

# テストサンプル

K0.1, 2019/08/24

# 目次

1. よく使う文法の紹介 .....	1
1.1. リスト .....	1
1.2. 段落 .....	2
1.3. セクション .....	2
1.4. ブロック .....	2
1.5. コードブロック .....	2
1.6. 脚注 .....	3
1.7. テキストフォーマット .....	3
1.8. マーカー .....	3
1.9. URL .....	3
1.10. コメント .....	3
1.11. 水平罫線・改行 .....	4
1.12. 表示・非表示 .....	5
1.13. 画像 .....	5
1.14. PlantUML .....	5
1.15. テーブル .....	5
1.16. 外部ファイル .....	7

# 1. よく使う文法の紹介

いくつかの文法の利用にはAttribute **:XXX:** の指定が必要です

## 1.1. リスト

先頭に **\*** を付けるとリストになる

- level 1
- level 1
  - level 2
    - level 3
- level 1

先頭に **.** を付けると番号付きリストになる

1. level 1
2. level 1
  - a. level 2
    - i. level 3
3. level 1

ラベル名に続けて **::** を付けるとラベル付きリストになる

### CPU

コンピューターの中心的な処理装置

### RAM

読み書き可能な主記憶装置

### SSD

フラッシュメモリを使用した補助記憶装置

### キーボード

キーを押すことで信号を送信する入力装置

### マウス

コンピューターのポインティングデバイス

### モニター

映像を表示する出力装置

## 1.2. 段落

- 通常の改行は無視
- 空行で別段落
- + で改行できる

例)

ただ改行しただけだと 文章はつながったままです

空行を設けると別段落扱いになります

改行させたいところで + をつければ  
改行できます

## 1.3. セクション

- = でタイトルを示す
- Level0(= が1個)は文章中でひとつしか使えない(ドキュメントタイトル扱い)
- Level1以上(= を2個以上重ねる)で自動的にナンバリングされる

