

VS Code Markdown 示例

- [Markdown 基础语法](#)
 - [1.标题](#)
 - [2.引用](#)
 - [3.字体格式](#)
 - [4.代码](#)
 - [5.列表](#)
 - [6.分割线](#)
 - [7.链接](#)
 - [8.图片](#)
 - [9.表格](#)
 - [10.注脚](#)
 - [11.html 标签](#)
 - [12.段落](#)
- [Markdown扩展 \(Markdown preview enhanced\)](#)
- [优先介绍](#)
 - [目录列表 \(TOC\)](#)
 - [导入外部文件 import](#)
 - [文档导出](#)
- [MPE插件安装](#)
- [flow流程图示例](#)
 - [定义元素的语法](#)
 - [连接元素的语法](#)
 - [位置控制](#)
 - [示例](#)
- [Mermaid时序图](#)
- [Mermaid甘特图](#)
- [Mermaid流程图](#)
 - [指定方向](#)
 - [节点形状](#)
 - [连线](#)
- [什么是 PlantUML](#)
- [软件安装](#)
- [java类图示例](#)
- [时序图示例](#)
- [用例图示例](#)
- [流程图示例](#)
- [流程图新语法示例](#)
- [组件图示例](#)
- [状态图示例](#)
- [PlantUML脑图](#)
 - [兼容OrgMode语法](#)
 - [兼容Markdown语法](#)
- [ditaa示例](#)

Markdown 基础语法

转自: [Markdown语法命令大全\(思维导图版\)](#)

1.标题

一级标题

二级标题

三级标题

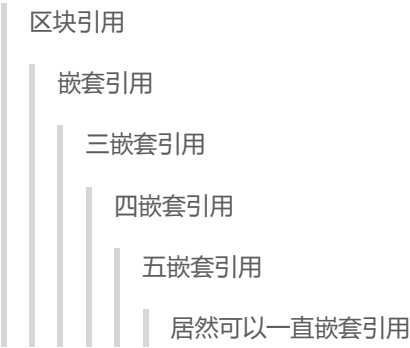
四级标题

五级标题

六级标题

这里的 {ignore=true} 语法目前请忽略。

2.引用



3.字体格式

加粗

斜体

删除线

标记

4.代码

- 1. 单行代码(前四个空格)
hello world
- 2. 行内代码
- 3. 代码段:(上下3点夹代码段)

```
static {  
  
}
```

代码块的扩展参考Markdown扩展部分

5.列表

1. 第一点(数字标点后有1空格)
我们测试
再来一行
2. 第三点 (数字编号错误并没有影响)
3. 第几点?

- 这一点
- item

有序列表间有空行的话可能会导致格式怪异?

1. 第一点(数字标点后有1空格)
2. 第三点 (数字编号错误并没有影响)
3. 第几点?

6.分割线

一行连用三个或者三个以上的星号、减号、或者下划线，就可以表示分割线，各家渲染有差异，有的支持等号表示双横线？

1

2

3

=====

7.链接

2、3行组合的写法没效果


[GitHub](#)

[GitHub][1]

[1]:<http://github.com>

[链接](#)

8.图片

 GitHub set up



惊叹号+图片本地地址

9.表格

Item	Value	Qty
Computer	1600 USD	5
Phone	12 USD	12
Pipe	1 USD	234

10.注脚

注脚^[1]

脚注^[2]

11.html 标签

值班人员	星期一	星期二	星期三
	李强	张明	王平

12.段落

一行文字即一个段落，段落是自动换行的，无需手动输入回车换行符（有道有两个空格换行的逻辑）

Markdown扩展（Markdown preview enhanced）

参考[MPE官方网站](#)

优先介绍

优先介绍下面三个功能，看到有三个这么好的扩展就忍不住安装MPE扩展了

目录列表（TOC）

参考[MPE官方介绍](#)

Markdown Preview Enhanced 支持你在 markdown 文件中创建目录，效果就像本教程的目录一样。
基本格式，通过编写 [front-matter](#) 来进行设置：

```
---
depth_from: 2
depth_to: 3
ordered: false
---
```

更详细的用法以后再学！

导入外部文件 import

参考[MPE官方介绍](#)

`import` 支持导入多种类型的文件，若和本教程一样导入的是markdown文件将会被分析处理然后被引用。支持导入特定行数，如 `{line_begin=11}` 表示从import指定文件的第11行开始导入，同理 `{line_end=-4}` 就表示....。

```
@import '你的文件' {line_begin=11}
```

更详细的用法以后再学！

文档导出

我们写好markdown文档预览时是像word一样的有格式的文档，可是我们要怎么样把预览的效果发给别人呢？这就是文档导出功能了。在markdown预览窗口 Preview xxx.md 中（需要你安装Markdown Preview Enhanced插件，并打开其预览窗口，具体参见下一小节的最后一段）右键即可看到好几个导出菜单，这里我们只介绍导出为HTML和PDF。

1. HTML》HTML(offline)，选择这个选项如果你要离线使用这个 html 文件
2. Chrome(Puppeteer)》PDF，导出为PDF
3. eBook》PDF，也可以用这个导出为PDF

导出PDF的多个菜单可以任选其一，

更详细的用法以后再学！

MPE插件安装

本教程我们需要安装的程序和插件

- vs code最左边的侧边栏第5个图标即是vs code扩展商店，所有vs code扩展插件都可以在这里搜索安装。
- 安装vs code插件 Markdown Preview Enhanced
- 安装vs code插件 PlantUML
- 安装 [graphviz](#)
graphviz 是个开源的图片渲染库。安装了这个库才能在 Windows 下实现把 PlantUML 脚本转换为图片。
- 安装jdk1.8或以上，这也是vs code的java开发插件 Java Language Support 支持的最低版本。安装多个版本的jdk不会冲突，所以若你电脑安装了jdk1.6，若使用Eclipse等java开发IDE，可以在IDE中指定工作区的jdk版本如jdk1.6。

安装好插件后，我们打开本教程中的文件："markdown [tutorial.md](#)"，即可查看完整的教程，打开文件后右键选择"Markdown Preview Enhanced: Open Preview to the Side"可以看到使用Markdown Preview Enhanced预览的效果。

flow流程图示例

定义元素的语法

格式： 变量=>操作块: 备注名

或

tag=>type: content:>url

tag就是元素名字，
type是这个元素的类型，有6中类型，分别为：
start # 开始
end # 结束
operation # 操作
subroutine # 子程序
condition # 条件
inputoutput # 输入或输出[平行四边形]
content就是在框框中要写的内容，注意type后的冒号与文本之间一定要有个空格。
url是一个连接，与框框中的文本相绑定

st=>start: 开始
e=>end: 结束
普通操作块 opration
op1=>operation: 第一个操作块
op2=>operation: 第二个操作块
判断块 condition
cond1=>condition: 第一个判断
cond2=>condition: 第二个判断
输入输出块 inputoutput[平行四边形]
io1=>inputoutput: 输入输出块1
io2=>inputoutput: 输入输出块2
子任务块
sub1=>subroutine: 子任务1
sub2=>subroutine: 子任务2

连接元素的语法

用->来连接两个元素，需要注意的是condition类型，因为他有yes和no两个分支，所以要写成

```
c2(yes)->io->e  
c2(no)->op2->e  
分着写  
c2(yes)->io  
io->e  
c2(no)->op2  
op2->e
```

