



小さくて簡単、でも美しいデザイン。

KiTTY

Beautiful Design for you ...

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

目次

第 1 章	イントロダクション	1
1.1	KiTTy とは	1
1.1.1	Versus L ^A T _E X	1
1.1.2	Versus Word	2
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	4
第 2 章	さあ始めよう	5
2.1	インストール	5
2.1.1	Kinx のインストール	5
2.1.2	Kinx Tiny Typwsetting パッケージのインストール	6
2.2	hello, world	6
第 3 章	機能概要	7
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	マルチカラム	8
3.1.4	箇条書き	9
3.1.5	数式	10
3.1.6	イメージ	11
3.1.7	グラフ（チャート）	13
3.1.8	テーブル	15
3.1.9	フォント	17
3.1.10	色	19
3.1.11	合字・特殊文字	19
3.1.12	プログラム・コード	20
3.1.13	タイトル（カバーページ）・目次	23
3.1.14	見出し	25
3.1.15	相互参照	25
3.1.16	引用	25
3.1.17	脚注	26
3.2	日本語用組版機能	26

3.2.1	日本語禁則処理	26
3.2.2	日本語ルビ（振り仮名）	27
3.3	PDF 機能	28
3.3.1	外部リンク	28
3.3.2	相互参照リンク	28
3.3.3	しおり	28
第 4 章	コマンド詳細	29
4.1	Markdown コマンド	29
4.1.1	パラグラフ・コマンド	29
4.1.2	インライン・コマンド	30
4.1.3	HTML コマンド	30
4.2	KiTTy コマンド	31
4.2.1	パラグラフ処理コマンド	31
4.2.2	単独処理コマンド	31
第 5 章	機能拡張方法	33
5.1	各種スタイル定義の追加	33
5.1.1	文書スタイルの追加	33
5.1.2	タイトル・スタイルの追加	34
5.1.3	チャプター・スタイルの追加	34
5.2	禁則処理	35
5.2.1	禁則処理の追加	35
5.3	フォント	37
5.3.1	新規フォントの追加	37
5.3.2	OS 組込みフォントの追加	37
5.4	コマンド	37
5.4.1	単独処理コマンドの定義	38
5.4.2	パラグラフ処理コマンドの定義	38
付録 A	プレ定義デザイン	41
A.1	タイトル・デザイン	41
A.1.1	StandardArticle	41
A.1.2	StandardBook	41
A.2	チャプター・デザイン	42
A.2.1	StandardBook	42
A.2.2	BigChapter1	42
A.2.3	BigChapter2	42

A.2.4 BigChapter3	42
付録 B スタイル	43
B.1 スタイル・パラメーター一覧	43
付録 C 色一覧	45
C.1 色名称および RGB/CMYK 対応表	45

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	12
図 3.3 F14 Tomcats	12
図 3.4 Radar Chart Example	14
図 3.5 Line Chart Example	14

表目次

表 3.1 イメージ・オプション	12
表 3.2 テーブル記述の例	15
表 3.3 テーブル・オプション	16
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	17
表 3.5 フォントサイズに指定可能な単位一覧	18
表 3.6 言語指定で指定できる値の一覧	21
表 3.7 box オプションで指定できる値の一覧	22
表 3.8 コード・スタイル値の一覧	23
表 3.9 見出しの指定方法	25
表 3.10 相互参照コマンド一覧	25
表 4.1 Markdown パラグラフ・コマンド	29
表 4.2 Markdown インライン・コマンド	30
表 4.3 HTML コマンド	30
表 4.4 パラグラフ処理コマンド	31
表 4.5 単独処理コマンド	32
表 5.1 OS 組込みフォントの検索パス	37
表 5.2 単独処理コマンドの返却オブジェクト種別	38
表 B.1 スタイル・パラメーター一覧（欧文用スタイル）	43

表 B.2	スタイル・パラメーター一覧（和文用スタイル）	43
表 C.1	色名称および RGB/CMYK 対応表	45

第 1 章

イントロダクション

まず KiTTY 自身のご紹介と類似製品との比較、およびそれらに対する KiTTY の価値と利用シーンについてご紹介いたします。また、サポート機能の一覧を簡単に提示いたします。

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 用のパッケージ (Kinx Tiny Typesetting) として単独で配布されており、Kinx と共に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。本文書をコンパイルするのに約 4 分ほどかかります。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「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。

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、通常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 では直接その見た目から「構造化されたもの」か「見た目だけ整っているのか」の区別が付きません。例えば、章番号がきちんと設定され、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かなど判別しづらいといった欠点もあります。特に、書き方に関わらず「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では正しく設定されていなかった、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

現時点で本格的な組版を利用したい場合は L^AT_EX を使いましょう。将来にわたって本格的な組版技術を学び、活用していきたい場合は Vivliostyle などの CSS 組版を学びましょう。その上で、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

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。

³ <https://vivliostyle.org/>

ちょっとした文書作成のために TeX 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。具体的な機能の内容に関しては、「[第 3 章 機能概要](#)」を参照してください。なお、カーニングは現在サポートしていません。

- ・ 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
- ・ マルチカラム
- ・ 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
- ・ タイトル・カバーページ・目次
- ・ 見出し
- ・ 相互参照
- ・ 引用
- ・ 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

⁴ つまり、私のような方。実際、当初の目的は自分のプロジェクトの簡易マニュアル作成のためでした。

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

KiTTY を実際に始めるための準備と、簡単な例を通した使い方をご紹介します。なお、準備に関しては Windows と Linux で異なりますが、あらかじめ準備されているフォントを同じように利用する限り Windows、Linux で同じ出力が得られるようになっていますので、どちらで使っても問題ありません。

2.1 インストール

KiTTY は現在 Kinx Tiny Typesetting という Kinx のパッケージとして配布されています。そのため、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. Kinx Tiny Typesetting パッケージのインストール

2.1.1 Kinx のインストール

まずはじめに、Kinx のインストールを行います。

2.1.1.1 Windows

Windows では Scoop を使います。以下のようにコマンドラインから実行します。

```
$ scoop bucket add kinx https://github.com/Kray-G/kinx
$ scoop install kinx
$ kinx --install-path
```

まず Kinx 用 Bucket の URL を登録します。この登録は最初に 1 度だけ必要です。Bucket が登録されていれば、`scoop install kinx` コマンドでインストールできます。Scoop でインストールした際は、パッケージコマンドへのパスを通すために `kinx --install-path` コマンドを実行してください。

2.1.1.2 Linux

Linux(Ubuntu) では、[Relases](#) ページから `.deb` ファイルをダウンロードします¹。ダウンロードしたディレクトリに移動し、次のようにインストールします。

```
$ sudo apt install ./kinx_1.1.0-0_amd64.deb
```

¹ ファイル名にバージョン番号が含まれます。必要なバージョンをダウンロードしてください

2.1.2 Kinx Tiny Typwsetting パッケージのインストール

2.1.2.1 Windows

Kinx のパッケージをインストールするには `kip` コマンドを使用します。コマンドプロンプトで以下のように実行することで、自動的に最新版数のパッケージがインストールされます。

```
$ kip install typesetting
```

2.1.2.2 Linux

Linux でも Windows と概ね同じですが、Linux では管理者権限が必要です。必要に応じて `sudo` コマンドを使用してください。

```
$ sudo kip install typesetting
```

2.2 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 Kinx Tiny Typesetting
% Your name
% October 7, 2020

# Greeting
hello, world
```

以下のように `kxkitty` コマンドを実行することで、`helloworld.pdf` が作成されます。

```
$ kxkitty helloworld.md
```

第 3 章

機能概要

ここでは機能の概要を説明いたします。本書自体 KiTTy で組版されていますので、本書で実現できていることはすべて実現可能です。まずは色々と試してみましょう。

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。

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。

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.

図 3.1 ハイフネーション・両端揃え

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。
- ・ 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。

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 マルチカラム

マルチカラムに対応しています。<set-column value="N"/> で N カラムに設定されます。元に戻す場合は <set-column value="1"/> と指定します。ただし、あまり N を大きくすると行幅が狭くなるのでレイアウトが崩れやすくなります。ページの最下段までテキストが到達した時点でカラムの先頭に戻ります。

高さを指定したい場合は、height 属性を使用します。height 属性には、例えば 10em のような形で値に単位を付けることが可能です。<set-column value="N" height="12em" /> で N カラムで高さを 12em に設定します。

脚注に関しては、カラムごとに出力されず常にページ全体として処理されます。以下は 2 カラムで構成する例です。夏目漱石「吾輩は猫である¹」の冒頭の一節です。

—

吾輩^{わがはい}は猫である。名前はまだ無い。

どこで生れたかとうと見当がつかぬ。何でも薄暗いじめじめした所でニャーニャー泣いていた事だけは記憶している。吾輩はここで始めて人間というものを見た。しかもあとで聞くとそれは書生という人間中で一番^{けんとう}癡^{どうあく}悪な種族であったそうだ。この書生というのは時々我々^{つかま}を捕えて煮て食うという話である。しかしその当時は何という考もなかったから別段恐いとも思わなかった。ただ彼の^{てのひら}掌に載せられてスーと持ち上げられた時何だかフワフワした感じがあったばかりである。掌の上で少し落ちついて書生の顔を見たのがいわゆる人間というもの^{みはじめ}の見始^{みはじめ}であろう。この時妙なものだと思った感じが今でも残っている。第一毛をもって装飾され

べきはずの顔がつるつるしてまるで^{やかん}葉缶だ。その後猫にもだいぶ逢ったがこんな片輪^{かたわ}には一度も^{でく}出会^{でく}わした事がない。のみならず顔の真中があまりに突起している。そうしてその穴の中から時々ぶうぶうと^{けむり}煙を吹く。どうも咽^むせぼくて実に弱った。これが人間の飲む^{たばこ}煙草というものである事はようやくこの頃知った。

この書生の掌^{うち}の裏^{うら}でしばらくはよい心持に坐っておったが、しばらくすると非常な速力で運転し始めた。書生が動くのか自分だけが動くのか分らないが^{むやみ}無暗に眼が廻る。胸が悪くなる。到底^{どうてい}助からないと思っていると、どさりと音がして眼から火が出た。それまでは記憶しているがあとは何の事やらいくら考え出そうとしても分らない。

ここでは高さを 28em に設定しています。高さを自動で調整することはできませんので、調整したい場合は個別に直接値を指定する必要があります。高さがページ下限に達した場合はそこで自動的に折り返されます。また、高さを指定しなければ自動的にページ下限で折り返されます。

¹「吾輩は猫である」(夏目漱石)

3.1.4 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

3.1.5 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。

3.1.5.1 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。

3.1.5.2 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した場合、例えば、数式 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。

3.1.6 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。

3.1.6.1 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。

