



Small and Easy, but Beautiful Design For You

KiTTY

Kray-G, Mr.Diamond Global Blue Publisher
September 18, 2020

目次

第 1 章 イン트로ダクション

1.1	KiTTy とは	1
1.1.1	Versus L ^A T _E X	1
1.1.2	Versus Word	1
1.1.3	Versus Vivliostyle	2
1.1.4	結論	2
1.2	サポート機能	3
1.2.1	組版機能	3
1.2.2	日本語用組版機能	3
1.2.3	PDF 機能	3

第 2 章 さあ始めよう

2.1	インストール	5
2.2	ビルド	5
2.3	hello, world	5

第 3 章 機能概要

3.1	組版機能	7
3.1.1	ハイフネーション・ジャスティフィケーション・行分割	7
3.1.2	ウィドウ／オーファン	7
3.1.3	箇条書き	8
3.1.4	数式	9
3.1.5	イメージ	10
3.1.6	グラフ（チャート）	12
3.1.7	テーブル	14
3.1.8	フォント	16
3.1.9	色	18
3.1.10	合字・特殊文字	18
3.1.11	プログラム・コード	18
3.1.12	相互参照	18
3.1.13	目次	18
3.1.14	タイトル・カバーページ	18
3.1.15	脚注	18
3.2	日本語用組版機能	18
3.2.1	日本語禁則処理	18
3.2.2	日本語ルビ	18

3.3 PDF 機能	18
3.3.1 外部リンク	18
3.3.2 相互参照リンク	18
3.3.3 しおり	18

第 4 章 コマンド

4.1 Markdown 処理コマンド	19
4.1.1 サポート Markdown コマンド	19
4.1.2 HTML コマンド	19
4.2 KiTTY コマンド	19
4.2.1 パラグラフ処理コマンド	19
4.2.2 単コマンド	19

第 5 章 機能拡張

5.1 スタイル	21
5.1.1 スタイルの追加	21
5.1.2 カバーページ・スタイル	21
5.1.3 チャプター・スタイル	21
5.2 禁則処理	21

図目次

図 3.1 ハイフネーション・両端揃え	7
図 3.2 イトトンボ	11
図 3.3 F14 Tomcats	11
図 3.4 Radar Chart Example	13
図 3.5 Line Chart Example	13

表目次

表 3.1 イメージ・オプション	11
表 3.2 テーブル記述の例	14
表 3.3 テーブル・オプション	15
表 3.4 ボールド、イタリック、ボールドイタリックの書き方	16
表 3.5 フォントサイズに指定可能な単位一覧	17

第 1 章

イントロダクション

1.1 KiTTY とは

KiTTY は Kinx Tiny Typesetting を意味し、Kinx で実装された簡易組版システムの名称です。Markdown 形式からの簡易トランスレーターを実装しているため、Markdown 形式で書かれたドキュメントを美しく組版することができます。本文書自体も Markdown で記載されているものを自動組版した一つの事例です。

考え方は L^AT_EX に近く、テキスト形式で管理している文書ファイルを美しく組版することを目的としています。より具体的には、本システムは L^AT_EX を置き換えることを目的とはしていませんが、以下を実現することによって、より個人的な利用シーンの中で、より簡単に利用できるようにすることを目的としています。

- 小さなシステムを維持すること
- それなりに美しく組版できること
- 直接 PDF ファイルを出力できること

KiTTY は小さなシステムながらある程度美しく組版できる機能を持ち、Markdown で書かれた文書から直接 PDF ファイルとして出力することができる組版システムです。

1.1.1 Versus L^AT_EX

L^AT_EX は巨大なシステムです。拡張性にも優れ、多くの人々に支えられた美しい文書を作成するための組版システムです。KiTTY も L^AT_EX と同じ目的を持つ組版システムですが、限られた機能しか提供しない代わりに小さなシステムとして提供されます。