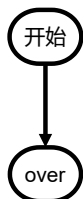
位置控制

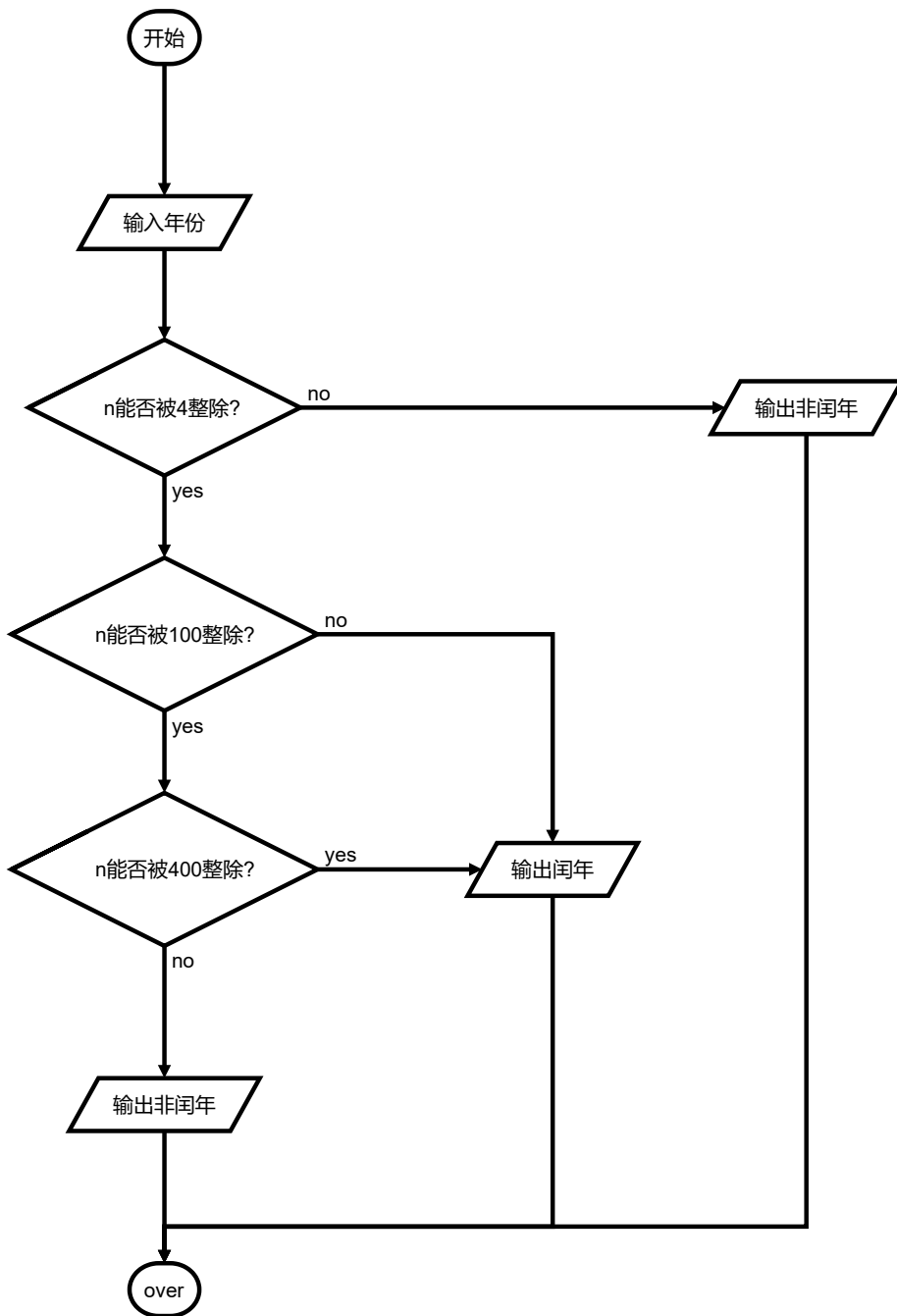
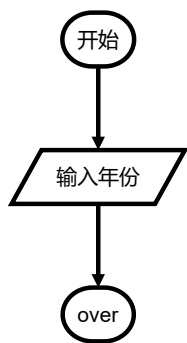
cond1(no)->op2(right)->op1 #控制 op2 位置置于右边，再由op2 返回 op1 (好像不能向左)

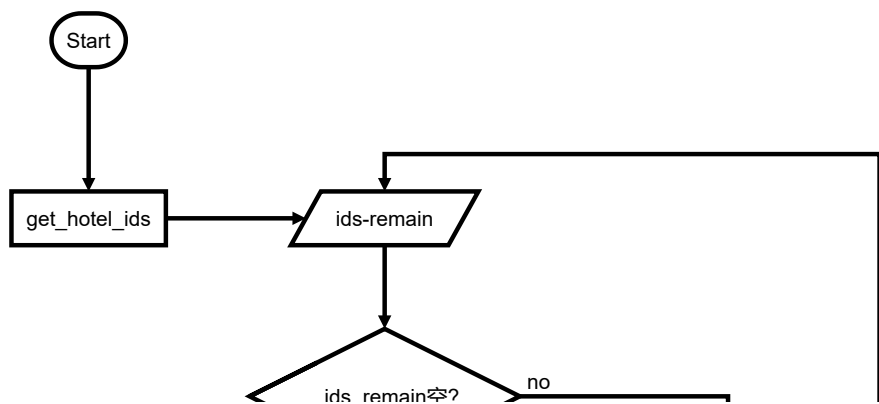
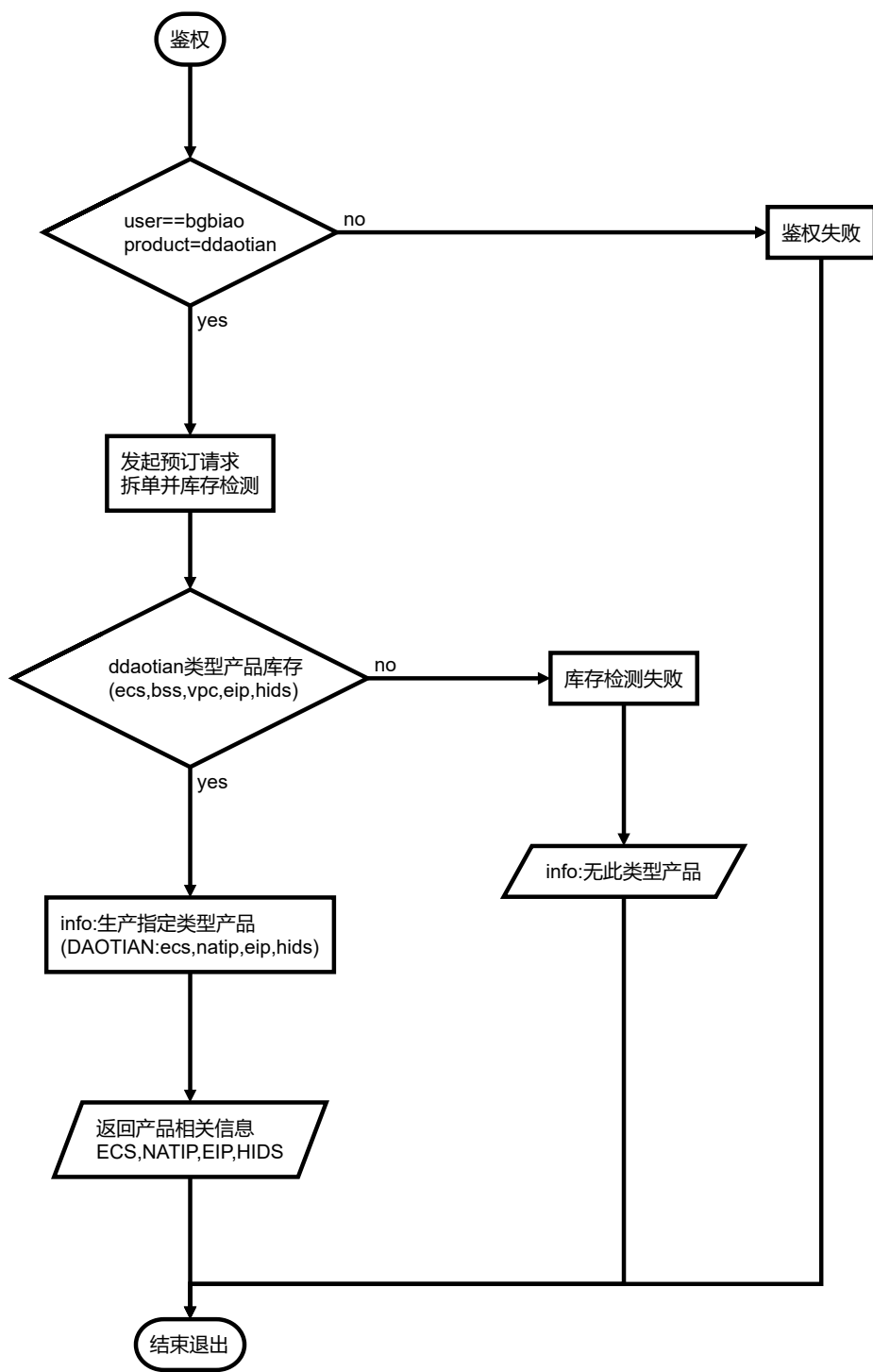
还可以这样 cond1(no,right)

