1. Copyright.

2. *linker_preamble_code* **thread.**
Extract linker's preamble code.

3. Fsm Clinker_preamble_code class.

4. Clinker_preamble_code constructor directive.

⟨ Clinker_preamble_code constructor directive 4 ⟩ ≡
```
empty_sdc_ = true;
chrs_.clear( );
```

5. Clinker_preamble_code op directive.

⟨ Clinker_preamble_code op directive 5 ⟩ ≡
```
parser__→set_use_all_shift_on( );
empty_sdc_ = true;
chrs_.clear( );
CAbs_lr1_sym * sym = parser__→start_token__;
switch (sym→enumerated_id__) {
case T_Enum :: T_LR1_eog_: break;
default: return;
}
parser__→set_use_all_shift_off( );
```

6. Clinker_preamble_code user-declaration directive.

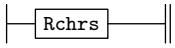
⟨ Clinker_preamble_code user-declaration directive 6 ⟩ ≡
```
public: std :: string chrs_;
  bool empty_sdc_;
```

7. Clinker_preamble_code user-prefix-declaration directive.

⟨ Clinker_preamble_code user-prefix-declaration directive 7 ⟩ ≡
```
#include "string.h"
#include "c_comments.h"
#include "c_string.h"
#include "c_literal.h"
#include "linker_id.h"
```

8. *Rlinker_preamble_code* **rule.**

```
Rlinker_preamble_code
```



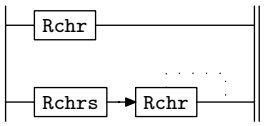
9. Rlinker_preamble_code op directive.

⟨ Rlinker_preamble_code op directive 9 ⟩ ≡

```
Clinker_preamble_code * fsm = ( Clinker_preamble_code * ) rule_info__.parser__→fsm_tbl__;
CAbs_lr1_sym * sym(0);
if (fsm→empty_sdc_ ≡ true) {
  sym = new Err_preamble_srce_code_not_present;
  sym→set_rc(*rule_info__.parser__→start_token__, __FILE__, __LINE__);
  RSVP(sym);
  rule_info__.parser__→set_stop_parse(true);
  return;
}
sym = new T_syntax_code(fsm→chrs_.c_str( ));
sym→set_rc(*rule_info__.parser__→start_token__, __FILE__, __LINE__);
RSVP(sym);
```

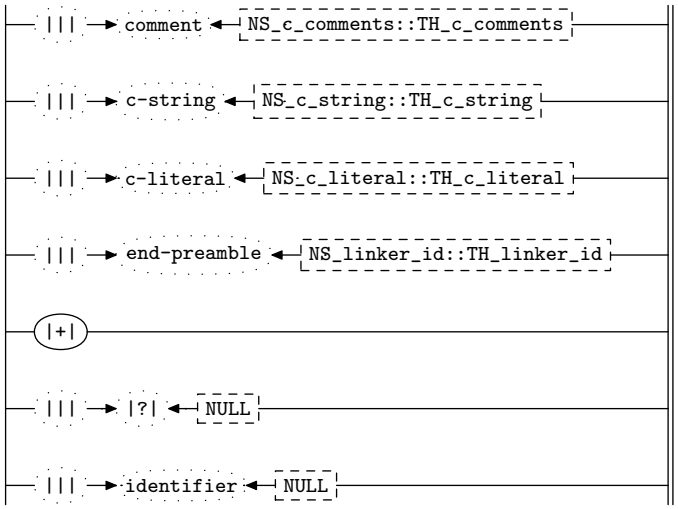
10. *Rchrs* rule.

Rchrs



11. *Rchr* rule.

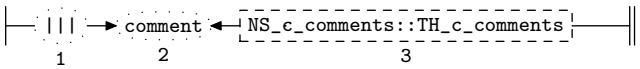
Rchr



12. Rchr op directive.

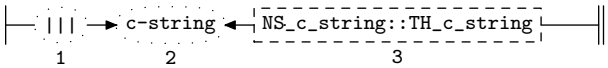
⟨ Rchr op directive 12 ⟩ ≡
```
if (rule_info__.parser__→stop_parse__ ≡ true) return;
if (rule_info__.parser__→use_all_shift__ ≡ false) return;      /* end-preamble seen */
CAbs_lr1_sym * la_sym = rule_info__.parser__→current_token( );
using namespace NS_yacco2_T_enum;
using namespace NS_yacco2_terminals;      /* watch for overrun */
if (la_sym→enumerated_id__ ≡ T_Enum :: T_LR1_eog_) {
  CAbs_lr1_sym * sym = new Err_end_preamble_kw_not_present;
  sym→set_rc(*la_sym, __FILE__, __LINE__);
  RSVP(sym);
  rule_info__.parser__→set_stop_parse(true);
}
```

13. *Rchr*'s subrule 1.