```
1 ![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。



3.1.6.2 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。

```
1 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。

3.1.6.3 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。7 ページに示す「[図 3.1 ハイフネーション・両端揃え](#)」の図はその一例です。



図 3.2 イトトンボ

左の図は Public Domain で配布されている図²です。このような形でイメージをフローティングさせることができます。フローティング形式でも版面の横幅に対するスケールとして `scale` の指定が可能ですが、版面の横幅の最大 70% までに補正（制限）されます³。

また、このようにフローティング中に複数の段落を配置することも可能です。最終的にフローティングされたイメージの下までパラグラフの文章が到達した際に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。



図 3.3 F14 Tomcats

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ることです。パラグラフの途中で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。

3.1.6.4 イメージ・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</code> , <code>right</code>	フローティング位置
<code>scale</code>	0.0～ 1.0	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
<code>box</code>	<code>BOX_NORMAL</code>	通常の太さの線で図を囲むようにボックスを表示
	<code>BOX_THIN</code>	細い線で図を囲むようにボックスを表示
	<code>BOX_THICK</code>	太い線で図を囲むようにボックスを表示
<code>padding</code>	実数	図とボックスの間の余白を指定

² <https://free-images.com/>

³ この 70% にはイメージとテキストの間の余白を含みます。

3.1.7 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。

3.1.7.1 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.js⁴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 Radar Chart Example のように出力されます。

⁴ <https://www.chartjs.org/>

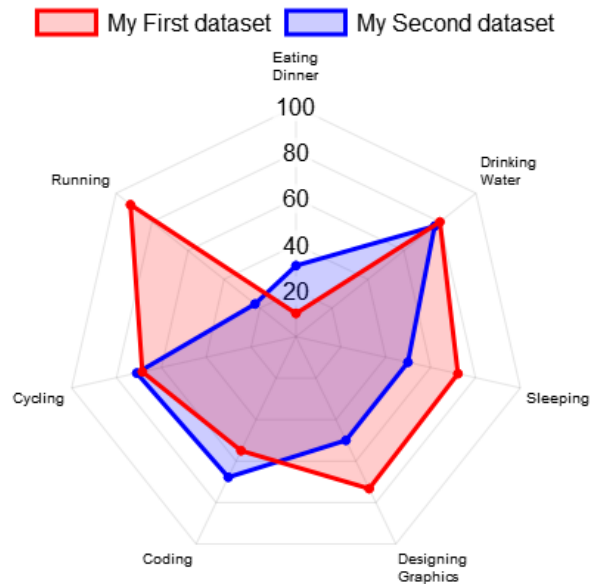


図 3.4 Radar Chart Example

3.1.7.2 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。

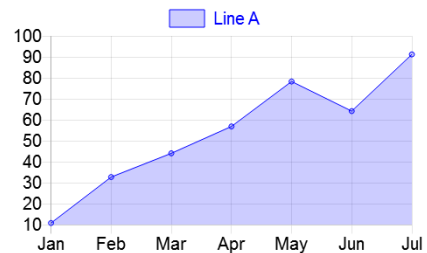


図 3.5 Line Chart Example

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  ``

```

3.1.8 テーブル

3.1.8.1 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)

3.1.8.2 テーブル・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。

3.1.9 フォント

3.1.9.1 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。

3.1.9.2 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⁵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.
```

これは、「Changing the font is available only with a scope like “*This is a pen.*,” and the font will be restored here.」となります。

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。

⁵ 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。

3.1.9.3 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。
pt	サイズをポイントで指定します。
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。

3.1.9.4 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。

3.1.10 色

3.1.10.1 文字色

文字の色を変えるには `\color` コマンドを使用します。以下にサンプルを示します。

```
1 * \color\red\{\bold{Red}.
2   This line should be colored by the name of `red`.}
3 * \color\green\{\bold{Green}.
4   This line should be colored by the name of `green`.}
5 * \color\blue\{\bold{Blue}.
6   This line should be colored by the name of `blue`.}
7 * \color\cyan1\{\bold{Cyan}.
8   This line should be colored by the name of `cyan1`.}
9 * \color\magenta1\{\bold{Magenta}.
10  This line should be colored by the name of `magenta1`.}
11 * \color\yellow\{\bold{Yellow}.
12  This line should be colored by the name of `yellow`.}
13 * \color[R=0,G=64,B=255]\{\bold{RGB}.
14  This line should be colored by RGB value of `RGB=0,64,255`.}
15 * \color[C=0.5,M=0.8,Y=0.2,K=0.0]\{\bold{CMYK}.
16  This line should be colored by CMYK value of `CMYK=0.5,0.8,0.2,0.0`.}
```

- **Red.** This line should be colored by the name of red.
- **Green.** This line should be colored by the name of green.
- **Blue.** This line should be colored by the name of blue.
- **Cyan.** This line should be colored by the name of cyan1.
- **Magenta.** This line should be colored by the name of magenta1.
- **Yellow.** This line should be colored by the name of yellow.
- **RGB.** This line should be colored by RGB value of RGB=0,64,255.
- **CMYK.** This line should be colored by CMYK value of CMYK=0.5,0.8,0.2,0.0.

サポートされる色の名称に関しては、「付録 C 色一覧」を参照してください。

3.1.11 合字・特殊文字

3.1.11.1 合字

以下の 5 種類の合字のみサポートしています。

- fi ... fi
- fl ... fl
- ff ... ff
- ffi ... ffi
- ffl ... ffl

3.1.11.2 「“」と「”」

「“」と「”」はいずれもシングルクォート 2 つで表現し、自動的に判断されます。

```
1 これは 'サンプル' です。
```

上記は、「これは“サンプル”です。」となります。

3.1.11.3 バッククォート

バッククォートは Markdown のコマンドと重なるため、特別な記法を用意しています。バッククォートの数を N として、<backqN /> の形で記載します。3 つのバッククォートを表現する場合は、<backq3 /> と記載します。「```」と出力されます。バッククォート 2 つの場合は <backq2 /> と記載します。「``」と出力されます。

3.1.12 プログラム・コード

3.1.12.1 記述方法

プログラムコードはブロック形式のコードブロックで記述します。デフォルトでは行番号が付加され、影付きのボックスで表現されます。

```
1 ```
2 class Test {
3     public test() {
4         # Test Method.
5     }
6 }
7 ```
```

上記のように記述すると以下のように出力されます⁶。

```
1 class Test {
2     public test() {
3         # Test Method.
4     }
5 }
```

行番号を外し、影を無くしてみましょう。以下のように記述します。

```
1 ```:lineNumber=false,box=BOX_NORMAL
2 class Test {
3     public test() {
4         # Test Method.
5     }
6 }
7 ```
```

⁶ どちらも同じ形ですので分かりづらいですが、```` のあるほうが記述例です。

すると、以下のように出力されます。

```
1 class Test {
2     public test() {
3         # Test Method.
4     }
5 }
```

3.1.12.2 プログラム・コード・オプション

言語指定

いくつかプログラミング言語、および特別な言語指定することで、事前定義された動作、およびハイライトを行います。なお、言語指定は``の直後に記載し、オプションとの区切りに:を使用します。

表 3.6 言語指定で指定できる値の一覧

言語	意味
math	数式として処理します。詳しくは「 3.1.5 数式 」を参照ください。
chart	グラフ（チャート）として処理します。詳しくは「 3.1.7 グラフ（チャート） 」を参照ください。
console	黒背景に白抜き文字で出力します。
JSON	JSON 形式のシンタックスを解釈してハイライトします。
c、c++、cpp	C/C++ 形式のシンタックスを解釈してハイライトします。
javascript、js	JavaScript 形式のシンタックスを解釈してハイライトします。
java	Java 形式のシンタックスを解釈してハイライトします。
kinx	Kinx 形式のシンタックスを解釈してハイライトします。
ruby	Ruby 形式のシンタックスを解釈してハイライトします。
python	Python 形式のシンタックスを解釈してハイライトします。

lineNumber

lineNumber オプションは true または false を指定します。box 内部での行番号を表示するかどうかの設定です。デフォルトでは表示しますが、言語指定されている場合はその言語のデザインに従います。

box

box オプションで指定可能な値は以下の通りです。デフォルトでは BOX_SHADOW ですが、言語指定されている場合はその言語のデザインに従います。

表 3.7 box オプションで指定できる値の一覧

値	意味
BOX_NORMAL	通常の太さの線でボックスを表示する。
BOX_THIN	細い線でボックスを表示する。
BOX_THICK	太い線でボックスを表示する。
BOX_SHADOW	影付きの線でボックスを表示する。(デフォルト)

color/bgcolor

color および bgcolor によって文字色と背景色を変更できます。指定できる色に関しては、「[付録 C 色一覧](#)」を参照してください。

3.1.12.3 シンタックス・ハイライト

言語指定をすることで、いくつかの言語でのシンタックス・ハイライトをサポートしています。指定できる言語は表 3.6 [言語指定で指定できる値の一覧](#)を参照してください。また、キーワード以外、カラーは同じにしています。ただし、言語ごとに以下のように変更可能です。

```
<code-style lang="ruby" name="box" value="shadow" />
<code-style lang="ruby" name="background-color" value="lightcyan1" />
```

このように指定すると Ruby のみ変更されます。

```
1 // This is a kinx block.
2 function fib(n) {
3     if (n < 3) return n;
4     return fib(n-2) + fib(n-1);
5 }
6 System.println(fib(34));
```

```
1 # This is a python block.
2 def fib(n):
3     if n < 3:
4         return n
5     else:
6         return fib(n-1) + fib(n-2)
7 print fib(34)
```