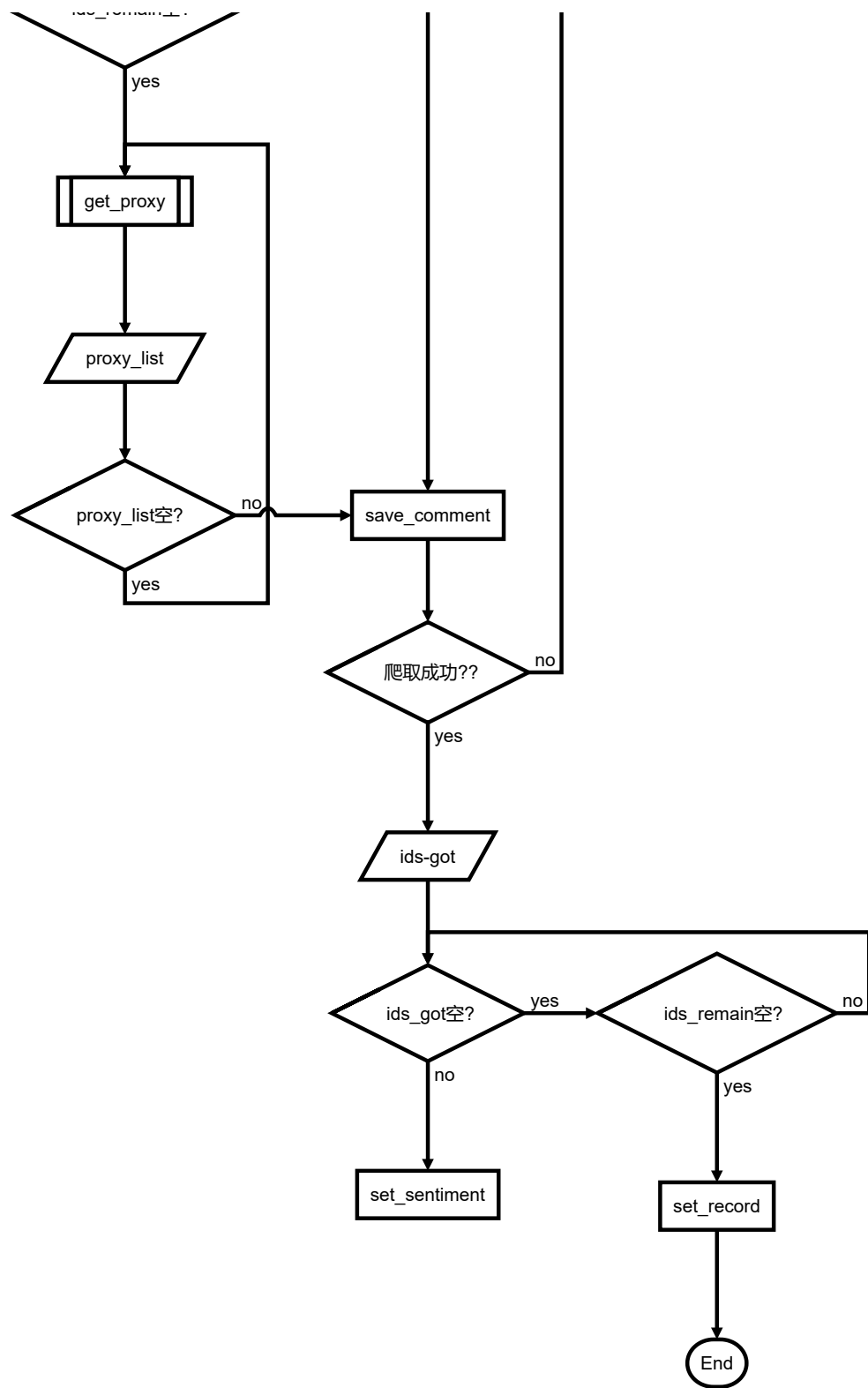
cond1(yes)->e

示例



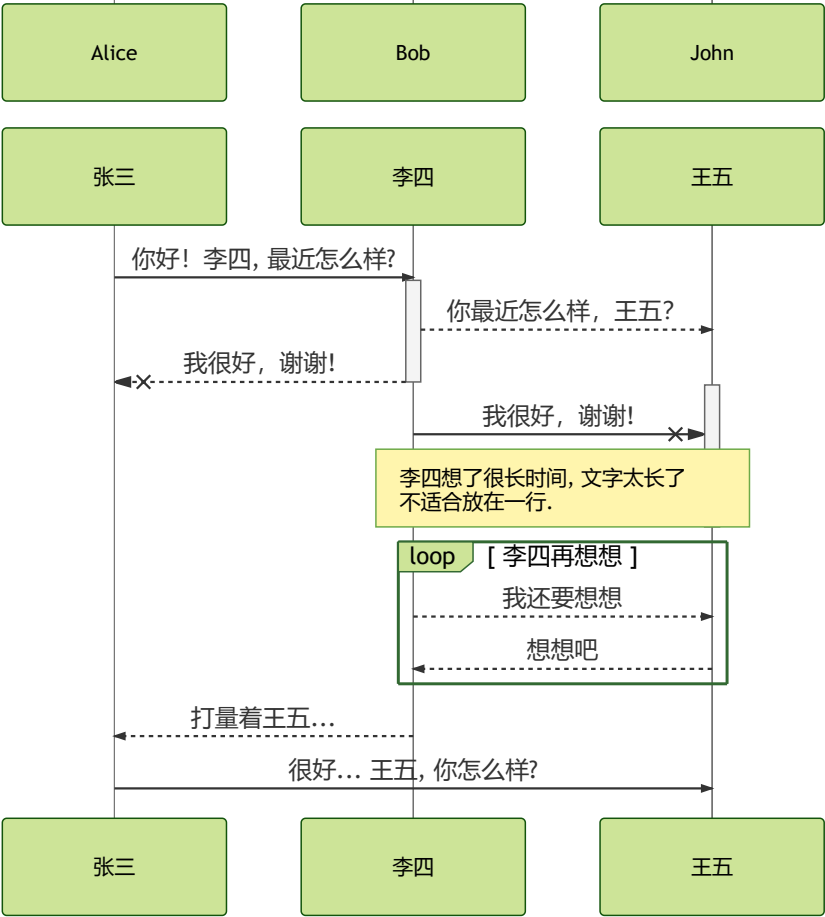
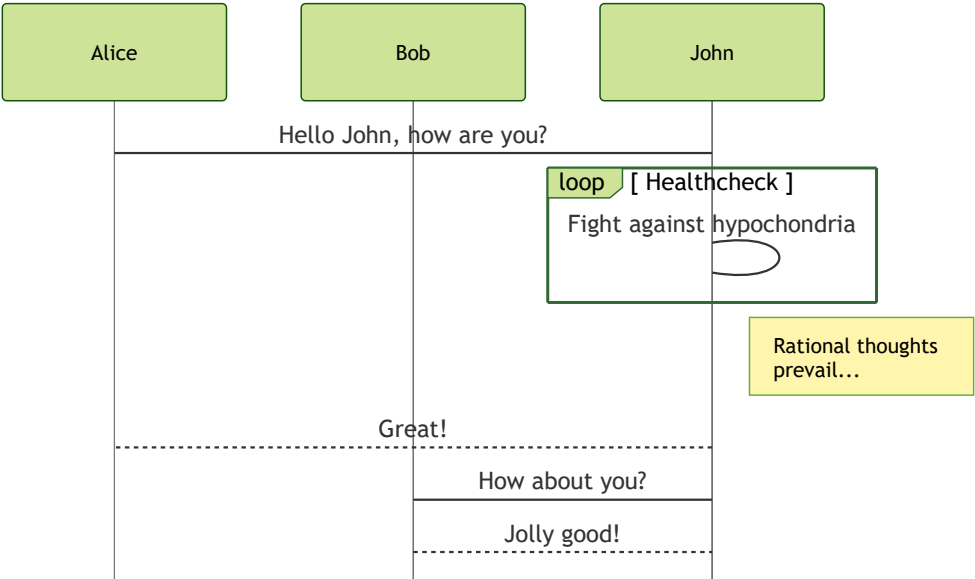


