LaTeX3 教程一：简介

项子越

ziyue.alan.xiang@gmail.com

https://github.com/xziyue/latex3-chinese-video

2020 年 8 月 25 日

- 提供类似现代编程语言的语法与命名
- 简化 LaTeX 的宏展开控制
- 标准化 LaTeX 各个功能的接口

- LaTeX3 并不是为一般用户而设计的：它是用来实现高级用户接口的一套底层方法。

```
\def\FV@GetLine{\@noligs\expandafter\FV@CheckScan\FancyVerbGetLine}
%% DG/SR modification end
\begingroup
\catcode`\^^M=\active%
\gdef\FancyVerbGetLine#1^^M{%
  \@nil%
  \FV@CheckEnd{#1}%
  \ifx\@tempa\FV@EnvironName%            % True if end is found
    \ifx\@tempb\FV@@@CheckEnd\else\FV@BadEndError\fi%
    \let\next\FV@EndScanning%
  \else%
    \def\FV@Line{#1}%
    \def\next{\FV@PreProcessLine\FV@GetLine}%
  \fi%
  \next}%
\endgroup
```

（节选自fancyvrb）

```
\newcommand{\testcmda}{abc}
\newcommand{\testcmdb}{def\testcmda}
\par\testcmdb
\renewcommand{\testcmda}{def}
\par\testcmdb
```

defabc
defdef

```
string a("abc");

string func_1(string *a){
    // 如果原 a 的值发生改变，那么返回值也会改变
    return "def" + *a;
}

string func_2(string a){
    // 即使原 a 的值发生改变，返回值也不会改变
    return "def" + a;
}
```

```
\par\uppercase{abcdefghi}
\newcommand{\testcmda}{def}
\par\uppercase{abc\testcmda ghi}
```

ABCDEFGHI
ABCdefGHI

- 整数
- 浮点数
- 布尔逻辑
- 凭据表（token list）
- 字符串
- 文件 IO

我们键入 tex 文件中的所有内容都可视作凭据表。

凭据表内的三类对象：

- 字符
- 命令
- 凭据表

```
{abc \def {abc\def} ghi}
```

{abc \def {abc\def} ghi}

LaTeX3 命名法提倡将函数的来源以及参数类型、变量的类型编码到其名字内。

- 可以更方便地让用户区别命令与变量
- 可以避免不同宏包之间命令的冲突
- LaTeX 并不拥有真正的类型系统，这样的命名方式可以让用户在编程时自行检查错误
- 只是一套指导意见，并不是强制性的要求

\< 作用域 >_< 介绍 >_< 类型 >　　\< 作用域 >__< 介绍 >_< 类型 >

作用域

- l：局部变量
- g：全局变量
- c：常量

```
\l_tmpa_tl
\g_tmpa_int
\c_left_brace_str
```

类型

- tl: 凭据表
- str：字符串
- int：整型
- fp：浮点数
- seq：队列
- dim：尺度/长度

⋮

`\<` 模块 `>_<` 介绍 `>:<` 参数列表 `>` `\__<` 模块 `>_<` 介绍 `>:<` 参数列表 `>`

常用的参数类型：

- N：接收一个命令，传递命令本身。
- V：与 N 类似，但是传递命令的值。
- n：接收一个凭据表。
- o：与 n 类似，但是对凭据表内的内容进行一次展开。
- x: 与 n 类似, 但是对凭据表内的内容进行递归展开。
- T/F：与 n 类似，用于判断语句中，根据判断结果执行 T/F 代码。
- c：接收一个凭据表，返回以其为名字的命令。
- p：参数列表（#1#2...）

```
\tl_item:Nn
\bool_if:nTF
\tl_put_right:Nx
```

如何使用 LaTeX3?

1. `\usepackage{expl3}`
2. 启用 LaTeX3 语法：`\ExplSyntaxOn`
3. 关闭 LaTeX3 语法：`\ExplSyntaxOff`

最小示例

```
\documentclass{article}
\usepackage{expl3}
\begin{document}
\ExplSyntaxOn
%LaTeX3 代码
\ExplSyntaxOff
\end{document}
```

关于`\ExplSyntaxOn`

- 所有空格及换行都会被忽略
- 下划线（_）和冒号（:）等同于英文字母

```
\ExplSyntaxOn
\cs_set:Npn \my_factorial:n #1 {
    \int_set:Nn \l_tmpa_int {1}
    \seq_clear:N \l_tmpa_seq
    \int_step_inline:nn {#1} {
        \seq_put_right:Nn \l_tmpa_seq {##1}
        \int_set:Nn \l_tmpa_int {\l_tmpa_int * ##1}
    }
    $\seq_use:Nn \l_tmpa_seq {\times} = \int_use:N \l_tmpa_int$
}
\par\my_factorial:n {3}
\par\my_factorial:n {7}
\ExplSyntaxOff
```

$1 \times 2 \times 3 = 6$

$1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 = 5040$