```
1 # This is a ruby block.
2 def fib(n)
3     return n if n < 3
4     fib(n-1) + fib(n-2)
5 end
6 p fib(34)
```

指定可能な属性名は以下の通りです。色に関しては、「[付録 C 色一覧](#)」を参照ください。

表 3.8 コード・スタイル値の一覧

属性名	デフォルト値	備考
box	"normal"	枠のスタイル。値は <code>noline</code> 、 <code>normal</code> 、 <code>thin</code> 、 <code>thick</code> 、 <code>shadow</code> のいずれか。
foreground-color	"black"	前景色。
background-color	"cornsilk1"	背景色。
comment-single-line	"grey50"	1行コメント。
comment-multi-line	"grey50"	複数行コメント。
string-multi-line	"darkred"	複数行文字列。Pythonのみ。
preprocessor	"grey30"	プリプロセッサ。C/C++のみ。
keyword	"dodgerblue2"	キーワード。
regex-literal	"red"	正規表現リテラル。
string-literal	"darkred"	文字列リテラル。
number	"lime"	数値。
function	"darkorange3"	関数呼び出し。
variable-capital	"green4"	大文字開始変数名。Variable のような形式。
variable	"cyan1"	小文字開始変数名。variable のような形式。

3.1.13 タイトル（カバーページ）・目次

3.1.13.1 タイトル表記内容の指定方法

KiTTY はタイトル（カバーページ）、および目次を自動的に作成します。現在のスタイル指定では、Book スタイルの際にカバーページとなるように設定されています。各種設定は最初のチャプターが現れる前までに設定します。以下の例は本文書の例です。以下のようにすることで、目次のタイトル、執筆者、日付を指定します。この順序は決まっているため、不要なものは%のみ記載します。また、%での設定の後に出てきた最初のパラグラフをサブタイトルと認識します。

```
1 % KiTTY
2 % Kray-G, Mr.Diamond Global Blue Publisher
3 % September 18, 2020
4
5 小さくて簡単、でも美しいデザイン。
```

3.1.13.2 タイトル・パラメータの設定

その他の設定は、最初のチャプターが現れる前までに `<param />` タグで指定します。以下が本文書での例です。

```
1 <param style="JBookA4"/>
2 <param titleSize="78.8"/>
3 <param subtitleSize="14.4"/>
4 <param backgroundImage="back.jpg"/>
```

設定できる項目はスタイルごとに異なります。ここでは JBookA4 スタイルで指定できる設定値となっています。これらの設定の意味は、以下の通りです。

- スタイルは JBookA4 を使用する。
- タイトルの文字サイズは 78.8pt とする。
- サブタイトルの文字サイズは 14.4pt とする。
- 表紙の背景画像として back.jpg を使用する。

3.1.13.3 目次の設定

目次を表示する指定は以下のように行います。その際、図目次 (lof)、表目次 (lot) も表示するよう指定しています⁷。

```
1 <toc with="lof,lot"/>
```

3.1.13.4 スタイルのカスタマイズ

各スタイルではデフォルトの設定値が存在します。それらを変更する場合は、`<style-info />` タグを使用して設定します。次の例は、チャプター・デザイン (A.2 チャプター・デザイン) を変更する例です。