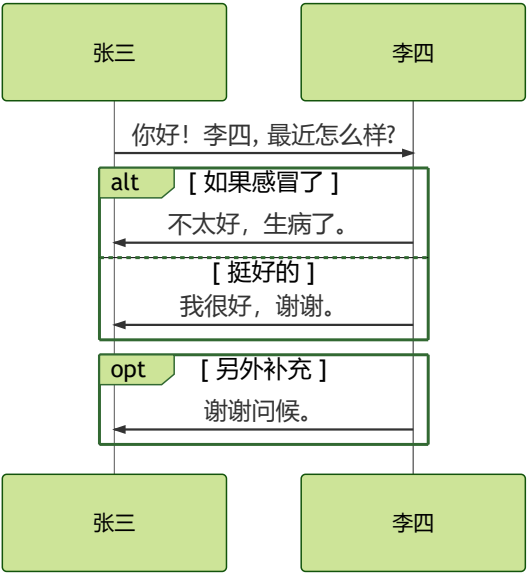




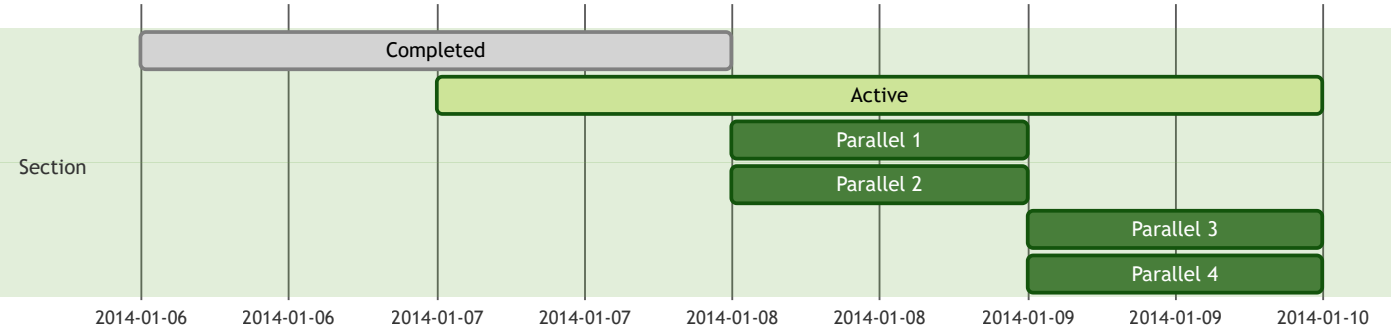
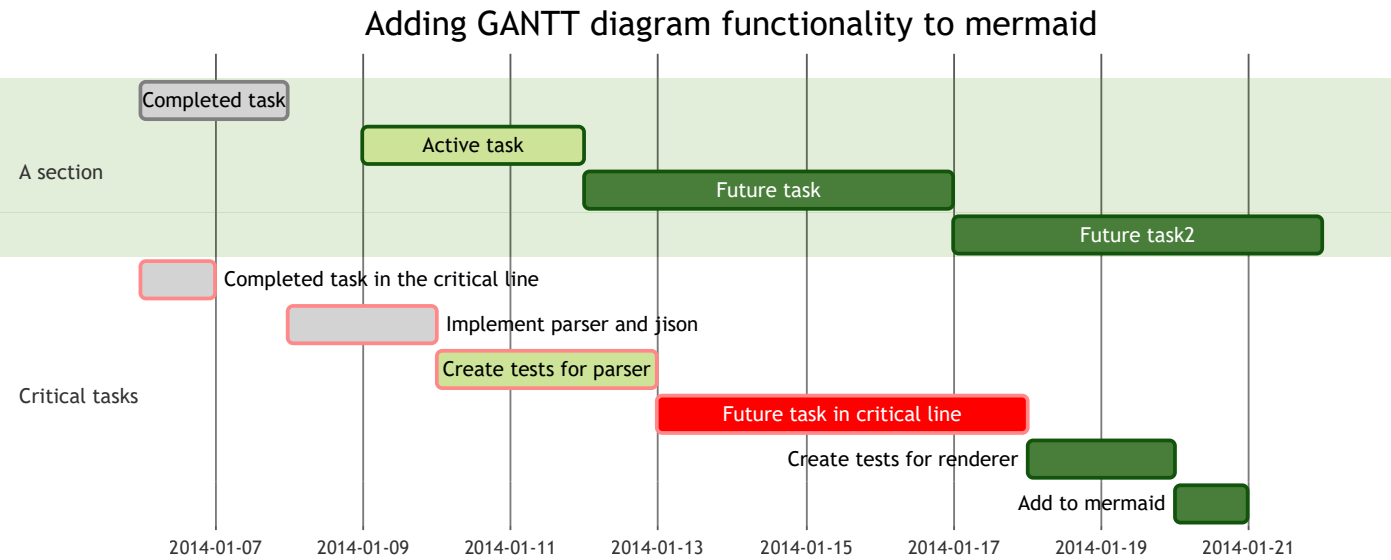
VS Code默认主题下，Mermaid Theme默认主题是dark，显示效果有问题，我设置为forset。