## 1.4. ブロック

- ---- や ==== など囲ってブロックを指定する
- ブロックのヘッダーに .XXX を付けるとタイトルが指定できる

例)

```
y = a × b + c
```

## 1.5. コードブロック

- ソースコードをハイライト表示できる
- ブロックのヘッダーに [source, 使用する言語] の形で指定します

例)C言語のサンプルコード

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
    puts("Hello World!");
    return EXIT_SUCCESS;
}
```

## 1.6. 脚注

- NOTE , TIP , IMPORTANT , CAUTION , WARNING の5種類
- ブロックのヘッダーに [NOTE] の形で指定します



ブロックの中に内容を書きます

## 1.7. テキストフォーマット

- 太字: 文字を \* で囲う
- モノスペース: 文字を ` で囲う

例)

太字の語句と 太字の文字

モノスペースの語句と モノスペースの文字

## 1.8. マーカー

- 蛍光ペン: 文字を # で囲う
- アンダーライン: 文字を # で囲い、頭に [.underline] を付ける
- 取り消し線: 文字を # で囲い、頭に [.line-through] を付ける
- 文字拡張: 文字を # で囲い、[.big] or [.small] を付ける
- 文字色: 文字を # で囲い、[color] を付ける

例)

冬よりかは 夏の方が 嫌い好きだ



PDF化時に文字の色が変わるようにする  
[https://blog.siwa32.com/asciidoctor\\_pdf\\_color/](https://blog.siwa32.com/asciidoctor_pdf_color/)  
→「2.2 asciidoctor-pdfのソースを修正する」

## 1.9. URL

- httpなどを自動で判定してリンクを生成してくれます  
<https://ja.wikipedia.org/wiki/AsciiDoc>
- 別名を指定する際は後ろに [xxx] を付与します  
[ここをクリック](#)

## 1.10. コメント

- // でコメントアウト
- // で囲うと複数行にわたってコメントアウト

## 1.11. 水平罫線・改行

- --- で水平罫線
- <<< で改ページ

## 1.12. 表示・非表示

- `ifdef::xxx[] ~ endif::[]` を使う
- 表示させる場合は、属性 `:xxx:` をAttributeに書く

この文章は外部用なので表示します

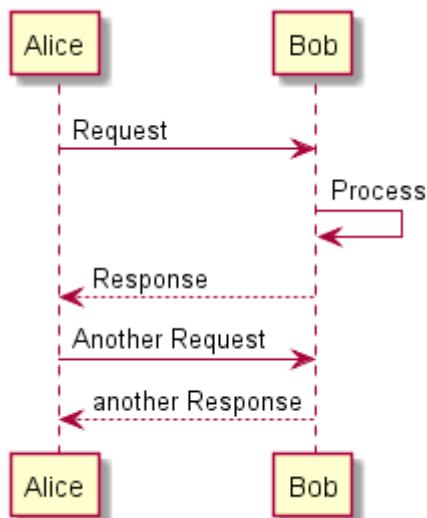
## 1.13. 画像

- `image::` に続けて画像ファイルを指定する
- 後ろに `[width, height]` を指定できる



## 1.14. PlantUML

- テキストからUML図を生成できる
- ブロックのヘッダーに `[plantuml, 任意の画像名]` の形で指定します



- プレビューするたびにpngが作成されてしまうので画像名を指定することをおススメする
- 編集時はAsciiDocのプレビューでなく、PlantUMLのプレビューを使った方が反応が圧倒的に早い  
<https://qiita.com/Ping/items/64930e8c21fb95bec095>  
→「PlantUML Serverリポジトリのclone」～「Visual Studio Code の設定を変更する」  
都度コマンドでサーバーを立ち上げる必要があるが、爆速でプレビューが可能  
`@startuml`～`@enduml`を、編集時は書いてPlantUMLでプレビューできるようにし、  
画像作成時は消してAsciiDocでプレビューしてpngを生成するような使い分けをおススメする

## 1.15. テーブル

- テーブルの区切りは `|===`

- セルの区切りは |

## ヘッダーで指定

cols属性で、文字揃え(align)、セル幅(width)、セル内スタイル(style)の指定が可能  
options属性で、先頭行をヘッダー行扱いにするか指定が可能

**cols="[align][width][style]", options="header"**

- alignの指定は [horizontal][.vertical]
  - horizontal: <(左詰め), ^(センタリング), >(右詰め)
  - vertical: .<(上), .^(中央), .>(下)
- widthの指定は [40,60]
  - %で合計が100になるように指定
- styleの指定は [エイリアス]
  - a: asciidoc記法
  - s: bold表示

## セル単位の指定

セル結合の指定が可能  
文字揃え(align)、セル内スタイル(style)の指定も可能

**[セル結合][align][style]**

- セル結合はセル区切りの頭に指定
  - 行方向: 結合セル数+
  - 列方向: .結合セル数+

header(0,1)	header(0,2)	header(0,3)
cell(1,1)	cell(1,2)	cell(1,3)
cell(2,1)		cell(2,3)
cell(3,1)		cell(3,3)
		cell(4,3)
		cell(5,3)
cell(6,1)	<ul style="list-style-type: none"> <li>• cell(6,2)</li> <li>• cell(6,3)</li> </ul>	
cell(7,1)	cell(7,2)	cell(7,3)

## フォーマットの指定

csv(カンマ区切り)形式で書くことが可能



header(0,1)	header(0,2)	header(0,3)
cell11	cell12	cell13
cell21	cell22	cell23
cell31	cell32	cell33

## 1.16. 外部ファイル

csvファイル (CSV UTF-8) や別のadocファイルをインクルード可能

header(0,1)	header(0,2)	header(0,3)
cell11	cell21	cell31
cell12	cell22	cell32
cell13	cell23	cell33

### 1.16.1. adocファイルのインクルード

章ごとにファイルを分けて管理してもよい