```
1 <style-info name="chapter.style" value="BigChapter3" />
```

変更可能なスタイル・パラメータの詳細に関しては、「[付録 B スタイル](#)」をご参照ください。

⁷ lof と lot は図目次・表目次 (List Of Figures、List of Tables) を意味します。図目次・表目次の両方を表示したい場合は、lof,lot のようにカンマ区切りで指定します。

3.1.14 見出し

章、節といった見出しは、通常の Markdown における # 記号で示します。それぞれ以下のように解釈されます。

表 3.9 見出しの指定方法

記号	内容
#	章 (Chapter)
##	節 (Section)
###	小節 (Sub Section)
####	少々節 (Sub Sub Section)

これらの見出しには自動的に番号が付加され、相互参照 (3.1.15 相互参照) 機能を使うことが可能になります。相互参照機能を利用する場合は、ラベルとして「見出し文字列」を直接指定することができます。

3.1.15 相互参照

相互参照機能をサポートしています。相互参照機能を利用できる対象は、以下の表の通りです。各対象に対し、例えば章番号やページ番号などを参照可能です。

表 3.10 相互参照コマンド一覧

参照方法	内容
<code>\ref{label}</code>	章番号、節番号、図表番号、数式番号
<code>\textref{label}</code>	それぞれのテキストの参照
<code>\nameref{label}</code>	それぞれの「番号+テキスト」の参照
<code>\pageref{label}</code>	それぞれに対するページの参照

なお、参照が先に現れた場合、その時点で番号などが解決できません。その場合は再度実行することで解決できるようになります。

3.1.16 引用

引用は以下のように行頭に ‘>’ を付けて記述します。引用中に Markdown のパラグラフ・コマンド (4.1.1 パラグラフ・コマンド) を使用することはできませんが、インライン・コマンド (4.1.2 インライン・コマンド)、KiTTY コマンド (4.2 KiTTY コマンド) を使用することはできます。また、引用はネストすることが可能です。その場合、引用は全て段落として認識されます。インデントをしたくない場合、段落の冒頭に `\\noindent` を付けることでインデントをしないようにすることができます。例えば、本書の冒頭の文章を一部ネストさせるように修正して引用してみましょう。

- 1 > **KitTy** は **Ki**nx** **T**iny** **Ty**pesetting** を意味し、
- 2 > **Kinx** で実装された簡易組版システムの名称です。
- 3 > `\noindent` 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、
- 4 > > 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。
- 5 > > 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。
- 6 >
- 7 > 考え方は `\LaTeX` に近く、
- 8 > 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。

以下のように引用されます。

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 \LaTeX 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。

3.1.17 脚注

脚注もサポートします。脚注は `[^label]` の形で参照し、独立したパラグラフの位置に `[^label]: ...` の形式で記述します。例えば、以下の通りです。

- 1 この文章に脚注^[^f1]を置きます。
- 2
- 3 ^[^f1]: これが脚注になります。

実際に記述すると、「この文章に脚注⁸を置きます。」となります。本ページの最下部に脚注として表示されているはずです。

3.2 日本語用組版機能

本機能は日本語独自の要件です。もし本章を必要としない場合⁹、次章（[3.3 PDF 機能](#)）へお進みください。

3.2.1 日本語禁則処理

以下の日本語の禁則処理を実施します。

- 行頭禁則文字
- 行末禁則文字
- グループルビの分離禁止

⁸ これが脚注になります。

⁹ 日本語マニュアルなので、この注意書きは不要と思います。が、念のため。

3.2.2 日本語ルビ（振り仮名）

3.2.2.1 ルビ（振り仮名）について

日本語の振り仮名に対応しています。「振り仮名」は「ルビ¹⁰」とも呼ばれます。振り仮名（ルビ）は以下の仕様にサポートしています。

- ・ 幅は親文字とルビのどちらか広いほうが採用されます。
- ・ モノルビ、グループルビの両方に対応しています。
- ・ 親文字、ルビ、どちらにも和文、英文両方が使えます。

これは日本語のみの要件です。日本語では、「振り仮名」または「ルビ」と呼ばれる書き方があります。これは、通常の文章の上側に小さな文字で、主に漢字の読み方を記載する書式のことを指します。例えば、「この本^{ほん}はとても面白^{おもしろ}いです。」のように使用されます。通常、幼い子供は漢字が読めません。また、大人であっても難しい漢字や普段使わない漢字の中には読めない漢字がある場合があります。そのような場合を考慮し、漢字の読み方が分かるよう「本^{ほん}」のような形でルビを振ります。

3.2.2.2 ルビの指定方法

KiTTy ではモノルビ、およびグループルビの2種類の形式をサポートします。モノルビは `\\ruby\[す|てき\\]{素敵}` という形で、各漢字に対応するように「|」記号を挿入して使用します。この場合、「素敵^{すてき}」のように表現され、それぞれの漢字の上部に個別にルビが振られます。もう一方のグループルビは `\\ruby\[すてき\\]{素敵}` といった形でそのままルビと親文字を指定します。グループルビの場合は「素敵^{すてき}」と表現され、対象となる全ての漢字の上に均等にルビが振られます。サンプルを見ると「て」の位置が「素」と「敵」の中間に位置していることが分かるでしょう。グループルビでは行分割の際に単語の分割は許可されませんが、モノルビの場合は各漢字の区切りで行分割することが可能です。

3.2.2.3 ルビの例

以下は芥川龍之介「蜘蛛の糸」の一節です。分かりやすいように引用形式で表現しておきましょう。まずはグループルビとしてルビを振る例です。

- 1 > ある日の事でございます。\\ruby\[おしゃかさま\\]{御釈迦様}は
- 2 > 極楽の\\ruby\[はすいけ\\]{蓮池}のふちを、
- 3 > 独りでぶらぶら御歩きになっていらっしゃいました。

これは以下のように出力されます。

ある日の事でございます。御釈迦^{おしゃかさま}様は極楽^{はすいけ}の蓮池のふちを、独りでぶらぶら御歩きになっていらっしゃいました。

次に全てモノルビとしてルビを振った例です。モノルビでは区切る場所に「|」を挿入します。

¹⁰ “（19世紀後半）イギリスから輸入された5.5ポイント活字の呼び名がruby（ルビー）であったことから、この活字を「ルビ活字」とよび、それによってつけられた（振られた）文字を「ルビ」とよぶようになった。” - [Wikipedia](#)

- 1 > ある日の事でございます。\\ruby\[お|しゃ|か|さま\\]{御釈迦様}は
- 2 > 極楽の\\ruby\[はす|いけ\\]{蓮池}のふちを、
- 3 > 独りでぶらぶら御歩きになっていらっしゃいました。

次のようになります。「御釈迦様（おしゃかさま）」はルビの乗り方が異なっていることが分かります。蓮池（はすいけ）」は位置があまり変わりませんが、「蓮（はす）」と「池（いけ）」の間で行分割が可能となります。

ある日の事でございます。御釈迦様は極楽の蓮池のふちを、独りでぶらぶら御歩きになっていらっしゃいました。

3.3 PDF 機能

3.3.1 外部リンク

外部リンクは直接 URL 文字列を記載することで URL に対するリンクを自動的に認識し、リンクを生成します。例えば、<https://github.com/Kray-G/kinx> と記載することで自動的に該当文字列（URL）上に外部リンクが生成されます。

また、Markdown のリンク構文もサポートしました。ただし、1行スタイルのみサポートし、分割スタイルは現状未サポートです。具体的にいうと、[リンクテキスト](URL) の形式のみサポートしています。例えば、[Kinx](<https://github.com/Kray-G/kinx>) と記載すると [Kinx](https://github.com/Kray-G/kinx) と出力され、リンク文字列上をクリックすることで外部リンク先に飛ぶことができます。

3.3.2 相互参照リンク

各見出しやラベルを付けた図や表に対して相互参照（[3.1.15 相互参照](#)）できますが、PDF として内部リンクが自動的に作成されます。したがって、PDF 上でクリックすることによって相互リンクされた見出しや図表へ飛ぶことができます。

3.3.3 しおり

PDF のしおり機能にも対応しています。各見出しは自動的に PDF のしおりとして作成されます。しおりに表示された見出しをクリックすることで、その場所に飛ぶことができます。

注意点として、PDF のしおり部分には KiTTY コマンドなどのコマンドを使用することはできません。したがって、見出し文字列には各種コマンドを利用しないようにしてください¹¹。

¹¹ 例外として TeX、LaTeX、KaTeX のみそれぞれ TeX、LaTeX、KaTeX に置き換えられます。

第 4 章

コマンド詳細

本章では利用可能なコマンドを説明します。コマンドには大きく、Markdown としてのコマンド、KiTTY としてのコマンド、の 2 種類があります。Markdown としてのコマンドは、実際の Markdown 記法でサポートされているもの、および HTML として KiTTY 用に特別に解釈されるものがあります。KiTTY としてのコマンドは、基本的にパラグラフ内で使用するコマンドですが、`'` で始まるコマンドとなります。この文字は Markdown ではエスケープ文字として扱われるため `\\command` の形で記載します。

4.1 Markdown コマンド

4.1.1 パラグラフ・コマンド

基本的には Markdown の記法を解釈しますが、個別に条件がある場合があります。以下の表を参照してください。

表 4.1 Markdown パラグラフ・コマンド

コマンド	内容
パラグラフ	空行で区切って記載。
ラインブレイク	行末に空白を 2 つ配置した行。
コードブロック	空白 4 文字で始まる行。または <code>```</code> で囲ったパラグラフ。
引用	<code>'</code> で始まる行。
見出し	<code>#</code> で始まる行。下線 (<code>==</code> 、 <code>--</code>) を使う形式は未サポート。
箇条書き	<code>*</code> で始まる行。数字での箇条書きは 1. 等の数値で始まる行。空行を挟むケースは不可。
表	通常の Markdown 形式での表形式をサポート。追加で指定したいパラメータは <code><context /></code> コマンドを使用する。
イメージ	パラグラフに <code>!...</code> のみ単独で記載された場合。単独で表れていない場合はインライン・コマンドとして扱われる。
脚注	<code>[^name]:</code> で始まる行。name は任意。(対応する脚注マークはインライン・コマンド)

4.1.2 インライン・コマンド

こちらも基本的には Markdown の記法を解釈しますが、個別に条件がある場合があります。以下の表を参照してください。

表 4.2 Markdown インライン・コマンド

コマンド	内容
イタリック	<code>*~*</code> で囲った文字列。
ボールド	<code>**~**</code> で囲った文字列。
ボールドイタリック	<code>***~***</code> で囲った文字列。
インライン・コード	<code>`~`</code> で囲った文字列。
インライン・イメージ	<code>!...</code> 形式で記載。
リンク	URL を自動認識。 <code>...</code> 形式と <code>[...][...]</code> 形式もサポート。
脚注	<code>[^name]</code> で記載。 <code>name</code> は任意。

4.1.3 HTML コマンド

Markdown 記法の中に、HTML で記載するコマンドがいくつか存在します。主に文章中には表示されず、パラメータの設定や制御を目的としたものです。

表 4.3 HTML コマンド

コマンド	内容
<code><toc /></code>	目次を表示する。最初のチャプターの前に指定する。 <code>with</code> 属性に <code>lot</code> 、 <code>lof</code> を指定可能。
<code><param /></code>	ページの初期値を設定する。最初のチャプターの前に指定する。
<code><context /></code>	一部の機能で必要な追加パラメータを一時的に設定する。使用可能なパラメータは、対象となる機能による。
<code><clear-float /></code>	フローティングされた図への回り込みを強制的に解除する。
<code><pagebreak /></code>	ページ区切りを強制的に挿入し、改ページ処理を行う。
<code><set-column /></code>	カラム数を設定する。 <code>value</code> 属性でカラム数を指定し、 <code>height</code> 属性で高さを指定する。
<code><style-info /></code>	スタイルに関するパラメータを変更する。 <code>name</code> 、 <code>value</code> 属性で指定する。
<code><appendix /></code>	Appendix の始まりを示す。
<code><include /></code>	挿入するファイルを <code>file</code> 属性で指定する。

4.2 KiTTY コマンド

4.2.1 パラグラフ処理コマンド

パラグラフ処理コマンドとは、{} 内をブロック（スコープ）として扱い、そのブロック内をパラグラフとして認識・処理するコマンドです。

表 4.4 パラグラフ処理コマンド

コマンド	内容
<code>\bigger{}</code>	フォントサイズを +1 する。
<code>\smaller{}</code>	フォントサイズを -1 する。
<code>\bold{}</code>	フォントをボールド体にする。 **~** で囲むのと同様。
<code>\itaric{}</code>	フォントをイタリック体にする。 <i>*~*</i> で囲むのと同様。
<code>\color[params]{} </code>	色を変更する。 red などの色名称、および R=r,G=g,B=b、C=c,M=m,Y=y,K=k による色指定が可能。
<code>\font[params]{} </code>	フォントを変更する。 <code>\font[size=1em]{} </code> のように指定する。 size ... フォントサイズを指定する。 name ... フォントを変更する。
<code>\footnote{}</code>	脚注を設定する。
<code>\monotype{}</code>	等倍フォントを使用するように変更する。
<code>\raise[params]{} </code>	Y 軸のオフセットを変更し、上に移動させる。height ... 位置オフセットを指定する。
<code>\lower[params]{} </code>	Y 軸のオフセットを変更し、下に移動させる。height ... 位置オフセットを指定する。
<code>\sans{}</code>	ゴシック体（Sans Serif 体）を使用するように変更する。
<code>\url[params]{} </code>	リンクテキストを生成する。param に URL を記載する。[Text](URL)と同様。

ここでの例では単純に {} とだけ記載されていますが、全てのコマンドで {} 内にパラグラフを記述できます。例えば、`\\sans{「これはサンセリフ・フォントです。」}` と記載すると、「これはサンセリフ・フォントです。」と表示されます。

4.2.2 単独処理コマンド

単独処理コマンドとは、{} 内もパラメータの一種として扱い、パラグラフ処理を行わないコマンドのことを指します。なお、[] も {} も省略可能ですが、次の単語との区切りを明確にする必要がある場合は、{} を付ける必要があります。例えば、`\\noindent{}` と記載します。

表 4.5 単独処理コマンド

コマンド	内容
<code>\TeX</code>	\TeX のロゴを出力する。
<code>\LaTeX</code>	$\mathrm{L}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ のロゴを出力する。
<code>\KaTeX</code>	$\mathrm{K}\mathrm{A}\mathrm{T}\mathrm{E}\mathrm{X}$ のロゴを出力する。
<code>\noindent</code>	インデント処理を打ち消す。インデントしないパラグラフを生成する。
<code>\apostrophe</code>	アポストロフィーを出力する。
<code>\copyright</code>	コピーライト (©) を出力する。
<code>\hspace</code>	区切りスペース (小文字 i の幅) サイズの空白を出力する。
<code>\hspace{width}</code>	<code>width</code> で指定された幅 (単位指定可) 分 X 座標を変更する (プラスの値で右方向)。
<code>\vspace{height}</code>	<code>height</code> で指定された高さ (単位指定可) 分 Y 座標を変更する (プラスの値で下方向)。
<code>\ref{label}</code>	<code>label</code> で指定された番号に対するクロス・リファレンスを生成する。
<code>\pageref{label}</code>	<code>label</code> で指定されたページに対するクロス・リファレンスを生成する。
<code>\textref{label}</code>	<code>label</code> で指定されたテキストに対するクロス・リファレンスを生成する。
<code>\nameref{label}</code>	<code>label</code> で指定された「番号+テキスト」に対するクロス・リファレンスを生成する。
<code>\pack{text}</code>	<code>text</code> の内容を改行させずひとまとまりとして認識させる。
<code>\ruby[Ruby]{Parent-Text}</code>	Parent-Text の上部にルビ (Ruby) を表記する。(左記表記での例: Parent-Text ^{Ruby})
<code>\arrow{direction}</code>	矢印を出力する。 <code>direction</code> には <code>left</code> 、 <code>right</code> 、 <code>up</code> 、 <code>down</code> 、 <code>left-right</code> 、 <code>up-down</code> 、 <code>left-up</code> 、 <code>right-up</code> 、 <code>right-down</code> 、 <code>left-down</code> のいずれかを指定。
<code>\unicode{code}</code>	<code>code</code> で示したユニコード文字を出力する。

第 5 章

機能拡張方法

KiTTY での機能追加の方法に関する概要を記載します。本章だけでは実際に追加するために必要な情報の全てを書ききれませんので、追加方法は別途まとめる予定です。現時点では大まかな機能拡張方法の概要と、追加の仕方のみ説明いたします。

5.1 各種スタイル定義の追加

5.1.1 文書スタイルの追加

文書全体のスタイル・ファイルはインストール・フォルダを `$INSTALL` として以下の位置に配置します。

```
1 $INSTALL/lib/std/typesetting/style/additional/*.kx
```

ここに配置するスタイルは順不同で読み込まれるため、スタイル同士で関連（クラス継承等）させることはできません。基準となる（親クラスとできる）標準スタイルは以下にあります。

