

**1. Copyright.**

Copyright © Dave Bone 1998 - 2015

**2. Error symbols vocabulary.**

Error symbols for the telling.

Due to my verbosity and *cweb*'s (TEX) black slug output some of my symbol keys are more cryptic in description as i wanted to title them with their enumeration symbol. Now, the enumeration symbol is removed from the index so there is no blob of ... If the truth be told, these descriptions should be indirect key values that allow for multilingual translation. One step at a time...

Upon experimentation with error keys having a suffixed Exxx indicator in the key itself whereby the xxx is a number: 001 and associating a translation file with the error construct, i nixed the ideas. It is simply a post error evaluation on the given error keys to translate them using some form of associated map. The error enumeration key with a language indicator would suffice. A simple database could be used. For example T\_Enum::T\_LR1\_err\_nested\_files\_exceeded\_ with a country code of "DE" for German would translate "nested files exceeded". Not finding a translation should default to the error token's literal key. See grammar "o2\_err\_hdr.lex" for a context sensitive approach to error reporting.

Ahh... grandeur. Now what about unicode?

**3. # T in list not eq.**

Enum: T\_Err\_no\_of\_native\_Ts\_in\_list\_not\_equal\_

Class: Err\_no\_of\_native\_Ts\_in\_list\_not\_equal

AB: N

AD: N

**4. # T in list not eq user-declaration directive.**

```
< # T in list not eq user-declaration directive 4 > ≡
public: Err_no_of_native_Ts_in_list_not_equal(unsigned long Value);
    unsigned long no() const;
private: unsigned long no_;
```

**5. # T in list not eq user-implementation directive.**

```
< # T in list not eq user-implementation directive 5 > ≡
Err_no_of_native_Ts_in_list_not_equal::Err_no_of_native_Ts_in_list_not_equal(unsigned long
    No)T_CTOR("no_terminals_in_list_not_equal,_chk_items_in_list",
    T_Enum::T_Err_no_of_native_Ts_in_list_not_equal_,0,false,false)
{
    no_ = No;
}
unsigned long Err_no_of_native_Ts_in_list_not_equal::no() const
{
    return no_;
}
```

**6. # threads in list not eq.**

Enum: T\_Err\_no\_of\_threads\_in\_list\_not\_equal\_

Class: Err\_no\_of\_threads\_in\_list\_not\_equal

AB: N

AD: N

**7. # threads in list not eq user-declaration directive.**

```
< # threads in list not eq user-declaration directive 7 > ≡
public: Err_no_of_threads_in_list_not_equal(unsigned long No);
    unsigned long no() const;
private: unsigned long no_;
```

**8. # threads in list not eq user-implementation directive.**

```

< # threads in list not eq user-implementation directive 8 > ≡
  Err_no_of_threads_in_list_not_equal :: Err_no_of_threads_in_list_not_equal(
    unsigned long
      No)T_CTOR("no_threads_in_list_not_equal,chk_list",
      T_Enum :: T_Err_no_of_threads_in_list_not_equal_, 0, false, false)
  {
    no_ = No;
  }
  unsigned long Err_no_of_threads_in_list_not_equal :: no() const
  {
    return no_;
  }

```

**9. ? ended subrule expr.**

Enum: T\_Err\_not\_T\_or\_R\_or\_eos\_in\_subrule\_expr\_

Class: Err\_not\_T\_or\_R\_or\_eos\_in\_subrule\_expr

AB: N

AD: N

**10. Empty file no grammar constructs present.**

Enum: T\_Err\_empty\_file\_

Class: Err\_empty\_file

AB: N

AD: N

**11. O<sub>2</sub> epsilon badly gened: 0 items in fsc lists.**

Enum: T\_Err\_epsilon\_pass\_thru\_

Class: Err\_epsilon\_pass\_thru

AB: N

AD: N

*O<sub>2</sub>* states that a thread has epsilon pass thru of thread calling. Example the called thread start state has an epsilon subrule. Sanity check or paranoia on my work.

**12. O<sub>2</sub> epsilon: T present, but no T list.**

Enum: T\_Err\_epsilon\_pass\_thru\_no\_Ts\_

Class: Err\_epsilon\_pass\_thru\_no\_Ts

AB: N

AD: N

*O<sub>2</sub>* states that a thread has epsilon pass thru of thread calling. Example the called thread start state has an epsilon subrule. Sanity check or paranoia on my work.