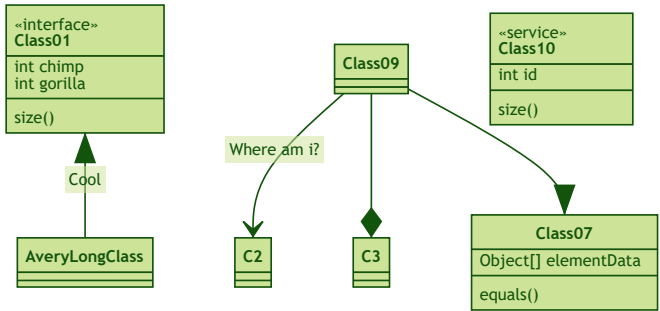
Mermaid时序图



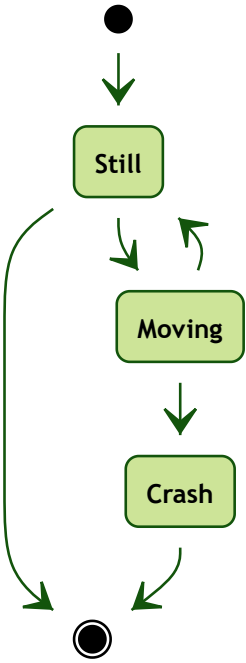


Mermaid甘特图

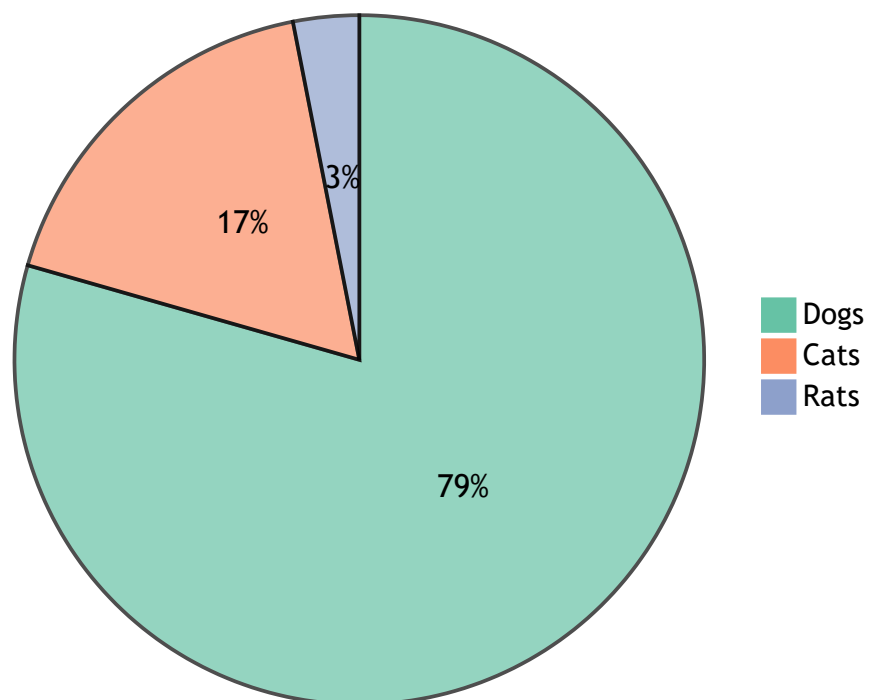




状态图

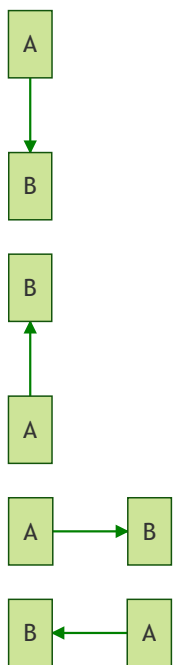


饼图



Mermaid流程图

指定方向



节点形状

这是直角四边形

这是圆角四边形

这是圆形

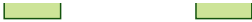
这是菱形

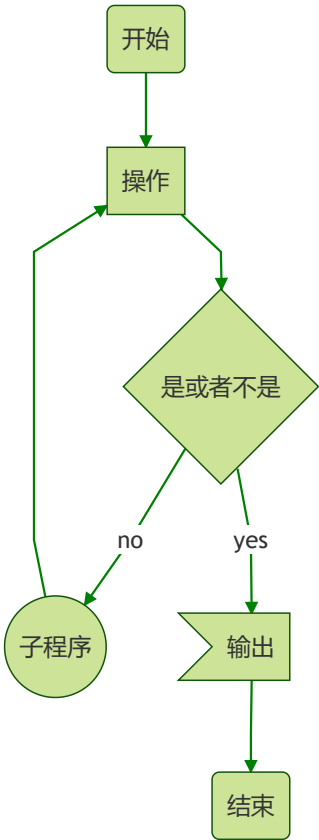
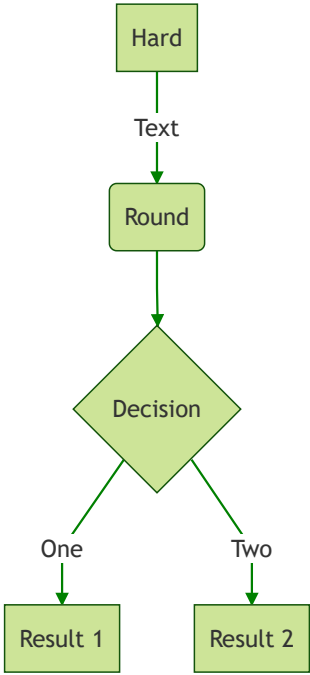
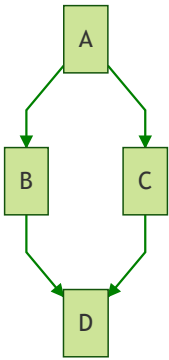
这是非对称图形

连线

- 箭头连接 A1-->B1
- 开放连接 A2---B2
- 标签连接 A3--text---B3 或者 A4---|text|B4
- 箭头标签连接 A5--text-->B5 或者 A6-->|text|B6
- 虚线开放连接 A7.-B7 或者 A8-.-B8 或者 A9..-B9
- 虚线箭头连接 A10.->B10 或者 A11-.->B11 或者 A12..->B12
- 标签虚线连接 A13-.text.-B13
- 标签虚线箭头连接 A14-.-text.->B14
- 粗线开放连接 A15===B15
- 粗线箭头连接 A16==>B16
- 标签粗线开放连接 A17==text===B17
- 标签粗线箭头连接 A18==text==>B18







什么是 PlantUML

PlantUML 是一个画图脚本语言，用它可以快速地画出：

- 时序图
- 流程图
- 用例图
- 状态图
- 组件图

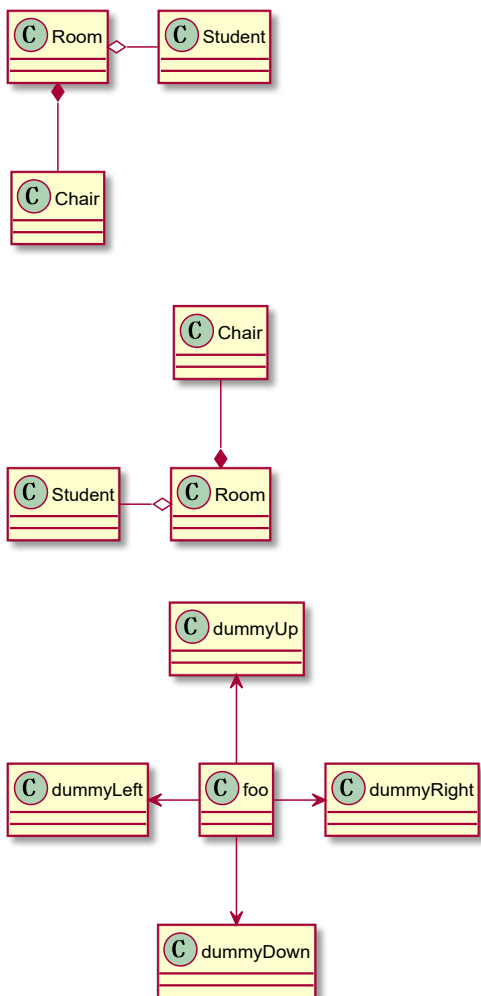
软件安装

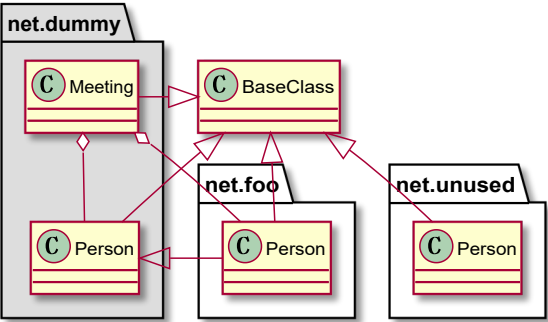
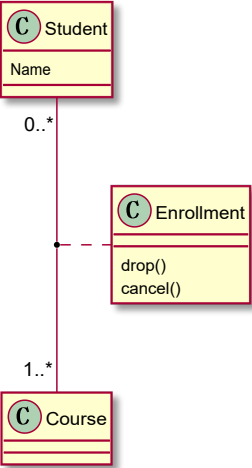
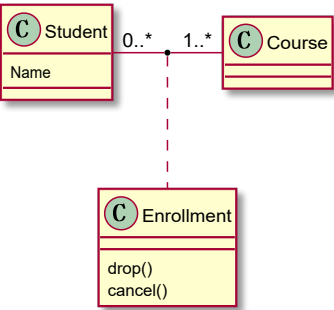
PlantUML应该可以单独使用，但是我是在VS Code中使用他，且是使用Markdown Preview Enhanced扩展来预览渲染效果。

- 安装VS Code插件Markdown Preview Enhanced和PlantUML插件
- 需要java支持，即要安装jdk
- 安装 [graphviz](#)

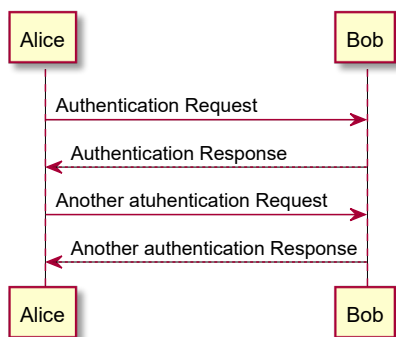
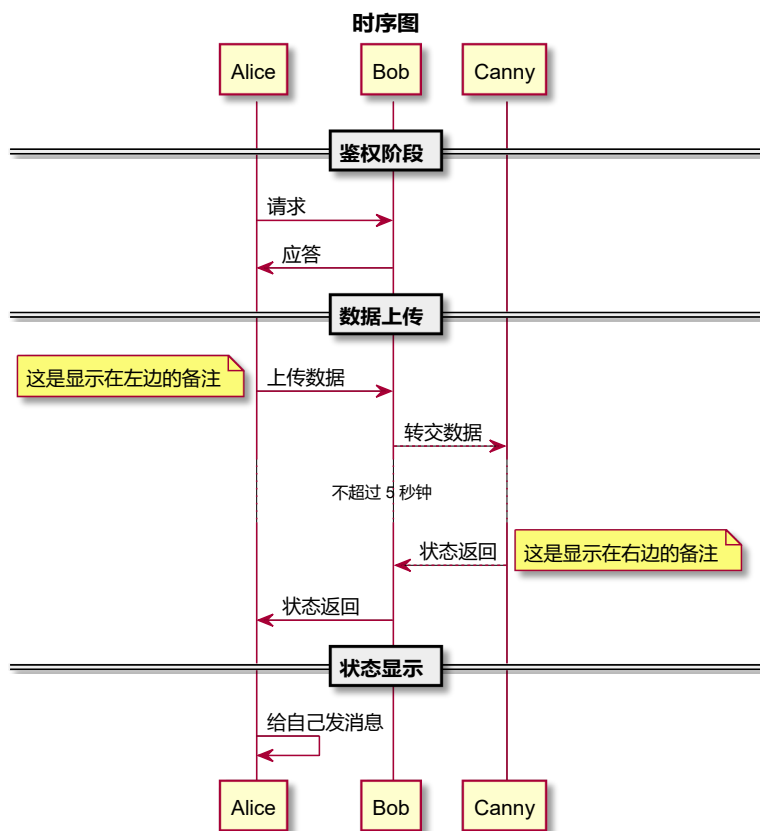
graphviz 是个开源的图片渲染库。安装了这个库才能在 Windows 下实现把 PlantUML 脚本转换为图片。

java类图示例





时序图示例

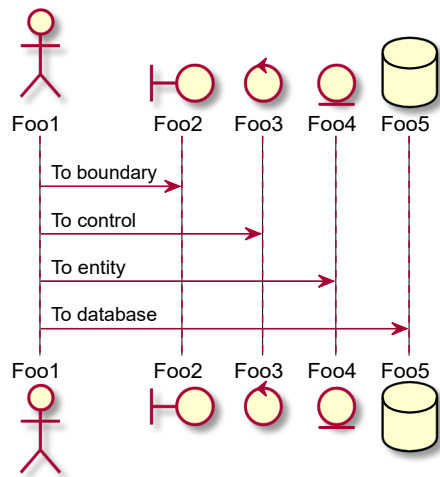
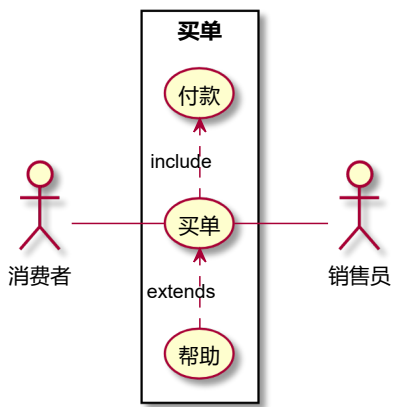


