VsCode中使用Emmet神器快速编写HTML代码

一、Emmet简述#

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

> 请注意在VsCode新版本中按Tab不再默认启用Emmet展开缩写!需要在首选项配置中将emmet.triggerExpansionOnTab设置为true值!

语法基本规则如下:

Copy

```
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。
```

> 本文仅介绍了在Html使用Emmet, 如果想Css缩写的语法请参考这里 https://docs.emmet.io/css-abbreviations/

二、基础用法#

- **元素(Elements)** 您可以使用元素的名称，如div或p来生成HTML标签。Emmet没有一组可用的标签名称，可以写任何单词并将其转换为标签。也就是只要知道元素的缩写,Emmet会自动转换成对应标签. 形如:

Copy

```
div => <div> div>
foo => <foo> foo>
html:5 => 将生成html5标准的包含body为空基本dom
html:xt => 生成XHTML过渡文档类型,DOCTYPE为XHTML
html:4s => 生成HTML4严格文档类型,DOCTYPE为HTML 4.01
a:mail          => <a href="mailto:">a>
a:link          => <a href="http://">a>
base            => <base href="">
br              => <br>
link            => <link rel="stylesheet" href="">
script:src      => <script src="">script>
form:get        => <form action="" method="get">form>
label           => <label for="">label>
input           => <input type="text">
inp             => <input type="text" name="" id="">
input:hidden    => <input type="hidden" name="">  input:h亦可
input:email     => <input type="email" name="" id="">
input:password  => <input type="password" name="" id="">
input:checkbox  => <input type="checkbox" name="" id="">
input:radio     => <input type="radio" name="" id="">
select          => <select name="" id="">select>
option          => <option value="">option>
bq              => <blockquote>blockquote>
btn             => <button>button>
btn:s           => <button type="submit">button>
btn:r           => <button type="reset">button>
```

- **文本操作符(Text)** 如果想在生成元素的同时添加文本内容可以使用{}

Copy

```
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>
```

- **属性操作符(Attribute operators)** 属性运算符用于修改输出元素的属性.
    - Id和Class (elem#id and elem.class)

```
Copy
div.test  => <div class="test">div>
div#pageId => <div id="pageId">div>
```

隐式标签则会自动联想生成对应元素,根据配置规则不同生成的结果也是不同的.

```
Copy
.class
=>
<div class>div>
em>.class
=>
<em><span class>span>em>
table>.row>.col
=>
<table>
    <tr class="row">
        <td class="col">td>
    tr>
table>
```

绑定多个类名用.符号连续起来即可

```
Copy
div.test1.test2.test3
=>
<div class="test1 test2 test3">div>
```

- 自定义属性使用 [attr1=" attr2="]

```
Copy
a[href='#' data-title='customer' target='_blank']
=>
<a href="#" data-title="customer" target="_blank">a>
```

- **嵌套操作符(Nesting operators)** 嵌套操作符用于将缩写元素放置在生成的树中,是否应放置在上下文元素的内部或附近.
  - 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

```
Copy
```

```
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.

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

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

- **分组操作符(Grouping)** 分组使用()来实现缩写的分离.比如这个例子,如果不加括号那么a将作为span的子级元素生成.加上括号a将于()内的元素同级.

```
Copy
div>(ul>li+span)>a
=>
<div>
    <ul>
        <li>li>
        <span>span>
    ul>
    <a href="">a>
div>
```

- **乘法(Multiplication)** 使用*N即可自动生成重复项.N是一个正整数.在使用时请注意*N所在位置,位置不同生成的结果不同.

Copy

```
ul>li*3
=>
<ul>
    <li>li>
    <li>li>
    <li>li>
ul>
```

- **自动计数(numbering)** 这个功能挺方便的对于生成重复项时增加一个序号,只需要加上$符号即可.

Copy

```
ul>li.item${item number:$}*3
<ul>
    <li class="item1">item number:1li>
    <li class="item2">item number:2li>
    <li class="item3">item number:3li>
ul>
```

如果生成两位数则使用两个连续的$$,更多位数以此类推... 使用@修饰符，可以更改编号方向（升序或降序）和基数（例如起始值）.注意这个操作符在$之后添加 @-表示降序,@+表示升序,默认使用升序. @N可以改变起始值.需要注意的是如果配合升降序使用的话N是放到+-符后.

Copy

```
ul>li.item$@-*3
=>
<ul>
    <li class="item3"></li>
    <li class="item2"></li>
    <li class="item1"></li>
</ul>
```

