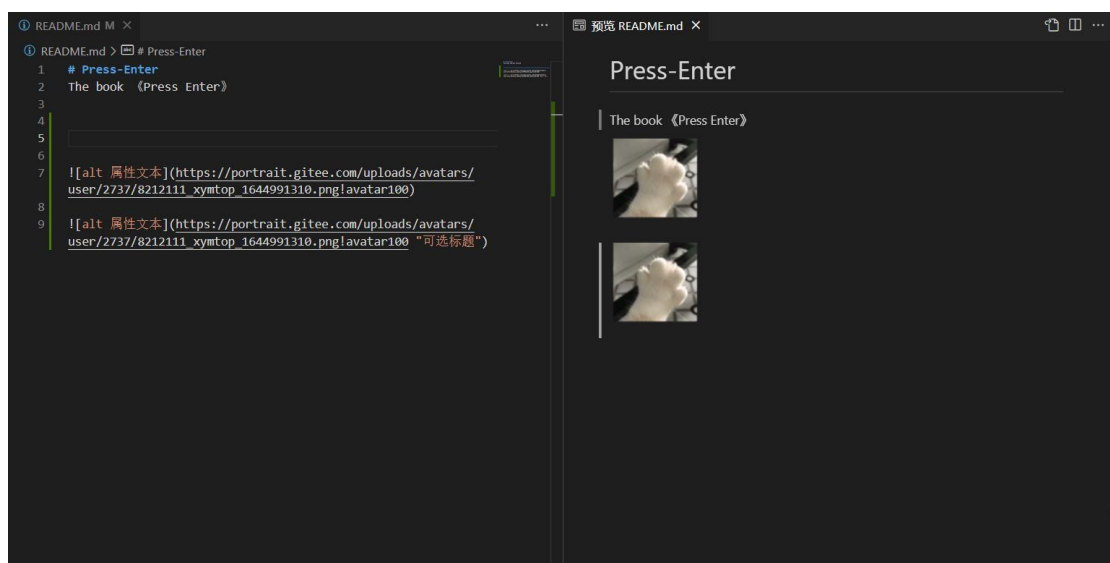
Markdown 提供链接展示组件，语法为：[链接名称](链接地址) 或者 <链接地址>



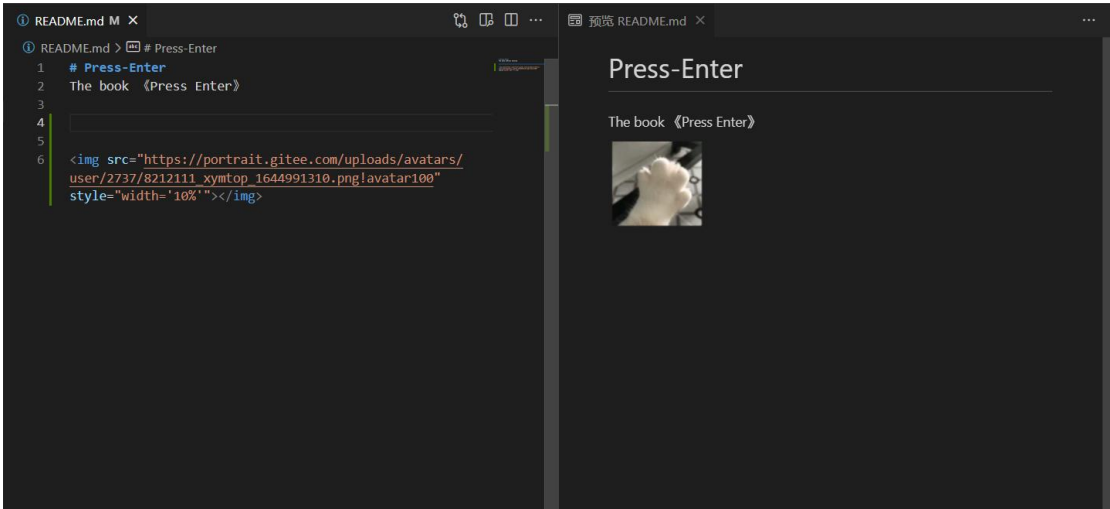
链接高级用法，变量赋值法



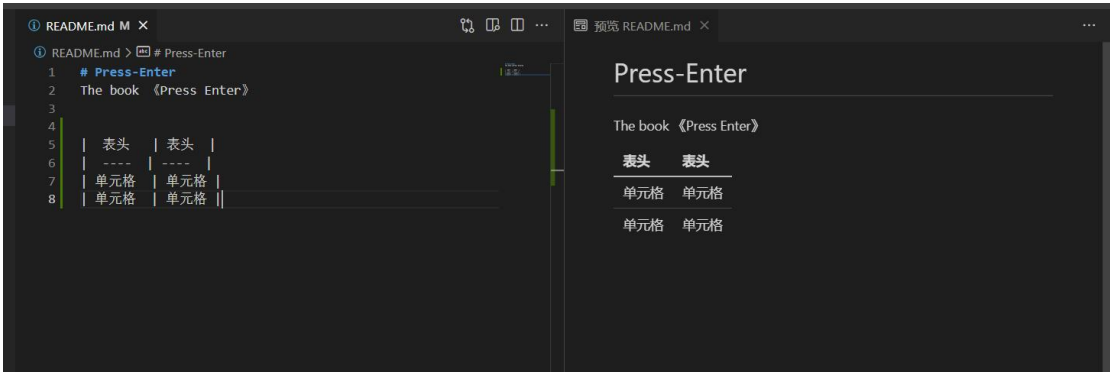
Markdown 中关于图片的展示，其中地址可以是在线地址或者本地地址，只要可访问即可。



在 markdown 中，因为图片不可以设置宽高属性值，所以我们可以借助 html 的语法来完成对图片的属性设置。

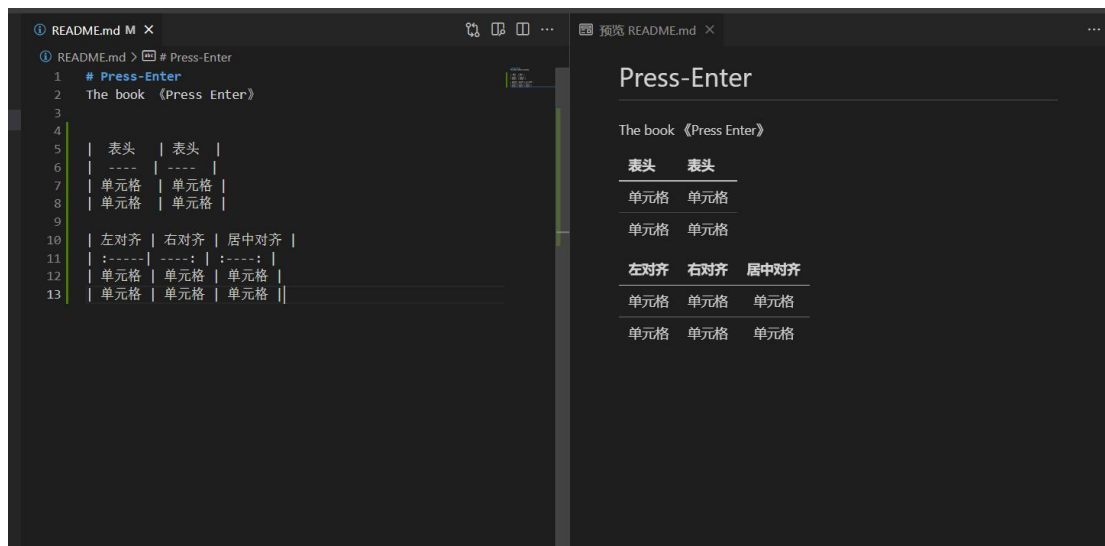


表格是我们日常使用比较频繁的工具，markdown 中也支持表格组件。Markdown 制作表格使用 | 来分隔不同的单元格，使用 - 来分隔表头和其他行。



为了美化表格，我们可以设置表格的对齐方式：

- : 设置内容和标题栏居右对齐。
- \_: 设置内容和标题栏居左对齐。
- =: 设置内容和标题栏居中对齐。



Markdown 中支持下列的符号加上 ‘\’ 来转义为相应的符号。

\	反斜线
`	反引号
*	星号
_	下划线
{ }	花括号
[ ]	方括号
( )	小括号
#	井字号
+	加号
-	减号
.	英文句点
!	感叹号

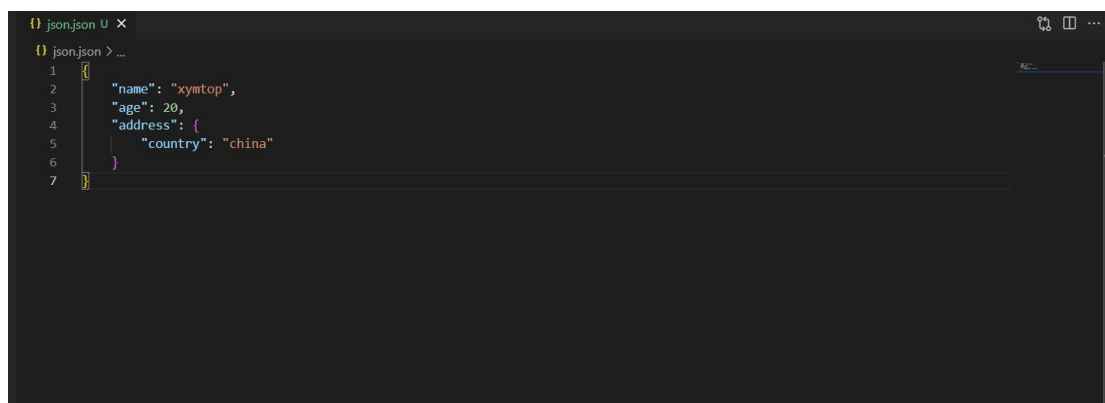
此外，markdown 中还直接支持 html 标签，以弥补 markdown 原生语法的不足。

常用HTML标签	标签	一般格式为: <标签名> 文本 </标签名>
换行	br	  </br>
加粗	b	<b>加粗</b>
斜体	i	<i>斜体</i>
下划线	u	u下划线<u>
删除线	del	<del>删除线</del>
加亮	mark	<mark>加亮</mark>
超链接	a 属性: href	<a href= '链接指向地址'>超链接</a> href="xxx":指定链接的目标。
文本下角	sub	文本<sub>下角</sub>
文本上角	sup	文本<sup>上角</sup>
小字体	small	<small>小字体</small>
大字体	big	<big>大字体</big>
打字机文本	tt	<tt>打字机文本</tt>

# JSON 与 XML

JSON (JavaScript Object Notation, JS 对象简谱) 是一种轻量级的数据交换格式。它基于 ECMAScript (European Computer Manufacturers Association, 欧洲计算机协会制定的 js 规范) 的一个子集, 采用完全独立于编程语言的文本格式来存储和表示数据。简洁和清晰的层次结构使得 JSON 成为理想的数据交换语言。易于人阅读和编写, 同时也易于机器解析和生成, 并有效地提升网络传输效率。