```
1 $INSTALL/lib/std/typesetting/style/basic/*.kx
```

`additional` フォルダ配下にはファイルを配置するだけで使用可能となりますが、`basic` 配下に追加する場合は、以下のファイルにロード処理を追加する必要がありますのでご注意ください。

```
1 $INSTALL/lib/std/typesetting/style/Styles.kx
```

したがって、通常は `additional` 配下にスタイル定義ファイルを追加するようにしてください。

5.1.1.1 スタイル定義ファイルの追加

以下がデフォルトで配置されているスタイル定義ファイル（`JArticleA4_2Cols`）の例です。

```
1 using typesetting.style.basic.JArticleA4;
2
3 namespace Typesetting {
4   namespace Style {
5
6     class JArticleA4_2Cols : Typesetting.Style.JArticleA4 {
7       @style.columns = 2;
8     }
9
10  } # namespace Style
11 } # namespace Typesetting
```

クラス名がスタイル名となります。また、名前空間として `Typesetting.Style` に属している必要があります。`JArticleA4` スタイルを継承し、カラム数を 2 に設定している例になります。現時点で全てのスタイルの元となる `ArticleA4` スタイルの定義は `basic/ArticleA4.kx` にあります。設定可能な項目は、本スタイル定義ファイルを参照してください。

5.1.2 タイトル・スタイルの追加

タイトルの表現方法をプラグインできます。タイトルページとして独立したページとすることも可能です。タイトルページのスタイル設定定義は以下の位置に存在します。

```
1 $INSTALL/lib/std/typesetting/style/title/*.kx
```

タイトル・スタイルを変更する場合は、以下のようにスタイル設定値を設定します。ここで指定される `yourstylename` は、(後述する方法でスタイルを追加した場合)「スタイル定義ファイルの拡張子を除くファイル名」となります。

```
1 <style-info name="title.style" value="yourstylename" />
```

5.1.2.1 タイトル・スタイルの例

例として、`StandardArticle` の定義例を見てみましょう。

`Typesetting.Style.Title[__FILE__.stem()]` という記述によって、ファイル名の拡張子を除いた部分をプロパティ名とし、`Typesetting.Style.Title` に関数オブジェクトとして登録するように記述します。

```
1 namespace Typesetting {
2   namespace Style {
3
4     Typesetting.Style.Title[__FILE__.stem()]
5       = function(info, context, core, title, opts) {
6         # 描画ロジックを記載。
7       };
8
9   } # namespace Style
10 } # namespace Typesetting
```

事前に定義されているタイトル・スタイルに関しては、「[A.1 タイトル・デザイン](#)」をご参照ください。

5.1.3 チャプター・スタイルの追加

チャプターのスタイルもプラグインできます。チャプターのスタイル設定定義は、インストール・フォルダを `$INSTALL` として以下の位置に存在します。


```
1 $INSTALL/lib/std/typesetting/style/chapter/*.kx
```

チャプター・スタイルを変更する場合は、以下のようにスタイル設定値を設定します。ここで指定される `yourstylename` は、(後述する方法でスタイルを追加した場合)「スタイル定義ファイルの拡張子を除くファイル名」となります。

```
1 <style-info name="chapter.style" value="yourstylename" />
```

5.1.3.1 チャプター・スタイルの例

例として、`StandardBook` の定義例を見てみましょう。

`Typesetting.Style.Chapter[__FILE__.stem()]` という記述によって、ファイル名の拡張子を除いた部分をプロパティ名とし、`Typesetting.Style.Chapter` に関数オブジェクトとして登録するように記述します。

```
1 namespace Typesetting {
2   namespace Style {
3
4     Typesetting.Style.Chapter[__FILE__.stem()]
5       = function(info, context, core, text, opts) {
6         # 描画ロジックを記載。
7       };
8
9   } # namespace Style
10 } # namespace Typesetting
```

事前に定義されているチャプター・スタイルに関しては、「[A.2 チャプター・デザイン](#)」をご参照ください。

5.2 禁則処理

5.2.1 禁則処理の追加

禁則処理は以下のフォルダに配置しています。なお、日本語内の英文に対する処理と英文のみの文章で使用する処理は同じであるため、英文用の処理は日本語用の処理と同じものを使用します。

```
1 $INSTALL/lib/std/typesetting/lang/*.kx
```

禁則処理を変更する場合は、以下のようにスタイル設定値を設定します。ここで指定される `yourprocname` は、(後述する方法で処理を追加した場合)「禁則処理定義ファイルの拡張子を除くファイル名」となります。

```
1 <style-info name="hyphenationRule" value="yourprocname" />
```

禁則処理定義ファイルに記載する内容としては、`Typesetting.insertGlue[__FILE__.stem()]` の形で関数オブジェクトを登録するようにプログラムを記載します。例えば、`ja.kx` であれば、`Typesetting.insertGlue.ja` に禁則処理のための関数オブジェクトを登録するように記述します。

```
1 Typesetting.insertGlue[__FILE__.stem()] = _function(info, Linebreak, wordlist, nodes) {
2   # 禁則処理のロジックを記載。
3
4   wordlist.each { |(node, i)|
5     # ...
6     nodes.push(Linebreak.Glue(0, Linebreak.Infinity, 0));
7     nodes.push(Linebreak.Penalty(0, -Linebreak.Infinity, 0));
8     nodes.push(node);
9     # ...
10  };
11  };
```

5.2.1.1 ルールの作成

`wordlist` に解析後の単語リストが格納されているため、その内容を見て `nodes` 配列に追加していきます。以下のような形でルールを作っていきます。

- ・ノードの前で改行を禁止する場合、以下の形になるようにノードの前に挿入する。

```
1 nodes.push(Linebreak.Penalty(0, Linebreak.Infinity, 0));
2 nodes.push(node); # 対象のノード
```

- ・ノードの後で改行を禁止する場合、以下の形になるようにノードの前に挿入する。

```
1 nodes.push(node); # 対象のノード
2 nodes.push(Linebreak.Penalty(0, Linebreak.Infinity, 0));
```

- ・ノードの前で必ず改行させる場合、以下の形になるようにノードの後に挿入する。

```
1 nodes.push(Linebreak.Glue(0, Linebreak.Infinity, 0));
2 nodes.push(Linebreak.Penalty(0, Linebreak.Infinity, 0));
3 nodes.push(node); # 対象のノード
```

- ・ノードの前にグルーを挿入する場合、以下の形になるようにノードの前に挿入する。

```
1 nodes.push(Linebreak.Glue(width, stretch, shrink));
2   # width ..... 空白の幅
3   # stretch ... 空白が伸びる場合の最大の伸び幅
4   # shrink .... 空白が縮む場合の最大の縮み幅
5 nodes.push(node); # 対象のノード
```

最終的に `nodes` 配列が禁則処理を含む単語配列となります。この情報を元にハイフネーション処理が行われます。

5.3 フォント

5.3.1 新規フォントの追加

フォントを追加する場合、以下の位置にフォントファイルを格納してください。なお、現時点では TrueType フォントのみをサポートしています¹。

```
1 $INSTALL/lib/fonts
```

5.3.2 OS 組込みフォントの追加

OS 用の組込みフォントを使用する場合は、フォントファイル名を指定してください。その際、Windows と Linux で検索パスが異なります。

表 5.1 OS 組込みフォントの検索パス

OS	検索パス
Windows	C:/Windows/Fonts
Linux	/usr/share/fonts/truetype

上記パスからの相対パスで指定してください。例えば、Windows で Times New Roman を使用する場合、以下のようになります。

```
1 <font-load info="Times,serif,regular,times.ttf" />
2
3 * This is a Regular Style of a default font.
4 * \\font\[name=Times\]{Times New Roman of Regular Style}
```

フォント情報は埋め込まれるので別の OS 上でも正しく表示することは可能です。ただし、当たり前の話ですが別の OS 上で検出できないフォントを使用した場合、組版処理自体はフォントが見つけられずに失敗しますのでご注意ください。

5.4 コマンド

KiTTY コマンドの追加も可能です。KiTTY コマンドは以下の場所に配置されています。

```
1 $INSTALL/lib/std/typesetting/command/inline/*.kx
2 $INSTALL/lib/std/typesetting/command/paragraph/*.kx
```

`inline` 配下が単独処理コマンド（[4.2.2 単独処理コマンド](#)）で、`paragraph` 配下がパラグラフ処理コマンド（[4.2.1 パラグラフ処理コマンド](#)）です。

¹ libharu の制限です。

この場所にファイルを配置することで、自動的に新しいコマンドが追加されて使用できるようになります。なお、KiTTY コマンドの実装はファイル名がコマンド名とはなっていますが、実際にはクラス名がコマンド名として使用されます。現状では全てクラス名とファイル名をある程度揃えています。コマンド名はクラス名と大文字小文字含めて完全一致するか、lower ケースで一致した場合に有効となります。

5.4.1 単独処理コマンドの定義

単独処理コマンドの定義の雛形は以下の通りです。必要な実装は `exec` メソッドです。

ここでは `WordSet` オブジェクトを返していますが、`Word` オブジェクト、`LineBreak.Box` オブジェクト、`LineBreak.Glue` オブジェクト、`LineBreak.Penalty` オブジェクト、または `null` を返すことができます。必要に応じて使い分けます。

```
1 namespace Typesetting {
2     namespace Command {
3
4         class Command(info_, context_) {
5             public exec(params) {
6                 var ws = new Typesetting.WordSet(info_);
7                 # ...
8                 return ws;
9             }
10        }
11
12    } # namespace Command
13 } # namespace Typesetting
```

表 5.2 単独処理コマンドの返却オブジェクト種別

返却オブジェクト	内容
<code>WordSet</code>	複数の文字や単語をまとめて返す際に使用します。
<code>Word</code>	単独の文字や単語を返します。
<code>LineBreak.Box</code>	テキスト幅が決まっている場合など、 <code>WordSet</code> で表現できない場合に使用します。
<code>LineBreak.Glue</code>	グルーを挿入します。
<code>LineBreak.Penalty</code>	ペナルティを挿入します。
<code>null</code>	単にコンテキストを変更する場合などに返します。文字や単語が存在しないことを伝えます。

5.4.2 パラグラフ処理コマンドの定義

パラグラフ処理コマンドの定義の雛形は以下の通りです。クラスが定義される名前空間が

ParagraphCommand となり、クラスで使用する実装が start、end、translate メソッドになります。なお、定義されていない場合は無視されます。

