

# 最大限 T<sub>E</sub>X 入門

---

北海道大学理学部 ひとみさん

令和 2 年 4 月 9 日

**T<sub>E</sub>X 概観**

**T<sub>E</sub>X の使い方**

**L<sup>A</sup>T<sub>E</sub>X の書き方**

**エラーへの対処**

**T<sub>E</sub>X のディレクトリ構成**

## [改訂第 7 版] L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 美文書作成入門 (奥村晴彦・黒木祐介)

- 3 年毎に改版  
(第 8 版は 2020 年?)
- 「とりあえずこれを読め」



👤 美文書何章に記述があるか適宜参照します 👤

# やらない話

- 文書を書くのに使う<sup>コマンド</sup>命令  
(美文書 3, 5, 6, 7-11 章の大半)
- <sup>コマンド</sup>命令の作成  
(美文書 4 章)
- 🍷 T<sub>E</sub>X 言語 🤢

発表資料は <http://www.circle9.work/tex/> で公開  
ソースも公開しています

# TEX 概觀

---

👤 美文書 1 章 👤

# TeXでできること、特徴

文字を並べた PDF を作成することができる。

- きれいな数式

$$\int_0^{\infty} \frac{\sin x}{\sqrt{x}} dx = \sum_{k=0}^{\infty} \frac{(2k)!}{2^{2k(k!)^2}} \frac{1}{2k+1} = \frac{\pi}{2}$$

- 相互参照
- 処理の自動化
- フリーソフト
- 様々な OS で利用可能
- 実体はテキストファイル

# TeX でできないこと

- 見たまま編集

Word などを使えば良い、もしくは LyX?

- 図の描画

ほかソフトで作ってから埋め込めば良い、もしくは TikZ?

- フォントを自在に扱う

LuaTeX や XeTeX で使える fontspec パッケージを使えば.....

# T<sub>E</sub>X とは何か

- 1978 年に Donald E. Knuth が発表
  - 相当に古い
- 組版システム
  - 組版するためのソフトウェア
  - 組版するためのプログラミング言語
- L<sup>A</sup>T<sub>E</sub>X
  - T<sub>E</sub>X とは別物
  - T<sub>E</sub>X のマクロ体系 (フォーマット)



$\text{T}_{\text{E}}\text{X}$  の仲間にはたくさんある (ナトカ $\text{T}_{\text{E}}\text{X}$ )

- 処理系..... $\text{T}_{\text{E}}\text{X}$  (ソフトウェア) を拡張したもの  
 $\epsilon$ - $\text{T}_{\text{E}}\text{X}$ , pdf $\text{T}_{\text{E}}\text{X}$ , X $\exists$  $\text{T}_{\text{E}}\text{X}$ , Lua $\text{T}_{\text{E}}\text{X}$ , p $\text{T}_{\text{E}}\text{X}$ , up $\text{T}_{\text{E}}\text{X}$  など
- フォーマット  
L $\text{A}$  $\text{T}_{\text{E}}\text{X}$ , plain  $\text{T}_{\text{E}}\text{X}$ , Con $\text{T}_{\text{E}}\text{X}$ t など

全部まとめて $\text{T}_{\text{E}}\text{X}$ と呼ぶことも多い

# T<sub>E</sub>X ディストリビューション

T<sub>E</sub>X に関する成果物は、CTAN に集められる

- <https://www.ctan.org>
- ボランティアで成り立っている

CTAN から様々な T<sub>E</sub>X ディストリビューションへ

- T<sub>E</sub>X Live (<http://www.tug.org/texlive/>)
- W32TeX (<http://w32tex.org/>)
- MiK<sub>T</sub>E<sub>X</sub> (<https://miktex.org>)<sup>1</sup>