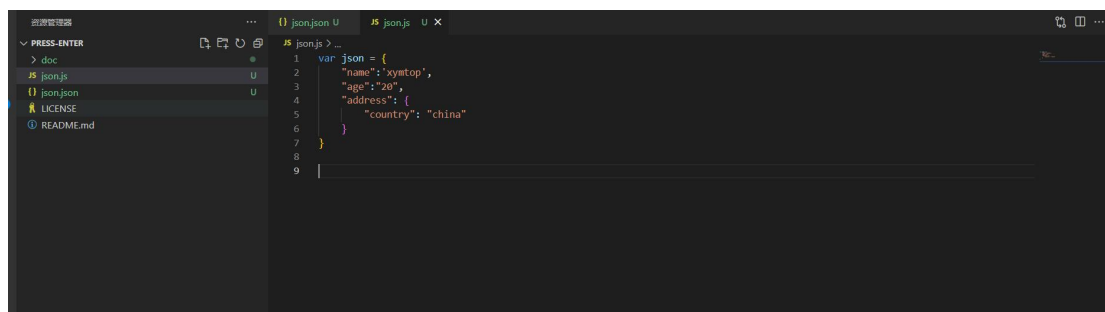
JSON (JavaScript Object Notation) 是一种轻量级的数据交换格式。易于人阅读和编写, 可以在多种语言之间进行数据交换。同时也易于机器解析和生成。它基于 JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999 的一个子集。



```
1 {  
2   "name": "xymtop",  
3   "age": 20,  
4   "address": {  
5     "country": "china"  
6   }  
7 }
```

JSON 是 Douglas Crockford 在 2001 年开始推广使用的数据格式, 在 2005 年-2006 年正式成为主流的数据格式, 雅虎和谷歌就在那时候开始广泛地使用 JSON 格式。任何支持的类型都可以通过 JSON 来表示, 例如字符串、数字、对象、数组等。但是对象和数组是比较特殊且常用的两种类型。

对象在 JS 中是使用花括号包裹 {} 起来的内容, 数据结构为 {key1: value1, key2: value2, ...} 的键值对结构。在面向对象的语言中, key 为对象的属性, value 为对应的值。键名可以使用整数和字符串来表示。值的类型可以是任意类型。




```
1 var json = {  
2   "name": "xymtop",  
3   "age": 20,  
4   "address": {  
5     "country": "china"  
6   }  
7 }  
8  
9
```

数组：数组在 JS 中是方括号 [] 包裹起来的内容，数据结构为 ["java", "javascript", "vb", ...] 的索引结构。在 JS 中，数组是一种比较特殊的数据类型，它也可以像对象那样使用键值对，但还是索引使用得多。同样，值的类型可以是任意类型。

简单地说，JSON 可以将 JavaScript 对象中表示的一组数据转换为字符串，然后就可以在网络或者程序之间轻松地传递这个字符串，并在需要的时候将它还原为各编程语言所支持的数据格式，例如在 PHP 中，可以将 JSON 还原为数组或者一个基本对象。在用到 AJAX 时，如果需要用到数组传值，这时就需要用 JSON 将数组转化为字符串。

对象是一个无序的“‘名称/值’对”集合。一个对象以 { 左括号开始，} 右括号结束。每个“名称”后跟一个: 冒号；“‘名称/值’对”之间使用逗号分隔。

和普通的 JS 数组一样，JSON 表示数组的方式也是使用方括号 []。



```
1 {  
2   "name": "xymtop",  
3   "age": 20,  
4   "address": {  
5     "country": "china"  
6   },  
7   "lang": ["js", "php", "java", "go"]  
8 }
```

在处理 JSON 格式的数据时，没有需要遵守的预定义的约束。所以，在同样的数据结构中，可以改变表示数据的方式，也可以使用不同方式表示同一事物。如前面所说，除了对象和数组，你也可以简单地使用字符串或者数字等来存储简单的数据，但这样并没有多大意义。

JSON 和 XML 的可读性可谓不相上下，一边是简易的语法，一边是规范的标签形式，很难分出胜负。

XML 天生有很好的扩展性，JSON 当然也有，没有什么 XML 可以扩展而 JSON 却不能扩展的。不过 JSON 在 Javascript 主场作战，可以存储 Javascript 复合对象，有着 xml 不可比拟的优势。

```

xml.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <country>
3      <name>中国</name>
4      <province>
5          <name>新疆维吾尔自治区</name>
6          <cities>
7              <school>新疆大学</school>
8              <school>新疆医科大学</school>
9          </cities>
10     </province>
11
12 </country>

```

XML 有丰富的编码工具，比如 Dom4j、Dom、SAX 等，JSON 也有提供的工具。无工具的情况下，相信熟练的开发人员一样能很快的写出想要的 xml 文档和 JSON 字符串，不过，xml 文档要多很多结构上的字符。

```

import java.net.URL;

import org.dom4j.Document;
import org.dom4j.DocumentException;
import org.dom4j.io.SAXReader;

public class Foo {

    public Document parse(URL url) throws DocumentException {
        SAXReader reader = new SAXReader();
        Document document = reader.read(url);
        return document;
    }
}

```

XML 的解析方式有两种：

一是通过文档模型解析，也就是通过父标签索引出一组标记。例如：  
`xmlData.getElementsByTagName("tagName")`，但是这样是要在预先知道文档结构的情况下使用，无法进行通用的封装。

另外一种方法是遍历节点（document 以及 childNodes）。这个可以通过递归来实现，不过解析出来的数据仍旧是形式各异，往往也不能满足预先的要求。凡是这样可扩展的结构数据解析起来一定都很困难。

JSON 也同样如此。如果预先知道 JSON 结构的情况下，使用 JSON 进行数据传递简直是太美妙了，可以写出很实用美观可读性强的代码。如果你是纯粹的前台开发人员，一定会非常喜欢 JSON。但是如果你是一个应用开发人员，就不是那么喜欢了，毕竟 xml 才是真正的结构化标记语言，用于进行数据传递。

而如果不知道 JSON 的结构而去解析 JSON 的话，那简直是噩梦。费时费力不说，代码也会变得冗余拖沓，得到的结果也不尽人意。但是这样也不影响众多前台开发人员选择 JSON。因为 json.js 中的 toJSONString() 就可以看到 JSON 的字符串结构。当然对于不是经常使用这个字符串的人，这样做仍旧是噩梦。常用 JSON 的人看到这个字符串之后，就对 JSON 的结构很明了了，就更容易的操作 JSON。以上是在 Javascript 中仅对于数据传递的 xml 与 JSON 的解析。在 Javascript 地盘内，JSON 毕竟是主场作战，其优势当然要远远优越于 xml。如果 JSON 中存储 Javascript 复合对象，而且不知道其结构的话，我相信很多程序员也一样是哭着解析 JSON 的。

除了上述之外，JSON 和 XML 还有另外一个很大的区别在于有效数据率。JSON 作为数据包格式传输的时候具有更高的效率，这是因为 JSON 不像 XML 那样需要有严格的闭合标签，这就让有效数据量与总数据包比大大提升，从而减少同等数据流量的情况下，网络的传输压力。

可扩展标记语言 (Extensible Markup Language, XML)，标准通用标记语言的子集，可以用来标记数据、定义数据类型，是一种允许用户对自己的标记语言进行定义的源语言。XML 是标准通用标记语言 可扩展性良好, 内容与形式分离, 遵循严格的语法要求, 保值性良好等优点。

在电子计算机中，标记指计算机所能理解的信息符号，通过此种标记，计算机之间可以处理包含各种的信息比如文章等。它可以用来标记数据、定义数据类型，是一种允许用户对自己的标记语言进行定义的源语言。它非常适合万维网传输，提供统一的方法来描述和交换独立于应用程序或供应商的结构化数据。是 Internet 环境中跨平台的、依赖于内容的技术，也是当今处理分布式结构信息的有效工具。早在 1998 年，W3C 就发布了 XML1.0 规范，使用它来简化 Internet 的文档信息传输。

可扩展标记语言与 Access, Oracle 和 SQL Server 等数据库不同，数据库提供了更强有力的数据存储和分析能力，例如：数据索引、排序、查找、相关一致性等，可扩展标记语言仅仅是存储数据。事实上它与其他数据表现形式最大的不同是：可扩展标记语言极其简单，这是一个看上去有点琐细的优点，但正是这点使它与众不同。

XML 的简单易学在任何应用程序中读/写数据，这使 XML 很快成为数据交换语言（此类语言主要包括 XML, JSON 等，常用于接口调用，配置文件，数据存储等场景），虽然不同的应用软件也支持其他的数据交换格式，但不久之后它们都将支持 XML，那就意味着程序可以更容易的与 Windows、Mac OS、Linux 以及其他平台下产生的信息结合，然后可以很容易加载 XML 数据到程序中并分析它，并以 XML 格式输出结果。

XML 有两个先驱：SGML 和 HTML，这两个语言都是非常成功的标记语言，但是都有一些与生俱来的缺陷。XML 正是为了解决它们的不足而诞生的。



早在 Web 未发明之前，SGML(Standard Generalized Markup Language，标准通用标记语言)就已存在，正如它的名称所言，SGML 是国际上定义电子文件结构和内容描述的标准。SGML 具有非常复杂的文档结构，主要用于大量高度结构化数据的访问和其他各种工业领域，在分类和索引数据中非常有用。

虽然 SGML 的功能很强大，但是它不适用于 Web 数据描述，而且 SGML 软件的价格非常昂贵；另外，SGML 十分庞大，既不容易学，又不容易使用，在计算机上实现也十分困难：不仅如此，几个主要的浏览器厂商都明确拒绝支持 SGML，这无疑是 SGML 在网上传播遇到的最大障碍。鉴于这些因素，Web 的发明者——欧洲核子物理研究中心的研究人员，根据当时（1989 年）的计算机技术，发明并推出了 HTML。

1989 年，HTML 诞生，它抛弃了 SGML 复杂庞大的缺点，继承了 SGML 的很多优点。HTML 最大的特点是简单性和跨平台性。

HTML 是一种界面技术，它只使用了 SGML 中很少的一部分标记，例如 HTML 4.0 中只定义了 70 余种标记。为了便于在计算机上实现，HTML 规定的标记是固定的，即 HTML 语法是不可扩展的。HTML 这种固定的语法使它易学易用，在计算机上开发 HTML 的浏览器也十分容易。正是由于 HTML 的简单性，使得基于 HTML 的 Web 应用得到了极大的发展。

随着 Web 应用的不断发展，HTML 的局限性也越来越明显地显现了出来，如 HTML 无法描述数据、可读性差、搜索时间长等。人们又把目光转向 SGML，再次改造 SGML 使之适应现在的网络需求。随着先辈的努力，1998 年 2 月 10 日，W3C(World Wide Web Consortium，万维网联盟)公布 XML 1.0 标准，XML 诞生了。