TIPS:

- 使用 title 来指定标题
- '->' 和 '-->' 来指示线条的形式
- 在每个时序后面加冒号 : 来添加注释
- 使用 note 来显示备注, 备注可以指定显示在左边或右边
- 使用 == xxx == 来分隔时序图
- 使用 ... 来表示延迟省略号
- 节点可以给自己发送消息, 方法是发送方和接收方使用同一个主体即可

用例图示例

用例图

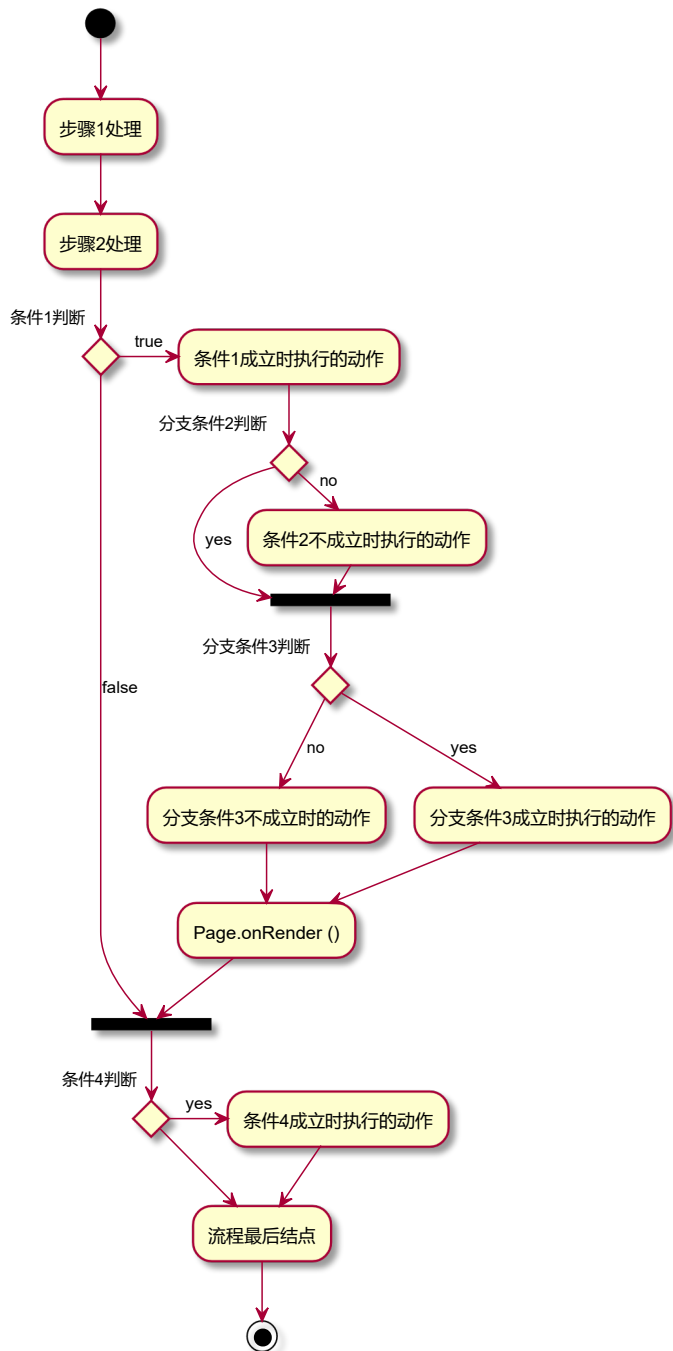


TIPS:

- 用例图是指由参与者（Actor）、用例（Use Case）以及它们之间的关系构成的用于描述系统功能的静态视图
- [百度百科](#)上有简易的入门资料，其中用例之间的关系 (include, extends) 是关键
- 使用 actor 来定义参与者
- 使用括号 (xxx) 来表示用例，用例用椭圆形表达
- 使用不同的线条表达不同的关系。包括参与者与用例的关系，用例与用例的关系