L^AT_EX の巨大さは、インストールの複雑さにもつながります。T_EX、L^AT_EX では様々な機能を提供するためにディストリビューションそのものが複数存在しています。それにより、ユーザーはまずどのディストリビューションを使うべきかで悩むことになります。KiTTY は、Kinx パッケージに標準で組み込まれており、Kinx をインストールすることですぐに使えるようになります。

ただし、小さく、そして簡単に使える代わりにトレードオフとして限られた機能¹しか提供されないといった欠点があります。また、組版スピードは非常に遅いです。パフォーマンスの改善は 1 つの課題ですが、小さなプロジェクトで個人的に利用することにフォーカスしています。

1.1.2 Versus Word

WYSIWYG² のワードプロセッサとして代表的な Word ですが、考え方が異なります。WYSIWYG では見たままの形式で編集可能ですが、その見た目情報を一緒に保存するために多くの場合バイナリ形式で保存されます。そのため、中に何が書かれているか知るには一般的に専用の

¹ 「限られた機能」に関しては、「第 3 章 機能概要」を参照してください。

² What You See Is What You Get の頭文字をとったもの。見たままのものを実際に作成出力するという意味。

ソフトウェア（この場合 Word）が必要となります。KiTTY は L^AT_EX 同様、テキストエディタさえあれば内容を知ることができ、編集することも可能です。

テキストで保存されるということは、別のソフトウェアで処理することも簡単であり、Git のようなバージョン管理システム上で差分を確認することも容易です。このことは特に、差分管理をバージョン管理システム上で実現したい場合には必須となる特徴です。

また、文書構造に関しても、Word では直接的にその見た目から「それが構造化されたもの」か「単に見た目としてそう見えているものか」の区別が付きません。例えば、章番号がきちんと章番号として設定され、文章の配置やレイアウトを変更した際に正しく番号を付け直してくれるよう配慮されているかどうか判別しづらいといった欠点もあります。特に、どのような書き方をしたとしても「見た目として正しく見えてしまっている」ということにより、他者の作成したファイルを編集する際に意図せず正しく設定してくれていなかった、といった不運もたびたび見られます。

KiTTY では、文書構造をテキストで表現する関係上、章やセクション、図、表などのリファレンスを常に正しく把握し、適切な番号付け、および相互参照機能を実現することができます。

その代わり、WYSIWYG のようにその場で出力後の体裁（見た目）を確認することができない、といった欠点があります。

1.1.3 Versus Vivliostyle

CSS 組版は Web 技術に基づいています。CSS 組版は今現在最も将来性のある取り組みでしょう。したがって、本格的に取り組むには CSS 組版がおすすめです。その中でも Vivliostyle³ は非常に有望なプロジェクトです。

KiTTY ではやはり「小さなシステム」という部分に価値を置いています。手軽に扱えることが重要です。時間が解決することではありますが、CSS 組版は現時点で標準化の推進と並行して活動が行われています。したがって仕様が未確定な部分も多く、今後仕様変更なども多く行われることが予想されます。また、大規模に標準化等を進めていますので、商用にも耐えうるシステムとなる一方でシステム自体は巨大になるでしょう。

KiTTY は小さく手軽に扱えることを一番のポイントとしています。

1.1.4 結論

要約すると、以下のケースにおいて KiTTY は有益でしょう。

- T_EX のような巨大なシステムではなく、小さな組版システムで簡単に利用したい
- Git のようなバージョン管理システムを使った差分管理をしたい
- 文書構造を常に適切に把握し、相互参照などを正確に実施したい
- サポートされている機能が十分であり、組版スピードも我慢できる

ちょっとした文書作成のために T_EX をフルセットで使うには巨大すぎる、と感じている方で、テキストで文書管理をしたい、と考えている方⁴のために本システムを作成しました。特に、Git で差分を含めた文書管理を行いたい場合、WYSIWYG で実現されているワープロソフトでの管理は

