

TEX の字句解析

荒田 実樹

2016 年 6 月 9 日

目次

TeX の処理の概要

字句解析とは

空白の扱い

改行の扱い

字句解析への介入

T_EX の処理の概要

T_EX の処理は大きく 4 段階に分けられる。

1. 字句解析

- ▶ 入力ファイルから「トークン」を切り出す
- ▶ コメントを読み飛ばしたりもする

2. 展開

- ▶ マクロの展開
- ▶ 条件分岐
- ▶ なかなか奥が深い

3. 実行

- ▶ 新しくマクロを定義する
(plain T_EX なら `\def`, L^AT_EX なら `\newcommand`)
- ▶ 字句解析器に介入できる (!)
- ▶ ほか、色々

4. 組版

字句解析とは？

普通のプログラミング言語のソースコードは、「トークン」と呼ばれる最小単位に分割できる。

```
1 | if (Math.PI > 3.14) {  
2 |     console.log("Hello world!");  
3 |     // This is a comment  
4 | }
```

字句解析とは？

普通のプログラミング言語のソースコードは、「トークン」と呼ばれる最小単位に分割できる。

```
1 | if (Math.PI > 3.14) {  
2 |     console.log("Hello world!");  
3 |     // This is a comment  
4 | }
```

トークンの種類：

キーワード、記号、識別子、数値リテラル、文字列リテラル、etc.
空白やコメントは読み飛ばされる。

T_EX の場合は？

T_EX におけるトークンの種類：

- ▶ 普通の文字（例：a, b, c, 0, 1, 2, @, +, |）
- ▶ 空白文字
- ▶ 特殊文字 \$ # ^ _ { } & ~ % \
- ▶ コントロールシーケンス（例：\foo）
 - ▶ T_EX のコマンドの名前に使える
 - ▶ 通常は、バックスラッシュ \ の後にアルファベットをいくつか続けたもの
 - ▶ バックスラッシュの後に記号一文字というパターンもある（例：\!, \\）

コメント（% から始まる）および一部の空白は、字句解析時に読み飛ばされる。

空白の扱い

T_EX では、場所によって空白が無視されたり無視されなかったりする。例えば、以下の状況では空白が無視される。

- ▶ コントロールシーケンスの直後
(例：`\foo {bar} {piyo}`)
- ▶ 行頭
- ▶ 空白文字の直後
- ▶ コマンドの引数の直前 (例：`\foo {bar} {piyo}`)
- ▶ 数式モード中の空白文字 (例： $\$a+b\$$)

空白の扱い

どの段階で無視される？

1. 字句解析で無視される空白（このスライドで解説）
 - ▶ コントロールシーケンスの直後（例：`\foo{bar}_\{piyo}`）
 - ▶ 行頭
 - ▶ 空白文字の直後
2. 展開時に無視される空白
 - ▶ コマンドの引数の直前など（例：`\foo_{bar}{piyo}`）
3. 実行時に無視される空白
 - ▶ 数式モード中の空白文字など（例：`$a+b$`）

空白の扱い — 無視される状況 その1

コントロールシーケンスの直後の空白は無視される。

- ▶ `\TeX!!!` と書いても `\TeX□!!!` と書いても同じ。
- ▶ ただし、記号一文字の場合は直後の空白は無視されない。
 - ▶ `keisan\%sugaku` → `keisan%sugaku`
 - ▶ `keisan\%_sugaku` → `keisan% sugaku`
- ▶ 空白文字一文字の場合は直後の空白は無視される。
 - ▶ `\renewcommand{_}{*}` した状況で、
 - ▶ `keisan_sugaku` → `keisan*sugaku`
 - ▶ `keisan_□sugaku` → `keisan*sugaku`

空白の扱い — 無視される状況 その2

行頭の空白は無視される。

例

以下の2つは等価。

```
1 | Happy
2 | \TeX{}\_Life!
```

```
1 | Happy
2 | \_ \_ \TeX{}\_Life!
```

ちなみに、この例では `\TeX` の後の空白を無視させないために `{}` を挟んでいる。

改行の扱い

改行は 1 個の半角空白として取り扱われる。

例

以下の 2 つは等価。

```
1 | a
2 | b
```

```
1 | a_b
```

- ▶ 半角空白を入れずにソースコード上で改行したい場合は、

```
1 | a%
2 | b
```

という風に、コメントを使うと良い。(ab と等価)

- ▶ 「コントロールシーケンスの直後の空白は無視される」ので、コントロールシーケンスの直後の改行も無視される。

改行の扱い — pT_EX の場合

pT_EX では、和文の直後の改行は無視される。

- ▶ さっきのルールによれば、以下の2つは T_EX 的には同じはず。

1		わあい
2		うすしお

1		わあい□うすしお
---	--	----------

- ▶ しかし、pT_EX では違う：前者は空白が入らない。

改行の扱い — 改段落

空行は、“`\par`” というコントロールシークエンスとして扱われる。

例

以下の2つは等価。

```
1 | 「イヤーツ！」  
2 |  
3 | 「グワーツ！」
```

```
1 | 「イヤーツ！」  
2 | \par  
3 | 「グワーツ！」
```