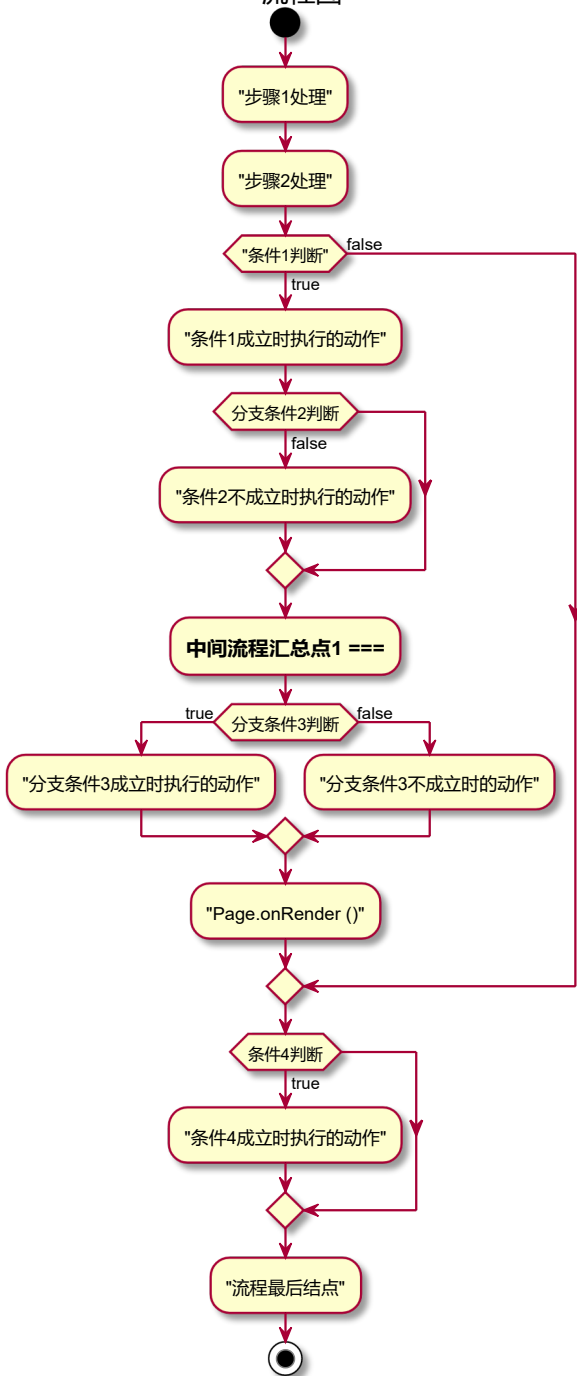
流程图示例

流程图

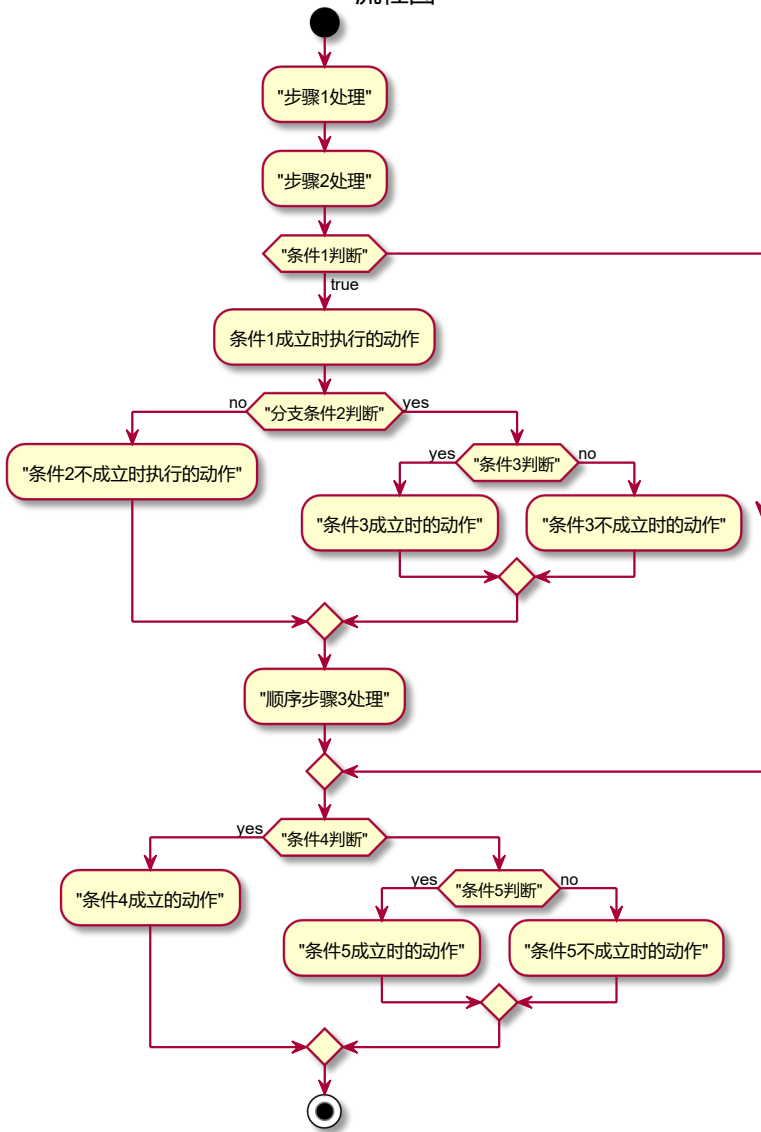


流程图新语法示例

流程图



流程图

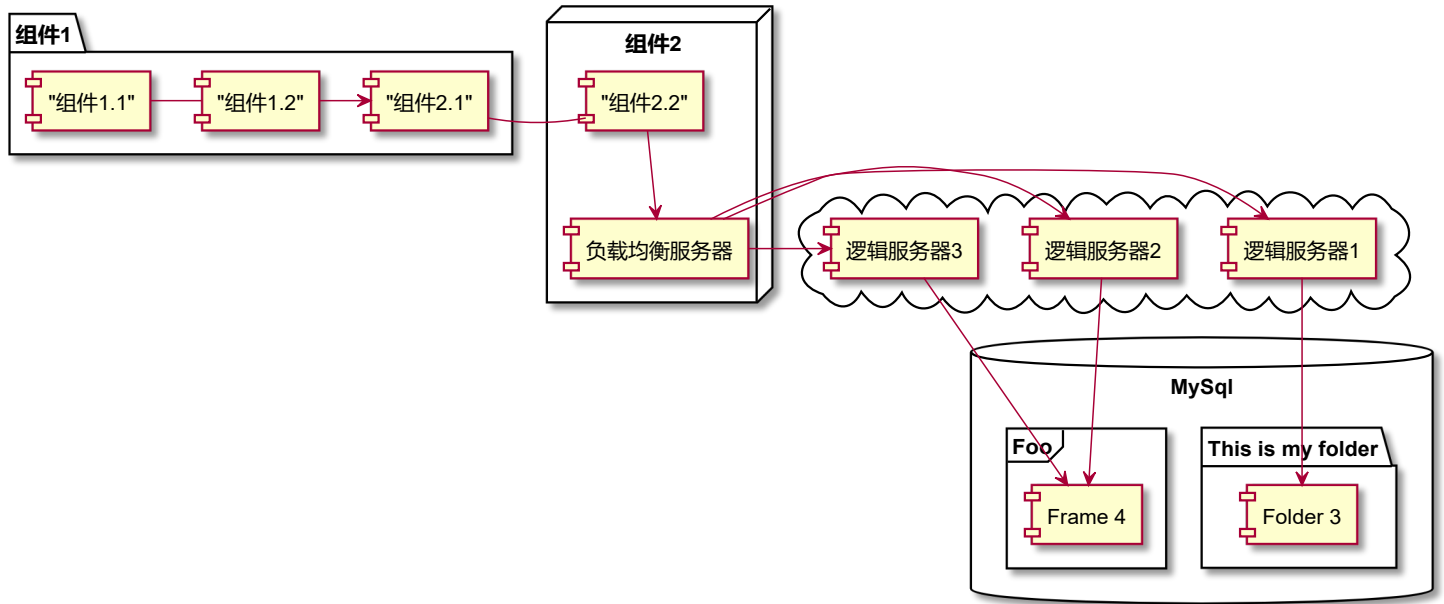


TIPS:

- 使用 start 来表示流程开始，使用 stop 来表示流程结束
- 顺序流程使用冒号和分号 :xxx; 来表示
- 条件语句使用 if ("condition 1") then (true/yes/false/no) 来表示
- 条件语句可以嵌套

组件图示例

组件图



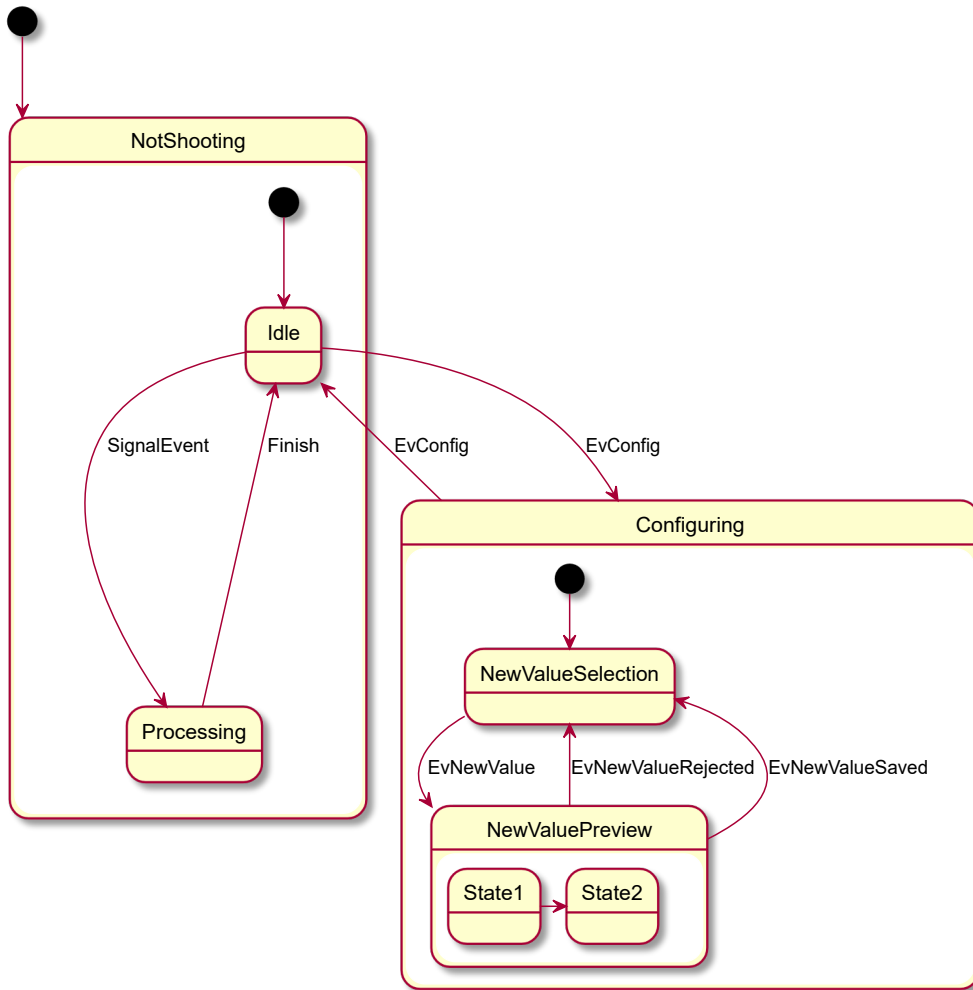
TIPS:

- 使用方括号 [xxx] 来表示组件
- 可以把几个组件合并成一个包，可以使用的关键字为 package, node, folder, frame, cloud, database。不同的关键字图形不一样。
- 可以在包内部用不同的箭头表达同一个包的组件之间的关系
- 可以在包内部直接表达达到另外一个包内部的组件的交互关系
- 可以在流程图外部直接表达包之间或包的组件之间的交互关系