```
1 namespace Typesetting {
2 namespace ParagraphCommand {
3
4     class Command(info_, context_) {
5         public start() {
6             # `{` が開始した際に実施する処理
7         }
8         public end() {
9             # 対応する `}` が現れた際に実施する処理
10        }
11        public translate(value) {
12            # `{` と `}` に挟まれたパラグラフ情報が value として入力される
13            # value はパラグラフ処理された後なので、WordSet オブジェクト等の配列
14            # value に対して変換が必要、または情報を追加する場合、ここで処理を行う
15            # end() 処理の前に呼ばれる
16        }
17    }
18
19 } # namespace Command
20 } # namespace Typesetting
```


付録 A

プレ定義デザイン

A.1 タイトル・デザイン

タイトルのデザインとして2種類のデザインを事前に用意しています。

A.1.1 StandardArticle

論文用のタイトル表現です。先頭ページの上部に表示されます。



A.1.2 StandardBook

独立した表紙を作成します。本書で
使用しているスタイルです。サブタイ
トルの設定、およびバックグラウンド
のイメージが出力可能です。

右の図は本書の表紙の例です。本ス
タイルではタイトルの上部にサブタイ
トルを出力し、タイトルの下にライン
を引きます。著者と日付はラインの
下側、右端に出力します。それ以外は
バックグラウンド・イメージになりま
す。バックグラウンド・イメージは最
上部よりマージン0で描画されます。

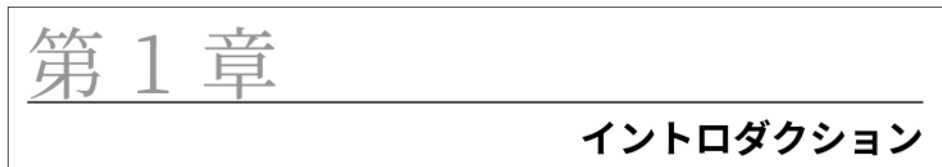


A.2 チャプター・デザイン

チャプターのデザインとして4種類のデザインを事前に用意しています。

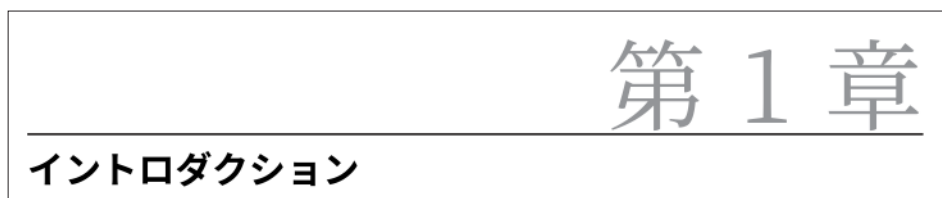
A.2.1 StandardBook

本書で使用しているスタイルです。大きめの章番号にアンダーラインを引き、ボールド体の見出しをラインの下側・右端に表記します。



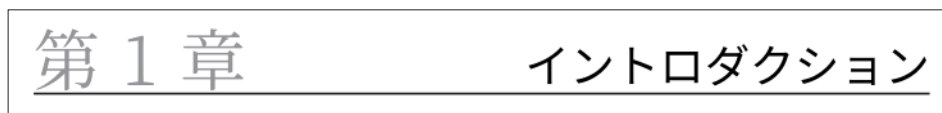
A.2.2 BigChapter1

StandardBookでの章番号を若干大きくし、見出し位置と入れ替えたものです。



A.2.3 BigChapter2

StandardBookで章番号を若干小さくして見出しを若干大きくし、通常の書体でアンダーラインの上に乗せたものです。



A.2.4 BigChapter3

章番号を極端に大きい番号だけの表記とし、通常の書体の見出しをその下に配置したものです。



付録 B

スタイル

B.1 スタイル・パラメーター一覧

スタイル・パラメータを `<style-info />` タグを使用して変更することが可能です。各パラメータは `<style-info name="name" value="value" />` の形式で指定します。

以下が欧文用スタイルのデフォルト値です。

表 B.1 スタイル・パラメーター一覧（欧文用スタイル）

パラメータ名	ArticleA4	BookA4
title.style	"StandardArticle"	"StandardBook"
abstract.title.text	"Abstract"	"Abstract"
chapter.style	-	"StandardBook"
hyphenationRule	"en"	"en"
toc.header	"Contents"	"Contents"
toc.lof	"List Of Figures"	"List Of Figures"
toc.lot	"List Of Tables"	"List Of Tables"
toc.appendix	"Appendix "	"Appendix "
table.label	"Table"	"Table"
image.label	"Fig"	"Fig"
image.fulllabel	"Figure"	"Figure"

次に、以下が和文用スタイルのデフォルト値となります。

表 B.2 スタイル・パラメーター一覧（和文用スタイル）

パラメータ名	JArticleA4	JBookA4
title.style	"StandardArticle"	"StandardBook"
abstract.title.text	"概要"	"本章の概要"
chapter.style	-	"StandardBook"
hyphenationRule	"ja"	"ja"
toc.header	"目次"	"目次"
toc.lof	"図目次"	"図目次"

パラメータ名	JArticleA4	JBookA4
toc.lot	"表目次"	"表目次"
toc.appendix	"付録 "	"付録 "
table.label	"表"	"表"
image.label	"図"	"図"
image.fulllabel	"図"	"図"

今後、対応可能なパラメータを随時増やしていく予定です。

付録 C

色一覧

C.1 色名称および RGB/CMYK 対応表

\color コマンドでは以下の色名称を使用できます。それぞれの RGB 値、CMYK 値との対応表も以下に示します。

表 C.1 色名称および RGB/CMYK 対応表

色名称	RGB 値	CMYK 値	色見本
aqua	[0x00, 0xff, 0xff]	[1.000, 0.000, 0.000, 0.000]	■色見本
aquamarine1	[0x87, 0xff, 0xd7]	[0.471, 0.000, 0.157, 0.000]	■色見本
aquamarine3	[0x5f, 0xd7, 0xaf]	[0.471, 0.000, 0.157, 0.157]	■色見本
black	[0x00, 0x00, 0x00]	[0.000, 0.000, 0.000, 1.000]	■色見本
blue	[0x00, 0x00, 0xff]	[1.000, 1.000, 0.000, 0.000]	■色見本
blue1	[0x00, 0x00, 0xff]	[1.000, 1.000, 0.000, 0.000]	■色見本
blue3	[0x00, 0x00, 0xd7]	[0.843, 0.843, 0.000, 0.157]	■色見本
blueviolet	[0x5f, 0x00, 0xff]	[0.627, 1.000, 0.000, 0.000]	■色見本
cadetblue	[0x5f, 0xaf, 0xaf]	[0.314, 0.000, 0.000, 0.314]	■色見本
chartreuse1	[0x87, 0xff, 0x00]	[0.471, 0.000, 1.000, 0.000]	■色見本
chartreuse2	[0x87, 0xd7, 0x00]	[0.314, 0.000, 0.843, 0.157]	■色見本
chartreuse3	[0x5f, 0xd7, 0x00]	[0.471, 0.000, 0.843, 0.157]	■色見本
chartreuse4	[0x5f, 0x87, 0x00]	[0.157, 0.000, 0.529, 0.471]	■色見本
cornflowerblue	[0x5f, 0x87, 0xff]	[0.627, 0.471, 0.000, 0.000]	■色見本
cornsilk1	[0xff, 0xff, 0xd7]	[0.000, 0.000, 0.157, 0.000]	■色見本
cyan1	[0x00, 0xff, 0xff]	[1.000, 0.000, 0.000, 0.000]	■色見本
cyan2	[0x00, 0xff, 0xd7]	[1.000, 0.000, 0.157, 0.000]	■色見本
cyan3	[0x00, 0xd7, 0xaf]	[0.843, 0.000, 0.157, 0.157]	■色見本
darkblue	[0x00, 0x00, 0x87]	[0.529, 0.529, 0.000, 0.471]	■色見本
darkcyan	[0x00, 0xaf, 0x87]	[0.686, 0.000, 0.157, 0.314]	■色見本
darkgoldenrod	[0xaf, 0x87, 0x00]	[0.000, 0.157, 0.686, 0.314]	■色見本
darkgreen	[0x00, 0x5f, 0x00]	[0.373, 0.000, 0.373, 0.627]	■色見本

色名称	RGB 值	CMYK 值	色見本
darkkhaki	[0xaf, 0xaf, 0x5f]	[0.000, 0.000, 0.314, 0.314]	■色見本
darkmagenta	[0x87, 0x00, 0xaf]	[0.157, 0.686, 0.000, 0.314]	■色見本
darkolivegreen1	[0xd7, 0xff, 0x87]	[0.157, 0.000, 0.471, 0.000]	■色見本
darkolivegreen2	[0xaf, 0xff, 0x5f]	[0.314, 0.000, 0.627, 0.000]	■色見本
darkolivegreen3	[0xaf, 0xd7, 0x5f]	[0.157, 0.000, 0.471, 0.157]	■色見本
darkorange	[0xff, 0x87, 0x00]	[0.000, 0.471, 1.000, 0.000]	■色見本
darkorange3	[0xd7, 0x5f, 0x00]	[0.000, 0.471, 0.843, 0.157]	■色見本
darkred	[0x87, 0x00, 0x00]	[0.000, 0.529, 0.529, 0.471]	■色見本
darkseagreen	[0x87, 0xaf, 0x87]	[0.157, 0.000, 0.157, 0.314]	■色見本
darkseagreen1	[0xd7, 0xff, 0xaf]	[0.157, 0.000, 0.314, 0.000]	■色見本
darkseagreen2	[0xaf, 0xff, 0xaf]	[0.314, 0.000, 0.314, 0.000]	■色見本
darkseagreen3	[0xaf, 0xd7, 0x87]	[0.157, 0.000, 0.314, 0.157]	■色見本
darkseagreen4	[0x5f, 0xaf, 0x5f]	[0.314, 0.000, 0.314, 0.314]	■色見本
darkslategray1	[0x87, 0xff, 0xff]	[0.471, 0.000, 0.000, 0.000]	■色見本
darkslategray2	[0x5f, 0xff, 0xff]	[0.627, 0.000, 0.000, 0.000]	■色見本
darkslategray3	[0x87, 0xd7, 0xd7]	[0.314, 0.000, 0.000, 0.157]	■色見本
darkturquoise	[0x00, 0xd7, 0xd7]	[0.843, 0.000, 0.000, 0.157]	■色見本
darkviolet	[0xaf, 0x00, 0xd7]	[0.157, 0.843, 0.000, 0.157]	■色見本