**13. T in list not defined in T-alphabet.**

Enum: T\_Err\_bad\_T\_in\_list\_

Class: Err\_bad\_T\_in\_list

AB: N

AD: N

*O<sub>2</sub><sup>linker</sup>* message. It indicates that the terminal is not defined in “T-alphabet” construct. This is the T vocabulary. If the terminal is not defined, it means that the grammar writer has not generated the vocabulary. So turn on the /t option and re-compile a grammar thus creating the updated list.

**14. T not found in stbl.**

Enum: T\_Err\_T\_not\_in\_stbl\_

Class: Err\_T\_not\_in\_stbl

AB: N

AD: N

Either the grammar writer forgot to add a new T definition or its a mistype.

---

**15. T not returned from a thread.**

Enum: T\_Err\_not\_T\_for\_rtned\_token\_from\_th\_

Class: Err\_not\_T\_for\_rtned\_token\_from\_th

AB: N

AD: N

**16. T-alphabet file does not exist.**

Enum: T\_Err\_T\_alphabet\_file\_does\_not\_exist\_

Class: Err\_T\_alphabet\_file\_does\_not\_exist

AB: N

AD: N

**17. T-alphabet file not present.**

Enum: T\_Err\_T\_alphabet\_file\_not\_present\_

Class: Err\_T\_alphabet\_file\_not\_present

AB: N

AD: N

**18. T-alphabet kw not present.**

Enum: T\_Err\_T\_alphabet\_kw\_not\_present\_

Class: Err\_T\_alphabet\_kw\_not\_present

AB: N

AD: N

**19. already defined AB tag.**

Enum: T\_Err\_already\_defined\_AB\_

Class: Err\_already\_defined\_AB

AB: N

AD: N

**20. already defined AD tag.**

Enum: T\_Err\_already\_defined\_AD\_

Class: Err\_already\_defined\_AD

AB: N

AD: N

**21. bad char.**

Enum: T\_Err\_bad\_char\_

Class: Err\_bad\_char

AB: N

AD: N

**22. bad char destructor directive.**

$\langle$  bad char destructor directive [22](#)  $\rangle \equiv$

`if (R->bad_char_  $\neq$  0) delete R->bad_char();`

**23. bad char user-declaration directive.**

```

⟨ bad char user-declaration directive 23 ⟩ ≡
public: Err_bad_char(CAbs_lr1_sym * Err_char);
        yacco2 :: CAbs_lr1_sym * bad_char() const;
        void zero_out_bad_char();
private: yacco2 :: CAbs_lr1_sym * bad_char_;

```

**24. bad char user-implementation directive.**

```

⟨ bad char user-implementation directive 24 ⟩ ≡
Err_bad_char :: Err_bad_char(CAbs_lr1_sym * Bad_char)T_CTOR("bad_char", T_Enum :: T_Err_bad_char_,
    &dctor_Err_bad_char, false, false)
{
    bad_char_ = Bad_char;
}
yacco2 :: CAbs_lr1_sym * Err_bad_char :: bad_char() const
{
    return bad_char_;
}
void Err_bad_char :: zero_out_bad_char()
{
    bad_char_ = 0;
}

```

**25. bad cmd-opt.**

```

Enum: T_Err_bad_cmd_lne_opt_
Class: Err_bad_cmd_lne_opt

```

AB: N

AD: N

**26. bad directive.**

```

Enum: T_Err_bad_directive_
Class: Err_bad_directive

```

AB: N

AD: N

**27. bad eos.**

```

Enum: T_Err_bad_eos_
Class: Err_bad_eos

```

AB: N

AD: N

**28. bad esc.**

```

Enum: T_Err_bad_esc_
Class: Err_bad_esc

```

AB: N

AD: N

**29. bad filename.**

```

Enum: T_Err_bad_filename_
Class: Err_bad_filename

```

AB: N

AD: N

**30. bad filename user-declaration directive.**

```

⟨bad filename user-declaration directive 30⟩ ≡
public: Err_bad_filename(std::string & File_name);
    Err_bad_filename(const char *File_name);
    std::string * file_name();
private: std::string file_name_;