XML 最初的设计目的是为了 EDI(Electronic Data Interchange，电子数据交换)，确切地说是为 EDI 提供一个标准数据格式。

当前的一些网站内容建设者们已经开始开发各种各样的 XML 扩展，比如数学标记语言 MathML、化学标记语言 CML 等。此外，一些著名的 IT 公司，如 Oracle、IBM 以及微软等都积极地投入人力与财力研发 XML 相关软件与服务支持，这无疑确定了 XML 在 IT 产业的重要地位。

XML 具有以下特点：

(1) XML 可以从 HTML 中分离数据。即能够在 HTML 文件之外将数据存储在 XML 文档中，这样可以使开发者集中精力使用 HTML 做好数据的显示和布局，并确保数据改动时不会导致 HTML 文件也需要改动，从而方便维护页面。XML 也能够将数据以“数据岛”的形式存储在 HTML 页面中，开发者依然可以把精力集中到使用 HTML 格式化和显示数据上。

(2) XML 可用于交换数据。基于 XML 可以在不兼容的系统之间交换数据，计算机系统和数据库系统所存储的数据有多种形式，对于开发者来说，最耗时间的工作就是在遍布网络的系统之间交换数据。把数据转换为 XML 格式存储将大大减少交换数据时的复杂性，还可以使这些数据能被不同的程序读取。

(3) XML 可应用于 B2B 中。例如在网络中交换金融信息，目前 XML 正成为遍布网络的商业系统之间交换信息所使用的主要语言，许多与 B2B 有关的完全基于 XML 的应用程序正在开发中。

(4) 利用 XML 可以共享数据。XML 数据以纯文本格式存储，这使得 XML 更易读、更便于记录、更便于调试，使不同系统、不同程序之间的数据共享变得更加简单。

(5) XML 可以充分利用数据。XML 是与软件、硬件和应用程序无关的，数据可以被更多的用户、设备所利用，而不仅仅限于基于 HTML 标准的浏览器。其他客户端和应用程序可以把 XML 文档作为数据源来处理，就像操作数据库一样，XML 的数据可以被各种各样的“阅读器”处理。

(6) XML 可以用于创建新的语言。比如，WAP 和 WML 语言都是由 XML 发展来的。WML(Wireless Markup Language, 无线标记语言)是用于标识运行于手持设备上(比如手机)的 Internet 程序的工具，它就采用了 XML 的标准。

总之，XML 使用一个简单而又灵活的标准格式，为基于 Web 的应用提供了一个描述数据和交换数据的有效手段。但是，XML 并非是用来取代 HTML 的。HTML 着重如何描述将文件显示在浏览器中，而 XML 与 SGML 相近，它着重描述如何将数据以结构化方式表示。

XML 文件格式是纯文本格式，在许多方面类似于 HTML，XML 由 XML 元素组成，每个 XML 元素包括一个开始标记(<)，一个结束标记(>)以及两个标记之间的内容，例如，可以将 XML 元素标记为价格、订单编号或名称。标记是对文档存储格式和逻辑结构的描述。在形式上，标记中可能包括注释、引用、字符数据段、起始标记、结束标记、空元素、文档类型声明(DTD)和序言。

具体规则如下：

#### 1、必须有声明语句

XML 声明是 XML 文档的第一句，其格式如下：

```
<?xml version="1.0" encoding="utf-8"?>
```

#### 2、注意大小写

在 XML 文档中，大小写是有区别的。“A”和“a”是不同的标记。注意在写元素时，前后标记的大小写要保持一致。最好养成一种习惯，或者全部大写，或者全部小写，或者大写第一个字母，这样可以减少因为大小写不匹配而产生的文档错误。

#### 3、XML 文档有且只有一个根元素

良好格式的 XML 文档必须有一个根元素，就是紧接着声明后面建立的第一个元素，其他元素都是这个根元素的子元素，根元素完全包括文档中其他所有的元素。根元素的起始标记要放在所有其他元素的起始标记之前；根元素的结束标记要放在所有其他元素的结束标记之后。

#### 4、属性值使用引号

在 HTML 代码里面，属性值可以加引号，也可以不加。但是 XML 规定，所有属性值必须加引号(可以是单引号，也可以是双引号，建议使用双引号)，否则将被视为错误。

#### 5、所有的标记必须有相应的结束标记

在 HTML 中，标记可以不成对出现，而在 XML 中，所有标记必须成对出现，有一个开始标记，就必须有一个结束标记，否则将被视为错误。

## 6、所有的空标记也必须被关闭

空标记是指标记对之间没有内容的标记，比如“”等标记。在 XML 中，规定所有的标记必须有结束标记。

(1)可扩展性方面：HTML 不允许用户自行定义他们自己的标识或属性，而在 XML 中，用户能够根据需要自行定义新的标识及属性名，以便更好地从语义上修饰数据。

(2)结构性方面：HTML 不支持深层的结构描述，XML 的文件结构嵌套可以复杂到任意程度，能表示面向对象的等级层次。

(3)可校验性方面：HTML 没有提供规范文件以支持应用软件对 HTML 文件进行结构校验，而 XML 文件可以包括一个语法描述，使应用程序可以对此文件进行结构校验。

虽然 XML 标准本身简单，但与 XML 相关的标准却种类繁多，W3C 制定的相关标准就有二十多个，采用 XML 制定的重要的电子商务标准就有十多个。这一方面说明 XML 确实是一种非常实用的结构化通用标记语言，并且已经得到广泛应用；另一方面，这又为了解这些标准带来一定的困难，除了标准种类繁多外，标准之间通常还互相引用，特别是应用标准，它们的制定不仅仅使用的是 XML 标准本身，还常常用到了其他很多标准。XML 标准的体系与 SGML 标准的体系非常相似，XML 相关标准也可分为元语言标准、基础标准、应用标准三个层次。

描述的是用来描述标准的元语言。在 XML 标准体系中就是 XML 标准，是整个体系的核心，其他 XML 相关标准都是用它制定的或为其服务的。

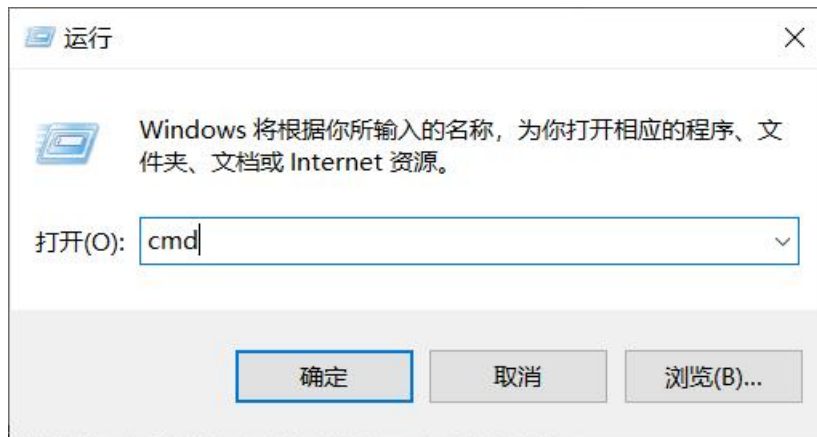
这一层次的标准是为 XML 的进一步实用化制定的标准，规定了采用 XML 制定标准时的一些公用特征、方法或规则。如 XML Schema 描述了更加严格地定义 XML 文档的方法，以便可以更自动化处理 XML 文档；XMLNamespace 用于保证 XML DTD 中名字的一致性，以便不同的 DTD 中的名字在需要时可以合并到一个文档中；XSL 是描述 XML 文档样式与转换的一种语言；XLink 用来描述 XML 文档中的超链接；XPointer 描述了定位到 XML 文档结构内部的方法；DOM 定义了与平台和语言无关的接口，以便程序和脚本动态访问和修改文档内容、结构及样式等。

XML 已开始被广泛接受，大量的应用标准，特别是针对因特网的应用标准，纷纷采用 XML 进行制定。有人甚至认为，XML 标准是因特网时代的 ASCII 标准。在这因特网时代，几乎所有的行业领域都与因特网有关。而它们一旦与因特网发生关系，都必然要有其行业标准，而这些标准往往采用 XML 来制定。

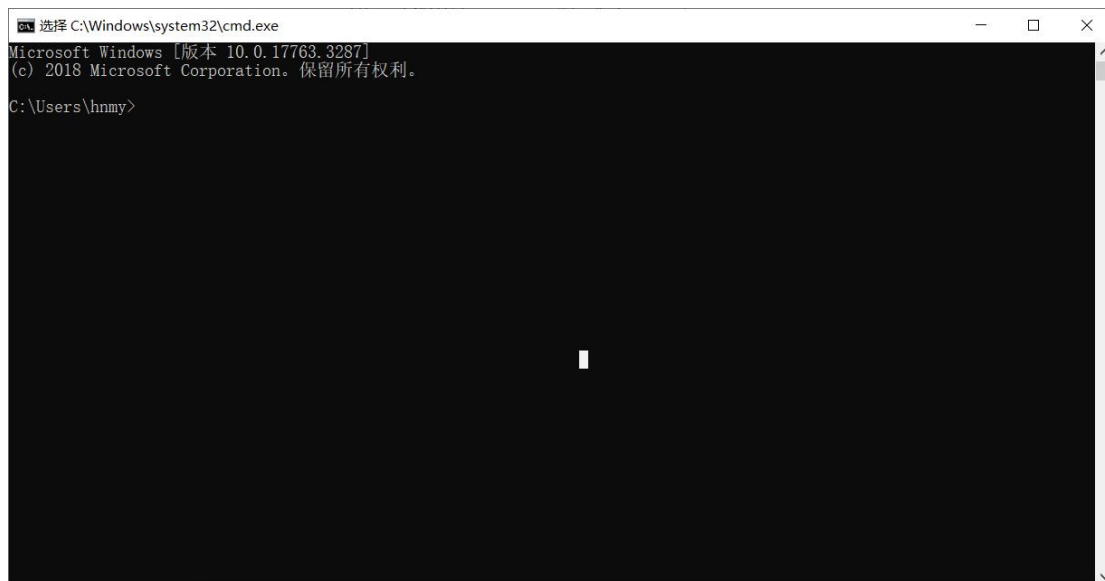
# BAT 脚本

Bat 脚本是专门在 windows 系统上运行的脚本文件, 后缀名可以为 .bat 或者 .cmd, 运行时调用 windows 系统中环境变量路径里面 cmd.exe 来执行命令, 与 linux 系统中的 shell 脚本类似, 可以进行批处理操作, 方便用户的操作。

首先, 我们打开 windows 电脑, 按下 win+R 键。



输入 cmd, 然后按回车键进入微软 windows 中自带的 command 命令行工具, 黑色的窗口。



与 linux 的终端类似, 我们可以在里面输入 command 命令并按下回车键执行, 例如, 我将演示如何在命令行打印出一条消息。

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.17763.3287]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\hnmmy>echo "press enter"
press enter

C:\Users\hnmmy>
```

## 访问网址

explorer <http://www.baidu.com>



## 打开文件夹或文件

打开文件夹

start D:\文件夹 1

打开文件

