打造VS Code(支持Markdown/UML/脑图/代码模板)

原创 丿寒风 已于 2023-12-28 09:55:14 修改 阅读量2.6w 收藏 111 点赞数 23
分类专栏： 开发工具 文章标签： markdown UML 思维导图 脑图

GitCode 开源社区 文章已被社区收录

开发工具 专栏收录该内容 0 订阅 19 篇文章

文章目录

Visual Studio Code是一个 文本编辑器，除了写代码和编辑普通文本外，借助插件还可以实现一些常用的图形化功能。下面介绍下，在 VS Code 中支持功能：

1. Markdown
2. 思维导图
3. UML

Markdown

Markdown 是一种轻量级的文本标记语言，使用简单符号即可编辑出带有丰富格式的内容，让编辑者专注于内容本身，而不用花费太多额外精力去调整文本简书、Github等网站都支持 Markdown 格式。语法参考

要在 VS Code 中支持 Markdown 格式，推荐安装以下插件：

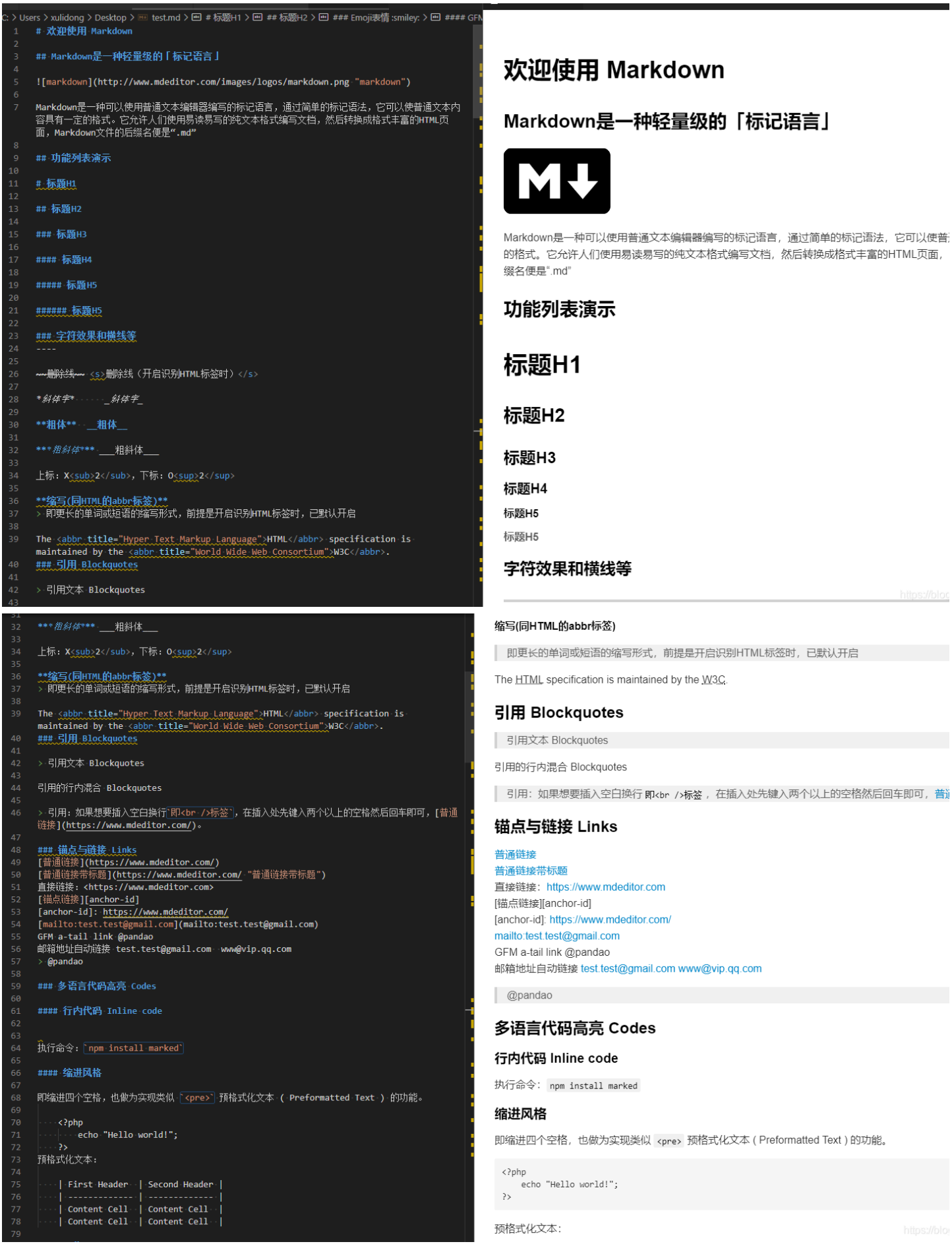
- Markdown All in One

文件后缀名保存为 `.md`，会自动预览，也可以用快捷键 `ctrl+shift + v` 显示预览。

觉得还不错

丿寒风 关注 23

以下内容是在网上找的一篇讲 Markdown 格式的示例文档，文档本身用也的 Markdown 格式，在 VS Code 中的显示效果如下：



```
C: > Users > xulidong > Desktop > test.md > # 标题H1 > ## 标题H2 > ### Emoji表情 :smiley: > #### GFM
# 欢迎使用 Markdown

## Markdown是一种轻量级的「标记语言」

![markdown](http://www.mdeditor.com/images/logos/markdown.png "markdown")

Markdown是一种可以使用普通文本编辑器编写的标记语言，通过简单的标记语法，它可以使普通文本内容具有一定的格式。它允许人们使用易读易写的纯文本格式编写文档，然后转换成格式丰富的HTML页面，Markdown文件的后缀名便是“.md”

## 功能列表演示

# 标题H1

## 标题H2

### 标题H3

#### 标题H4

##### 标题H5

###### 标题H5

### 字符效果和横线等
---

~~删除线~~ <s>删除线（开启识别HTML标签时）</s>

*斜体字*      _斜体字_

**粗体**  __粗体__

***粗斜体*** ___粗斜体___

上标：X<sub>2</sub>，下标：O<sup>2</sup>

**缩写(同HTML的abbr标签)**
> 即更长的单词或短语的缩写形式，前提是开启识别HTML标签时，已默认开启

The <abbr title="Hyper Text Markup Language">HTML</abbr> specification is maintained by the <abbr title="World Wide Web Consortium">W3C</abbr>.
### 引用 Blockquotes

> 引用文本 Blockquotes

引用的行内混合 Blockquotes

> 引用：如果想要插入空白换行`即<br />标签`，在插入处先键入两个以上的空格然后回车即可，[普通链接](https://www.mdeditor.com/)。

### 锚点与链接 Links
[普通链接](https://www.mdeditor.com/)
[普通链接带标题](https://www.mdeditor.com/ "普通链接带标题")
直接链接：<https://www.mdeditor.com>
[锚点链接][anchor-id]
[anchor-id]: https://www.mdeditor.com/
[mailto:test.test@gmail.com](mailto:test.test@gmail.com)
GFM a-tail link @pandao
邮箱地址自动链接 test.test@gmail.com www@vip.qq.com
> @pandao

### 多语言代码高亮 Codes

#### 行内代码 Inline code

执行命令：`npm install marked`

#### 缩进风格

即缩进四个空格，也做为实现类似 `<pre>` 预格式化文本 ( Preformatted Text ) 的功能。

    <?php
        echo "Hello world!";
    ?>
预格式化文本：

    | First Header  | Second Header |
    | ------------- | ------------- |
    | Content Cell  | Content Cell  |
    | Content Cell  | Content Cell  |
```