---

<sup>1</sup>日本語できない模様

# T<sub>E</sub>X ディストリビューション

T<sub>E</sub>X 本体やパッケージ以外にも、関連するバイナリも収録されている

- dvi ウェア (後述)
- kpathsea (後述)
- texdoc

texdoc → ドキュメントを検索するコマンド

例: `texdoc latex`

```
texdoc platexsheet-jsclasses
```

でコマンド一覧を表示

ワトソン氏 (朝倉卓人) 作成

# TeX Live について

最も普及している TeX ディストリビューション

膨大な数のパッケージやバイナリが含まれる

晩春に名前が変わる大型アップデート

2 月頃に更新停止 (frozen) ・次年度版の pretest

2019 年 4 月 31 日 TeX Live 2018 → TeX Live 2019

バイナリの更新は原則大型アップデート時のみ

パッケージ (テキストファイル) の更新は frozen 時以外はいつでも

大型アップデート時はインストールし直す必要

# TEXの使い方

---

👤 美文書 2 章 👤

# TeXのインストール

Windows なら W32TeX  
それ以外なら TeX Live をインストール

TeX Wiki<sup>2</sup>で調べてください。

インストールには数時間かかります。

---

<sup>2</sup><https://texwiki.texjp.org>

# TeX Live のアップデート

(TeX Live をインストールした場合)

```
sudo tlmgr update --self --all
```

上のコマンドで TeX Live をアップデート

定期的に行おう

年度が変わる大型アップデート時には再インストール

自分で書いた TeX ソースを、TeX 処理系に処理させることで、PDF ファイルを得る。

## のだが

歴史的経緯で、TeX 処理系は、PDF ファイルではなく、**dvi ファイル**を出力する<sup>3</sup>。

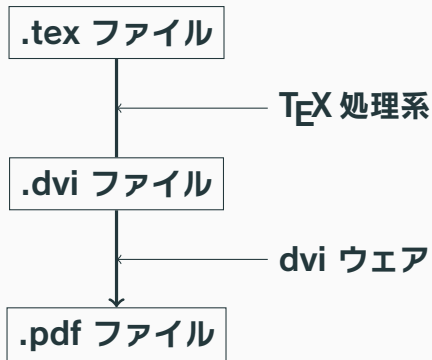
出てきた dvi ファイルを dvi ウェアで処理することによって、最終的な PDF を得ることができる。

---

<sup>3</sup>新しい処理系には、直接 PDF を出力するものもある。  
例: pdfTeX, LuaTeX, XeTeX



# TEXの使い方



dvi ファイルを pdf ファイルに変換するソフトウェア<sup>4</sup>

- dvi<sub>ps</sub>pdf, dvips など

使う処理系、フォーマット、dvi ウェアで、tex ファイルの書き方が変わるので注意

---

<sup>4</sup>PDF 以外にも、PostScript や、各種画像ファイルに変換するものもある

# 日本で一般的な方法

日本では

- p $\text{\LaTeX}$  + dvipdfmx
- up $\text{\LaTeX}$  + dvipdfmx

あたりが主流

Lua $\text{\LaTeX}$  も広まってきている

このスライドでは、主に p $\text{\LaTeX}$  + dvipdfmx を例にして話す

# L<sup>A</sup>T<sub>E</sub>X の書き方

---

👤 美文書 3 章 👤

## Listing 1: sample.tex

```
1 \documentclass[12pt,dvipdfmx]{jsarticle}
2 \usepackage[T1]{fontenc}
3 \usepackage{graphicx,xcolor}
4 \usepackage{otf}
5 \usepackage{newpxtext,newpxmath}
6 \usepackage{amsmath}
7 \usepackage[a6paper]{geometry}
8 \begin{document}
9 吾輩は\TeX である。名前はまだない。
10 \[e^{i\pi}=-1\]
11 \end{document}
```

**BOM なし UTF-8 で保存しましょう**

コマンドラインで以下を実行

```
platex sample
```

```
dvipdfmx sample
```

吾輩は TEX である。名前はまだない。

$$e^{i\pi} = -1$$

コマンド  
**命令**

\ で始まる、英文字（と和文文字）の列

もしくは \ のあとに数字か記号ひとつ

`\TeX` や `\^` など（コントロールワード 制御語とコントロールシンボル 制御文字）

**環境** `\begin{ナトカ}` と `\end{ナトカ}` で囲まれたもの

**コメント** `%` から行末まではコメント扱い（無視される）

## 特殊な文字

以下の文字は特殊文字

`% \ ^ _ ~ { } # & $`

## コマンド 命令についての注意

- <sup>コマンド</sup>命令の引数は { } で括る
- オプションな引数は [ ] で括る

{ } はカッコの対応を確認されるが、  
[ ] はカッコの対応を確認しない

例:

`\lstinputlisting[caption=[1]]{foo.tex}` は  
caption=[1 だけが [ ] に入っている判定  
→ [ ] に含めたい全体を { } で括ると解決  
`\lstinputlisting[{caption=[1]}]{foo.tex}`