```
start D:\文件夹1\test.txt
```

## 删除文件

删除当前目录下的 **test.txt** 文件

```
del test.txt
```

删除上级目录下的 **test.txt** 文件

```
del ..\test.txt
```

删除当前目录 **TEST** 文件夹下的所有 **.o** 文件

```
del .\TEST\*.o
```

## 复制文件

复制当前目录下所有 **txt** 文件到文件夹 **1**，文件夹 **1** 需要已经创建 **copy \*.txt** 文件夹 **1**

复制文件 **1** 到文件夹 **1**、文件 **2** 到文件夹 **2**、支持多个文件操作，同时支持上级及下级文件路径

```
copy file1.txt 文件夹1  
copy file2.txt 文件夹2
```

复制桌面文件到 **D** 盘根目录，使用绝对路径

```
copy C:\Users\user\Desktop\welcome.txt D:\
```

## 重命名

修改文件扩展名，所有 **txt** 扩展名改为 **mp3** 扩展名

```
ren *.txt *.mp3  
ren *.gif *.jpg
```

修改文件名称，把 **aa.txt** 改为 **bb.c**

```
ren aa.txt bb.c
```

## 创建文件夹

创建三个文件夹 1

```
md 文件夹 1  
md 文件夹 2  
md 文件夹 3
```

## 创建文件

当前目录创建 a.txt 文件

```
cd.>a.txt
```

## 把 hex 文件的第一行之后的内容写入新文件

```
more +1 "..\OBJ\output.hex">"..\OBJ\flash_after_del_hex_line1.hex"
```

## 删除 Keil 编译产生的垃圾文件

```
%删除 OBJ 目录下的多余文件%  
del ..\OBJ\*.lnp /s  
::del ..\OBJ\*.opt /s  ::不允许删除 JLINK 的设置  
del ..\OBJ\*.__i /s  
del ..\OBJ\*.crf /s  
del ..\OBJ\*.o /s  
del ..\OBJ\*.d /s
```

```
%删除 USER 目录下的多余文件%  
del *.map /s  
del *.lst /s  
del *.dep /s
```

echo 编译产生的其他文件已经删除

## 提取文件名

提取当前目录下扩展名为 mp3 的文件名，输出到 mp3 文件名.txt

```
dir *.mp3 /b>mp3 文件名.txt
```

提取当前目录下的“深度睡眠”文件夹下的所有 mp3 文件名到文件

```
dir .\深度睡眠\*.mp3 /b>mp3 文件名.txt
```

提取当前目录下所有文件的文件名到 a.txt

```
dir c:\*.* >a.txt
```

## 输出文件的绝对路径信息

输出当前目录下 mp3 文件

```
dir *.mp3/b/s>MP3 文件信息.txt
```

MP3 文件信息.txt 的内容

```
D:\Music\深度睡眠\01.细水长流.mp3
D:\Music\深度睡眠\02.花絮轻撒.mp3
D:\Music\深度睡眠\03.爱的轮回.mp3
D:\Music\深度睡眠\04.月影摇曳.mp3
D:\Music\深度睡眠\05.逐梦.mp3
D:\Music\深度睡眠\06.萦绕天使.mp3
D:\Music\深度睡眠\07.夜宴.mp3
```

## 开启电脑热点

开启电脑无线，设置用户名和密码

```
netsh wlan set hostednetwork mode=allow LAPTOP key=1234567890
netsh wlan start hostednetwork
pause
```

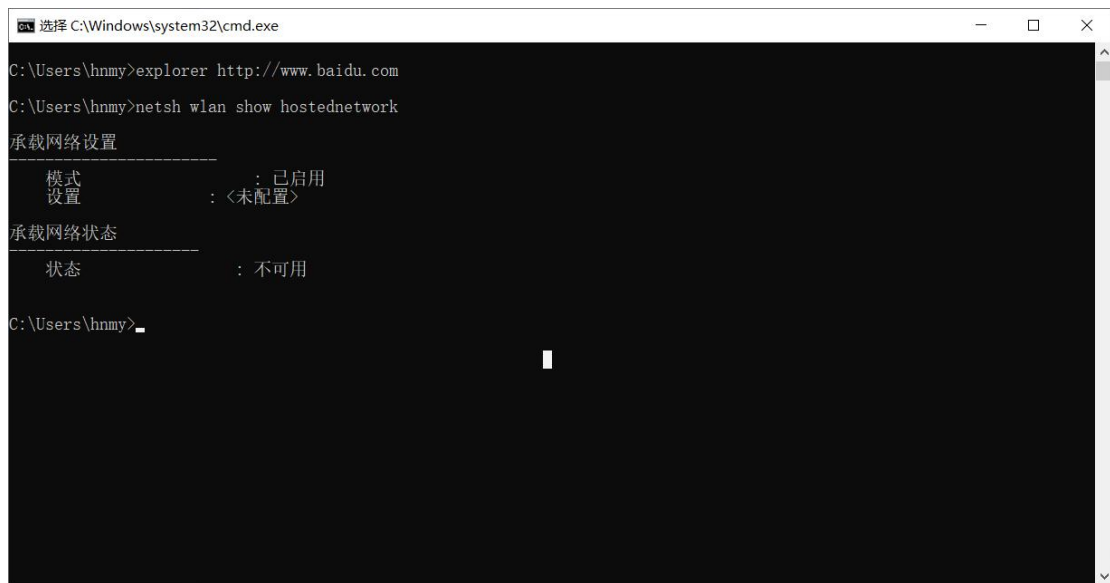
## 关闭热点

```
netsh wlan stop hostednetwork
pause
```

## 查看当前热点信息

```
netsh wlan show hostednetwork
Pause
```





```
C:\Windows\system32\cmd.exe
C:\Users\hnmy>explorer http://www.baidu.com
C:\Users\hnmy>netsh wlan show hostednetwork
承载网络设置
-----
模式                : 已启用
设置                : <未配置>
承载网络状态
-----
状态                : 不可用
C:\Users\hnmy>
```

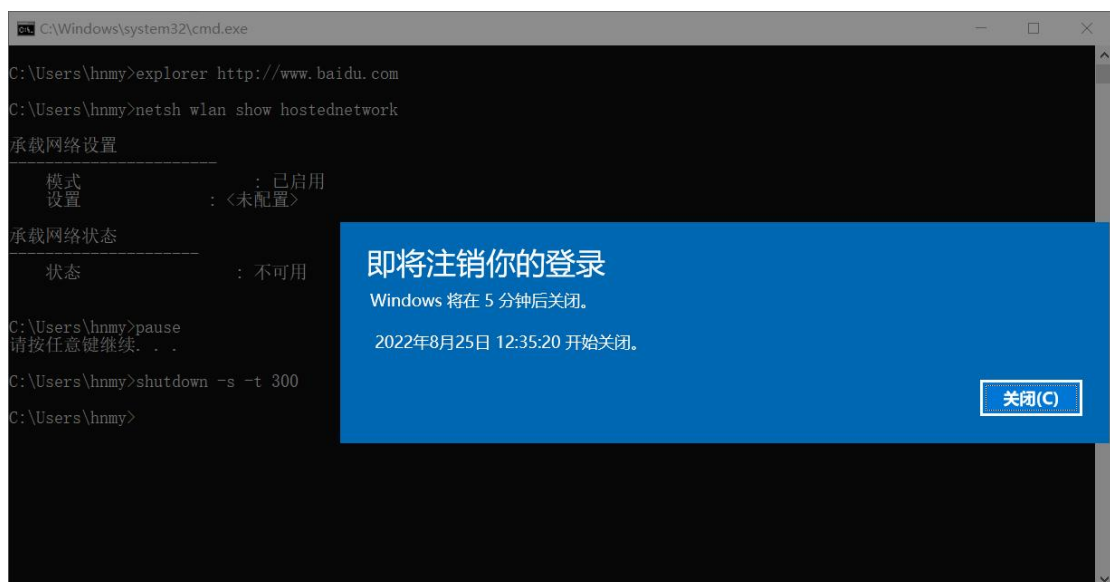
## 执行另一个批处理文件