状态图示例

我们一般使用状态图来画状态机

状态图



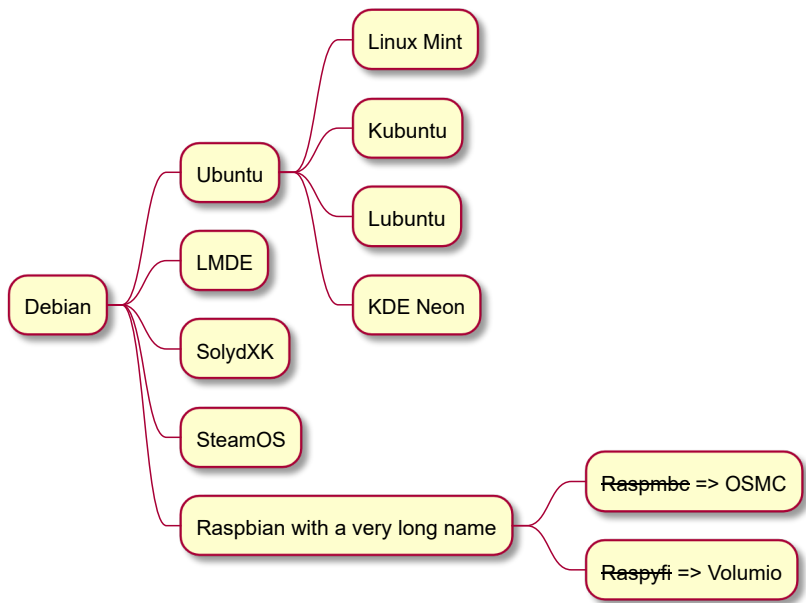
TIPS:

- 使用 [*] 来表示状态的起点
- 使用 state 来定义子状态图
- 状态图可以嵌套
- 使用 scale 命令来指定生成的图片的尺寸

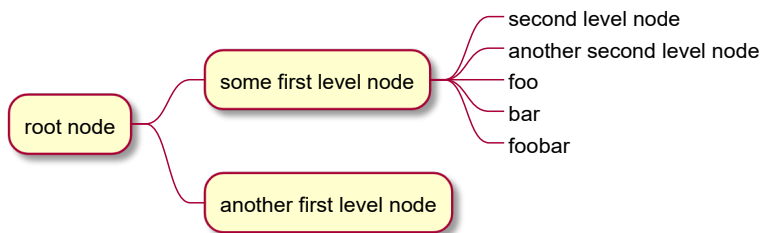
不需要去记这些标记，在需要的时候去使用它，通过不断地使用来熟悉不同的图的语法。可以下载 PlantUML 官方文档 作为参考，遇到问题的时候翻一翻，这样很快就可以学会使用 PlantUML 高效地画图。

PlantUML脑图

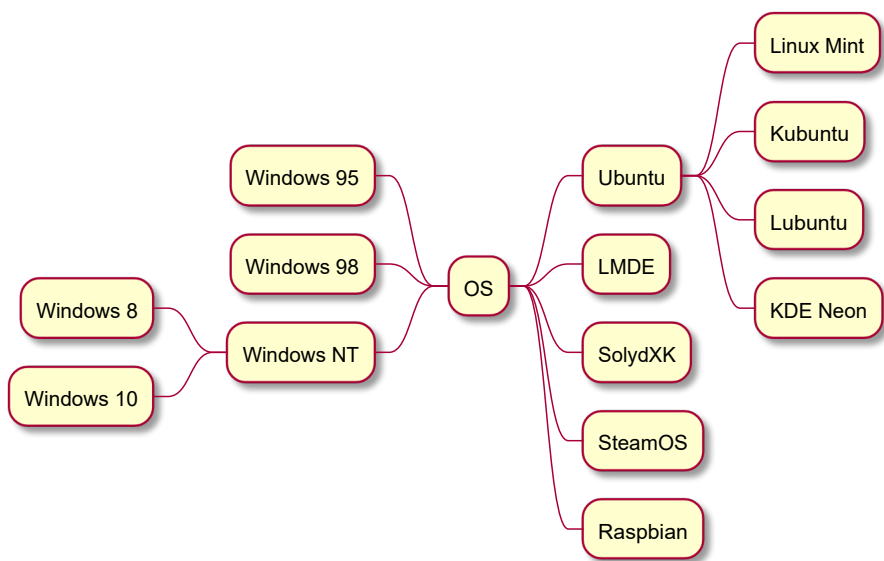
兼容OrgMode语法



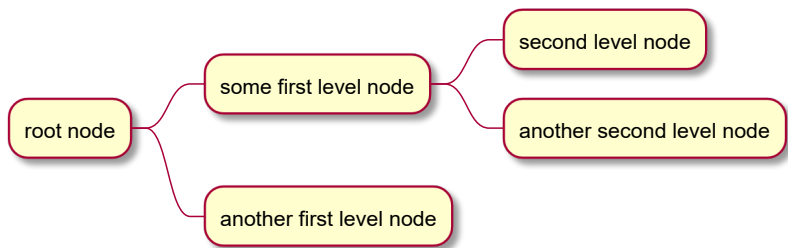
你可以用下划线去除外边框。



你可以使用下面的运算符来决定图形方向。



兼容Markdown语法



你可以同时使用图形的左右两侧

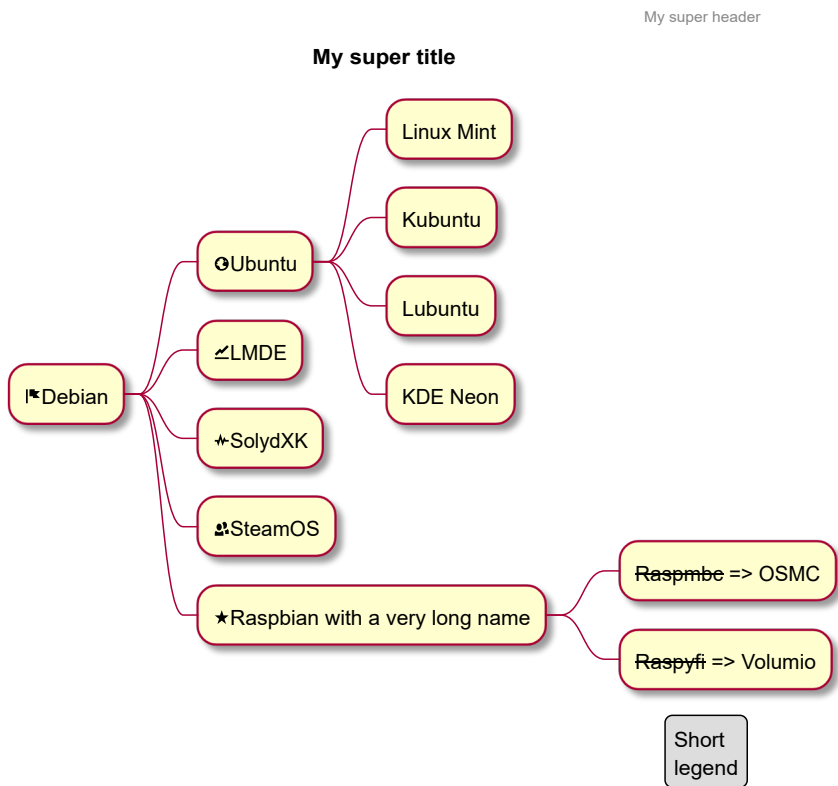
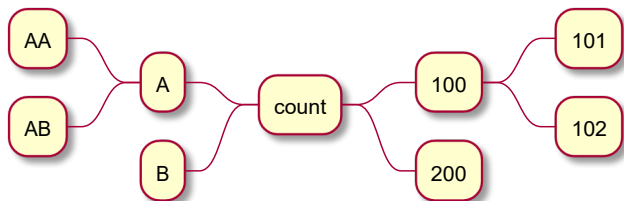
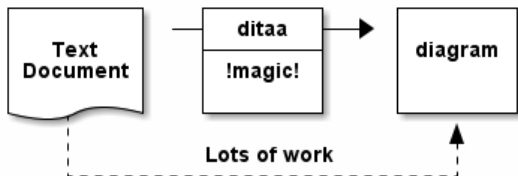


figure 1
My super footer

ditaa示例

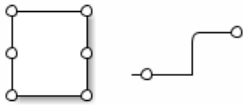


Color codes

RED	BLU
GRE	PNK
BLK	YEL

Things to do

- Cut the grass
- Buy jam
- Fix car
- Make website



1. 头上两个角 ↩
2. demo ↩