³ <https://vivliostyle.org/>

⁴ つまり、私のような方。実は、自分のオープンソース・プロジェクトの簡易マニュアル作成のために作りました。

大変困難です。主に、ワープロソフトでは管理したくないけれど高機能な組版ソフトは大がかりすぎる、といった利用シーンを想定しています。

1.2 サポート機能

1.2.1 組版機能

KiTTy は組版機能として、以下の機能をサポートしています。なお、カーニングは現在サポートされていません。

- ・ ハイフネーション・ジャスティフィケーション・行分割
- ・ ウィドウ／オーファン
- ・ 箇条書き
- ・ 数式
- ・ イメージ
- ・ グラフ（チャート）
- ・ テーブル
- ・ フォント
- ・ 色
- ・ 合字・特殊文字
- ・ プログラム・コード
- ・ 相互参照
- ・ 目次
- ・ タイトル・カバーページ
- ・ 脚注

1.2.2 日本語用組版機能

基本的な組版機能に加え、以下の日本語特有の処理が組み込まれています。日本語以外の言語への拡張は私自身に知見が乏しく言語ごとの固有の拡張ポイントを意識していないため、大幅な修正、もしくは機能追加が必要かもしれません。ただし、ソースコードは公開されているので必要に応じて拡張することは可能でしょう。

- ・ 日本語禁則処理
- ・ 日本語ルビ

1.2.3 PDF 機能

印刷した際には表面に現れてきませんが、以下の機能を PDF 機能としてサポートしています。

- ・ 外部リンク
- ・ 相互参照リンク
- ・ しおり

第 2 章

さあ始めよう

2.1 インストール

インストールは以下の 2 ステップを実施します。

1. Kinx のインストール
2. KiTTY 追加モジュールのインストール

2.2 ビルド

通常、ビルドから実施する必要はありません。既にインストーラが提供されており、手順にしたがってインストールを実施することで本システムを利用することができます。あえてビルドから実行したい、といった場合は以下の手順によってビルドを実施できます。

2.3 hello, world

次の文書を作成し、helloworld.md ファイルとして保存します。

```
hello, world
```


第 3 章

機能概要

3.1 組版機能

3.1.1 ハイフネーション・ジャスティフィケーション・行分割

Franklin M. Liang のアルゴリズムに基づくハイフネーションをサポートしています。また、ハイフネーションに伴うジャスティフィケーション（両端揃え）機能をサポートしています。

This Kinx TT has supported some kind of TeX algorithms, so the final output would be very beautiful. You can check it on your eyes yourself as this document was generated by this system. On the other hand, there are some bad points below as a trade off.

行分割は Knuth-Plass Line Breaking アルゴリズムを採用しています。本アルゴリズムは、Box、Glue、Penalty によって分割位置をコント

図 3.1 ハイフネーション・両端揃え

ロールするアルゴリズムであり、TeX で実装されているアルゴリズムと同様です。これらハイフネーション・アルゴリズムも行分割アルゴリズムも、今のところ組版システムでは最良の方法として知られている方法です。ただし、実装自体は Kinx で改めて行われているため、必ずしも出力結果は TeX での出力と同一にはならない場合があります。

3.1.2 ウィドウ／オーファン

ウィドウ、およびオーファンに対するペナルティ処理を一部ですが実施します。全てのケースで有効ではありませんのでご注意ください。具体的には以下のケースで有効です。

- ・セクション名がページの最後に取り残されるケースを抑止。
 - この場合、セクション名ごと次のページに追い出されます。
- ・複数行パラグラフにおいて、最後の行のみ次のページに送られるケースを抑止。
 - この場合、最後の 2 行分が次のページに追い出されます。
- ・複数行パラグラフにおいて、最初の行のみ前のページに残るケースを抑止。
 - この場合、全ての行が次のページに追い出されます。
 - この処理の結果としてセクションが残る場合、セクション自体も次のページに追い出されます。

これらの処理は自動的に行われます。特に文書内に指示を記載する必要はありません。ただし、全てのケースで正しく動作をする訳ではありませんので、うまくレイアウトされない場合は必要に応じて `<pagebreak/>` コマンドを使って改ページを行ってください。

3.1.3 箇条書き

箇条書きは記号によるものと番号付きのものが利用できます。次の例は記号による箇条書きの例です。

```
1 * レベル1
2   * レベル2
3     * レベル3
4       * レベル4
```