```
call c:\code\run.bat
```

## 自动关机

300s 后自动关机

```
shutdown -s -t 300
```



## 取消自动关机

```
shutdown -a
```

## 立刻重启

```
shutdown -r -t 0
```

## 自动休眠

60s 后休眠

```
shutdown -h -t 60
```

## 隐藏文件夹

```
attrib +s +h D:\Secret
```

## 取消隐藏文件夹

```
attrib -s -h D:\Secret
```

## attrib 命令

```
attrib +/ -r
```

```
attrib +/ -a
```

```
attrib +/ -s
```

```
attrib +/ -h
```

+ 设置属性

- 清除属性

r 只读属性

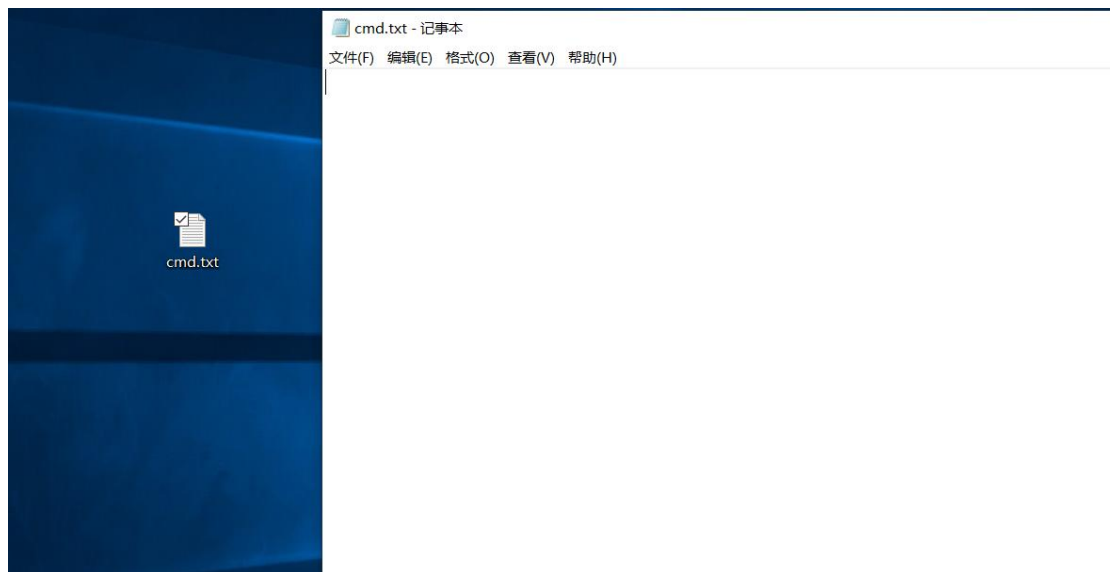
a 存档属性

s 系统属性

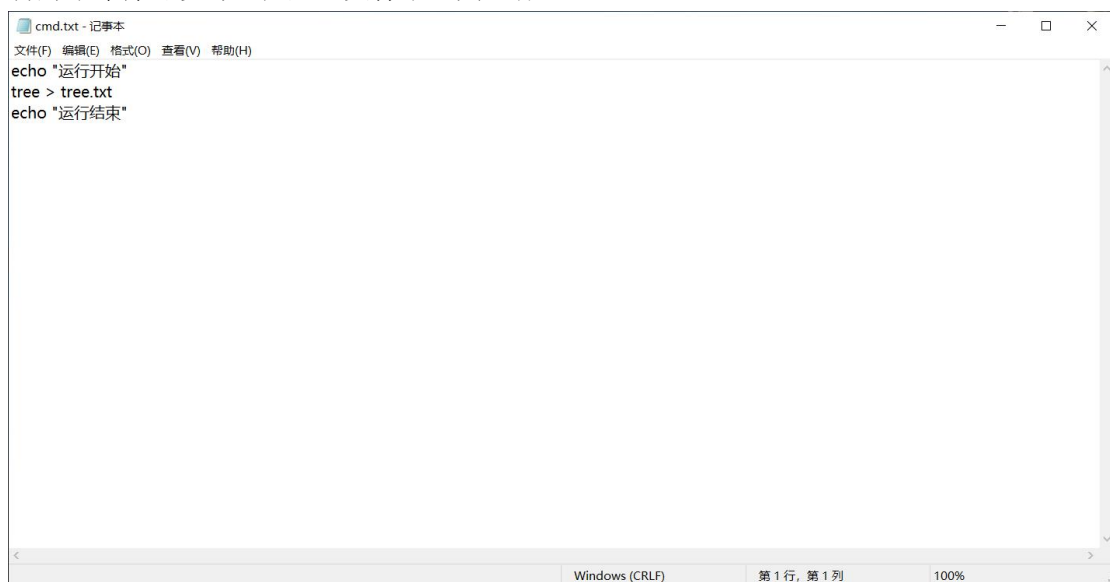
h 隐藏属性

我们可以直接在 **cmd** 里面运行命令，我们也可以将命令提前写在文本里面，修改文本为相应的格式，最后让 **cmd** 程序统一批处理的运行命令。

首先，在桌面新建一个文本文件，用记事本打开。



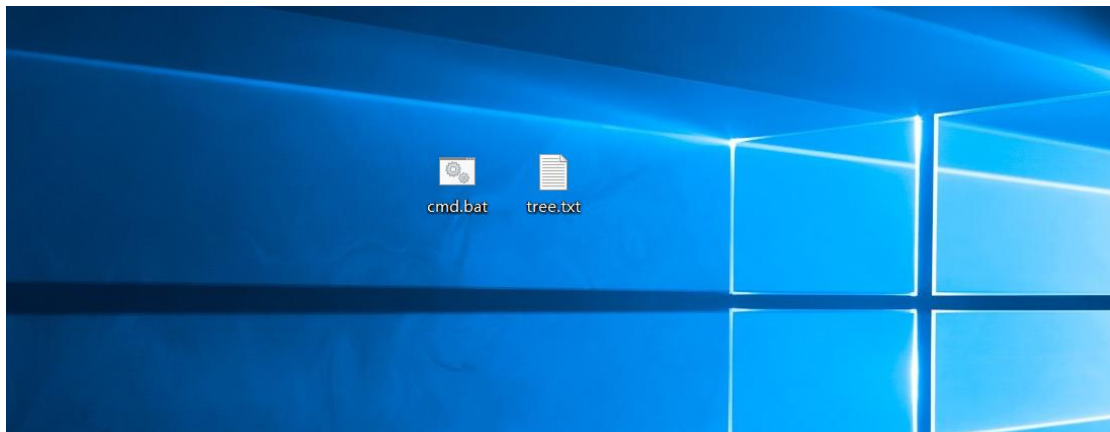
然后按顺序写入我们需要执行的命令行命令，理论上来说，我们前面尝试过在黑窗口运行成功的命令都可以写入批处理文件中让其运行。



最后，我们将 txt 文件修改后缀为 bat 或者 cmd，双击运行查看效果

```
选择 C:\Windows\system32\cmd.exe
D:\desktop\desktop>echo "运行开始"
"运行开始"
D:\desktop\desktop>tree 1>tree.txt
```

运行结束后，将会在桌面产生一个 **tree.txt** 的文件，这就是我们批处理脚本运行成功的成果，里面将会扫描当前文件夹的目录树并写入到 **tree.txt** 里面。



```
tree.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
| | | | | icons-svg
| | | | | | es
| | | | | | | asn
| | | | | | | inline-namespaced-svg
| | | | | | | filled
| | | | | | | outlined
| | | | | | | twotone
| | | | | | | inline-svg
| | | | | | | filled
| | | | | | | outlined
| | | | | | | twotone
| | | | | | lib
| | | | | | | asn
| | | | | | icons-vue
| | | | | | | es
| | | | | | | components
| | | | | | | icons
| | | | | | lib
| | | | | | components
| | | | | | icons
| | | | | @antfu
| | | | | | utils
| | | | | | dist
```

# 环境变量

环境变量 (environment variables) 一般是指在操作系统中用来指定操作系统运行环境的一些参数，如：临时文件夹位置和系统文件夹位置等。

环境变量是在操作系统中一个具有特定名字的对象，它包含了一个或者多个应用程序所将使用到的信息。例如 Windows 和 DOS 操作系统中的 path 环境变量，当要求系统运行一个程序而没有告诉它程序所在的完整路径时，系统除了在当前目录下面寻找此程序外，还应到 path 中指定的路径去找。用户通过设置环境变量，来更好的运行进程。

环境变量相当于给系统或用户应用程序设置的一些参数，具体起什么作用这当然和具体的环境变量相关。比如 path，是告诉系统，当要求系统运行一个程序而没有告诉它程序所在的完整路径时，系统除了在当前目录下面寻找此程序外，还应到哪些目录下去寻找；再如 tc 或 vc++ 中，set include=path1;path2; 是告诉编译程序到哪里去找 .h 类型的文件；当然不仅仅是指定什么路径，还有其它的作用的，如 set dircmd=/4 设置一个环境变量的作用是在使用 dir 命令时会把 /4 作为缺省的参数添加到你的 dir 命令之后，就像你的每个命令都加了 /4 参数，它实际上是给命令解释程序 command 设置的一个环境变量，并且是给 dir 这个内部命令设置的。

DWORD GetEnvironmentVariable(LPCSTR lpName, LPSTR lpBuffer, DWORD dSize), 参数 lpName 是你要求查询的环境变量的名，lpBuffer 是返回你所指定的环境变量的值的，dSize 是告诉这个函数 lpBuffer 可以存放多少个字节。

分析本地故障时原因很可能就是因为环境变量中的默认路径被删除的结果，默认路径一经设置，当前系统如有程序运行时需要某些 DLL 或 EXE 文件，以及 Active 控件时就会到所有默认路径中去查找，如果在这些目录中查找到相应的程序则自动加载，查找不到则报告缺少某某文件的错误信息。

很多朋友会在自己的计算机上安装双系统，例如 C 盘安装 Windows XP，D 盘安装 Windows 7。可是某些软件往往只在 Windows XP 系统中安装，Windows 7 系统中是无法正常使用的，比较麻烦却有效的方法是再安装一遍。当我们了解了环境变量中的用途后就可以很好解决双系统的软件共用问题。

为什么在 Windows XP 中安装了的软件在 Windows 7 下无法运行呢 (绿色软件除外)。原因是安装软件时往往须要向系统目录中复制某些文件，而使用另外一个系统时会由于缺少这些文件而无法运行。因此，我们可以通过设置环境变量的方法来解决这个问题。

总之一句话，环境变量就是告诉系统哪个软件在哪个地方！！！！

以 Java 环境为例，我们编译 Java 代码需要用到 Java 给我们提供的 Java 的 jdk，我们可以在 cmd 里面输入 java 来查看 cmd 的返回数据。

```
C:\Windows\system32\cmd.exe

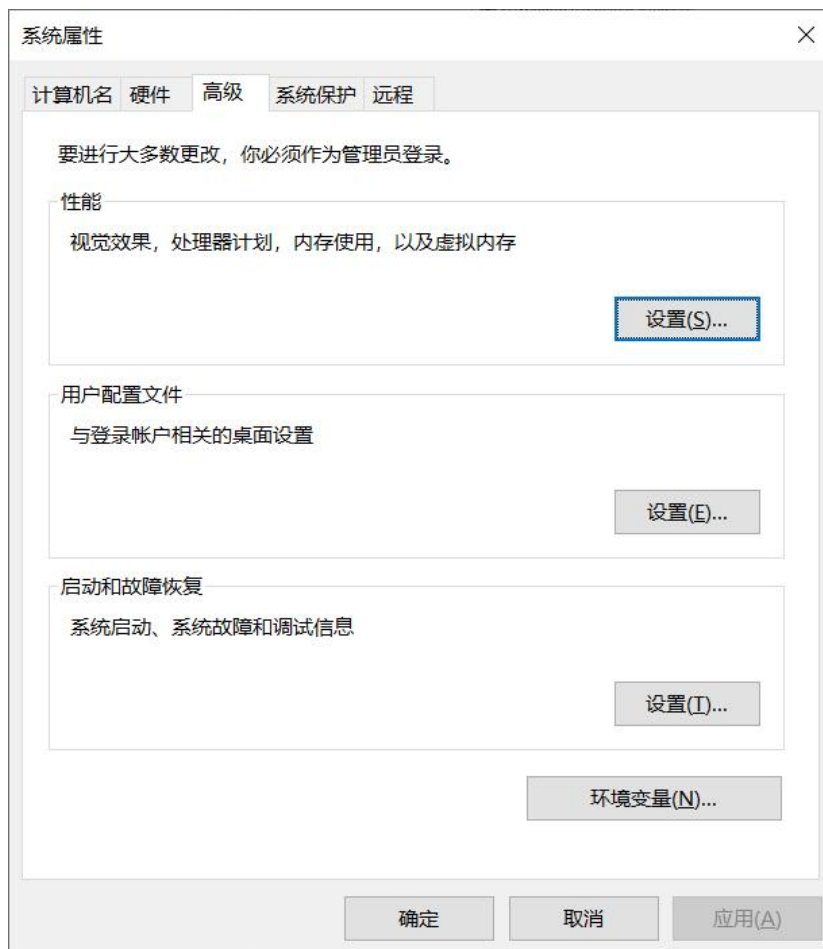
C:\Users\hnmmy>java
用法: java [-options] class [args...]
       (执行类)
或 java [-options] -jar jarfile [args...]
       (执行 jar 文件)