deeppink1	[0xff, 0x00, 0xaf]	[0.000, 1.000, 0.314, 0.000]	■色見本
deeppink2	[0xff, 0x00, 0x5f]	[0.000, 1.000, 0.627, 0.000]	■色見本
deeppink3	[0xd7, 0x00, 0x87]	[0.000, 0.843, 0.314, 0.157]	■色見本
deeppink4	[0xaf, 0x00, 0x5f]	[0.000, 0.686, 0.314, 0.314]	■色見本
deepskyblue1	[0x00, 0xaf, 0xff]	[1.000, 0.314, 0.000, 0.000]	■色見本
deepskyblue2	[0x00, 0xaf, 0xd7]	[0.843, 0.157, 0.000, 0.157]	■色見本
deepskyblue3	[0x00, 0x87, 0xd7]	[0.843, 0.314, 0.000, 0.157]	■色見本
deepskyblue4	[0x00, 0x5f, 0xaf]	[0.686, 0.314, 0.000, 0.314]	■色見本
dodgerblue1	[0x00, 0x87, 0xff]	[1.000, 0.471, 0.000, 0.000]	■色見本
dodgerblue2	[0x00, 0x5f, 0xff]	[1.000, 0.627, 0.000, 0.000]	■色見本
dodgerblue3	[0x00, 0x5f, 0xd7]	[0.843, 0.471, 0.000, 0.157]	■色見本
fuchsia	[0xff, 0x00, 0xff]	[0.000, 1.000, 0.000, 0.000]	■色見本
gold1	[0xff, 0xd7, 0x00]	[0.000, 0.157, 1.000, 0.000]	■色見本

色名称	RGB 值	CMYK 值	色見本
gold3	[0xd7, 0xaf, 0x00]	[0.000, 0.157, 0.843, 0.157]	■色見本
green	[0x00, 0x80, 0x00]	[0.502, 0.000, 0.502, 0.498]	■色見本
green1	[0x00, 0xff, 0x00]	[1.000, 0.000, 1.000, 0.000]	■色見本
green3	[0x00, 0xd7, 0x00]	[0.843, 0.000, 0.843, 0.157]	■色見本
green4	[0x00, 0x87, 0x00]	[0.529, 0.000, 0.529, 0.471]	■色見本
greenyellow	[0xaf, 0xff, 0x00]	[0.314, 0.000, 1.000, 0.000]	■色見本
grey	[0x80, 0x80, 0x80]	[0.000, 0.000, 0.000, 0.498]	■色見本
grey0	[0x00, 0x00, 0x00]	[0.000, 0.000, 0.000, 1.000]	■色見本
grey100	[0xff, 0xff, 0xff]	[0.000, 0.000, 0.000, 0.000]	
grey11	[0x1c, 0x1c, 0x1c]	[0.000, 0.000, 0.000, 0.890]	■色見本
grey15	[0x26, 0x26, 0x26]	[0.000, 0.000, 0.000, 0.851]	■色見本
grey19	[0x30, 0x30, 0x30]	[0.000, 0.000, 0.000, 0.812]	■色見本
grey23	[0x3a, 0x3a, 0x3a]	[0.000, 0.000, 0.000, 0.773]	■色見本
grey27	[0x44, 0x44, 0x44]	[0.000, 0.000, 0.000, 0.733]	■色見本
grey3	[0x08, 0x08, 0x08]	[0.000, 0.000, 0.000, 0.969]	■色見本
grey30	[0x4e, 0x4e, 0x4e]	[0.000, 0.000, 0.000, 0.694]	■色見本
grey35	[0x58, 0x58, 0x58]	[0.000, 0.000, 0.000, 0.655]	■色見本
grey37	[0x5f, 0x5f, 0x5f]	[0.000, 0.000, 0.000, 0.627]	■色見本
grey39	[0x62, 0x62, 0x62]	[0.000, 0.000, 0.000, 0.616]	■色見本
grey42	[0x6c, 0x6c, 0x6c]	[0.000, 0.000, 0.000, 0.576]	■色見本
grey46	[0x76, 0x76, 0x76]	[0.000, 0.000, 0.000, 0.537]	■色見本
grey50	[0x80, 0x80, 0x80]	[0.000, 0.000, 0.000, 0.498]	■色見本
grey53	[0x87, 0x87, 0x87]	[0.000, 0.000, 0.000, 0.471]	■色見本
grey54	[0x8a, 0x8a, 0x8a]	[0.000, 0.000, 0.000, 0.459]	■色見本
grey58	[0x94, 0x94, 0x94]	[0.000, 0.000, 0.000, 0.420]	■色見本
grey62	[0x9e, 0x9e, 0x9e]	[0.000, 0.000, 0.000, 0.380]	■色見本
grey63	[0xaf, 0x87, 0xaf]	[0.000, 0.157, 0.000, 0.314]	■色見本
grey66	[0xa8, 0xa8, 0xa8]	[0.000, 0.000, 0.000, 0.341]	■色見本
grey69	[0xaf, 0xaf, 0xaf]	[0.000, 0.000, 0.000, 0.314]	■色見本
grey7	[0x12, 0x12, 0x12]	[0.000, 0.000, 0.000, 0.929]	■色見本
grey70	[0xb2, 0xb2, 0xb2]	[0.000, 0.000, 0.000, 0.302]	■色見本

色名称	RGB 值	CMYK 值	色見本
grey74	[0xbc, 0xbc, 0xbc]	[0.000, 0.000, 0.000, 0.263]	■色見本
grey78	[0xc6, 0xc6, 0xc6]	[0.000, 0.000, 0.000, 0.224]	■色見本
grey82	[0xd0, 0xd0, 0xd0]	[0.000, 0.000, 0.000, 0.184]	■色見本
grey84	[0xd7, 0xd7, 0xd7]	[0.000, 0.000, 0.000, 0.157]	■色見本
grey85	[0xda, 0xda, 0xda]	[0.000, 0.000, 0.000, 0.145]	■色見本
grey89	[0xe4, 0xe4, 0xe4]	[0.000, 0.000, 0.000, 0.106]	■色見本
grey93	[0xee, 0xee, 0xee]	[0.000, 0.000, 0.000, 0.067]	■色見本
honeydew2	[0xd7, 0xff, 0xd7]	[0.157, 0.000, 0.157, 0.000]	■色見本
hotpink	[0xff, 0x5f, 0xd7]	[0.000, 0.627, 0.157, 0.000]	■色見本
hotpink2	[0xd7, 0x5f, 0xaf]	[0.000, 0.471, 0.157, 0.157]	■色見本
hotpink3	[0xd7, 0x5f, 0x87]	[0.000, 0.471, 0.314, 0.157]	■色見本
indianred	[0xd7, 0x5f, 0x5f]	[0.000, 0.471, 0.471, 0.157]	■色見本
indianred1	[0xff, 0x5f, 0x87]	[0.000, 0.627, 0.471, 0.000]	■色見本
khaki1	[0xff, 0xff, 0x87]	[0.000, 0.000, 0.471, 0.000]	■色見本
khaki3	[0xd7, 0xd7, 0x5f]	[0.000, 0.000, 0.471, 0.157]	■色見本
lightcoral	[0xff, 0x87, 0x87]	[0.000, 0.471, 0.471, 0.000]	■色見本
lightcyan1	[0xd7, 0xff, 0xff]	[0.157, 0.000, 0.000, 0.000]	■色見本
lightcyan3	[0xaf, 0xd7, 0xd7]	[0.157, 0.000, 0.000, 0.157]	■色見本
lightgoldenrod1	[0xff, 0xff, 0x5f]	[0.000, 0.000, 0.627, 0.000]	■色見本
lightgoldenrod2	[0xff, 0xd7, 0x87]	[0.000, 0.157, 0.471, 0.000]	■色見本
lightgoldenrod3	[0xd7, 0xaf, 0x5f]	[0.000, 0.157, 0.471, 0.157]	■色見本
lightgreen	[0x87, 0xff, 0x87]	[0.471, 0.000, 0.471, 0.000]	■色見本
lightpink1	[0xff, 0xaf, 0xaf]	[0.000, 0.314, 0.314, 0.000]	■色見本
lightpink3	[0xd7, 0x87, 0x87]	[0.000, 0.314, 0.314, 0.157]	■色見本
lightpink4	[0x87, 0x5f, 0x5f]	[0.000, 0.157, 0.157, 0.471]	■色見本
lightsalmon1	[0xff, 0xaf, 0x87]	[0.000, 0.314, 0.471, 0.000]	■色見本
lightsalmon3	[0xd7, 0x87, 0x5f]	[0.000, 0.314, 0.471, 0.157]	■色見本
lightseagreen	[0x00, 0xaf, 0xaf]	[0.686, 0.000, 0.000, 0.314]	■色見本
lightskyblue1	[0xaf, 0xd7, 0xff]	[0.314, 0.157, 0.000, 0.000]	■色見本
lightskyblue3	[0x87, 0xaf, 0xd7]	[0.314, 0.157, 0.000, 0.157]	■色見本
lightslateblue	[0x87, 0x87, 0xff]	[0.471, 0.471, 0.000, 0.000]	■色見本

色名称	RGB 值	CMYK 值	色見本
lightslategrey	[0x87, 0x87, 0xaf]	[0.157, 0.157, 0.000, 0.314]	■色見本
lightsteelblue	[0xaf, 0xaf, 0xff]	[0.314, 0.314, 0.000, 0.000]	■色見本
lightsteelblue1	[0xd7, 0xd7, 0xff]	[0.157, 0.157, 0.000, 0.000]	■色見本
lightsteelblue3	[0xaf, 0xaf, 0xd7]	[0.157, 0.157, 0.000, 0.157]	■色見本
lightyellow3	[0xd7, 0xd7, 0xaf]	[0.000, 0.000, 0.157, 0.157]	■色見本
lime	[0x00, 0xff, 0x00]	[1.000, 0.000, 1.000, 0.000]	■色見本