欢迎使用 Markdown

Markdown是一种轻量级的「标记语言」

Markdown是一种可以使用普通文本编辑器编写的标记语言，通过简单的标记语法，它可以使普通文本内容具有一定的格式。它允许人们使用易读易写的纯文本格式编写文档，然后转换成格式丰富的HTML页面，Markdown文件的后缀名便是“.md”

功能列表演示

标题H1

标题H2

标题H3

标题H4

标题H5

标题H5

字符效果和横线等

~~删除线~~ 删除线（开启识别HTML标签时）

斜体字 *斜体字*

粗体 **粗体**

粗斜体 ***粗斜体***

上标：X2，下标：O2

缩写(同HTML的abbr标签)

> 即更长的单词或短语的缩写形式，前提是开启识别HTML标签时，已默认开启

The HTML specification is maintained by the W3C.

引用 Blockquotes

> 引用文本 Blockquotes

引用的行内混合 Blockquotes

> 引用：如果想要插入空白换行 `即<br />标签` ，在插入处先键入两个以上的空格然后回车即可，普通

锚点与链接 Links

普通链接
普通链接带标题
直接链接：https://www.mdeditor.com
[锚点链接][anchor-id]
[anchor-id]: https://www.mdeditor.com/
mailto:test.test@gmail.com
GFM a-tail link @pandao
邮箱地址自动链接 test.test@gmail.com www@vip.qq.com

> @pandao

多语言代码高亮 Codes

行内代码 Inline code

执行命令：`npm install marked`

缩进风格

即缩进四个空格，也做为实现类似 `<pre>` 预格式化文本 (Preformatted Text) 的功能。

```
<?php
    echo "Hello world!";
?>
```

预格式化文本：

觉得还不错

丿寒风 关注 23

```
#### 缩进风格

即缩进四个空格，也做为实现类似 `<pre>` 预格式化文本 ( Preformatted Text ) 的功能。

    <?php
        echo "Hello world!";
    ?>
预格式化文本：

    | First Header  | Second Header |
    | ------------- | ------------- |
    | Content Cell  | Content Cell  |
    | Content Cell  | Content Cell  |

#### JS代码
```javascript
function test() {
    console.log("Hello world!");
}
```

HTML 代码 HTML codes
```html
<!DOCTYPE html>
<html>
    <head>
        <mate charest="utf-8" />
        <meta name="keywords" content="Editor.md, Markdown, Editor" />
        <title>Hello world!</title>
        <style type="text/css">
            body{font-size:14px;color:#444;font-family: "Microsoft Yahei", Tahoma,
            "Hiragino Sans GB", Arial;background:#fff;}
            ul{list-style: none;}
            img{border:none;vertical-align: middle;}
        </style>
    </head>
    <body>
        <h1 class="text-xxl">Hello world!</h1>
        <p class="text-green">Plain text</p>
    </body>
</html>
```
图片 Images

图片加链接 (Image + Link):
```

缩进风格

即缩进四个空格，也做为实现类似 `<pre>` 预格式化文本 (Preformatted Text) 的功能。

```
<?php
    echo "Hello world!";
?>
```

预格式化文本：

```
| First Header  | Second Header |
| ------------- | ------------- |
| Content Cell  | Content Cell  |
| Content Cell  | Content Cell  |
```

JS代码

```
function test() {
    console.log("Hello world!");
}
```

HTML 代码 HTML codes

```
<!DOCTYPE html>
<html>
    <head>
        <mate charest="utf-8" />
        <meta name="keywords" content="Editor.md, Markdown, Editor" />
        <title>Hello world!</title>
        <style type="text/css">
            body{font-size:14px;color:#444;font-family: "Microsoft Yahei", Tahoma, "Hira
            ul{list-style: none;}
            img{border:none;vertical-align: middle;}
        </style>
    </head>
    <body>
        <h1 class="text-xxl">Hello world!</h1>
        <p class="text-green">Plain text</p>
    </body>
</html>
```

图片 Images

图片加链接 (Image + Link):

> Follow your heart.

https://blog

```
[![](https://www.mdeditor.com/images/logos/markdown.png)](https://www.mdeditor.com/images/logos/markdown.png "markdown")

> Follow your heart.

---
### 列表 Lists

#### 无序列表（减号）Unordered Lists (-)

- 列表一
- 列表二
- 列表三

#### 无序列表（星号）Unordered Lists (*)

* 列表一
* 列表二
* 列表三

#### 无序列表（加号和嵌套）Unordered Lists (+)
+ 列表一
+ 列表二
    + 列表二-1
    + 列表二-2
    + 列表二-3
+ 列表三
    * 列表一
    * 列表二
    * 列表三

#### 有序列表 Ordered Lists (-)

1. 第一行
2. 第二行
3. 第三行

#### GFM task list

- [x] GFM task list 1
- [x] GFM task list 2
- [ ] GFM task list 3
    - [ ] GFM task list 3-1
    - [ ] GFM task list 3-2
    - [ ] GFM task list 3-3
- [ ] GFM task list 4
    - [ ] GFM task list 4-1
    - [ ] GFM task list 4-2
```

无序列表（减号）Unordered Lists (-)

- 列表一
- 列表二
- 列表三

无序列表（星号）Unordered Lists (*)

- 列表一
- 列表二
- 列表三

无序列表（加号和嵌套）Unordered Lists (+)

- 列表一
- 列表二
  - 列表二-1
  - 列表二-2
  - 列表二-3
- 列表三
  - 列表一
  - 列表二
  - 列表三

有序列表 Ordered Lists (-)