```

**31. bad filename user-implementation directive.**

```

⟨bad filename user-implementation directive 31⟩ ≡
    Err_bad_filename::Err_bad_filename(std::string & File_name)T_CTOR("bad_filename",
        T_Enum::T_Err_bad_filename_, 0, false, false)
    {
        file_name_ += File_name.c_str();
    }
    Err_bad_filename::Err_bad_filename(const char *File_name)T_CTOR("bad_filename",
        T_Enum::T_Err_bad_filename_, 0, false, false)
    {
        file_name_ += File_name;
    }
    std::string * Err_bad_filename::file_name()
    {
        return &file_name_;
    }

```

**32. bad filename for Errors vocabulary header.**

Enum: T\_Err\_bad\_errors\_hdrfilename\_

Class: Err\_bad\_errors\_hdrfilename

AB: N

AD: N

**33. bad filename for Errors vocabulary header user-declaration directive.**

```

⟨bad filename for Errors vocabulary header user-declaration directive 33⟩ ≡
public: Err_bad_errors_hdrfilename(std::string & File_name);
    Err_bad_errors_hdrfilename(const char *File_name);
    std::string * file_name();
private: std::string file_name_;

```

**34. bad filename for Errors vocabulary header user-implementation directive.**

```

⟨bad filename for Errors vocabulary header user-implementation directive 34⟩ ≡
  Err_bad_errors_hdrfilename :: Err_bad_errors_hdrfilename(std :: string &
    File_name)T_CTOR("bad_filename_for_Errors_vocabulary_header",
    T_Enum :: T_Err_bad_errors_hdrfilename_, 0, false, false)
  {
    file_name_ += File_name.c_str();
  }
  Err_bad_errors_hdrfilename :: Err_bad_errors_hdrfilename(const char
    *File_name)T_CTOR("bad_filename_for_Errors_vocabulary_header",
    T_Enum :: T_Err_bad_errors_hdrfilename_, 0, false, false)
  {
    file_name_ += File_name;
  }
  std :: string * Err_bad_errors_hdrfilename :: file_name()
  {
    return &file_name_;
  }

```

**35. bad filename for Errors vocabulary implementation.**

Enum: T\_Err\_bad\_errors\_impfilename\_

Class: Err\_bad\_errors\_impfilename

AB: N

AD: N

**36. bad filename for Errors vocabulary implementation user-declaration directive.**

```

⟨bad filename for Errors vocabulary implementation user-declaration directive 36⟩ ≡
public: Err_bad_errors_impfilename(std :: string & File_name);
  Err_bad_errors_impfilename(const char *File_name);
  std :: string * file_name();
private: std :: string file_name_;

```

**37. bad filename for Errors vocabulary implementation user-implementation directive.**

```

⟨bad filename for Errors vocabulary implementation user-implementation directive 37⟩ ≡
  Err_bad_errors_impfilename :: Err_bad_errors_impfilename(std :: string &
    File_name)T_CTOR("bad_filename_for_Errors_vocabulary_implementation",
    T_Enum :: T_Err_bad_errors_impfilename_, 0, false, false)
  {
    file_name_ += File_name.c_str();
  }
  Err_bad_errors_impfilename :: Err_bad_errors_impfilename(const char
    *File_name)T_CTOR("bad_filename_for_Errors_vocabulary_implementation",
    T_Enum :: T_Err_bad_errors_impfilename_, 0, false, false)
  {
    file_name_ += File_name;
  }
  std :: string * Err_bad_errors_impfilename :: file_name()
  {
    return &file_name_;
  }

```

**38. bad filename to output cpp.**

Enum: T\_Err\_bad\_fsmcpp\_filename\_

Class: Err\_bad\_fsmcpp\_filename

AB: N

AD: N

**39. bad filename to output cpp user-declaration directive.**

⟨bad filename to output cpp user-declaration directive 39⟩ ≡

**public:** *Err\_bad\_fsmcpp\_filename*(*std::string* & *File\_name*);*Err\_bad\_fsmcpp\_filename*(**const char** \**File\_name*);*std::string* \* *file\_name*();**private:** *std::string* *file\_name\_;***40. bad filename to output cpp user-implementation directive.**