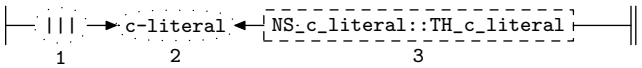
⟨ Rchr subrule 1 op directive 13 ⟩ ≡
```
Clinker_preamble_code * fsm = ( Clinker_preamble_code * ) rule_info__.parser__→fsm_tbl__;
T_comment * k = sf→p2__;
k→set_auto_delete(true);      /* delete it when it pops from stack */
fsm→chrs_ += k→comment_data( )→c_str( );
```

14. *Rchr*'s subrule 2.



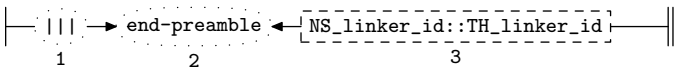
⟨ Rchr subrule 2 op directive 14 ⟩ ≡
```
Clinker_preamble_code * fsm = ( Clinker_preamble_code * ) rule_info__.parser__→fsm_tbl__;
T_c_string * k = sf→p2__;
k→set_auto_delete(true);      /* delete it when it pops from stack */
  /* add back the bounding double quotes */
fsm→chrs_ += '"';
fsm→chrs_ += k→c_string( )→c_str( );
fsm→chrs_ += '"';
fsm→empty_sdc_ = false;
```

15. *Rchr*'s subrule 3.



⟨ Rchr subrule 3 op directive 15 ⟩ ≡
```
Clinker_preamble_code * fsm = ( Clinker_preamble_code * ) rule_info__.parser__→fsm_tbl__;
T_c_literal * k = sf→p2__;
k→set_auto_delete(true);      /* delete it when it pops from stack */
  /* add back the bounding single quotes */
fsm→chrs_ += "'";
fsm→chrs_ += k→c_literal( )→c_str( );
fsm→chrs_ += "'";
fsm→empty_sdc_ = false;
```

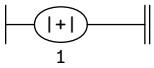
16. *Rchr*'**s subrule 4.**



⟨ Rchr subrule 4 op directive 16 ⟩ ≡

```
Clinker_preamble_code * fsm = ( Clinker_preamble_code * ) rule_info__.parser__→fsm_tbl__;
rule_info__.parser__→set_use_all_shift_off ( );
if (fsm→empty_sdc_ ≡ true) {
  CAbs_lr1_sym * sym = new Err_preamble_srce_code_not_present;
  sym→set_rc(*rule_info__.parser__→start_token__, __FILE__, __LINE__);
  RSVP(sym);
  rule_info__.parser__→set_stop_parse(true);
  return;
}
```

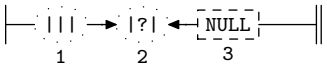
17. ***Rchr*'s subrule 5.**



⟨ Rchr subrule 5 op directive 17 ⟩ ≡

```
Clinker_preamble_code * fsm = ( Clinker_preamble_code * ) rule_info__.parser__→fsm_tbl__;
CAbs_lr1_sym * k = sf→p1__;
switch (k→enumerated_id__) {
case T_Enum :: T_T_eol_:
  {
    fsm→chrs_ += "\n";
    break;
  }
case T_Enum :: T_raw_sp_:
  {
    fsm→chrs_ += "␣";
    break;
  }
case T_Enum :: T_raw_ff_:
  {
    fsm→chrs_ += "\f";
    break;
  }
case T_Enum :: T_raw_vt_:
  {
    fsm→chrs_ += "\v";
    break;
  }
case T_Enum :: T_raw_ht_:
  {
    fsm→chrs_ += "\t";
    break;
  }
case T_Enum :: T_raw_cr_:
  {
    fsm→chrs_ += "\n";
    break;
  }
case T_Enum :: T_raw_lf_:
  {
    fsm→chrs_ += "\n";
    break;
  }
default:
  {
    fsm→chrs_ += *k→id__;
    fsm→empty_sdc_ = false;
    break;
  }
}
```

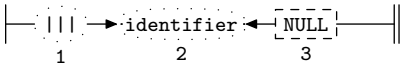
18. *Rchr*'**s subrule 6.**



⟨ Rchr subrule 6 op directive 18 ⟩ ≡

```
RSVP(sf→p2__);
rule_info__.parser__→set_stop_parse(true);
```

19. *Rchr*'**s subrule 7.**



⟨ Rchr subrule 7 op directive 19 ⟩ ≡

```
Clinker_preamble_code * fsm = ( Clinker_preamble_code * ) rule_info__.parser__→fsm_tbl__;
T_identifier * k = sf→p2__;
k→set_auto_delete(true);      /* delete it when it pops from stack */
fsm→chrs_ += k→identifier( )→c_str( );
```