これは以下のように整形されます。

- レベル1
 - レベル2
 - レベル3
 - * レベル4

次の例は番号付き箇条書きの例です。数値ラベルは自動的に補正されます。

```
1 1. レベル1
2   1. レベル2
3     1. レベル3
4       1. レベル4
5       1. レベル4
```

これは以下のように整形されます。

1. レベル1
 - (a) レベル2
 - i. レベル3
 - A. レベル4
 - B. レベル4

また、両者を混在させることも可能です。次の例は混在させた場合の箇条書きの例です。

```
1 * レベル1
2   1. レベル2
3     * レベル3
4       1. レベル4
```

これは以下のように整形されます。

- レベル1
 - (a) レベル2
 - レベル3
 - A. レベル4

3.1.4 数式

KiTTy は K_AT_EX を内蔵しており、数式を表現することも可能です。数式はスタンドアロン形式とインライン形式の 2 つの表現方法があります。

3.1.4.1 スタンドアロン形式

スタンドアロン形式はコードブロックの形式で記載し、1 行で表現されます。その際、言語として `math` を指定します。

```
1 ``math:label=Math1
2 \begin{aligned}
3   \int_{-\infty}^{\infty} f(x) dx &= \sqrt{\pi}
4 \end{aligned}
5 ``
```

上記のように記載すると、以下のように表現されます。label オプションは付けなくても問題ありませんが、ラベルを付けておくことで数式 1 のように数式への参照を行うことが可能です。

$$\int_{-\infty}^{\infty} f(x) dx = \sqrt{\pi} \quad (1)$$

ただし、K_AT_EX はラベル機能を持っていないため、ラベル機能は KiTTy によって実現されています。したがって、2 つの式を表現する場合にはラベルを自分自身でコントロールする必要があります。

```
1 ``math:label=Math2(0.2)/Math3(0.6)
2 \begin{aligned}
3   E &= mc^2 \\
4   m &= \frac{m_0}{\sqrt{1-\frac{v^2}{c^2}}}
5 \end{aligned}
6 ``
```

上記のようにすることで、ラベルの配置位置を数式の上端からそれぞれ 20%、60% の位置に Math2、Math3 のラベルを配置します。

$$E = mc^2 \quad (2)$$

$$m = \frac{m_0}{\sqrt{1 - \frac{v^2}{c^2}}} \quad (3)$$

これによって、`\ref{Math2}` と記載することで数式 2 への参照を、`\ref{Math3}` と記載することで数式 3 への参照を作成することが可能となります。

3.1.4.2 インライン形式

インラインで数式を扱う場合は \$ で囲みます。例えば、 $E = mc^2$ と記載すると、 $E = mc^2$ と表現されます。また、インテグラルなどの高さのある表記をインラインで記載すると、例えば 1 と同じ $\int_{-\infty}^{\infty} f(x) dx = \sqrt{\pi}$ を記載すると、 $\int_{-\infty}^{\infty} f(x) dx = \sqrt{\pi}$ と表現されます。なお、\$ で括った中では Markdown の記法と重なるため、\ や _ を \ でエスケープする必要があることにご注意ください。

仮に大きな形式で表現したい場合は `\displaystyle` を先頭につけて記載します。例えば、 $\int_{-\infty}^{\infty} f(x) dx = \sqrt{\pi}$ と `\displaystyle` を付けて記載すると、 $\int_{-\infty}^{\infty} f(x) dx = \sqrt{\pi}$ と表現されます。ただし、行の高さが揃わないためあまりお勧めするものではありません。

3.1.5 イメージ

イメージは Markdown のイメージ形式で記載しますが、alt 部分にオプションを指定し、`![options](path)` の形で記載します。スタンドアロン形式での挿入、インラインでの図の挿入、およびテキストを周りに配置する形でのフローティング形式で挿入することが可能です。