# L<sup>A</sup>T<sub>E</sub>X 文書の構造

```
1 % クラスファイル( jsarticle.cls )を読み込む
2 \documentclass[dvipdfmx]{jsarticle}
3
4 % プリアンブル
5 %   パッケージ( ナツカ.sty )の読み込みや
6 %   文書全体の設定
7
8 \begin{document}
9 %   文書本体
10 \end{document}
```

```
\documentclass[dvipdfmx]{jsarticle}
```

**クラスファイルを読み込む → 版面構成の定義など**  
実体は `natcls.cls` というテキストファイル

## 主要なクラスファイル

- `jsarticle`, `jsreport`, `jsbook` (新ドキュメントクラス)
- `jlreq` (日本語組版処理の要件<sup>5</sup>対応)
- `beamer` (スライド用 日本語するには工夫が必要)
- `jarticle`, `jreport`, `jbook` (s なし) は非推奨

---

<sup>5</sup><https://www.w3.org/TR/jlreq/ja/>

```
\documentclass[dvipdfmx]{jsarticle}
```

## [ ] 中はオプション設定

フォントサイズ、見開きの設定など

```
\documentclass[12pt,dvipdfmx]{jsarticle}
```

**必ず使う dvi ウェアをオプションに設定する**

```
\begin{document}  
  ⋮  
\end{document}
```

文書本体は `\begin{document}` と `\end{document}` の間に書く

打ち込んだ文字がそのまま出力される（特殊文字は除く）

命令を利用できる

# 文書を書くときの注意

## 改行の扱い

- 改行は空白扱い
- 和文文字直後の改行は無視（空白にもならない）
- 連続した改行 → 改段落
- % は改行文字も含めて、行末まで無視する  
→ 空白は入らない

メール的なフォーマットで書ける

1行が長くなったら改行、改段落は空行

# 文書を書くときの注意

コントロールシーケンス

## 制御 綴 や空白の扱い

- 空白はいくつつなげても 1 つに吸収される
- 行頭行末の空白は無視される
- `\_` や `~` で空白を出力できる (`~` は行分割されない)
- コントロールワード 制御 語 コントロールワード 直後の空白は コントロールワード 制御 語 の区切りでしかない  
→ 無視される
- コントロールシンボル 制御 文 字 直後の空白は無視されない

`\TeX_ Live` → `TEX Live`  
`\TeX\ Live` → `TEX Live`

`\TeX_ Live` → `TEX Live`  
`{\TeX}_ Live` → `TEX Live`

# 文書を書くときの注意

その他の注意、使える命令、環境は

- `texdoc platexsheet-jsclasses`
- 美文書作成入門

を参照

**ググるより先に上を読みましょう**

ググって出てくる情報は軒並み古くて怪しい<sup>6</sup>

---

<sup>6</sup>ディスプレイ数式を  $\$ \$ \sim \$ \$$  で囲んだり、`\begin{eqnarray}`を使ったり

`\documentclass` から `\begin{document}` の間  
→ **プリアンブル (preamble)**

パッケージの読み込み・文書全体の設定をする

```
\usepackage[a4paper]{geometry}
```

→ `geometry` パッケージを、`a4paper` オプション付きで読み込む

本文を書くことはできない

逆に、プリアンブルでしか使えないコマンドもある

```
\usepackage など
```



# パッケージとは

様々な便利機能を提供

他のプログラミング言語で言うところのライブラリ

実体は、`hatカ.sty` というテキストファイル

例: ゆきだるま 🌨 を書きたい!

→ `scsnowman` パッケージ

→ `\usepackage{scsnowman}`

→ `\scsnowman[scale=3,hat,arms,buttons]`

→  素敵!

# パッケージの使い方

## 1. 用途からパッケージを探す

ググるしかない もしくは CTAN でググる<sup>7</sup>

## 2. プリアンブルで

`\usepackage[オプション]{パッケージ名}`

## 3. 使う

## 4. 使い方がわからなくなるので `texdoc パッケージ名`

---

<sup>7</sup>英語なので厳しい; ググるを誤用してるのは承知です

# おすすめプリアンブル

```
1 % フォントエンコード( 文字化けしないように )
2 \usepackage[T1]{fontenc}
3 % 図の挿入、色を扱う
4 \usepackage{graphicx,xcolor}
5 % フォントをアイコンジにしてくれる
6 \usepackage{otf}
7 % フォントを変更( デフォルトはサイズ指定に不具合 )
8 \usepackage{newpxtext,newpxmath} % Palatino
9 %%% \usepackage{newtxtext,newtxmath} % Times
10 %%% \usepackage{lmodern} % Latain Modern
11 % 数学するなら必要
12 \usepackage{amsmath}
13 % 用紙サイズの設定
14 \usepackage[a4paper]{geometry}
```