```
ul>li.item$@-10*3 => <ul> <li class="item12"></li> <li
class="item11"></li> <li class="item10"></li> </ul>
```

上述的操作是可以搭配使用进而得出酷炫的效果,使用时请注意空格的问题,缩写代码不要有空格否则是不会进行转换的. 另外如果你的编辑器中已经有了一些html智能提示代码段,比如我的VsCode还装了HTML Snippets插件,这个与Emmet语法有部分冲突,使用Tab键时会优先使用插件的代码提示,建议禁用. 组合起来看看效果:

Copy

```
table.table-row[role='table']>(thead>tr>td{item $@120}*5)+(tbody>tr>
(td.item$$@-)lorem10*5)
```

这段目的在于生成一个类名为table-row,且自定义了属性role的table标签,内部包含了thead与tbody,分别生成5个td. thead中td的内容是item加上自增序号,自增序号基数从120开始. tbody中td拥有一个名为item加降序自增符号类名,且每个td内容随机填充10个单词.

Copy

```
class="table-row" role="table">
    <thead>
        <tr>
            <td>item 120td><td>item 121td><td>item 122td><td>item 123td><td>item
124td>tr>thead><tbody>
        <tr>
            <td>
                <td class="item05">Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Odit, assumenda.td>td><td><td class="item04">Magnam possimus
molestias ipsum animi rem placeat, ut obcaecati laudantium.td>td><td><td
class="item03">Consequuntur, labore ad optio cupiditate iusto dolores fugit
quidem officiis.td>td><td><td class="item02">Veniam, explicabo consequuntur
blanditiis at dicta fuga ratione eos beatae.td>td><td><td class="item01">Fuga
voluptatum illo quis ducimus ad eveniet non. Saepe, eveniet.td>td>tr>tbody>
```

再来个配合嵌套元素和计数的大栗子.

Copy

```
div.nav>(nav#navbar>(ul>li>(a[href="/xxx/product/$" data-
index=$]>lorem4)*5))+div.btn[type='button']>span{--}^^div#main
```

这个有点复杂,看下效果:

Copy

```
class="nav">
    <nav id="navbar">
        <ul>
            <li>
                <a href="/xxx/product/1" data-index="1">Lorem ipsum dolor sit.a>
                <a href="/xxx/product/2" data-index="2">Dolor vel, quia quas.a>
                <a href="/xxx/product/3" data-index="3">Qui hic, corrupti eum!a>
                <a href="/xxx/product/4" data-index="4">Necessitatibus
perspiciatis, corrupti. Praesentium!a>
                <a href="/xxx/product/5" data-index="5">Nostrum quos, voluptate.
Velit!a>
            li>
        ul>
    nav>
    <div class="btn" type="button"><span>--span>div>

<div id="main">div>
```

三、进阶高级用法#

- **模拟文本/随机文本** 在开发时经常要填充一些文本内容占位,Emmet内置了Lorem Ipsum功能来实现.loremN或者lipsumN,N表示生成的单词数,正整数.可以不填.

Copy

```
lorem
=> Lorem ipsum dolor sit amet, consectetur adipisicing elit. Suscipit quia
commodi vero sint omnis fugiat excepturi reiciendis necessitatibus totam
asperiores, delectus saepe nulla consequuntur nostrum! Saepe suscipit recusandae
repellendus assumenda.

p>lorem4
=>
<p>Lorem ipsum dolor sit.p>

(p>lorem4)*3
=>
<p>Lorem ipsum dolor sit.p>
<p>Labore aperiam, consequuntur architecto.p>
<p>Quidem nisi, cum odio!p>
```

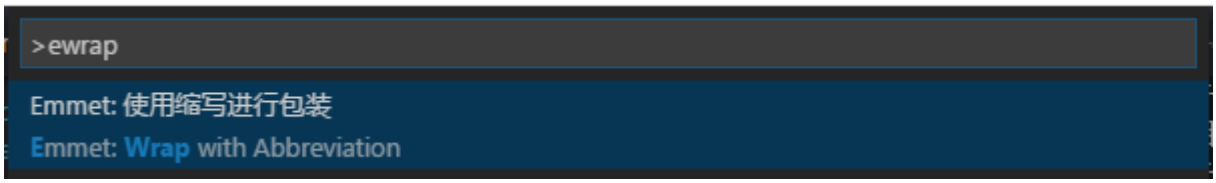
- **包装文本** 听起来可能有点绕,通俗点解释就是把一段指定的文本包装成我们想要的结构.注意这个功能需要编辑器的支持,举个大栗子: 比如PM给了这样一段文本

Copy