3.1.5.1 スタンドアロン・イメージ

全幅で表示させるには前後を空行の形にし、独立したパラグラフで記載します。

```
![scale=0.6](kinxlogo.png)
```


上記のように記載すると以下のように図が挿入されます。scale=0.6 の指定により版面の横幅の 60% の大きさに補正されて表示されます。また、縦横の比率は維持されます。



3.1.5.2 インライン・イメージ

インラインの例です。インラインで図を挿入する場合、文中に直接以下のように書きます。

```
ファイルアイコンは ![scale=0.08,offsetY=-5.0](zip256.png) になります。
```

この場合、「ファイルアイコンは  になります。」と表現されます。図の元のサイズに応じて scale と offsetY を適宜調整してください。

3.1.5.3 フローティング・イメージ

イメージをフローティングさせるには、オプションに `float=left` または `float=right` を指定します。p.7 に示す「[図 3.1 ハイフネーション・両端揃え](#)」の図はフローティング・イメージの一例です。



図 3.2 イトトンボ

イメージの下までパラグラフの文章が到達した場合、自動的にテキスト幅が版面の幅に戻り、自然な形でテキストが配置されます。

また、右のイメージのようにパラグラフの右側に配置することも可能です。イメージは上記と同様 Public Domain のものを使わせていただいています。

1 つ注意点としては、パラグラフの先頭とイメージの上端の位置を合わせる必要があります。パラグラフの途中にフローティングさせることはできません。あるパラグラフを開始する際にフローティングすべきイメージがあれば、パラグラフの左右どちらか指定した場所にイメージを配置します。



図 3.3 F14 Tomcats

3.1.5.4 オプション

オプションは以下のものを使用できます。

表 3.1 イメージ・オプション

オプション	値	意味
<code>float</code>	<code>left, right</code>	フローティング位置
<code>scale</code>	<code>0.0 ~ 1.0</code>	実数、版面の幅に対する拡大率
<code>caption</code>	キャプション	図のキャプション

⁵ <https://free-images.com/>

⁶ この 70% にはイメージとテキストの間の余白を含みます。

3.1.6 グラフ（チャート）

グラフ（チャート）も挿入可能です。グラフもスタンドアロン形式とフローティング形式の両形式をサポートしています。

3.1.6.1 スタンドアロン・グラフ

スタンドアロンでグラフを表示するには、コードブロックで **chart** を指定します。例えば以下のように JSON データで記載します。width や height などのグラフの情報に加え、Chart.js⁷ のデータそのものを options フィールドに記載します。

```
1  ``chart
2  {
3      width: 800,
4      height: 400,
5      fontSize: 16,
6      scale: 1.0,
7      caption: "Radar Chart Example",
8      options: {
9          type: "radar",
10         data: {
11             labels: ["Eating", "Dinner"], ["Drinking", "Water"],
12                  "Sleeping", ["Designing", "Graphics"],
13                  "Coding", "Cycling", "Running"],
14             datasets: [{
15                 label: "My First dataset",
16                 backgroundColor: "rgba(255, 0, 0, 0.2)",
17                 borderColor: "red",
18                 pointBackgroundColor: "red",
19                 data: [ 10.1, 80.0, 72.2, 73.3, 55.0, 68.5, 92.0 ]
20             }, {
21                 label: "My Second dataset",
22                 backgroundColor: "rgba(0, 0, 255, 0.2)",
23                 borderColor: "blue",
24                 pointBackgroundColor: "blue",
25                 data: [ 30.9, 77.1, 49.9, 50.0, 67.8, 71.0, 22.8 ]
26             }]
27         },
28         options: {
29             legend: {
30                 position: "top",
31             },
32             scale: {
33                 ticks: {
34                     beginAtZero: true
35                 }
36             }
37         }
38     }
39 }
40 ...
```