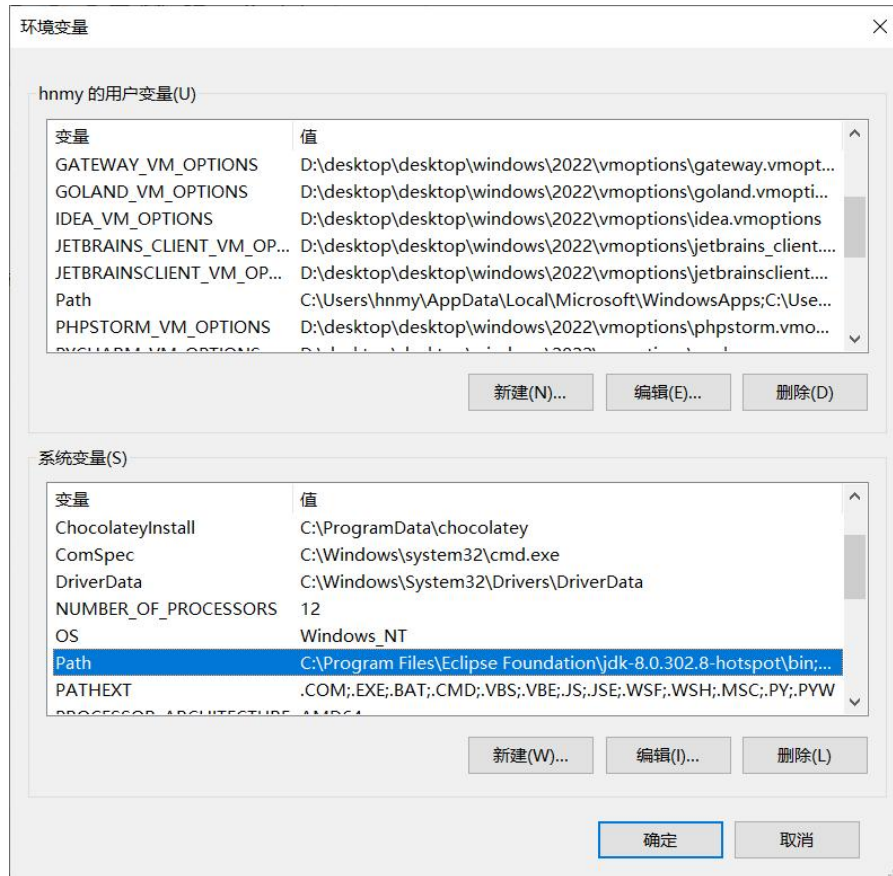
其中选项包括:
  -d32          使用 32 位数据模型 (如果可用)
  -d64          使用 64 位数据模型 (如果可用)
  -server       选择 "server" VM
                 默认 VM 是 server.

  -cp <目录和 zip/jar 文件的类搜索路径>
  -classpath <目录和 zip/jar 文件的类搜索路径>
               用 : 分隔的目录, JAR 档案
               和 ZIP 档案列表, 用于搜索类文件。

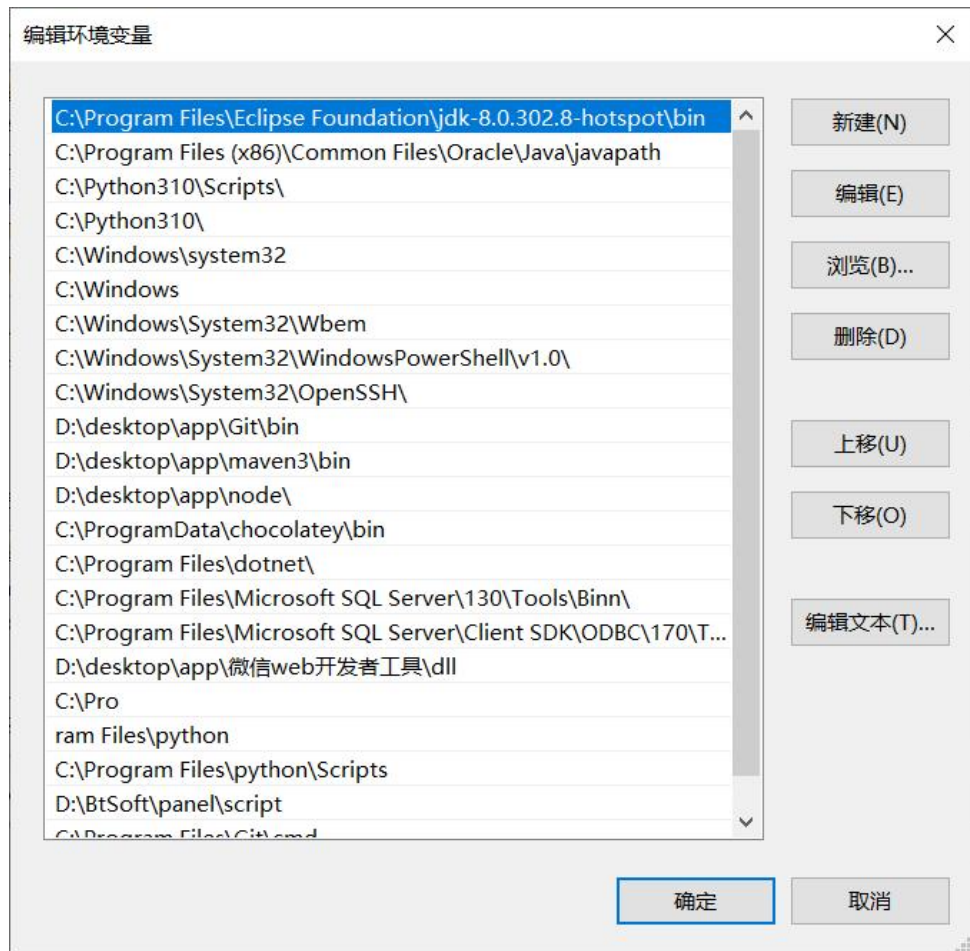
  -D<名称>=<值> 设置系统属性
  -verbose:[class|gc|jni] 启用详细输出
  -version      输出产品版本并退出
  -version:<值> 警告: 此功能已过时, 将在
                 未来发行版中删除。
                 需要指定的版本才能运行
  -showversion  输出产品版本并继续
  -jre-restrict-search | -no-jre-restrict-search
                 警告: 此功能已过时, 将在
                 未来发行版中删除。
                 在版本搜索中包括/排除用户专用 JRE
```

事实上, 我们输入 Java 命令后, 我们实际上运行的程序为环境变量里面的 Java.exe 文件, 并且将我们后面的参数传递给 Java.exe, 然后该文件运行后做出反应, 返回数据, 该数据将展示在 cmd 命令行里面, 这就是我们看到的内容。





在这里我们可以看到我们 Java 所在的目录



其实，我们也可以用命令来查找我们的 Java 在哪个地方，这个命令就是 `where java`，这样我们就可以看到我们的 Java 文件的目录的绝对目录。

```
C:\Windows\system32\cmd.exe
C:\Users\hnmmy>where java
C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe
C:\Program Files (x86)\Common Files\Oracle\Java\javapath\java.exe
C:\Users\hnmmy>
```

同理可得，我们可以查看其他软件的位置

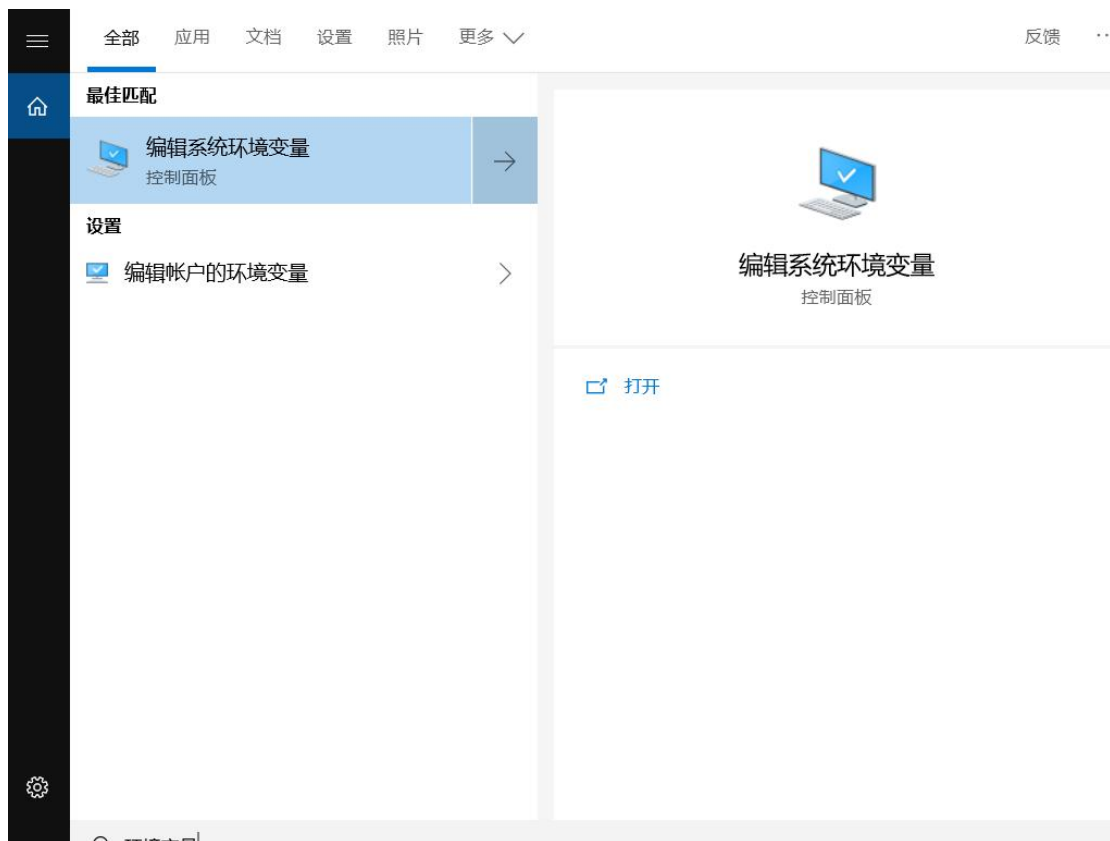