```
首页
产品介绍
相关案例
```

```
关于我们
联系我们
而我们预期的效果是这样
<nav>
    <ul>
        <li>首页li>
        <li>产品介绍li>
        <li>相关案例li>
        <li>关于我们li>
        <li>联系我们li>
    ul>
nav>
```

1. 选中文本,按下`ctrl+shift+p`打开命令窗口输入ewrap
2. 选择`Emmet:使用缩写进行包装(Wrap with Abbreviation)`选项



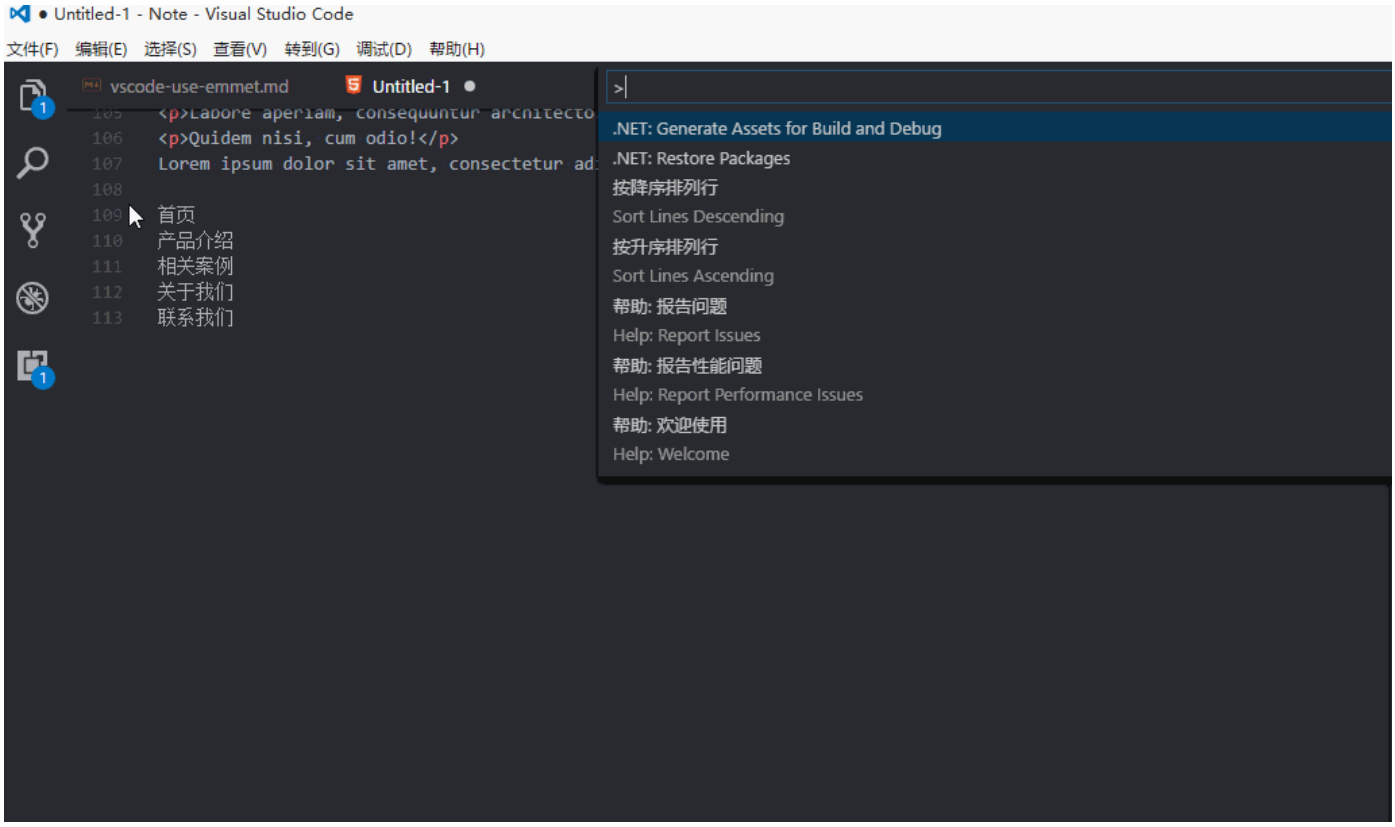
3. 输入缩写字符`nav>ul>li*`按下回车键即可看到效果. 当然也可以在菜单=>编辑=>Emmet(M)..然后输入.

> 这里需要的注意的地方是输入的缩写代码中*所在位置不同得到的效果也是不同的.

另外如果给的文本带有序号的情况,我们也是可以通过缩写来处理,而不是手动删除,主要用的是`|t`来处理. 比如:

```
Copy
    1.首页
    2.产品介绍
    3.相关案例
    4.关于我们
    5.联系我们
    输入包装字符命令
    nav>ul>li*|t
    即可看到生成的html中自动去掉了序号
```

针对上边说的几种情况来演示一下.



利用好Emmet来快速编写html代码,能提升工作效率不用在一个个敲闭合标签,简直是我等偷懒党神器.赶快去耍耍吧.