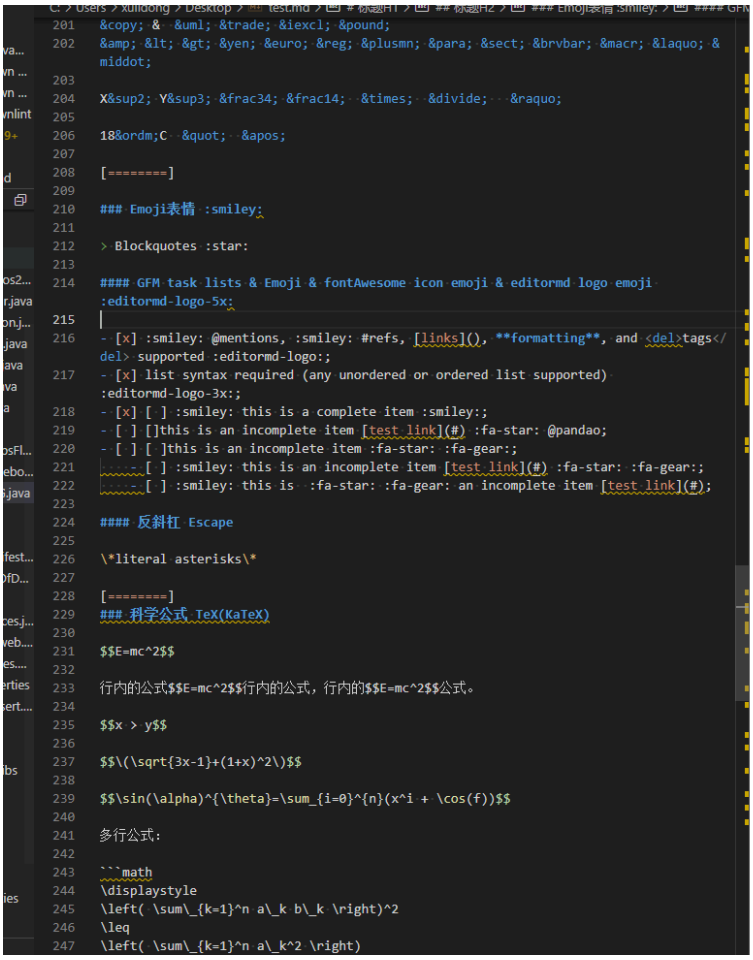
1. 第一行
2. 第二行
3. 第三行

GFM task list

- [x] GFM task list 1
- [x] GFM task list 2
- [ ] GFM task list 3
  - [ ] GFM task list 3-1
  - [ ] GFM task list 3-2
  - [ ] GFM task list 3-3
- [ ] GFM task list 4
  - [ ] GFM task list 4-1
  - [ ] GFM task list 4-2

觉得还不错

丿寒风 关注 23



```
&copy; &amp; &uml; &trade; &iexcl; &pound;
&amp; &lt; &gt; &yen; &euro; &reg; &plusmn; &para; &sect; &brvbar; &macr; &laquo; &middot;

X&sup2; Y&sup3; &frac34; &frac14; &times; &divide; &raquo;

18&ordm;C &quot; &apos;

[========]

### Emoji表情 :smiley:

> Blockquotes :star:

#### GFM task lists & Emoji & fontAwesome icon emoji & editormd logo emoji :editormd-logo-5x:

- [x] :smiley: @mentions, :smiley: #refs, [links](), **formatting**, and <del>tags</del> supported :editormd-logo:;
- [x] list syntax required (any unordered or ordered list supported) :editormd-logo-3x:;
- [x] [ ] :smiley: this is a complete item :smiley:;
- [ ] []this is an incomplete item [test link](#) :fa-star: @pandao;
- [ ] [ ]this is an incomplete item :fa-star: :fa-gear:;
    - [ ] :smiley: this is an incomplete item [test link](#) :fa-star: :fa-gear:;
    - [ ] :smiley: this is :fa-star: :fa-gear: an incomplete item [test link](#);

#### 反斜杠 Escape

\*literal asterisks\*

[========]
### 科学公式 TeX(KaTeX)

$$E=mc^2$$

行内的公式$$E=mc^2$$行内的公式，行内的$$E=mc^2$$公式。

$$x > y$$

$$\(\sqrt{3x-1}+(1+x)^2\)$$

$$\sin(\alpha)^{\theta}=\sum_{i=0}^{n}(x^i + \cos(f))$$

多行公式:

```math
\displaystyle
\left( \sum_{k=1}^n a_k b_k \right)^2
\leq
\left( \sum_{k=1}^n a_k^2 \right)
```
```

Emoji表情 😃

> Blockquotes ☆

GFM task lists & Emoji & fontAwesome icon emoji & editormd logo emoji logo-5x:

- [x] 😃 @mentions, 😃 #refs, links, **formatting**, and ~~tags~~ supported :editormd-logo:;
- [x] list syntax required (any unordered or ordered list supported) :editormd-logo-3x:;
- [x] [ ] 😃 this is a complete item 😃;
- [ ] []this is an incomplete item test link ★ @pandao;
- [ ] [ ]this is an incomplete item ★ ✪;
    - [ ] 😃 this is an incomplete item test link ★ ✪;
    - [ ] 😃 this is ★ ✪ an incomplete item test link;

反斜杠 Escape

literal asterisks

[========]

科学公式 TeX(KaTeX)

$$E = mc^2$$

行内的公式

$$E = mc^2$$

行内的公式，行内的

$$E = mc^2$$

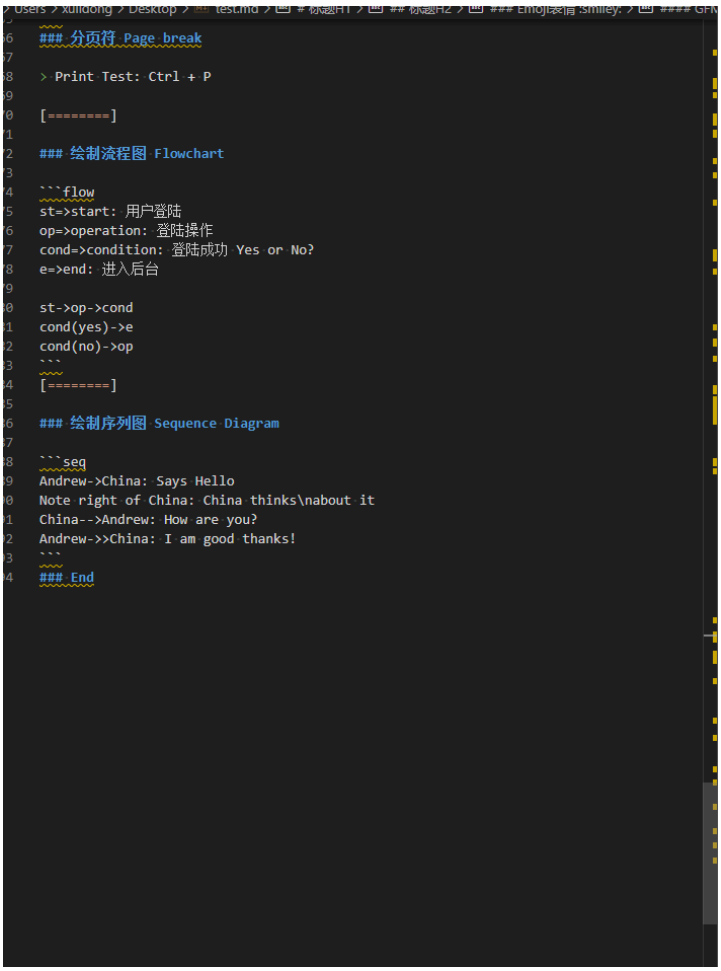
公式。

$$x > y$$

ParseError: KaTeX parse error: Can't use function '\(' in math mode at position 1: \(\sqrt{3x-1}+(1

$$\sin(\alpha)^{\theta} = \sum_{i=0}^{n}(x^i + \cos(f))$$

多行公式:



```
### 分页符 Page break

> Print Test: Ctrl + P

[========]

### 绘制流程图 Flowchart