これは図 3.4 のように出力されます。

⁷ <https://www.chartjs.org/>

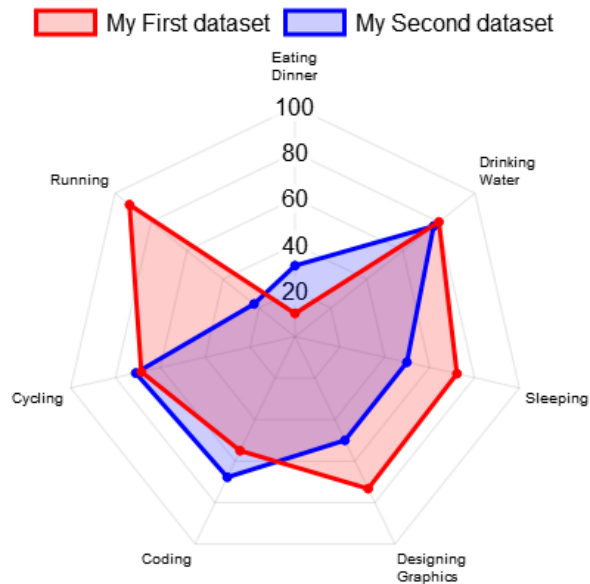


図 3.4 Radar Chart Example

3.1.6.2 フローティング・グラフ

グラフはイメージの配置と同様、テキストの中にフローティングさせることも可能です。

オプションで `float: { right: true }` といった形で指定すると、その後に続くパラグラフに対してフローティングさせることができます。

ここでは折れ線グラフを記載しています。scale はイメージと同様、版面の横幅に対するスケールを表します。スタンドアロン形式、フローティング形式いずれの場合でも、グラフにもキャプションを付けることが可能で、図として挿入されます。目次を出力する場合、図目次にも反映されます。また、イメージをフローティング形式で配置させた場合と同様に、パラグラフはグラフの下側に自然な形で自動的に取り囲むように配置されます。

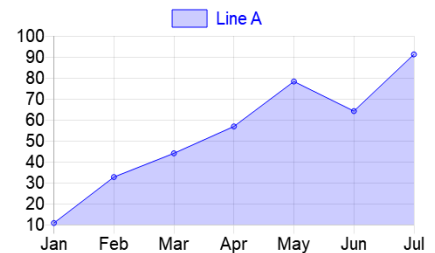


図 3.5 Line Chart Example

上記は以下のように記載されます。なお、ここでは紙面の都合上、省略しますが、options は Chart.js のオプションです。

```

1  ``chart
2  {
3      float: { right: true },
4      width: 480, height: 300, scale: 0.4, caption: "Line Chart Example",
5      options: {
6          type: "line",
7          ...(省略)
8      }
9  }
10 ``

```

3.1.7 テーブル

3.1.7.1 Markdown テーブル

テーブルもサポートします。通常の Markdown の形式で表を記載することにより、自動的にテーブル出力します。

例えば、先ほどの「表 3.1 イメージ・オプション」のテーブルは次のように記載しています。直接 Markdown の表形式で表現できないオプションは、<context /> タグで指定します。

```
1 <context label="Table:ImageOptions"/>
2 <context caption="イメージ・オプション"/>
3 | オプション | 値 | 意味 |
4 | ----- | ----- | ----- |
5 | `float` | `left`, `right` | フローティング位置 |
6 | `scale` | 0.0 ~ 1.0 | 実数、版面の幅に対する拡大率 |
7 | `caption` | キャプション | 図のキャプション |
```

通常の Markdown テーブルのように記載することで、右寄せ、中寄せ等も可能です。また、数式を含めることも可能です。セルの内容が長くなりすぎる場合、<context cell-i-j="..." /> の形で <context /> タグに追い出すことも可能です。この時、 (i, j) は Body 部分（ヘッダは含まない）の左上を $(0, 0)$ として記載します。

```
1 <context label="Table:TableExample"/>
2 <context caption="テーブル記述の例"/>
3 <context vline-left="single"/>
4 <context vline-right="single"/>
5 <context vline-inside="single"/>
6 <context hline-header="double"/>
7 <context hline-inside="single"/>
8 <context cell-2-1="\displaystyle\int_{-\infty}^{\infty} f(x) dx = \sqrt{\pi}" />
9 | 左寄せ | 中寄せ | 右寄せ |
10 | :----- | :-----: | -----: |
11 | A1 | Aligned to the center. | Aligned to the right. |
12 | A2 | Cell $(1,1)$ | Cell $(1,2)$ |
13 | A3 | - | Cell $(2,2)$ |
```