⟨bad filename to output cpp user-implementation directive 40⟩ ≡

*Err\_bad\_fsmcpp\_filename* :: *Err\_bad\_fsmcpp\_filename*(*std::string* &  
*File\_name*)T\_CTOR("bad\_filename\_to\_output\_cpp", *T\_Enum* :: *T\_Err\_bad\_fsmcpp\_filename\_*, 0,  
*false*, *false*){  
*file\_name\_* += *File\_name.c\_str*();

}

*Err\_bad\_fsmcpp\_filename* :: *Err\_bad\_fsmcpp\_filename*(**const char**  
\**File\_name*)T\_CTOR("bad\_filename\_to\_output\_cpp", *T\_Enum* :: *T\_Err\_bad\_fsmcpp\_filename\_*,  
0, *false*, *false*){  
*file\_name\_* += *File\_name*;

}

*std::string* \* *Err\_bad\_fsmcpp\_filename* :: *file\_name*(){  
**return** &*file\_name\_;*

}

**41. bad filename to output enumeration header.**

Enum: T\_Err\_bad\_enum\_filename\_

Class: Err\_bad\_enum\_filename

AB: N

AD: N

**42. bad filename to output enumeration header user-declaration directive.**

⟨bad filename to output enumeration header user-declaration directive 42⟩ ≡

**public:** *Err\_bad\_enum\_filename*(*std::string* & *File\_name*);*Err\_bad\_enum\_filename*(**const char** \**File\_name*);*std::string* \* *file\_name*();**private:** *std::string* *file\_name\_;*



**43. bad filename to output enumeration header user-implementation directive.**

```

⟨ bad filename to output enumeration header user-implementation directive 43 ⟩ ≡
  Err_bad_enum_filename :: Err_bad_enum_filename(std :: string &
    File_name)T_CTOR("bad_filename_to_output_enumeration_header",
    T_Enum :: T_Err_bad_enum_filename_, 0, false, false)
  {
    file_name_ += File_name.c_str();
  }
  Err_bad_enum_filename :: Err_bad_enum_filename(const char
    *File_name)T_CTOR("bad_filename_to_output_enumeration_header",
    T_Enum :: T_Err_bad_enum_filename_, 0, false, false)
  {
    file_name_ += File_name;
  }
  std :: string * Err_bad_enum_filename :: file_name()
  {
    return &file_name_;
  }

```

**44. bad filename to output grammar header.**

Enum: T\_Err\_bad\_fsmheader\_filename\_

Class: Err\_bad\_fsmheader\_filename

AB: N

AD: N

**45. bad filename to output grammar header user-declaration directive.**

```

⟨ bad filename to output grammar header user-declaration directive 45 ⟩ ≡
public: Err_bad_fsmheader_filename(std :: string & File_name);
  Err_bad_fsmheader_filename(const char *File_name);
  std :: string * file_name();
private: std :: string file_name_;

```

**46. bad filename to output grammar header user-implementation directive.**

```

⟨ bad filename to output grammar header user-implementation directive 46 ⟩ ≡
  Err_bad_fsmheader_filename :: Err_bad_fsmheader_filename(std :: string &
    File_name)T_CTOR("bad_filename_to_output_grammar_header",
    T_Enum :: T_Err_bad_fsmheader_filename_, 0, false, false)
  {
    file_name_ += File_name.c_str();
  }
  Err_bad_fsmheader_filename :: Err_bad_fsmheader_filename(const char
    *File_name)T_CTOR("bad_filename_to_output_grammar_header",
    T_Enum :: T_Err_bad_fsmheader_filename_, 0, false, false)
  {
    file_name_ += File_name;
  }
  std :: string * Err_bad_fsmheader_filename :: file_name()
  {
    return &file_name_;
  }

```

**47. bad filename to output sym.**

Enum: T\_Err\_bad\_fsmsym\_filename\_

Class: Err\_bad\_fsmsym\_filename

AB: N

AD: N

**48. bad filename to output sym user-declaration directive.**

⟨bad filename to output sym user-declaration directive 48⟩ ≡

**public:** Err\_bad\_fsmsym\_filename(std::string & File\_name);

Err\_bad\_fsmsym\_filename(const char \*File\_name);

std::string \* file\_name();