```flow
st=>start: 用户登陆
op=>operation: 登陆操作
cond=>condition: 登陆成功 Yes or No?
e=>end: 进入后台

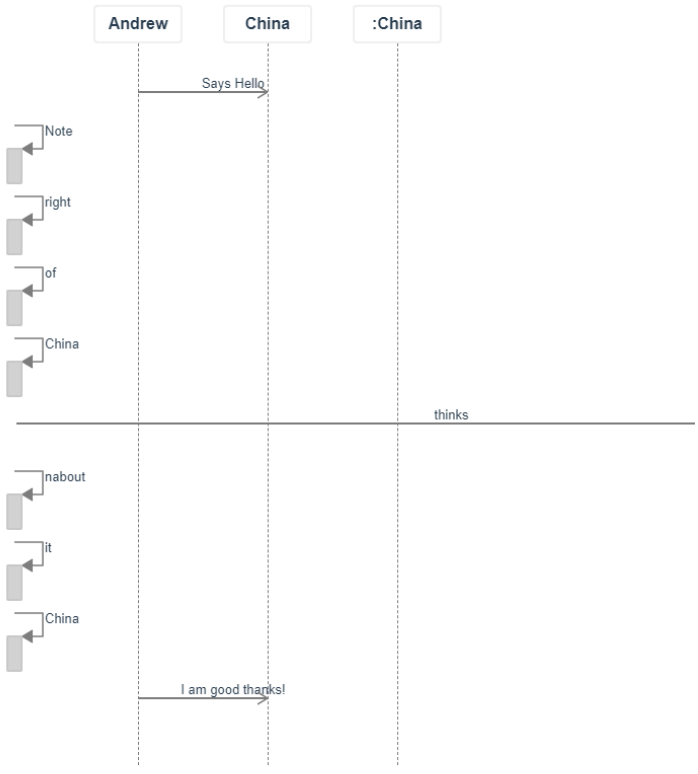
st->op->cond
cond(yes)->e
cond(no)->op
```
[========]

绘制序列图 Sequence Diagram

```seq
Andrew->China: Says Hello
Note right of China: China thinks\nabout it
China-->Andrew: How are you?
Andrew->>China: I am good thanks!
```
End
```

[========]

绘制序列图 Sequence Diagram



End

为了更好的支持 Markdown 格式，另外推荐两个选装插件：

- Markdownlint Markdown 语法格式检测和语法错误提

觉得还不错
丿寒风 关注
23

- Markdown Preview Enhanced 显示功能增强

本文也是用 VS Code 书写

2023/6/29补充

Markdown 是 HTML 的子集，不支持的内容还可以直接使用 HTML 写，下面是一个复杂表格的例子。

```
<table>
  <tr>
    <td>列名1</td>
    <td>列名2</td>
    <td>列名3</td>
  </tr>
  <tr>
    <td>行首1</td>
    <td colspan="2">合并列2-3</td>
  </tr>
  <tr>
    <td rowspan="2">行首2-3</td>
    <td>列名2</td>
    <td>列名3</td>
  </tr>
  <tr>
    <td>列名2</td>
    <td>列名3</td>
  </tr>
</table>
```

生成的效果如下：

<table>
<tr><td>列名1</td><td>列名2</td><td>列名3</td></tr>
<tr><td>行首1</td><td colspan="2">合并列2-3</td></tr>
<tr><td rowspan="2">行首2-3</td><td>列名2</td><td>列名3</td></tr>
<tr><td>列名2</td><td>列名3</td></tr>
</table>