20. First Set Language for O_2^{linker}.

```
/*
 File: linker_preamble_code.fsc
 Date and Time: Fri Jan  2 15:33:42 2015
*/
transitive   y
grammar-name "linker_preamble_code"
name-space   "NS_linker_preamble_code"
thread-name  "TH_linker_preamble_code"
monolithic   n
file-name    "linker_preamble_code.fsc"
no-of-T      569
list-of-native-first-set-terminals 1
   LR1_all_shift_operator
end-list-of-native-first-set-terminals
list-of-transitive-threads 4
  NS_c_literal::TH_c_literal
  NS_c_comments::TH_c_comments
  NS_linker_id::TH_linker_id
  NS_c_string::TH_c_string
end-list-of-transitive-threads
list-of-used-threads 4
  NS_c_comments::TH_c_comments
  NS_c_literal::TH_c_literal
  NS_c_string::TH_c_string
  NS_linker_id::TH_linker_id
end-list-of-used-threads
fsm-comments
"\\olinker's lexer of preamble code section:\n similar to Pass3 lexer."
```

21. Lr1 State Network.

⇒ State: 1 state type: ^{s}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
c	Rchr			3	3	1	|||	c-literal NS_c_literal::TH_c_literal		1	2	5	
c	Rchr			3	7	1	|||	identifier NULL		1	2	7	
c	Rchr			3	1	1	|||	comment NS_c_comments::TH_c_comments		1	2	4	
c	Rchr			3	2	1	|||	c-string NS_c_string::TH_c_string		1	2	6	
c	Rchr			3	4	1	|||	end-preamble NS_linker_id::TH_linker_id		1	2	8	
c	Rchr			3	6	1	|||	|?| NULL		1	2	3	
c	Rchr			3	5	1	|+|			1	9	9	
c	Rchrs			2	2	1	Rchrs	*Rchr*		1	10	11	
c	Rlinker_preamble_code			1	1	1	Rchrs			1	10	10	
c	Rchrs			2	1	1	Rchr			1	12	12	

⇒\|\|\| *arbitration−code*: ϵ State: 2 state type: ^{s}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	6	2	|?|			1	3	3	
t	Rchr			3	1	2	comment			1	4	4	
t	Rchr			3	3	2	c-literal			1	5	5	
t	Rchr			3	2	2	c-string			1	6	6	
t	Rchr			3	7	2	identifier			1	7	7	
t	Rchr			3	4	2	end-preamble			1	8	8	

⇒\|?\| State: 3 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	6	3				1	0	3	1

⇒*comment* State: 4 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	1	3				1	0	4	1

⇒*c−literal* State: 5 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	3	3				1	0	5	1

⇒*c−string* State: 6 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	2	3				1	0	6	1

⇒*identifier* State: 7 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	7	3				1	0	7	1

⇒*end−preamble* State: 8 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	4	3				1	0	8	1

⇒\|+\| State: 9 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchr			3	5	2				1	0	9	1

$\Rightarrow^{Rchrs}$ State: 10 state type: $^{s/r}$

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rlinker_preamble_code			1	1	2				1	0	10	1
c	Rchr			3	3	1	|||	c-literal NS_c_literal::TH_c_literal		10	2	5	
c	Rchr			3	7	1	|||	identifier NULL		10	2	7	
c	Rchr			3	1	1	|||	comment NS_c_comments::TH_c_comments		10	2	4	
c	Rchr			3	2	1	|||	c-string NS_c_string::TH_c_string		10	2	6	
c	Rchr			3	4	1	|||	end-preamble NS_linker_id::TH_linker_id		10	2	8	
c	Rchr			3	6	1	||| |?|	NULL		10	2	3	
c	Rchr			3	5	1	|+|			10	9	9	
t	Rchrs			2	2	2	Rchr			1	11	11	

$\Rightarrow^{Rchr}$ State: 11 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchrs			2	2	3				1	0	11	1

$\Rightarrow^{Rchr}$ State: 12 state type: ^{r}

	←	rule	→	R#	sr#	Po	←	subrule element	→	Brn	Gto	Red	LA
t	Rchrs			2	1	2				1	0	12	1

22. Index.

linker_preamble_code Grammar

Date: January 2, 2015 at 15:36

File: linker_preamble_code.lex Ns: NS_linker_preamble_code

Version: 1.0 Debug: false

Grammar Comments: Type: Thread

O_2^{linker}'s lexer of preamble code section: similar to Pass3 lexer.

1 element(s) in Lookahead Expression below

eolr