**private:** std::string file\_name\_;**49. bad filename to output sym user-implementation directive.**

⟨bad filename to output sym user-implementation directive 49⟩ ≡

Err\_bad\_fsmsym\_filename::Err\_bad\_fsmsym\_filename(std::string &  
File\_name)T\_CTOR("bad\_filename\_to\_output\_sym", T\_Enum::T\_Err\_bad\_fsmsym\_filename\_, 0,  
false, false){  
file\_name\_ += File\_name.c\_str();  
}Err\_bad\_fsmsym\_filename::Err\_bad\_fsmsym\_filename(const char  
\*File\_name)T\_CTOR("bad\_filename\_to\_output\_sym", T\_Enum::T\_Err\_bad\_fsmsym\_filename\_,  
0, false, false){  
file\_name\_ += File\_name;  
}

std::string \* Err\_bad\_fsmsym\_filename::file\_name()

{  
return &file\_name\_;  
}**50. bad filename to output tbl.**

Enum: T\_Err\_bad\_fsmtbl\_filename\_

Class: Err\_bad\_fsmtbl\_filename

AB: N

AD: N

**51. bad filename to output tbl user-declaration directive.**

⟨bad filename to output tbl user-declaration directive 51⟩ ≡

**public:** Err\_bad\_fsmtbl\_filename(std::string & File\_name);

Err\_bad\_fsmtbl\_filename(const char \*File\_name);

std::string \* file\_name();

**private:** std::string file\_name\_;

**52. bad filename to output tbl user-implementation directive.**

```

⟨bad filename to output tbl user-implementation directive 52⟩ ≡
  Err_bad_fsmtbl_filename :: Err_bad_fsmtbl_filename(std :: string &
    File_name)T_CTOR("bad_filename_to_output_tbl", T_Enum :: T_Err_bad_fsmtbl_filename_, 0,
    false, false)
  {
    file_name_ += File_name.c_str();
  }
  Err_bad_fsmtbl_filename :: Err_bad_fsmtbl_filename(const char
    *File_name)T_CTOR("bad_filename_to_output_tbl", T_Enum :: T_Err_bad_fsmtbl_filename_, 0,
    false, false)
  {
    file_name_ += File_name;
  }
  std :: string * Err_bad_fsmtbl_filename :: file_name()
  {
    return &file_name_;
  }

```

**53. bad int-no.**

Enum: T\_Err\_bad\_int\_no\_

Class: Err\_bad\_int\_no

AB: N

AD: N

**54. bad int-no range.**

Enum: T\_Err\_bad\_int\_no\_range\_

Class: Err\_bad\_int\_no\_range

AB: N

AD: N

**55. bad operator la expr: not a + or -.**

Enum: T\_Err\_bad\_operator\_in\_la\_expr\_

Class: Err\_bad\_operator\_in\_la\_expr

AB: N

AD: N

Thread's lookahead expression malformed.

**56. bad pos of t def.**

Enum: T\_Err\_cannot\_define\_term\_after\_sufx\_dir\_

Class: Err\_cannot\_define\_term\_after\_sufx\_dir

AB: N

AD: N

**57. bad pos of terminals-refs.**

Enum: T\_Err\_terminals\_refs\_def\_after\_terminals\_

Class: Err\_terminals\_refs\_def\_after\_terminals

AB: N

AD: N

**58. bad term in la expr.**

Enum: T\_Err\_bad\_term\_in\_la\_expr\_

Class: Err\_bad\_term\_in\_la\_expr

AB: N

AD: N

Must be a Rule or a T term within the subrule expression.

**59. bad term in subrule expr.**

Enum: T\_Err\_not\_T\_or\_R\_in\_subrule\_expr\_

Class: Err\_not\_T\_or\_R\_in\_subrule\_expr

AB: N

AD: N

**60. bad thread expr.**

Enum: T\_Err\_bad\_th\_qualifier\_

Class: Err\_bad\_th\_qualifier

AB: N

AD: N

**61. bad univ-seq.**

Enum: T\_Err\_bad\_univ\_seq\_

Class: Err\_bad\_univ\_seq

AB: N

AD: N

**62. bad univ-seq user-declaration directive.**`<bad univ-seq user-declaration directive 62> ≡``public: Err_bad_univ_seq(unsigned long Value);``unsigned long bad_univ_seq() const;``private: unsigned long bad_univ_seq_;`**63. bad univ-seq user-implementation directive.**`<bad univ-seq user-implementation directive 63> ≡``Err_bad_univ_seq::Err_bad_univ_seq(unsigned long Bad_univ_seq)T_CTOR("bad_univ_seq",  
T_Enum::T_Err_bad_univ_seq_, 0, false, false)``{  
bad_univ_seq_ = Bad_univ_seq;  
}``unsigned long Err_bad_univ_seq::bad_univ_seq() const  
{  
return bad_univ_seq_;  
}`**64. command line chaffe.**