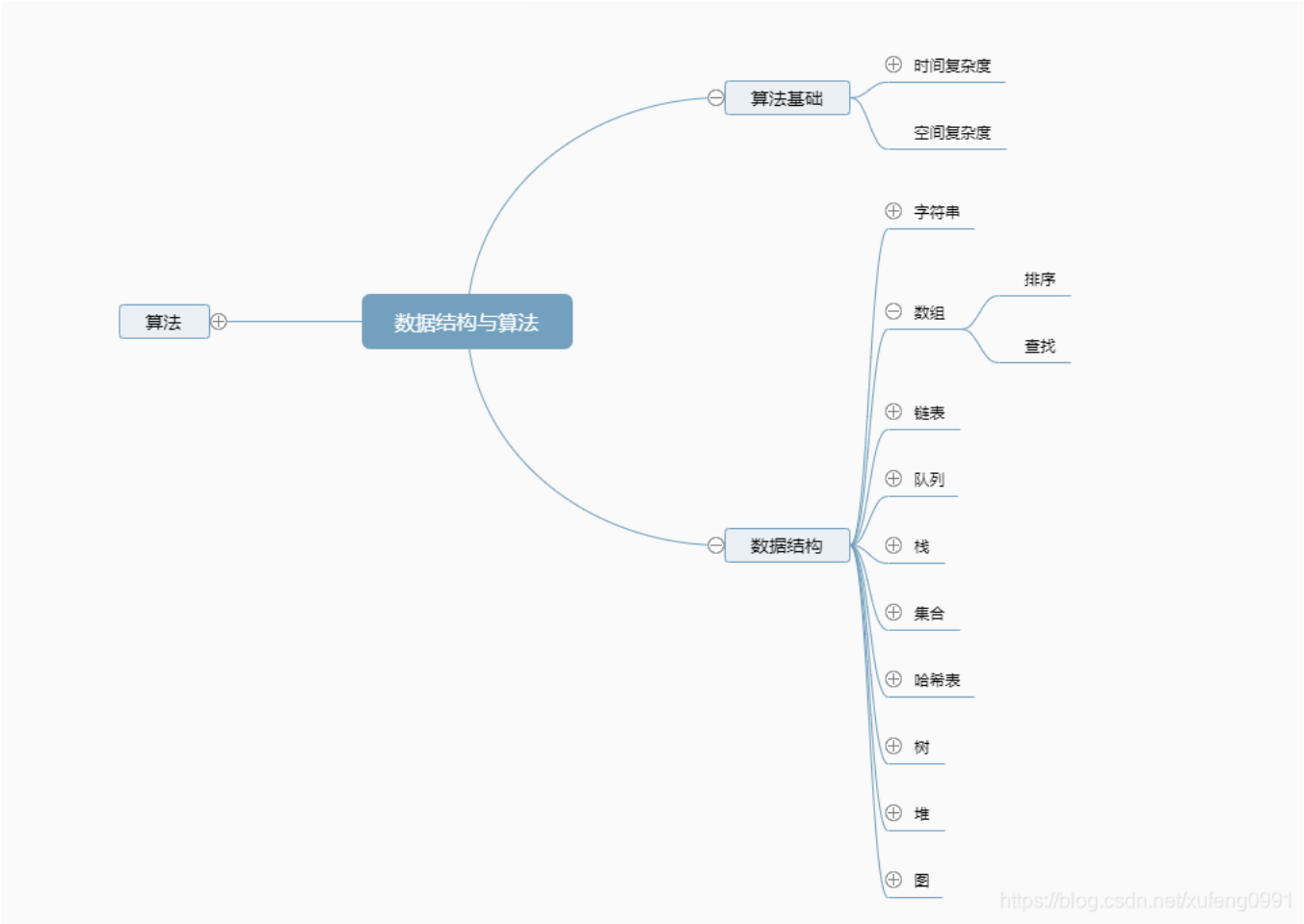
脑图

思维导图又叫脑图，是一种图形化思维的工具。百度脑图相信很多人都用过，推荐以下插件，在 VS Code 中画思维导图，体验跟百度脑图一样，这个插件

- vscode-mindmap

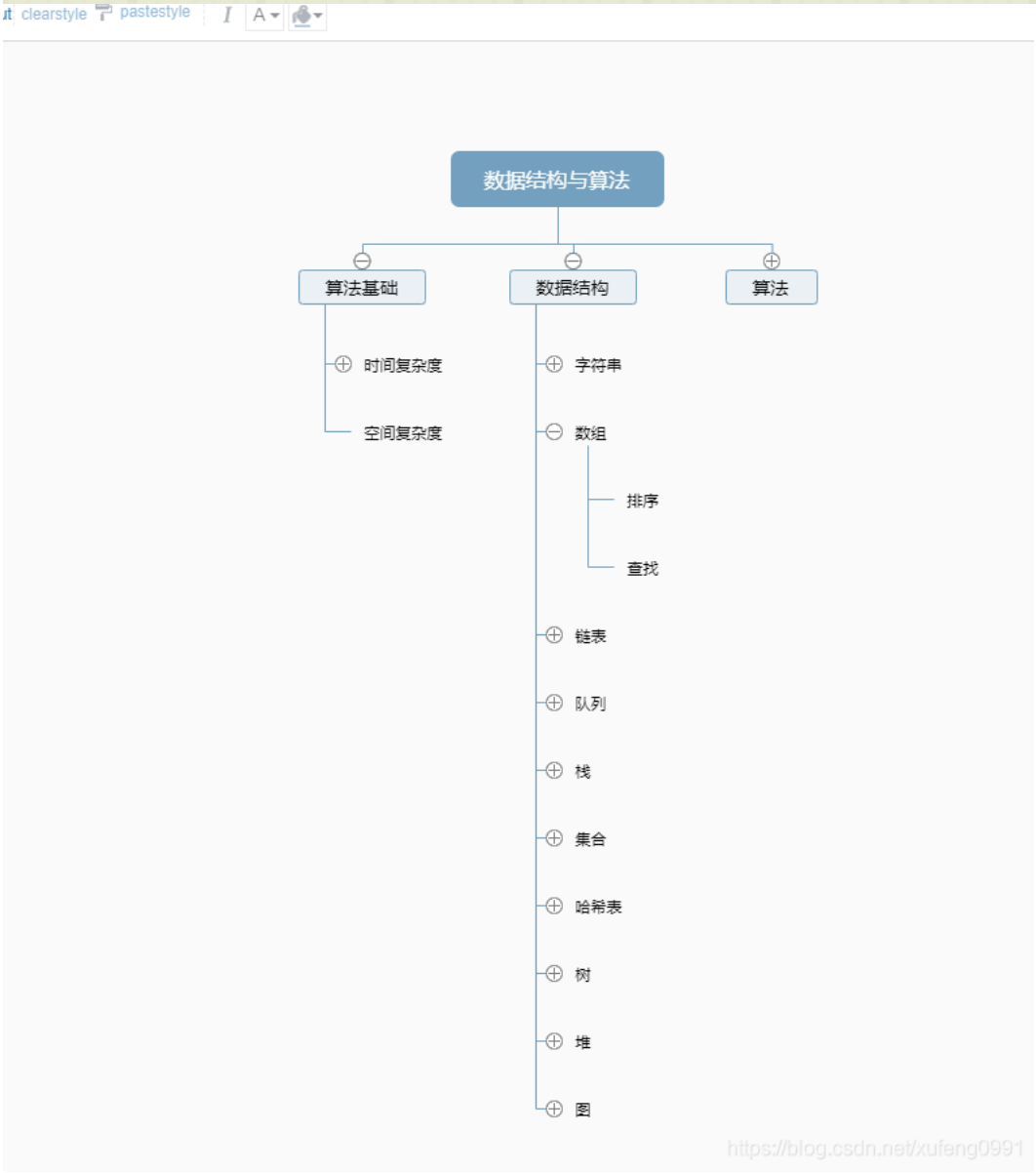
文件后缀名以保存为`.km`，会自动显示，也可以用快捷键`ctrl/cmd + m`显示。跟 Markdown 不一样，思维导图的编辑，是在图形化的界面中，不需要写文
这个插件整理的算法与数据结构的知识点，也分享出来：