上記は以下のように出力されます。

表 3.2 テーブル記述の例

左寄せ	中寄せ	右寄せ
A1	Aligned to the center.	Aligned to the right.
A2	Cell (1,1)	Cell (1,2)
A3	$\int_{-\infty}^{\infty} f(x)dx = \sqrt{\pi}$	Cell (2,2)

3.1.7.2 オプション

Markdown テーブル表記で表現できなかったパラメータは、`<context />` タグで指定します。指定できる項目の一覧を「表 3.3 テーブル・オプション」に示します。これらの値は一時的に利用され、参照された後自動的に削除されます。したがって、複数のテーブルで共有して使用できませんので、テーブルごとに設定します。

ただし、`label`、`caption` 以外はデフォルト値を変更できます。その場合、項目名に `-default` を指定して設定してください。例えば、`vline-left-default` とします。

表 3.3 テーブル・オプション

オプション	値	意味
<code>label</code>	相互参照ラベル	相互参照で指定するラベルを設定する。
<code>caption</code>	キャプション	表のキャプション。
<code>vline-left</code>	<code>single</code> , <code>double</code> , <code>false</code>	表の左側に縦罫線を出力するかの指定をする（デフォルト <code>false</code> ）。
<code>vline-right</code>	同上	表の右側に縦罫線を出力するかの指定をする（デフォルト <code>false</code> ）。
<code>vline-inside</code>	同上	表中に縦罫線を出力するかの指定をする（デフォルト <code>false</code> ）。
<code>hline-top</code>	同上	表の上側に横罫線を出力するかの指定をする（デフォルト <code>single</code> ）。
<code>hline-bottom</code>	同上	表の下側に横罫線を出力するかの指定をする（デフォルト <code>single</code> ）。
<code>hline-header</code>	同上	表のヘッダ行の下側に横罫線を出力するかの指定をする（デフォルト <code>single</code> ）。
<code>hline-inside</code>	同上	表中に横罫線を出力するかの指定をする（デフォルト <code>false</code> ）。
<code>cell-i-j</code>	テキスト	セル内容を別定義する。 <code>i</code> 、 <code>j</code> はセル位置で、 <code>i</code> が行、 <code>j</code> が列を示す。それぞれ 0 始まりで指定する。
<code>limit-column</code>	整数	セルサイズの最小幅を指定された列が折り返さない幅とする。
<code>limit-width</code>	実数	セルサイズの最小幅を指定された値とする。

3.1.8 フォント

3.1.8.1 ボールド、イタリック、ボールドイタリック

Bold、*Italic*、***BoldItalic*** は通常の Markdown と同様に記述できます。以下のように記述します。なお、日本語にイタリック体はありませんのでご注意ください。

表 3.4 ボールド、イタリック、ボールドイタリックの書き方

Markdown	出力	意味
Bold	Bold	ボールド体で表現する。
<i>*Italic*</i>	<i>Italic</i>	イタリック体で表現する。
<i>***BoldItalic***</i>	<i>BoldItalic</i>	ボールドかつイタリック体で表現する。

3.1.8.2 フォントの利用

デフォルトで用意されていないフォント・ファイルは明示的にロードして利用します。ロードの仕方は以下の通りです。カンマ区切りで以下の 4 つのパラメータを指定します。

```
1 <font-load info="Name,type,shape,FileName.ttf" />
```

1 度ロードすれば、それ以降の文章中に Name を使用して自由に利用できます。各パラメータの意味は以下の通りです。

- Name は識別子として任意の名前を付けられます。
- type はフォントのタイプを表し、`serif`、`sans`、`monotype` の中から選択します。
- shape はフォントの形を表し、`regular`、`bold`、`italic`、`bolditalic` の中から選択します。
 - `regular` は通常の文章で使用されます。
 - `bold`、`italic`、`bolditalic` はそれぞれボールド、イタリック、ボールドイタリックに対応します。

