TOKENS

tokens as I see them

context 2020 meeting

About tokens

- Like nodes, it's a common term used in programming.
- In TeX The Program tokens and nodes are therefore omni-present.
- For most users they are irrelevant concepts.
- But we will explain them anyway.
- Let's try to avoid the snobbish token-speak sometimes heard in the community.
- So . . . I won't correct you as long as you don't correct me.
- Let's now enter the world of tokens in the naïve way.

What are tokens

- It is an internal data structure, effectively a (32 bit) integer.
- This integer encodes a command (opcode) and an char code (operand).
- But often it's not a character but more a sub command.
- Input is converted into tokens.
- Tokens are either expanded (interpreted) or stored.
- When they are stored they are part of a larger data structure, a memory word.
- Token memory is an array of such memory words.
- The token memory ‘word’ has two integers: a token value and an index into token memory.
- That way TEX can have forward linked lists of tokens.
- A hash table maps control sequences onto indices into token memory.

Some implementation details

- Sometimes there is special head token at the start.
- A head token makes for easier appending of extra tokens.
- Shared lists use the head node for a reference count.
- Original TEX uses global temporary lists.
- This is needed when we expand (nested) and need to report issues.
- This is not needed when we just serialize (which we do a lot in LuaTEX).
- So, this is all optimized for performance and memory consumption.
- Freed tokens are collected in a cache so tokens can get scattered.
- In LuaMetaTEX we stay as close to original TEX as possible.
- But the Lua interfaces force us to occasionally divert.

A schematic view of tokens

A token value:

cmd	chr

Token memory:

1	info	link
2	info	link
3	info	link
n	info	link

Looking up control sequences

- A very visible to-be-token is a `\controlsequence`.
- When read, the name will be looked up in the hash table.
- When found its value will point to the table of equivalents.
- That table keeps track of:
  - the type (cmd)
  - the current level (grouping)
  - the current meaning (token list)

The (big) table of equivalents (simplified)

main hash	null control sequence
	128K hash entries
	frozen control sequences
	special sequences (undefined)
registers	17 internal & 64K user glues
	4 internal & 64K user mu glues
	12 internal & 64K user tokens
	2 internal & 64K user boxes
	116 internal & 64K user integers
	0 internal & 64K user attribute
	22 internal & 64K user dimensions
specifications	5 internal & 0 user
extra hash	additional entries (grows dynamic)

The hash table (simplified)

The hash table runs parallel to the main hash. On the todo list is is to move the registers to its own tables and make them dynamic.

1	string index	equivalents or (next > n) index
2	string index	equivalents or (next > n) index
n	string index	equivalents or (next > n) index
n + 1	string index	equivalents or (next > n) index
n + 2	string index	equivalents or (next > n) index
n + m	string index	equivalents or (next > n) index

Equivalents (registers direct, macros indirect i.e. token lists):

1	level	type	value
2	level	type	value
3	level	type	value
n	level	type	value

Other data management

- Grouping is handles by a nesting stack.
- Nested conditionals (`\if...`) have their own stack.
- The values before assignments are saved ion the save stack.
- Also other local changes (housekeeping) ends up in the save stack.
- Token lists and macro aliases have references pointers (reuse).
- Attributes, being linked node lists, have their own management.

Example 1: in the input

```
\luatokentable{1 \bf{2} 3\what {!}}
```

given token list:

30789	12	49	other char	1	U+00031	
185711	10	32	spacer			
501761	132	0	protected call			bf
82491	1	123	left brace			
489346	12	50	other char	2	U+00032	
378571	2	125	right brace			
501943	10	32	spacer			
501949	12	51	other char	3	U+00033	
502165	119	0	undefined cs			what
501845	1	123	left brace			
502074	12	33	other char	!	U+00021	
501934	2	125	right brace			

Example 1: in the input

```
\luatokentable{a \the\scratchcounter b \the\parindent \hbox to 10pt{x}}
```

given token list:

347356	11	97	letter	a	U+00061	
501735	10	32	spacer			
113	129	0	the			the
501930	85	257	register int			scratchcounter
30818	11	98	letter	b	U+00062	
114	10	32	spacer			
30792	129	0	the			the
501811	88	0	internal dimen			parindent
448988	30	10	make box			hbox
501936	11	116	letter	t	U+00074	
430669	11	111	letter	o	U+0006F	
502102	10	32	spacer			
385326	12	49	other char	1	U+00031	
502014	12	48	other char	0	U+00030	
501877	11	112	letter	p	U+00070	
501804	11	116	letter	t	U+00074	
502091	1	123	left brace			
501955	11	120	letter	x	U+00078	
187935	2	125	right brace			

Example 2: user registers

```
\scratchtoks{foo \framed{\red 123}456}

\luatokentable\scratchtoks
```

token register: scratchtoks

502253	11	102	letter	f	U+00066	
502220	11	111	letter	o	U+0006F	
501834	11	111	letter	o	U+0006F	
502230	10	32	spacer			
501726	134	0	tolerant protected call			framed
489431	1	123	left brace			
503085	132	0	protected call			red
502216	12	49	other char	1	U+00031	
378567	12	50	other char	2	U+00032	
502243	12	51	other char	3	U+00033	
501954	2	125	right brace			
501838	12	52	other char	4	U+00034	
501933	12	53	other char	5	U+00035	
297090	12	54	other char	6	U+00036	