觉得还不错
丿寒风 关注
23



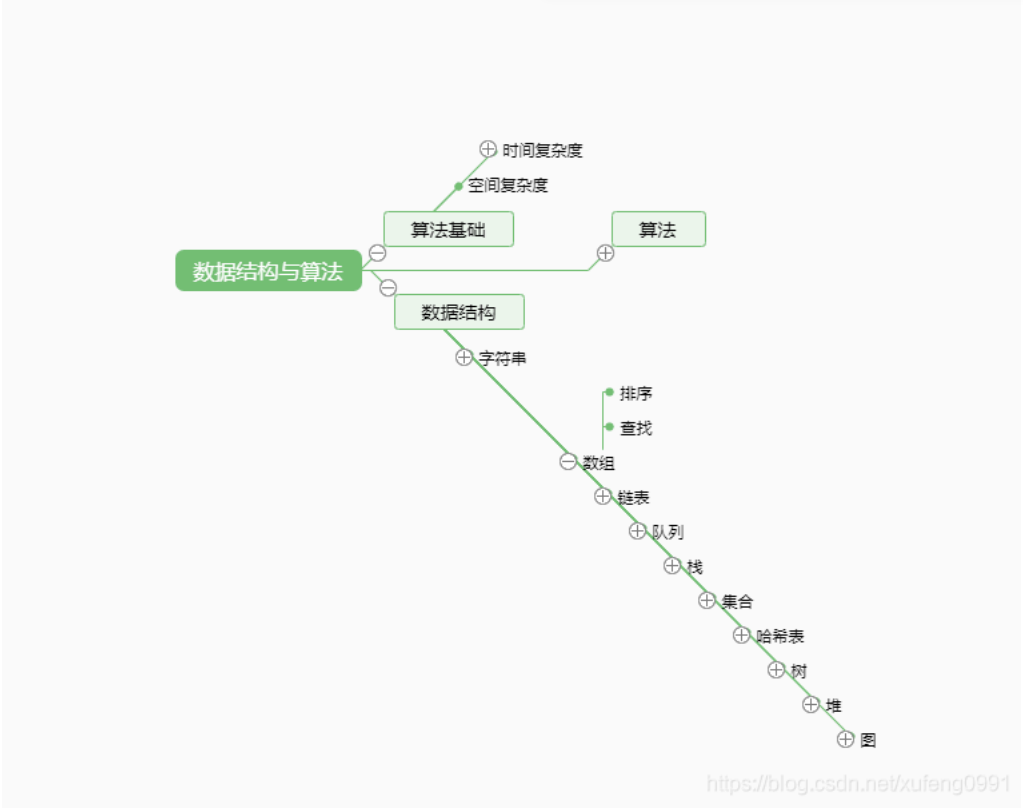
可以切换各种不同的显示样式:

觉得还不错
丿寒风 关注
23



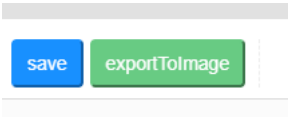
也可以选择不同的配色：

觉得还不错
丿寒风 关注
23



缺点是不会自动保存，按快捷键 `ctrl/cmd+s` 保存。

可以导出成图片：



UML

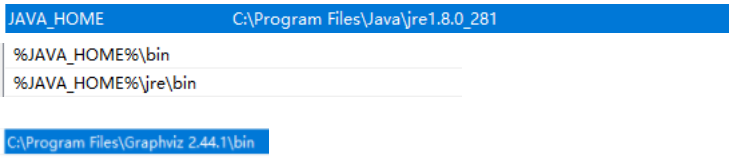
UML 即 统一建模语言，是一种面向对象分析和设计的建模工具，广泛使用的有时序图和类图。利用插件，也可以在 VS Code 中画 UML 图。PlanUML是一UML 图绘制库，可以使用文本描述，绘制出对应的 UML 图，非常适合程序员使用。在 VS Code 中支持 PlantUML，需要安装插件：

- PlantUML

同时，还必须安装：

- Java 环境(下载地址)。
- Graphviz-Dot。

配置环境变量如下：



将文件名以 `*.pu, *.puml` 后缀结尾，即可使用PlantUML 语法画图，使用快捷键 `Alt+D` 编译预览。

下面是一个简单的时序图示例：

```
@startuml test
== build ==
Main -> Builder: createEntity()
Builder -> EntityFactory: create()
EntityFactory -> Entity: constructor()
== run ==
Main -> Main: update()
Main -> Entity: update()
Entity -> Entity: render()
Entity -> Entity: emitEvent(EVENT_RENDER)
```

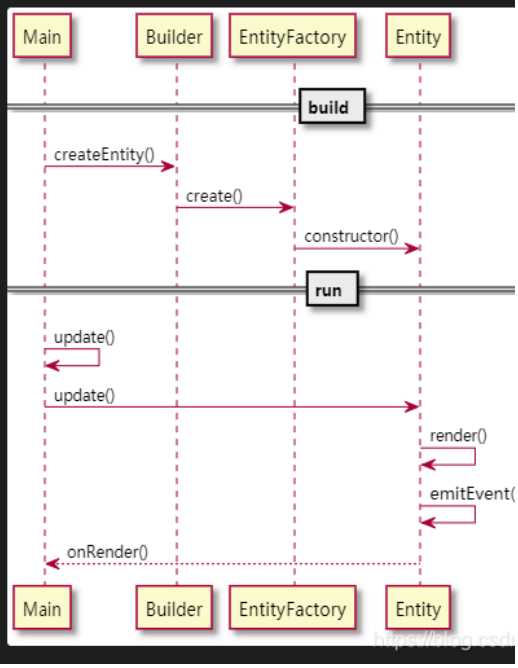
觉得还不错
丿寒风 关注
23

```
Entity --> Main: onRender()

@enduml
```

```
@startuml test
== build ==
Main -> Builder: createEntity()
Builder -> EntityFactory: create()
EntityFactory -> Entity: constructor()
== run ==
Main -> Main: update()
Main -> Entity: update()
Entity -> Entity: render()
Entity -> Entity: emitEvent(EVENT_RENDER)
Entity --> Main: onRender()

@enduml
```



下面是一个类图的示例：

```
@startuml test

Class ClassA {
    -- 属性 --
    +String publicAttr
    #int protectedAttr
    -long privateAttr
    .. 其他格式 ..
    +A: String
    #B: Number
    -C: Boolean
    == 方法 ==
    +getAttr()
    #setAttr()
    -readAttr()
}

note top: 在顶部注释说明

Class ClassB {

}

note right: 在右边注释说明

ClassA <-- ClassB:关联
ClassA <.. ClassB : 依赖
ClassA o-- ClassB:聚集
ClassA <|-- ClassB:泛化

```

觉得还不错
丿寒风 关注 23

```
ClassA <|.. ClassB:实现
@enduml
```

```
@startuml test

Class ClassA {
    -- 属性 --
    +String publicAttr
    #int protectedAttr
    -long privateAttr
    .. 其他格式 ..
    +A: String
    #B: Number
    -C: Boolean
    == 方法 ==
    +getAttr()
    #setAttr()
    -readAttr()
}

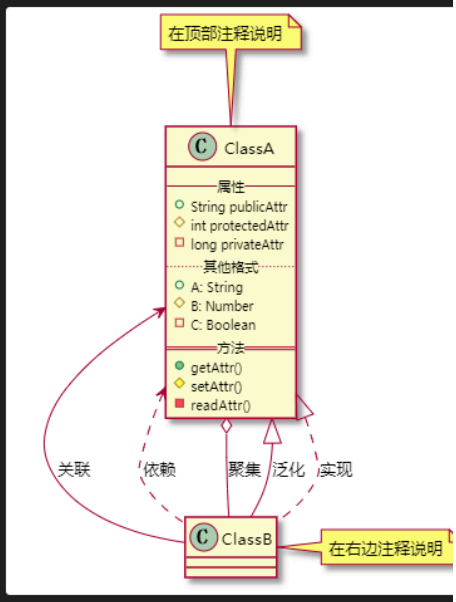
note top: 在顶部注释说明

Class ClassB {

}

note right: 在右边注释说明

ClassA <-- ClassB:关联
ClassA <.. ClassB : 依赖
ClassA o-- ClassB:聚集
ClassA <|-- ClassB:泛化
ClassA <|.. ClassB:实现
@enduml
```