```
C:\Windows\system32\cmd.exe

C:\Users\hnmmy>where git
D:\desktop\app\Git\bin\git.exe
C:\Program Files\Git\cmd\git.exe
C:\Users\hnmmy>
```

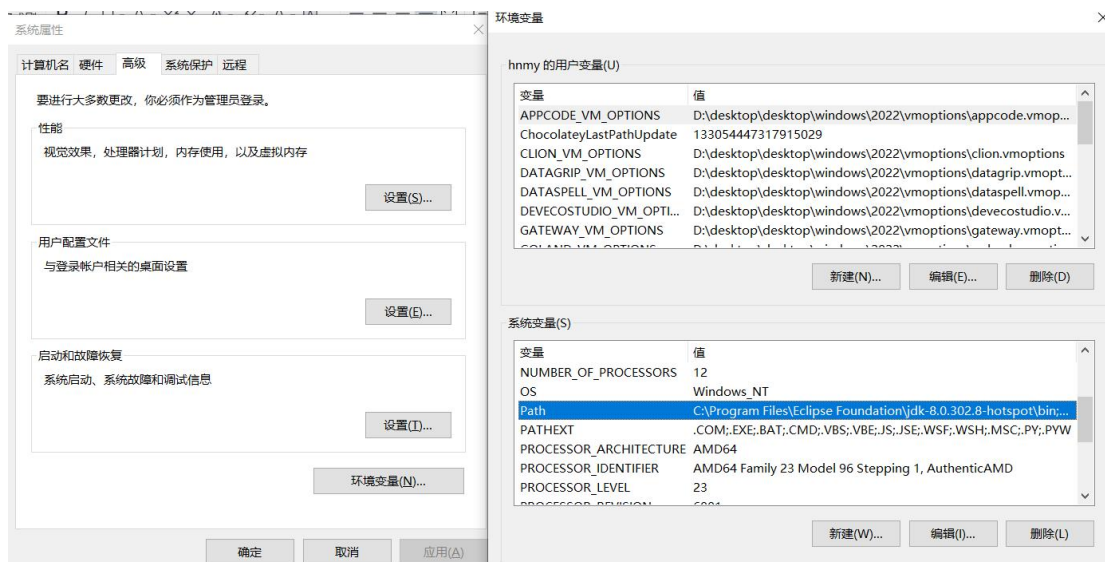
那么，如何配置环境变量呢？

我们可以直接按 **win** 键，然后输入 环境变量 4 个字来进行搜索，然后按回车键就行，就可以打开环境变量的主页面。

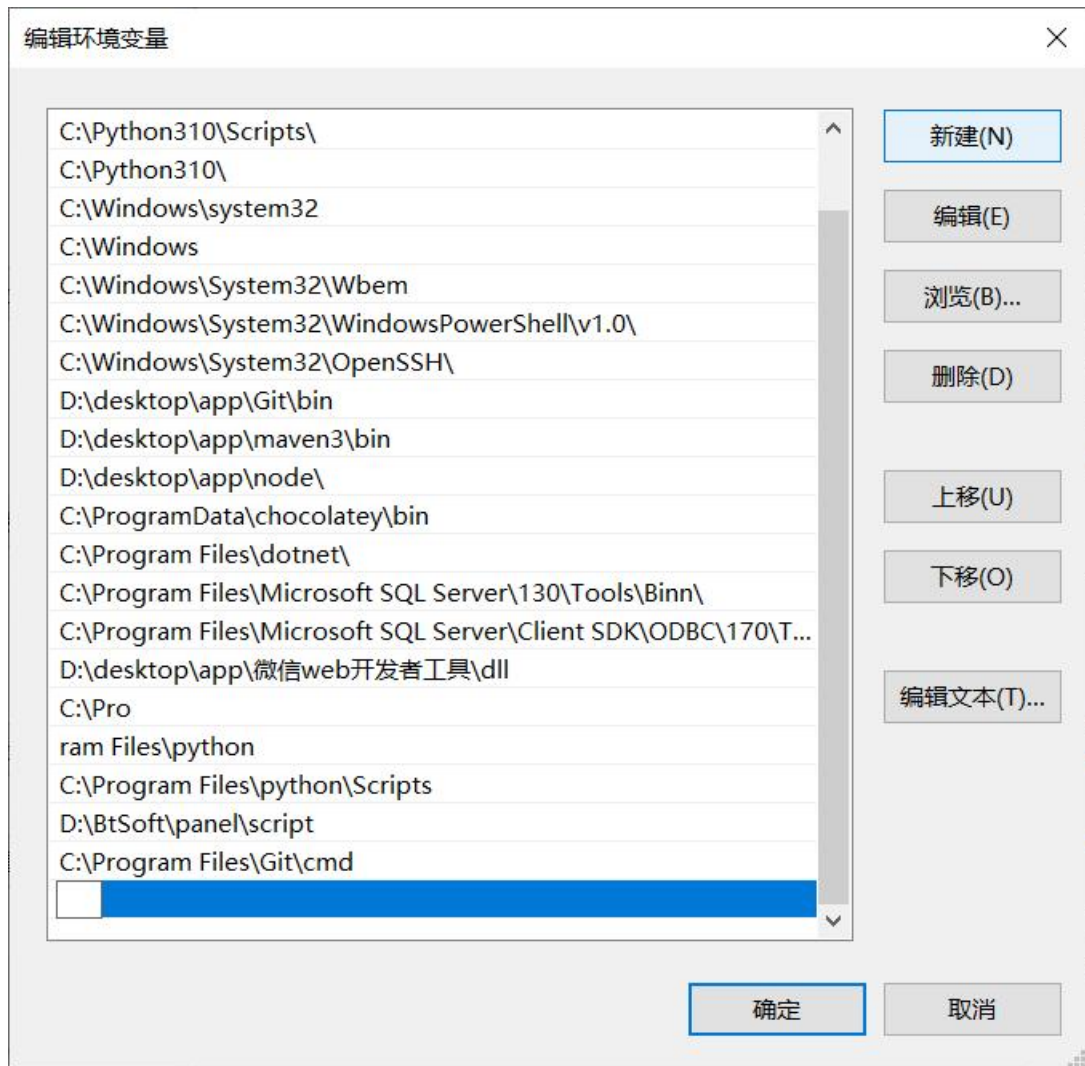




然后我们点击右下角的 环境变量 按钮，进入环境变量编辑页面，后面点击 Path 进入配置页面。



最后点击右上角新增按钮，将目录地址粘贴进入输入框即可



# 开源协议

在开源协议方面，本人认为自己的对于开源协议的解释并不算太好，所以参考前辈对于开源协议的解释，并标注内容来源，表示尊重作者著作权。

作者: pdai

开源不等于免费！为了加速我们的开发，我们会使用开源的软件和源码； 为避免商业风险，需要在使用时了解第三方如软件协议，版本，和已知 CVE 风险等；本文旨在从开源软件再发布过程使用权限的角度入手，总结各个常见开源协议的异同，方便理解。

大部分人都希望作品能够被多数人分享查阅。这样不仅提高自己业界的知名度，同时也方便了需要的人为开源做出了贡献。但是代码一旦被贴出来，任何人都可以看到并获取，之后发生的事情你就无法控制了。

所以为了公开分享你的代码，同时又让你对代码保留一定权利，在作品中声明一个许可协议是非常有必要的。有协议和没声明协议的裸代码是有非常重要区别的，一般作品当中没声明协议的默认为 Copy right 的，也就是版权保留。此种情况表明他人没有任何授权，不得复制分发修改使用等等。有了协议的声明，在未来你的维权上面会方便很多，让你的作品在分享的同时保留了自身的一些权利。

License 是软件的授权许可，里面详尽表述了你获得代码后拥有的权利，可以对别人的作品进行何种操作，何种操作又是被禁止的。

## 软件协议可分为开源和商业

- 对于商业协议，或者叫法律声明、许可协议，每个软件会有自己的一套行文，由软件作者或专门律师撰写。因为涉及到以后侵权打官司这种事情，这种商业条款的行文是非常严谨而讲究的，读起来很晦涩难懂。
- 对于开源协议，要知道开源不等于免费，也不等于没有约束。虽然相对商业协议要更加简明，但对于很多人来说还是像在看天书一样。

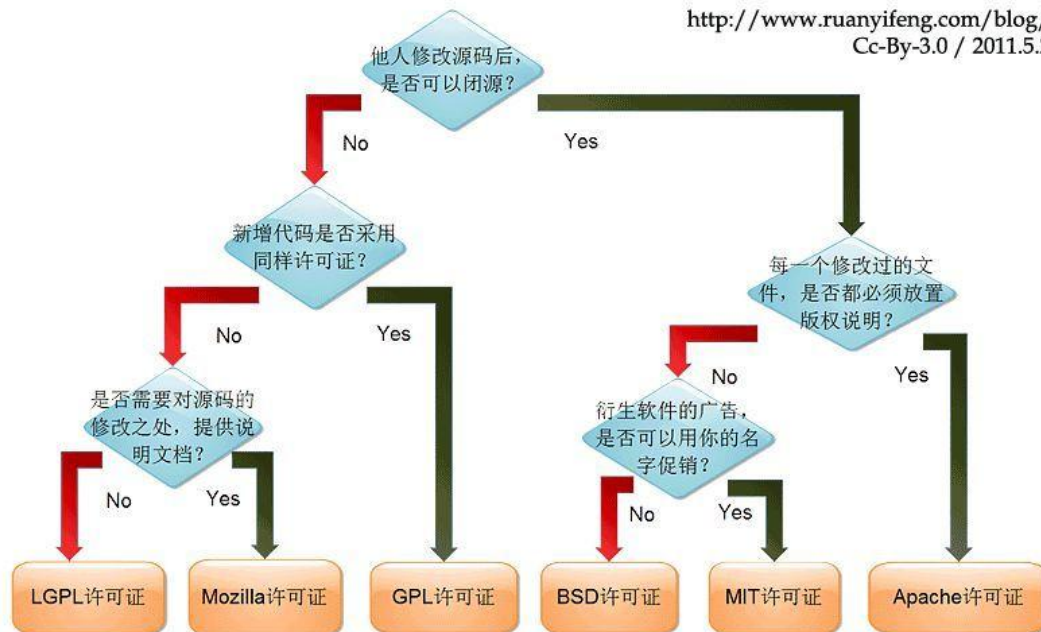
## 协议列表

首先一共有哪些公开的协议: <https://opensource.org/licenses/alphabetical>

## 常用协议

**最流行的六种**----GPL、BSD、MIT、Mozilla、Apache 和 LGPL。

乌克兰程序员 Paul Bagwell，画了一张分析图，说明应该怎么选择，只用两分钟，你就能搞清楚这六种许可证之间的最大区别。 下面是阮一峰中文翻译版本：



## 1. Apache 许可协议

Apache 许可证(Apache License)，是一个在 Apache 软件基金会发布的自由软件许可证，最初为 Apache [http 服务器](#)而撰写。Apache 许可证要求被授权者保留版权和放弃权利的申明，但它不是一个反版权的许可证。

此许可证最新版本为“版本 2”，于 2004 年 1 月发布。Apache 许可证在 Apache 社区内外被广泛使用。Apache 基金会下属所有项目都使用 Apache 许可证，许多非 Apache 基金会项目也使用了 Apache 许可证：据统计，截至 2008 年 4 月，在 sourceforge 上有超过 3000 个项目使用了 Apache 许可证。

Apache 许可协议, 2.0 版本, 授予了用户大量的权利。这些权利可以应用于拷贝权，也可以用于专利权。因为很多许可协议只能适用于拷贝权，不适用于专利权，所以这个灵活性就成了让有专利的开发者们选择许可协议时的一个显著参考因素 (要想明白两者之间的不同，请参考 [How Stuff Works](#) 上的这篇文章 )。

下面是关于 Apache 许可协议所允许的事项的详细说明：

- **权利永恒。** 一旦被授权，权利永久不失。
- **权利无疆界。** 在一个国家里被授权，形同于在所有国家被授权。例如，你在美国，但许可权最初在印度被授予，你同样可以使用这个被授权的程序。
- **授权无需付费和支付酬劳。** 你既不需要在使用之前支付任何的费用，也无需在每次使用时支付任何的费用，或者其它类似情况。
- **权利不排他。** 使用这种许可协议下的软件时，不妨碍你使用其它软件。

- **权利不可变更。** 权利一旦授予，不可剥夺。也就是说，你在使用这个软件的过程中，你无需担心这种情况：当你开发出了令人羡慕的基于这种授权软件的衍生产品时，有人突然跳出来对你说，抱歉，你将不再被允许使用这个程序。

(在这个协议里有个条款声明：如果你控告别人在这个许可协议下的产品有侵犯专利的行为，那你的授权将会自动终止，但这只是适用于有专利权的作品。只要你不搞有专利作品的诉讼，你永远无需担心这种问题。)

- **对再分发的作品还有个特殊要求，总的就是说要给予这些程序的作者和许可协议的维护者适当的名誉。**

## 2. MIT 许可协议

MIT 协议应该是在流行的开源协议中最简短的、使用最广泛的一种协议。它的条款非常的宽松，而且跟其它协议相比更自由。MIT 协议是目前最少限制的协议。

它基本上就是任何人可以对这个协议下的软件的做任何的事情，只要你能认可这个协议。这种协议最基本的条款 ( the information that it is provided without warranty, which comprises the final paragraph)如下：

特此授权，任何人都可免费获得这个软件以及相关文档 (the Software) 的拷贝，可以无限制的使用这个软件，包括无限制的权利去使用、复制、修改、合并、发布、附加从属协议，以及/或者出售软件的拷贝，同时，为了让软件的提供者有权利做到这些，下面的条件必须遵守：

上面的拷贝权声明和许可声明必须包含在所有的这个软件拷贝里和实际分署部分里。

**这也就是说：**

- 你可以随意使用，复制，修改这个软件。没有人能够阻止你在任何工程里使用它，你可以复制任意次数、以任何形式，或按你的愿望修改它。
- 你可以向外免费发放，或出售。你可以随意的分发它，没有任何限制。
- 唯一的限制是你必须接受协议条款。

## 3. BSD 许可协议

BSD 协议有很多分支，它们都代表了一种宽松的自由软件协议，相对其它协议，例如 GPL，来说，它们对软件的传播给予了更少的限制。

在这种协议的各种版本中，有两个版本格外的重要：新 BSD 协议/修订版 BSD 协议和简化 BSD 协议/FreeBSD 协议。这两类协议都实现的对 GPL 兼容的自由软件协议，而且被 Open Source Initiative 认可为开源软件协议。

新 BSD 协议(3-clause license)无任何限制的允许你以任何目的二次分发这种软件，唯一的要求是必须保留拷贝权的声明和协议里的软件权利放弃条款。这种协议还

有一个限制，未经许可不得使用这个作品的所有曾经捐助者的署名。新 BSD 协议和简化 BSD 协议的最主要的区别是后者删除了署名条款。

BSD 开源协议是一个给予使用者很大自由的协议。基本上使用者可以“为所欲为”，可以自由的使用，修改源代码，也可以将修改后的代码作为开源或者专有软件再发布。

但“为所欲为”的前提当你发布使用了 BSD 协议的代码，或则以 BSD 协议代码为基础做二次开发自己的产品时，需要满足三个条件：

- 如果再发布的产品中包含源代码，则在源代码中必须带有原来代码中的 BSD 协议。
- 如果再发布的只是二进制类库/软件，则需要在类库/软件的文档和版权声明中包含原来代码中的 BSD 协议。
- 不可以用开源代码的作者/机构名字和原来产品的名字做市场推广。
- BSD 代码鼓励代码共享，但需要尊重代码作者的著作权。BSD 由于允许使用者修改和重新发布代码，也允许使用或在 BSD 代码上开发商业软件发布和销售，因此是对商业集成很友好的协议。而很多的公司企业在选用开源产品的时候都首选 BSD 协议，因为可以完全控制这些第三方的代码，在必要的时候可以修改或者二次开发。

