

The `l3tl-build` package: building token lists*

The L^AT_EX3 Project[†]

Released 2015/07/14

1 `l3tl-build` documentation

This module provides no user function: it is meant for kernel use only.

There are two main ways of building token lists from individual tokens. Either in one go within an `x`-expanding assignment, or by repeatedly using `\tl_put_right:Nn`. The first method takes a linear time, but only allows expandable operations. The second method takes a time quadratic in the length of the token list, but allows expandable and non-expandable operations.

The goal of this module is to provide functions to build a token list piece by piece in linear time, while allowing non-expandable operations. This is achieved by abusing `\toks`: adding some tokens to the token list is done by storing them in a free token register (time $O(1)$ for each such operation). Those token registers are only put together at the end, within an `x`-expanding assignment, which takes a linear time.¹ Of course, all this must be done in a group: we can't go and clobber the values of legitimate `\toks` used by L^AT_EX 2_ε.

Since none of the current applications need the ability to insert material on the left of the token list, I have not implemented that. This could be done for instance by using odd-numbered `\toks` for the left part, and even-numbered `\toks` for the right part.

*This file describes v5662, last revised 2015/07/14.

[†]E-mail: latex-team@latex-project.org

¹If we run out of token registers, then the currently filled-up `\toks` are put together in a temporary token list, and cleared, and we ultimately use `\tl_put_right:Nx` to put those chunks together. Hence the true asymptotic is quadratic, with a very small constant.

1.1 Internal functions

<code>__tl_build:Nw</code>	<code>__tl_build:Nw <tl var> ...</code>
<code>__tl_gbuild:Nw</code>	<code>__tl_build_one:n {<tokens>} ...</code>
<code>__tl_build_x:Nw</code>	<code>__tl_build_one:n {<tokens>} ...</code>
<code>__tl_gbuild_x:Nw</code>	<code>...</code>
	<code>__tl_build_end:</code>

Defines the $\langle tl\ var \rangle$ to contain the contents of $\langle tokens1 \rangle$ followed by $\langle tokens2 \rangle$, *etc.* This is built in such a way to be more efficient than repeatedly using `\tl_put_right:Nn`. The code in “...” does not need to be expandable. The commands `__tl_build:Nw` and `__tl_build_end:` start and end a group. The assignment to the $\langle tl\ var \rangle$ occurs just after the end of that group, using `\tl_set:Nn`, `\tl_gset:Nn`, `\tl_set:Nx`, or `\tl_gset:Nx`.

<code>__tl_build_one:n</code>	<code>__tl_build_one:n {<tokens>}</code>
<code>__tl_build_one:(o x)</code>	

This function may only be used within the scope of a `__tl_build:Nw` function. It adds the $\langle tokens \rangle$ on the right of the current token list.

<code>__tl_build_end:</code>	Ends the scope started by <code>__tl_build:Nw</code> , and performs the relevant assignment.
-------------------------------	---

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

T	
T _E X and L ^A T _E X 2 _ε commands:	
<code>\toks</code>	<u>1</u> , <u>1</u> , <u>1</u> , <u>1</u> , <u>1</u>
tl commands:	
<code>__tl_build:Nw</code>	<u>2</u> , <u>2</u> , <u>2</u> , <u>2</u> , <u>2</u>
<code>__tl_build_end:</code>	<u>2</u> , <u>2</u> , <u>2</u>
<code>__tl_build_one:n</code>	<u>2</u> , <u>2</u> , <u>2</u> , <u>2</u>
<code>__tl_build_x:Nw</code>	<u>2</u>
<code>__tl_gbuild:Nw</code>	<u>2</u>
<code>__tl_gbuild_x:Nw</code>	<u>2</u>
<code>\tl_gset:Nn</code>	<u>2</u>
<code>\tl_gset:Nx</code>	<u>2</u>
<code>\tl_put_right:Nn</code>	<u>1</u> , <u>2</u>
<code>\tl_put_right:Nx</code>	<u>1</u>
<code>\tl_set:Nn</code>	<u>2</u>
<code>\tl_set:Nx</code>	<u>2</u>