MD 中插入 UML

除了在单独的 uml 文件中，也可以将 UML 的绘制代码嵌入到 Markdown 中，格式如下：

```
# markdown 中嵌入 uml

## 格式

可以将 plantuml 作为代码，嵌入到 markdown 文本中，如：

``` plantuml
@startuml
title markdown 中嵌入 uml
package test
{
    class A {

    }

    class B {

    }

    B -> A: method
}
@enduml
```

结果

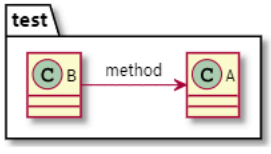
如右图显示
```

markdown 中嵌入 uml

格式

可以将 plantuml 作为代码，嵌入到 markdown 文本中，如：

markdown 中嵌入 uml



结果

如右图显示

https://blog.csdn.net/xufeng0991

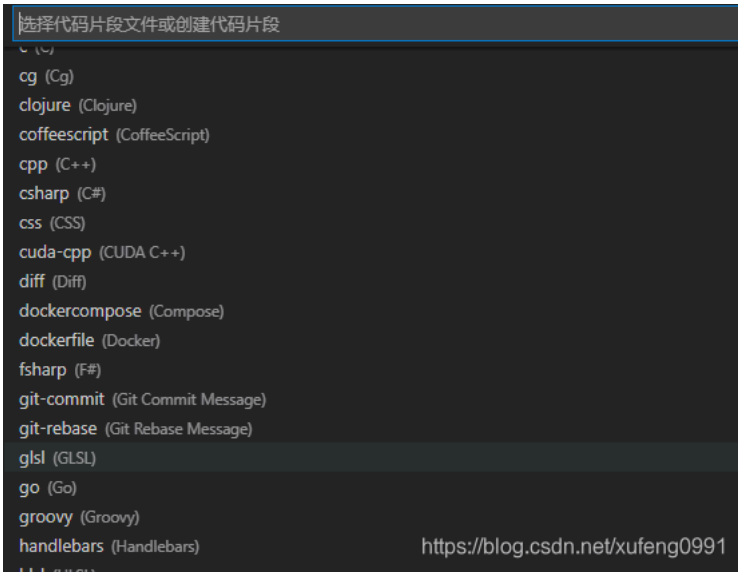
代码 模板

在开发的过程中，很多代码片段都是重复的，如果能自动

觉得还不错
丿寒风 关注
23

1. 设置-》用户代码片段

2. 选择要设置模板编程语言，比如 typescript



3. 会打开一个对应编程预言的 json 文件，如 typescript.json，其默认内容如下，是一个简单的示例：

```
{
    // Place your snippets for typescript here. Each snippet is defined under a snippet name and has a prefix, body and
    // description. The prefix is what is used to trigger the snippet and the body will be expanded and inserted. Possible variables
    // $1, $2 for tab stops, $0 for the final cursor position, and ${1:label}, ${2:another} for placeholders. Placeholders with the
    // same ids are connected.
    // Example:
    // "Print to console": {
    //  "prefix": "log",
    //  "body": [
    //      "console.log('$1');",
    //      "$2"
    //  ],
    //  "description": "Log output to console"
    // }

}
```

默认内容是一个简单的示例，参照例子，添加自己的模板，如

```
    {
        // Place your snippets for typescript here. Each snippet is defined under a snippet name and has a prefix, body and
    1   // description. The prefix is what is used to trigger the snippet and the body will be expanded and inserted. Possible
    2   // $1, $2 for tab stops, $0 for the final
    3   // same ids are connected.
    4
```

觉得还不错
丿寒风 关注
23

```
    // Example:
    // "Print to console": {
    //  "prefix": "log",
    //  "body": [
    //      "console.log('$1');",
    //      "$2"
    //  ],
    //  "description": "Log output to console"
    // }
    "ts file template": {
        "prefix": "ts",
        "body": [
            "",
            "",
            "/**",
            " * ${2: description}",
            " *",
            " * @export",
            " * @class ${1:ClassName}",
            " * @author xuld",
            " * @date $CURRENT_YEAR-$CURRENT_MONTH-$CURRENT_DATE",
            " */",
            "",
            "export class ${1:ClassName} extends $3",
            "{",
            "    $0",
            "}",
        ],
        "description": "my typescript file template"
    },
    "constructor": {
        "prefix": "con",
        "body": [
            "constructor()",
            "{",
            "    $0",
            "}",
        ],
        "description": "my typescript constructor template"
    },
    "ts region template": {
        "prefix": "re",
        "body": [
            "//#region ${1:setter getter}",
            "$0",
            "//#endregion ${1:setter getter}",
        ],
        "description": "my typescript region template"
    },
    "getter setter": {
        "prefix": "gs",
        "body": [
            "public get ${1:attrName}(): ${2:Type}",
            "{",
            "    return this._${1:attrName};",
            "}",
            "public set ${1:attrName}(value: ${2:Type})",
            "{",
            "    this._${1:attrName} = value;$0",
            "}",
        ],
        "description": "my typescript getter setter template"
    },
    "singleton instance": {
        "prefix": "ins",
        "body": [
            "private static _instance: ${1:ClassName} = null;",
            "public static get instance(): ${1:ClassName}",
            "{",
            "    if (this._instance == null)",
            "    {",
```

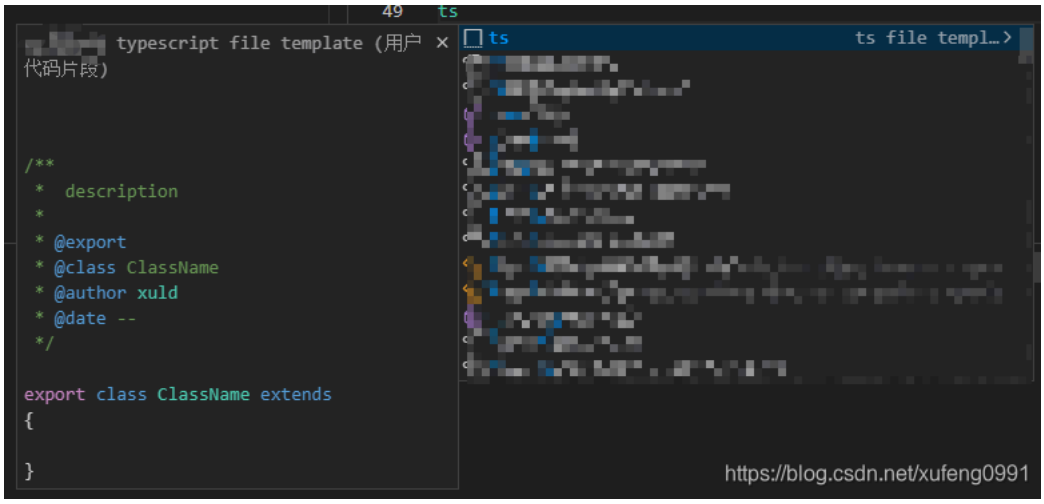
觉得还不错
丿寒风 关注
23

```
            "        this._instance = new ${1:ClassName}();",
            "    }",
            "    return this._instance;",
            "}",
        ],
        "description": "my typescript singleton instance template"
    },
}
```