#### 4. GPL 许可协议

我们很熟悉的 Linux 就是采用了 GPL。GPL 协议和 BSD，Apache Licence 等鼓励代码重用的许可很不一样。GPL 的出发点是代码的开源/免费使用和引用/修改/衍生代码的开源/免费使用，但不允许修改后和衍生的代码做为闭源的商业软件发布和销售。

这也就是为什么我们能免费的各种 linux，包括商业公司的 linux 和 linux 上各种各样的由个人，组织，以及商业软件公司开发的免费软件了。

GPL 协议的主要内容是只要在一个软件中使用（“使用”指类库引用，修改后的代码或者衍生代码）GPL 协议的产品，则该软件产品必须也采用 GPL 协议，既必须也是开源和免费。这就是所谓的“传染性”。GPL 协议的产品作为一个单独的产品使用没有任何问题，还可以享受免费的优势。

由于 GPL 严格要求使用了 GPL 类库的软件产品必须使用 GPL 协议，对于使用 GPL 协议的开源代码，商业软件或者对代码有保密要求的部门就不适合集成/采用作为类库和二次开发的基础。

其它细节如再发布的时候需要伴随 GPL 协议等和 BSD/Apache 等类似。

#### 5. LGPL 许可协议

LGPL 是 GPL 的一个为主要为类库使用设计的开源协议。和 GPL 要求任何使用/修改/衍生之 GPL 类库的软件必须采用 GPL 协议不同。LGPL 允许商业软件通过类库引用 (link) 方式使用 LGPL 类库而不需要开源商业软件的代码。这使得采用 LGPL 协议的开源代码可以被商业软件作为类库引用并 发布和销售。

但是如果修改 LGPL 协议的代码或者衍生，则所有修改的代码，涉及修改部分的额外代码和衍生的代码都必须采用 LGPL 协议。因此 LGPL 协议的开源 代码很适合作为第三方类库被商业软件引用，但不适合希望以 LGPL 协议代码为基础，通过修改和衍生的方式做二次开发的商业软件采用。

GPL/LGPL 都保障原作者的知识产权，避免有人利用开源代码复制并开发类似的产品。

## 6. MPL 许可协议

MPL 是 The Mozilla Public License 的简写，是 1998 年初 Netscape 的 Mozilla 小组为其开源软件项目设计的软件许可证。

MPL 许可证出现的最重要原因就是，Netscape 公司认为 GPL 许可证没有很好地平衡开发者对源代码的需求和他们利用源代码获得的利益。同著名的 GPL 许可证和 BSD 许可证相比，MPL 在许多权利与义务的约定方面与它们相同（因为都是符合 OSI 认定的开源软件许可证）。

但是，相比而言 MPL 还有以下几个显著的不同之处：

- MPL 虽然要求对于经 MPL 许可证发布的源代码的修改也要以 MPL 许可证的方式再许可出来，以保证其他人可以在 MPL 的条款下共享源代码。但是，在 MPL 许可证中对“发布”的定义是“以源代码方式发布的文件”，这就意味着 MPL 允许一个企业在自己已有的源代码库上加一个接口，除了接口程序的源代码以 MPL 许可证的形式对外许可外，源代码库中的源代码就可以不用 MPL 许可证的方式强制对外许可。这些，就为借鉴别人的源代码用做自己商业软件开发的行为留了一个豁口。

- MPL 许可证第三条第 7 款中允许被许可人将经过 MPL 许可证获得的源代码同自己其他类型的代码混合得到自己的软件程序。

对软件专利的态度，MPL 许可证不像 GPL 许可证那样明确表示反对软件专利，但是却明确要求源代码的提供者不能提供已经受专利保护的源代码（除非他本人是专利权人，并书面向公众免费许可这些源代码），也不能在将这些源代码以开放源代码许可证形式许可后再去申请与这些源代码有关的专利。

### 对源代码的定义

- 而在 MPL（1.1 版本）许可证中，对源代码的定义是：“源代码指的是对作品进行修改最优先择取的形式，它包括：所有模块的所有源程序，加上有关的接口的



定义，加上控制可执行作品的安装和编译的‘原本’（原文为‘Script’），或者不是与初始源代码显著不同的源代码就是被源代码贡献者选择的从公共领域可以得到的程序代码。”

- MPL 许可证第 3 条有专门的一款是关于对源代码修改进行描述的规定，就是要求所有再发布者都得有一个专门的文件就对源代码程序修改的时间和修改的方式有描述。

## 小结

### GPL 协议、LGPL 协议与 BSD 协议的法律区别。

简而言之，GPL 协议就是一个开放源代码协议，软件的初始开发者使用了 GPL 协议并公开软件的源程序后，后续使用该软件源程序开发软件者亦应当根据 GPL 协议把自己编写的源程序进行公开。GPL 协议要求的关键在于开放源程序，但并不排斥软件作者向用户收费。

虽然如此，很多大公司对 GPL 协议还是又爱又恨，爱的是这个协议项下的软件历经众多程序员千锤百炼的修改，已经非常成熟完善，恨的是必须开放自己后续的源程序，导致竞争对手也可以根据自己修改的源程序开发竞争产品。

正因大公司对 GPL 协议在商业上存在顾虑，因此，另两种协议被采用的更多，第一种是 LGPL（亦称 GPL V2）协议，可以翻译为更宽松的 GPL 协议。与 GPL 协议的区别为，后者如果只是对 LGPL 软件的程序库的程序进行调用而不是包含其源代码时，相关的源程序无需开源。

调用和包含的区别类似在互联网网页上对他人网页内容的引用：如果把他人的内容全部或部分复制到自己的网页上，就类似包含，如果只是贴一个他人网页的网址链接而不引用内容，就类似调用。有了这个协议，很多大公司就可以把很多自己后续开发内容的源程序隐藏起来。

第二种是 BSD 协议（类似的还有 MIT 协议）。BSD 协议鼓励软件的作者公开自己后续开发的源代码，但不强求。在 BSD 协议项下开发的软件，原始的源程序是开放源代码的，但使用者修改以后，可以自行选择发布源程序或者二进制程序（即目标程序），当然，使用者有义务把自己原来使用的源程序与 BSD 协议在软件对外发布时一并发布。因为比较灵活，所以 BSD 深受大公司的欢迎。

## 参考文章

常用开源软件许可协议简介

<https://www.cnblogs.com/yefengmeander/p/4173673.html> 开源协议在再发布过程中的授权总结 <http://blog.csdn.net/bqlyj/article/details/60322655> 如何为你的代码选择一个开源协议 - 刘哇勇的部落格

[http://www.cnblogs.com/Wayou/p/how\\_to\\_choose\\_a\\_license.html](http://www.cnblogs.com/Wayou/p/how_to_choose_a_license.html) 重要开源协议的比较(BSD,Apache,GPL,LGPL,MIT) – TechBirds 在路上

[http://blog.csdn.net/techbirds\\_bao/article/details/8785413](http://blog.csdn.net/techbirds_bao/article/details/8785413) 游云庭律师的回答 - 主流开源协议之间有何异同? - 知乎

<https://www.zhihu.com/question/19568896/answer/12284793> Choose a license

<https://choosealicense.com/licenses/> Understanding Copyright And License

<https://www.smashingmagazine.com/2011/06/understanding-copyright-and-lic...>

开源协议 - 百度百科

<http://baike.baidu.com/item/%E5%BC%80%E6%BA%90%E5%8D%8F%E8%AE%AE> 如何选择开源许可证?

[http://www.ruanyifeng.com/blog/2011/05/how\\_to\\_choose\\_free\\_software\\_licenses](http://www.ruanyifeng.com/blog/2011/05/how_to_choose_free_software_licenses)

更多文章请参考 <http://www.cnblogs.com/pengdai/>