**LaTeX** を理解するまでは、これをそのまま使おう

# 書き方まとめ

```
1 \documentclass[12pt,dvipdfmx]{jsarticle}
2 % プリアンブル
3 \usepackage[T1]{fontenc}
4 \usepackage{graphicx,xcolor}
5 \usepackage{otf}
6 \usepackage{newpxtext,newpxmath}
7 \usepackage{amsmath}
8 \usepackage[a4paper]{geometry}
9
10 \begin{document}
11 ドキュメント本文
12 \end{document}
```

# エラーへの対処

---

👤 美文書 2 章 9 節 👤

⚠️ご注意ください⚠️



**エラー対処が上手かどうかで  
作業効率が激変します**



# エラーに遭遇する

$\text{T}_{\text{E}}\text{X}$  はプログラミング言語なので、書き方を間違えるとエラーが出る

$\backslash\text{TEX}$  と書いてしまうと.....

! Undefined control sequence.

1.3  $\backslash\text{TEX}$

「？」と聞かれるので、   のどれかを押す

# エラーが出たら



処理を中断して終了



処理を継続、ログは標準出力しない



処理を継続、再びエラーが出ると止まる



を数回連打するのがおすすめ

大抵、複数のエラーが混入しているため

連続して 5 回以上エラーが出てきたら  するべし



## ? 以外のプロンプトの場合

Enter file name:




\usepackage でパッケージ名を間違えたときに出がち

 を押して Enter

---

\*

\end{document} を忘れたときに出がち

1. \stop と打って Enter
2. \aaa (未定義の コントロールシーケンス 制御 綴) を打って Enter  
→ ? のプロンプト → 
3.  

# エラーメッセージの見方

! You can't use 'macro parameter character #' in horizontal mode.

1.3 O-oooooooooooo #

AAAAE-A-A-I-A-U-

?

**! エラーメッセージ**

**1. 行数  $\text{\TeX}$  が読み込んだもの**

**まだ読み込んでいないもの**

**エラーが出た行に戻って治せばいいのだが.....**

# エラーへの対処

## 大体のエラーの原因

- ・ コントロールシーケンス 制御 綴 の綴りのマチガイ
- ・ 環境の閉じ忘れ
- ・ ものの不均衡 (`{ }`、`$ $`、`\left \right` など)
- ・ コマンド 命令の用法のマチガイ

エラーが起きた行付近で上がらないか確認

コマンドの用法のマチガイ →

`texdoc <パッケージ名>` で確認

# 対処しにくいエラー

## おさらい

$\text{\LaTeX}$  は  $\text{\TeX}$  のフォーマット (マクロ体系)

→  $\text{\LaTeX}$  レベルのエラーと、 $\text{\TeX}$  レベルのエラーがある

起きたエラーによっては、原因が特定しにくい

例: ! Missing number, treated as zero.

処理中に外部ファイルを読み込むこともある

→ 行番号が、どのファイルの行番号かわからなくなる

# エラーを起こさないために

- タイプセットを細かく行う
- 開いた環境はすぐ閉じる
- 全角空白「」を使わない  
段落頭の字下げは `\parindent` で設定  
欧文クラスで、一番最初のパラグラフを字下げしたい場合 → `indentfirst` パッケージ
- `\verb` 命令もなるべく避ける  
コマンド  
命令の引数にあるとエラー (`\verb` の呪い)

それでも意味不明なエラーが起きる

# パッケージの衝突

```
1 \documentclass{jsarticle}
2 %%% 略
3 \usepackage{mathabx} % いろんな記号を使いたい
4 \usepackage{yhmath} % 大きいカッコを綺麗にしたい
5 \begin{document}
6 \[e^{i\pi}=-1\]
7 \end{document}
```



! LaTeX Error: Command `\iint` already defined.  
Or name `\end...` illegal, see p.192 of the manual.  
1.645 ...d{\iint}{\DOTSI\protect\MultiIntegral{2}}

**mathabx と ymash が同じ<sup>コマンド</sup>命令を定義 → エラー**

パッケージを読み込む順番を変えたら誤魔化せる場合も

→ 読み込む順番を変えてみる

→ どうしようもなければ諦める

パッケージが日本語対応してなくてエラーが起きる場合も

→ (u)pL<sup>A</sup>T<sub>E</sub>X なら plautopatch パッケージ<sup>8</sup>を試してみる

---

<sup>8</sup><https://aminophen.github.io/slide/hytexconf18.pdf>

# エラーが解消できなくてどうしようもないときは

## とりあえずエラーメッセージでググってみる

これで解決できたら苦労しないんだよなあ わかりにくいエラーメッセージが嫌ならば、SATySFj.....?