Example 3: internal variables

```
\luatokentable\everypar
```

internal token variable: everypar				
43736	132	0	protected call	dotagsetparcounter
30802	132	0	protected call	page_otr_command_synchronize_side_floats
501867	132	0	protected call	checkindentation
502079	131	0	call	showparagraphnumber
385312	132	0	protected call	restoreinterlinepenalty
30830	131	0	call	flushnotes
30846	132	0	protected call	registerparoptions
502257	131	0	call	flushpostponednodedata
297088	131	0	call	typo_delimited_repeat
30807	131	0	call	spac_paragraphs_flush_intro
502205	131	0	call	typo_initial_handle
502148	131	0	call	typo_firstline_handle
502047	131	0	call	spac_paragraph_wrap
501730	132	0	protected call	spac_paragraph_freeze

Example 4: macro definitions

```
\protected\def\whatever#1[#2](#3)\relax{oeps #1 and #2 & #3 done ## error}

\luatokentable\whatever
```

protected control sequence: whatever

502150	19	49	match		argument 1
502566	12	91	other char	[	U+0005B
502973	19	50	match		argument 2
502001	12	93	other char	]	U+0005D
502284	12	40	other char	(	U+00028
512079	19	51	match		argument 3
289563	12	41	other char	)	U+00029
502646	16	1114112	relax		relax
501900	20	0	end match		
502549	11	111	letter	o	U+0006F
512264	11	101	letter	e	U+00065
501818	11	112	letter	p	U+00070
512204	11	115	letter	s	U+00073
385349	10	32	spacer		
502112	21	1	parameter reference		
502167	10	32	spacer		
502219	11	97	letter	a	U+00061
30871	11	110	letter	n	U+0006E
502979	11	100	letter	d	U+00064
501769	10	32	spacer		
502972	21	2	parameter reference		
385317	10	32	spacer		
385301	12	38	other char	&	U+00026
112034	10	32	spacer		
501733	21	3	parameter reference		
502000	10	32	spacer		
501767	11	100	letter	d	U+00064
502182	11	111	letter	o	U+0006F
385345	11	110	letter	n	U+0006E
502194	11	101	letter	e	U+00065
501801	10	32	spacer		
512305	6	35	parameter		
512279	10	32	spacer		
491751	11	101	letter	e	U+00065
512420	11	114	letter	r	U+00072
385306	11	114	letter	r	U+00072
502485	11	111	letter	o	U+0006F
209355	11	114	letter	r	U+00072

Example 5: commands

```
\luatokentable\startitemize
```

```
frozen instance protected control sequence: startitemize
151441  134    0  tolerant protected call                startitemgroup
502981   12   91  other char                [  U+0005B
502630   11  105  letter                    i  U+00069
503086   11  116  letter                    t  U+00074
501833   11  101  letter                    e  U+00065
502770   11  109  letter                    m  U+0006D
502440   11  105  letter                    i  U+00069
489254   11  122  letter                    z  U+0007A
502027   11  101  letter                    e  U+00065
501925   12   93  other char                ]  U+0005D
```

Example 6: commands

```
\luatokentable\doifelse
```

permanent protected control sequence: doifelse

512157	19	49	match	argument 1	
502550	19	50	match	argument 2	
512123	20	0	end match		
489211	126	21	if test		iftok
30847	1	123	left brace		
502042	21	1	parameter reference		
502446	2	125	right brace		
501849	1	123	left brace		
30853	21	2	parameter reference		
501968	2	125	right brace		
501904	120	0	expand after		expandafter
30779	131	0	call		firstoftwoarguments
154308	126	3	if test		else
209351	120	0	expand after		expandafter
501864	131	0	call		secondoftwoarguments
501824	126	2	if test		fi

Example 7: nothing

```
\luatokentable\relax
```

primitive control sequence: relax				
512299	16	1114112	relax	relax

Example 8: Hashes

```
\edef\foo#1#2{(#1)(\letterhash)(#2)}  \luatokentable\foo
```

control sequence: foo					
501719	19	49	match		argument 1
512303	19	50	match		argument 2
30839	20	0	end match		
501738	12	40	other char	(	U+00028
502887	21	1	parameter reference		
512866	12	41	other char	)	U+00029
502640	12	40	other char	(	U+00028
297096	12	35	other char	#	U+00023
512170	12	41	other char	)	U+00029
112001	12	40	other char	(	U+00028
502926	21	2	parameter reference		
512256	12	41	other char	)	U+00029

Example 9: Nesting

```
\def\foo#1{\def\foo##1{(#1)(##1)}}   \luatokentable\foo
```

```
control sequence: foo
503175    19    49  match                         argument 1
512052    20     0  end match
501967   115     1  def                                       def
512382   131     0  call                                      foo
489258     6    35  parameter
501892    12    49  other char               1    U+00031
501963     1   123  left brace
502911    12    40  other char               (    U+00028
512254    21     1  parameter reference
111995    12    41  other char               )    U+00029
502962    12    40  other char               (    U+00028
502319     6    35  parameter
512259    12    49  other char               1    U+00031
30858     12    41  other char               )    U+00029
503199     2   125  right brace
```