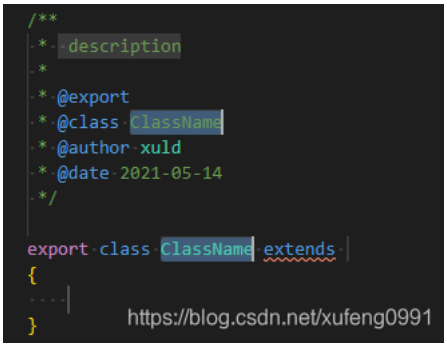
配置说明：

- json 的 key，简单的功能描述
- json 的 value，是配置具体的值
  - prefix ，输入之后弹窗自动补全的文本
  - body，补全的内容，其中
    - $0 表示 按tab建最后停留的位置
    - $1 表示默认鼠标停留的位置
    - $n 表示按tab键之后依次停留的位置
    - ${n-内容}，表示tab键移动时选中的文本，可以有多个，会同时选中
    - 支持变量，如：
      - $CURRENT_YEAR 年
      - $CURRENT MONTH 月
      - $CURRENT_DATE 日
  - description 功能的详细描述

4. 使用示例，比如如上配置，输入 ts 时，会出现提示：



按enter键，自动补全，按tab键可以在配置$n的位置自动切换，如下：



觉得还不错
丿寒风 关注
23

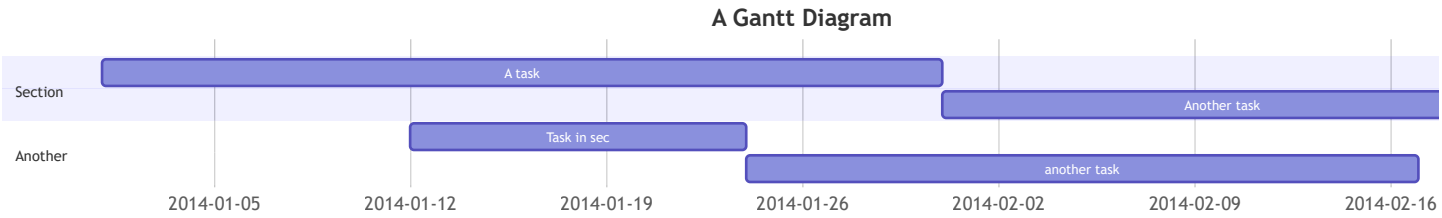
mermaid 语法

PlantUML 不支持甘特图，饼状图等，脑图也不是很好用，mermaid 可以作为补充，支持：

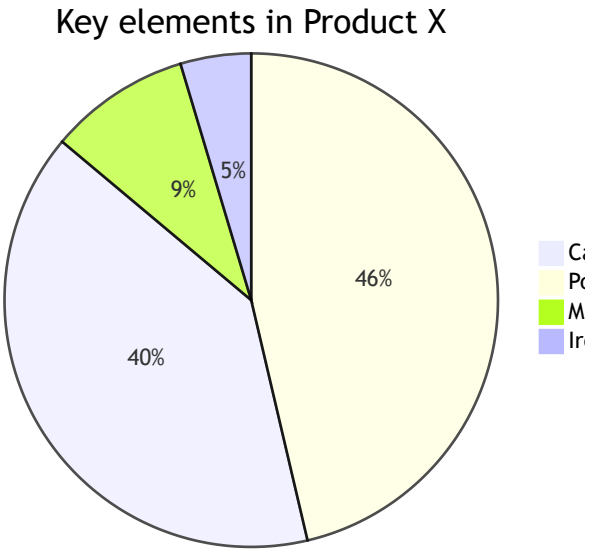
- 甘特图：使用 gantt 关键字，具体用法后文有示例
- 饼状图：使用 pie 关键字，具体用法后文示例
- 流程图：使用 graph 关键字，也可以用来画脑图
- 序列图：使用 sequenceDiagram 关键字
- 类图：使用 classDiagram 关键字
- 状态图：使用 stateDiagram 关键字
- 用户旅程图：使用 journey 关键字

语法说明

甘特图



饼状图



脑图（流程图）

语法

```
graph 方向描述
    图表中的其他语句...
```

方向

方向描述

方向	意义	翻译
TB	from Top to Bottom	从上到下
BT	from Bottom to Top	从下到上
RL	from Right to Left	
TB		

觉得还不错
丿寒风 关注
23

节点

节点定义

表述	说明
id[文字]	矩形节点
id(文字)	圆角矩形节点
id((文字))	圆形节点
id>文字]	右向旗帜状节点
id{文字}	菱形节点

连线

节点间的连线

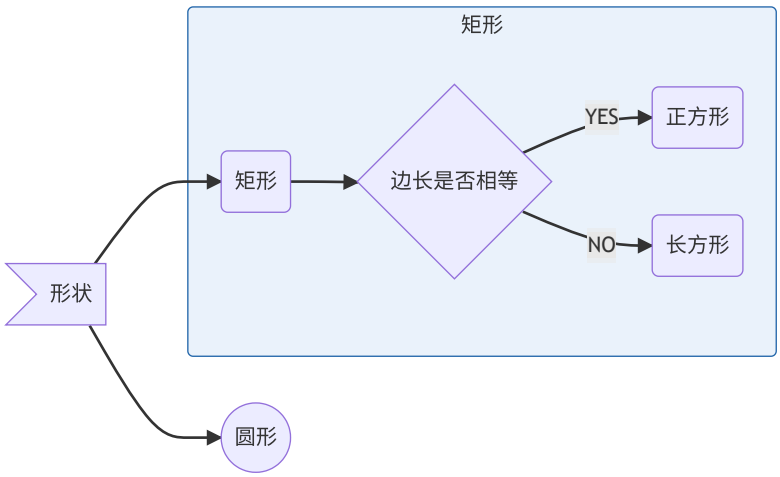
表述	说明
>	添加尾部箭头
-	不添加尾部箭头
—	单线
—text—	单线上加文字
==	粗线
==text==	粗线加文字
-.-	虚线
-.text.-	虚线加文字

节点的文字中包含标点符号，需要时用双引号包起来

子图表

```
subgraph 子图表名称
    子图表中的描述语句...
end
```

脑图示例



小结

VS Code 本身只是一个文本编辑工具，通过丰富的插件拓

觉得还不错
丿寒风 关注
23