## わからなければ詳しい人に聞く

TeX Forum<sup>9</sup> で質問 Twitter でつぶやくのも実は有用

## 実はバグを踏んでいる可能性も

---

<sup>9</sup><https://oku.edu.mie-u.ac.jp/tex/>



## わかりにくいエラー①

[a] 真鍋 \\ [b] いつき

→! Missing number, treated as zero.

\\ (強制改行) 命令は、実はオプション引数をもつ

→\\[<長さ>]

\\{} のように {} で区切ると解決

[a] 真鍋 \\{} [b] いつき

→[a] 真鍋

[b] いつき

## わかりにくいエラー②

```
\section{$\overrightarrow{\mbox{ぶーん}}$}
```

→! Illegal parameter number in definition of \reserved@a.

エラーが起きる原因 → 🤔<sup>10</sup>

`\section` や `\caption` で変なエラーが出たら、  
引数に入ってるヤバそうな命令コマンドに `\protect` を前置

```
\section{$\protect\overrightarrow{\mbox{ぶーん}}$}
```

→  $\overrightarrow{\text{ぶーん}}$

---

<sup>10</sup>`\section` の引数は動くので、脆弱な`\overrightarrow` は保護しなければならない

# TEXのディレクトリ構成

---

 美文書 付録 B 3 節 

TEX 関連ファイルを入れるディレクトリ構成

TEXMF ← TEX+ METAFONT<sup>11</sup>

複数の TEXMF ツリーを使い分けるのが主流  
多重 TEXMF ツリー

確認方法: `kpsewhich -var-value TEXMF`

---

<sup>11</sup>METAFONT は Knuth が作ったフォント記述言語

# 多重 TEXMF ツリーの利点

ディストリビューションが用意したファイルと、自分がインストールしたファイルを分離できる

ディストリビューションを更新しても、自分のインストールしたファイルは削除されない

ディストリビューションが用意したファイル

→ `kpsewhich -var-value TEXMFDIST`

自分がインストールするファイル

→ `kpsewhich -var-value TEXMFLOCAL`

全ユーザーが使える

→ `kpsewhich -var-value TEXMFHOME`

そのユーザーが使える

TEXMF ツリーからファイルを検索する

kpathsea ← Karl Berry 氏によって作られた path searching

例: `kpsehwhich hmtrump.sty`

→/usr/local/texlive/texmf-local/tex/latex/local/hmtrump.sty

TEXMF ツリーに作られた <sup>ファイル一覧</sup> `ls-R` を見て検索する

TEXMF ツリーに変更 → <sup>ファイル一覧</sup>Is-R を更新する必要<sup>12</sup>

```
sudo mktexlsr
```

sudo tlmgr update --self --all 後も必要だが、

最近は tlmgr が自動でやっているっぽい

---

<sup>12</sup>Is-R を使わない運用方法もあるらしいですが、やったことがないのでわかりません

# パッケージをインストールする

ディストリビューションに含まれないパッケージを使いたい

→ 自分で TEXMF ツリーに入れる必要

作業ディレクトリに置いてもよいけれども

正しい場所に入れなければ正常に使えない



# パッケージをインストールする場所

かなり複雑なので **TeX Wiki<sup>13</sup>参照**

中身を覗いてみればなんとなくわかる

**まずはパッケージドキュメントを確認**

**あまり失敗しない方法**

ドキュメントに記載がない場合

- ドキュメントは `$TEXMFLOCAL/doc/latex/パッケージ名`
- その他は `$TEXMFLOCAL/tex/latex/パッケージ名`

にディレクトリを作って、コピー

フォント関連などはもっと複雑で、上記の通りでは無理です.....

---

<sup>13</sup><https://texwiki.texjp.org/?TeX%20> のディレクトリ構成

**とりあえず美文書は読んでください**

**もっと詳しく知りたい場合**

- **T<sub>E</sub>X Wiki**

<https://texwiki.texjp.org>

- **Acetaminophen's diary**

<http://acetaminophen.hatenablog.com>

**以下のブログは、もっと沼にハマりたい人向け**

- **ラングラグー**

<https://blog.wtsnjp.com>

- **マクロツイーター**

<https://zrbabbler.hatenablog.com>