- ▶ `\par` は通常は改段落コマンドを表す。
- ▶ 空行が2つあったら `\par` 2つになる。(改段落コマンドは2つ以上あっても効果は同じ)

改行の扱い — 改段落

空白のみからなる行も空行として扱われる (“\par” になる)。

例

以下の2つは等価。つまり、どちらも改段落される。

```
1 | 「イヤーツ！」  
2 |  
3 | 「グワーツ！」
```

```
1 | 「イヤーツ！」  
2 | □□□  
3 | 「グワーツ！」
```

しかし、次のように、コメントのみからなる行は改段落にならない。

```
1 | 「イヤーツ！」  
2 | %  
3 | 「グワーツ！」
```

字句解析への介入

T_EX では、実行中のプログラムが自身の字句解析に介入できる！

- ▶ T_EX のプログラムの実行結果によって、特殊文字の扱いを変更できる。
- ▶あるいは、新たに特殊文字を作ったりできる。

字句解析への介入 — 例

L^AT_EX のコマンドにも、字句解析に介入するものがある。

- ▶ 特殊文字を無効化するコマンド・環境
 - ▶ `\verb` コマンド
 - ▶ `verbatim` 環境
 - ▶ `comment` 環境
 - ▶ `alltt` 環境：`\`, `{`, `}`以外の特殊文字を無効化する
- ▶ `\makeatletter`, `\makeatother` コマンド
 - ▶ コマンド名に “@” を使えるようにする。

字句解析への介入 — 制限

すでに字句解析が終わった部分へは介入できない。

例

```
1 | \newcommand{\foo}[1]{#1}  
2 | \verb+\hello_ world+  
3 | \foo{\verb+\hello_ world+}
```

- ▶ 2 番目の `\verb` は期待通りに動作するか？

字句解析への介入 — 制限

すでに字句解析が終わった部分へは介入できない。

例

```
1 | \newcommand{\foo}[1]{#1}
2 | \verb+\hello_world+
3 | \foo{\verb+\hello_world+}
```

- ▶ 2 番目の `\verb` は期待通りに動作するか？ → No!
- ▶ 2 番目の `\verb` が実行される時点で、後ろの部分が `+ \hello w o r l d +` という風に字句解析済み。
- ▶ 特殊文字を無効化できずに、エラーになる。
- ▶ `\verb` 等の「字句解析に介入する」コマンドは、マクロの引数の中では使えない（重要）

カテゴリーコード

TeXでは、字句解析時に読み取った文字に対して「カテゴリーコード」と呼ばれる整数を割り当てる。

この「カテゴリーコード」によって、その文字がどう扱われるかが決まる。

- 0. Escape character: \
- 1. Beginning of group: {
- 2. End of group: }
- 3. Math shift: \$
- 4. Alignment tab: &
- 5. End of line
- 6. Parameter character: #
- 7. Superscript: ^
- 8. Subscript: _

(続く)

カテゴリーコード

(続き)

10. Space

11. Letter: アルファベット。

- ▶ コントロールシーケンスを読む際に、アルファベットとして扱われる。

12. Other: 記号や数字。

13. Active: 半角チルダ (~) .

- ▶ バックスラッシュなしで、単独でコマンド名として使える文字の事。 `\newcommand{~}{...}` みたいなことができる。
- ▶ これに対し、半角チルダ以外の記号、例えば @ では `\newcommand{@}{...}` とはできない。

14. Comment character: %

字句解析への介入 — 悪用例

カテゴリーコードを変更すると、例えば、普通の文字・記号をコマンド名として使える。

例

文章中の句読点「、。」を、カンマとピリオド「,.」に変える

```
1 | \catcode‘、 =13_\def、 {, }%  
2 | \catcode‘。 =13_\def。 {, . }%  
3 | 最近、学校が好きだ。 %→「最近, 学校が好きだ.」になる
```



「自分が何をやっているかわかっていて」「何が起こっても責任を取れる」上級者向け。

説明しなかったこと & 参考文献

字句解析について説明しなかったこと

- ▶ カテゴリーコード 9 (ignored) と カテゴリーコード 15 (invalid character)
- ▶ Superscript 2 つ (\sim) による制御文字の挿入 (例: $\sim M$)

参考文献

-  Donald Knuth, *The T_EXbook*, Addison-Wesley, 1984.
特に Chapter 7: How T_EX Reads What You Type が字句解析について扱っている。
-  Victor Eijkhout, *T_EX by Topic*, Addison-Wesley, 1991.
著者の Web サイトから PDF を無料でダウンロード可能。