利用する場合は、以下の例のように `\font` コマンドで指定して利用します。なお、コマンド名と “[” および “]” をエスケープする必要がありますのでご注意ください。以下の例は `Parisienne-Regular.ttf` フォント⁸を利用して出力する例です。

```
1 <font-load info="Parisienne,serif,regular,Parisienne-Regular.ttf" />
2 Changing the font is available only with a scope like
3 '\font\[name=Parisienne\]{This is a pen.},'
4 and the font will be restored here.
```

Changing the font is available only with a scope like “*This is a pen.*,” and the font will be restored here.

上記例の通り、`\font` コマンドで囲まれたスコープ内でのみ有効です。

⁸ KiTTY のパッケージに含まれてはいますが、デフォルトでロードされません。このように明示的にロードして利用します。

3.1.8.3 フォント・サイズ（直接指定）

フォント・サイズは `\font` コマンドの `size` パラメータを指定します。

```
1 「この後フォントが 7pt に \\font\[size=7pt\]{縮小} します。
2 また、この後フォントが 15pt に \\font\[size=15pt\]{拡大} します。」
3 と表現されます。
4 また、例えば \\font[size=1.2em]{サイズ 1.2 倍} と記載すると、
5 \\font[size=1.2em]{サイズ 1.2 倍} と表現されます。
```

「この後フォントが 7pt に 縮小 します。また、この後フォントが 15pt に 拡大 します。」と表現されます。また、例えば `\font[size=1.2em]{サイズ 1.2 倍}` と記載すると、サイズ 1.2 倍 と表現されます。

サイズは上記の例の通り単位を指定することができます。指定できる単位は以下の通りです。

表 3.5 フォントサイズに指定可能な単位一覧

単位	意味
em	現在のフォントサイズを基準（1.0）とした相対サイズで指定します。
ex	現在のフォントの小文字の高さを基準とした相対サイズで指定します。
px	サイズをピクセルで指定します。
pc	サイズをパイカ（1pc = 1/6インチ）で指定します。
mm	サイズをミリメートルで指定します。
cm	サイズをセンチメートルで指定します。
in	サイズをインチで指定します。

3.1.8.4 フォント・サイズ（相対指定）

相対的に指定するには `\bigger`、`\smaller` を使います。それぞれ、指定したスコープ内のフォントサイズが +1 または -1 されます。単位はポイント（pt）です。

```
1 次の文章は \\bigger の例です。
2 \\bigger{これが「\\bigger{これが「\\bigger{これが「文章」です}」です}」です、
3 となります。
4 また、\\smaller は、
5 \\smaller{これが「\\smaller{これが「\\smaller{これが「文章」です}」です}」です、
6 となります。
```

次の文章は `\bigger` の例です。これが「これが「これが「文章」です」です」です、となります。また、`\smaller` は、これが「これが「これが「文章」です」です」です、となります。

3.1.9 色

3.1.9.1 文字色

文字の色を変えるには、以下のようにします。

```
1  \\color\[options\]{This is a pen.}
```

3.1.10 合字・特殊文字

3.1.11 プログラム・コード

3.1.12 相互参照

3.1.13 目次

3.1.14 タイトル・カバーページ

3.1.15 脚注

3.2 日本語用組版機能

3.2.1 日本語禁則処理

3.2.2 日本語ルビ

3.3 PDF 機能

3.3.1 外部リンク

3.3.2 相互参照リンク

3.3.3 しおり

第 4 章

コマンド

4.1 Markdown 処理コマンド

4.1.1 サポート Markdown コマンド

4.1.2 HTML コマンド

4.2 KiTTY コマンド

4.2.1 パラグラフ処理コマンド

4.2.2 単コマンド

第 5 章

機能拡張

5.1 スタイル

5.1.1 スタイルの追加

5.1.2 カバーページ・スタイル

5.1.3 チャプター・スタイル

5.2 禁則処理