magenta1	[0xff, 0x00, 0xff]	[0.000, 1.000, 0.000, 0.000]	■色見本
magenta2	[0xff, 0x00, 0xd7]	[0.000, 1.000, 0.157, 0.000]	■色見本
magenta3	[0xd7, 0x00, 0xd7]	[0.000, 0.843, 0.000, 0.157]	■色見本
maroon	[0x80, 0x00, 0x00]	[0.000, 0.502, 0.502, 0.498]	■色見本
mediumorchid	[0xaf, 0x5f, 0xd7]	[0.157, 0.471, 0.000, 0.157]	■色見本
mediumorchid1	[0xff, 0x5f, 0xff]	[0.000, 0.627, 0.000, 0.000]	■色見本
mediumorchid3	[0xaf, 0x5f, 0xaf]	[0.000, 0.314, 0.000, 0.314]	■色見本
mediumpurple	[0x87, 0x87, 0xd7]	[0.314, 0.314, 0.000, 0.157]	■色見本
mediumpurple1	[0xaf, 0x87, 0xff]	[0.314, 0.471, 0.000, 0.000]	■色見本
mediumpurple2	[0xaf, 0x87, 0xd7]	[0.157, 0.314, 0.000, 0.157]	■色見本
mediumpurple3	[0x87, 0x5f, 0xd7]	[0.314, 0.471, 0.000, 0.157]	■色見本
mediumpurple4	[0x5f, 0x5f, 0x87]	[0.157, 0.157, 0.000, 0.471]	■色見本
mediumspringgreen	[0x00, 0xff, 0xaf]	[1.000, 0.000, 0.314, 0.000]	■色見本
mediumturquoise	[0x5f, 0xd7, 0xd7]	[0.471, 0.000, 0.000, 0.157]	■色見本
mediumvioletred	[0xaf, 0x00, 0x87]	[0.000, 0.686, 0.157, 0.314]	■色見本
mistyrose1	[0xff, 0xd7, 0xd7]	[0.000, 0.157, 0.157, 0.000]	■色見本
mistyrose3	[0xd7, 0xaf, 0xaf]	[0.000, 0.157, 0.157, 0.157]	■色見本
navajowhite1	[0xff, 0xd7, 0xaf]	[0.000, 0.157, 0.314, 0.000]	■色見本
navajowhite3	[0xaf, 0xaf, 0x87]	[0.000, 0.000, 0.157, 0.314]	■色見本
navy	[0x00, 0x00, 0x80]	[0.502, 0.502, 0.000, 0.498]	■色見本
navyblue	[0x00, 0x00, 0x5f]	[0.373, 0.373, 0.000, 0.627]	■色見本
olive	[0x80, 0x80, 0x00]	[0.000, 0.000, 0.502, 0.498]	■色見本
orange1	[0xff, 0xaf, 0x00]	[0.000, 0.314, 1.000, 0.000]	■色見本
orange3	[0xd7, 0x87, 0x00]	[0.000, 0.314, 0.843, 0.157]	■色見本
orange4	[0x87, 0x5f, 0x00]	[0.000, 0.157, 0.529, 0.471]	■色見本

色名称	RGB 值	CMYK 值	色見本
orangered1	[0xff, 0x5f, 0x00]	[0.000, 0.627, 1.000, 0.000]	■色見本
orchid	[0xd7, 0x5f, 0xd7]	[0.000, 0.471, 0.000, 0.157]	■色見本
orchid1	[0xff, 0x87, 0xff]	[0.000, 0.471, 0.000, 0.000]	■色見本
orchid2	[0xff, 0x87, 0xd7]	[0.000, 0.471, 0.157, 0.000]	■色見本
palegreen1	[0xaf, 0xff, 0x87]	[0.314, 0.000, 0.471, 0.000]	■色見本
palegreen3	[0x87, 0xd7, 0x87]	[0.314, 0.000, 0.314, 0.157]	■色見本
paleturquoise1	[0xaf, 0xff, 0xff]	[0.314, 0.000, 0.000, 0.000]	■色見本
paleturquoise4	[0x5f, 0x87, 0x87]	[0.157, 0.000, 0.000, 0.471]	■色見本
palevioletred1	[0xff, 0x87, 0xaf]	[0.000, 0.471, 0.314, 0.000]	■色見本
pink1	[0xff, 0xaf, 0xd7]	[0.000, 0.314, 0.157, 0.000]	■色見本
pink3	[0xd7, 0x87, 0xaf]	[0.000, 0.314, 0.157, 0.157]	■色見本
plum1	[0xff, 0xaf, 0xff]	[0.000, 0.314, 0.000, 0.000]	■色見本
plum2	[0xd7, 0xaf, 0xff]	[0.157, 0.314, 0.000, 0.000]	■色見本
plum3	[0xd7, 0x87, 0xd7]	[0.000, 0.314, 0.000, 0.157]	■色見本
plum4	[0x87, 0x5f, 0x87]	[0.000, 0.157, 0.000, 0.471]	■色見本
purple	[0xaf, 0x00, 0xff]	[0.314, 1.000, 0.000, 0.000]	■色見本
purple3	[0x5f, 0x00, 0xd7]	[0.471, 0.843, 0.000, 0.157]	■色見本
purple4	[0x5f, 0x00, 0xaf]	[0.314, 0.686, 0.000, 0.314]	■色見本
red	[0xff, 0x00, 0x00]	[0.000, 1.000, 1.000, 0.000]	■色見本
red1	[0xff, 0x00, 0x00]	[0.000, 1.000, 1.000, 0.000]	■色見本
red3	[0xd7, 0x00, 0x00]	[0.000, 0.843, 0.843, 0.157]	■色見本
rosybrown	[0xaf, 0x87, 0x87]	[0.000, 0.157, 0.157, 0.314]	■色見本
royalblue1	[0x5f, 0x5f, 0xff]	[0.627, 0.627, 0.000, 0.000]	■色見本
salmon1	[0xff, 0x87, 0x5f]	[0.000, 0.471, 0.627, 0.000]	■色見本
sandybrown	[0xff, 0xaf, 0x5f]	[0.000, 0.314, 0.627, 0.000]	■色見本
seagreen1	[0x5f, 0xff, 0xaf]	[0.627, 0.000, 0.314, 0.000]	■色見本
seagreen2	[0x5f, 0xff, 0x5f]	[0.627, 0.000, 0.627, 0.000]	■色見本
seagreen3	[0x5f, 0xd7, 0x87]	[0.471, 0.000, 0.314, 0.157]	■色見本
silver	[0xc0, 0xc0, 0xc0]	[0.000, 0.000, 0.000, 0.247]	■色見本
skyblue1	[0x87, 0xd7, 0xff]	[0.471, 0.157, 0.000, 0.000]	■色見本
skyblue2	[0x87, 0xaf, 0xff]	[0.471, 0.314, 0.000, 0.000]	■色見本

色名称	RGB 値	CMYK 値	色見本
skyblue3	[0x5f, 0xaf, 0xd7]	[0.471, 0.157, 0.000, 0.157]	■色見本
slateblue1	[0x87, 0x5f, 0xff]	[0.471, 0.627, 0.000, 0.000]	■色見本
slateblue3	[0x5f, 0x5f, 0xd7]	[0.471, 0.471, 0.000, 0.157]	■色見本
springgreen1	[0x00, 0xff, 0x87]	[1.000, 0.000, 0.471, 0.000]	■色見本
springgreen2	[0x00, 0xff, 0x5f]	[1.000, 0.000, 0.627, 0.000]	■色見本
springgreen3	[0x00, 0xd7, 0x5f]	[0.843, 0.000, 0.471, 0.157]	■色見本
springgreen4	[0x00, 0x87, 0x5f]	[0.529, 0.000, 0.157, 0.471]	■色見本
steelblue	[0x5f, 0x87, 0xaf]	[0.314, 0.157, 0.000, 0.314]	■色見本
steelblue1	[0x5f, 0xd7, 0xff]	[0.627, 0.157, 0.000, 0.000]	■色見本
steelblue3	[0x5f, 0x87, 0xd7]	[0.471, 0.314, 0.000, 0.157]	■色見本
tan	[0xd7, 0xaf, 0x87]	[0.000, 0.157, 0.314, 0.157]	■色見本
teal	[0x00, 0x80, 0x80]	[0.502, 0.000, 0.000, 0.498]	■色見本
thistle1	[0xff, 0xd7, 0xff]	[0.000, 0.157, 0.000, 0.000]	■色見本
thistle3	[0xd7, 0xaf, 0xd7]	[0.000, 0.157, 0.000, 0.157]	■色見本
turquoise2	[0x00, 0xd7, 0xff]	[1.000, 0.157, 0.000, 0.000]	■色見本
turquoise4	[0x00, 0x87, 0x87]	[0.529, 0.000, 0.000, 0.471]	■色見本
violet	[0xd7, 0x87, 0xff]	[0.157, 0.471, 0.000, 0.000]	■色見本
wheat1	[0xff, 0xff, 0xaf]	[0.000, 0.000, 0.314, 0.000]	■色見本
wheat4	[0x87, 0x87, 0x5f]	[0.000, 0.000, 0.157, 0.471]	■色見本
white	[0xff, 0xff, 0xff]	[0.000, 0.000, 0.000, 0.000]	
yellow	[0xff, 0xff, 0x00]	[0.000, 0.000, 1.000, 0.000]	■色見本
yellow1	[0xff, 0xff, 0x00]	[0.000, 0.000, 1.000, 0.000]	■色見本
yellow2	[0xd7, 0xff, 0x00]	[0.157, 0.000, 1.000, 0.000]	■色見本
yellow3	[0xd7, 0xd7, 0x00]	[0.000, 0.000, 0.843, 0.157]	■色見本
yellow4	[0x87, 0xaf, 0x00]	[0.157, 0.000, 0.686, 0.314]	■色見本