Enum: T\_Err\_cmd\_line\_chaffe\_

Class: Err\_cmd\_line\_chaffe

AB: N

AD: N

**65. comment-overrun.**

Enum: T\_Err\_comment\_overrun\_

Class: Err\_comment\_overrun

AB: N

AD: N

**66. dup ? sufx directive.**

Enum: T\_Err\_dup\_sufx\_directive\_

Class: Err\_dup\_sufx\_directive

AB: N

AD: N

**67. dup-entry in sym-table.**

Enum: T\_Err\_dup\_entry\_in\_sym\_table\_

Class: Err\_dup\_entry\_in\_sym\_table

AB: N

AD: N

**68. duplicate T\_enum phrase.**

Enum: T\_Err\_already\_processed\_T\_enum\_phrase\_

Class: Err\_already\_processed\_T\_enum\_phrase

AB: N

AD: N

**69. duplicate directive.**

Enum: T\_Err\_duplicate\_directive\_

Class: Err\_duplicate\_directive

AB: N

AD: N

**70. duplicate error phrase.**

Enum: T\_Err\_already\_processed\_error\_phrase\_

Class: Err\_already\_processed\_error\_phrase

AB: N

AD: N

**71. duplicate fsm phrase.**

Enum: T\_Err\_already\_processed\_fsm\_phrase\_

Class: Err\_already\_processed\_fsm\_phrase

AB: N

AD: N

**72. duplicate logical name.**

Enum: T\_Err\_duplicate\_logical\_name\_

Class: Err\_duplicate\_logical\_name

AB: N

AD: N

**73. duplicate lrk phrase.**

Enum: T\_Err\_already\_processed\_lrk\_phrase\_

Class: Err\_already\_processed\_lrk\_phrase

AB: N

AD: N

**74. duplicate parallel-parse phrase.**

Enum: T\_Err\_already\_processed\_pp\_phase\_

Class: Err\_already\_processed\_pp\_phase AB: N AD: N

**75. duplicate raw character phrase.**

Enum: T\_Err\_already\_processed\_rc\_phase\_

Class: Err\_already\_processed\_rc\_phase AB: N AD: N

**76. duplicate rule phrase.**

Enum: T\_Err\_already\_processed\_rule\_phase\_

Class: Err\_already\_processed\_rule\_phase AB: N AD: N

**77. duplicate terminal phrase.**

Enum: T\_Err\_already\_processed\_T\_phase\_

Class: Err\_already\_processed\_T\_phase AB: N AD: N

**78. duplicate-entry in alphabet.**

Enum: T\_Err\_dup\_entry\_in\_alphabet\_

Class: Err\_dup\_entry\_in\_alphabet AB: N AD: N

**79. emitfile file does not exist.**

Enum: T\_Err\_emitfile\_file\_does\_not\_exist\_

Class: Err\_emitfile\_file\_does\_not\_exist AB: N AD: N

**80. emitfile file not present.**

Enum: T\_Err\_emitfile\_file\_not\_present\_

Class: Err\_emitfile\_file\_not\_present AB: N AD: N

**81. emitfile kw not present.**

Enum: T\_Err\_emitfile\_kw\_not\_present\_

Class: Err\_emitfile\_kw\_not\_present AB: N AD: N

**82. file of T-alphabet not present.**

Enum: T\_Err\_file\_of\_T\_alphabet\_not\_present\_

Class: Err\_file\_of\_T\_alphabet\_not\_present AB: N AD: N

**83. file-name file does not exist.**

Enum: T\_Err\_filename\_file\_does\_not\_exist\_

Class: Err\_filename\_file\_does\_not\_exist

AB: N

AD: N

**84. fsc file does not exist.**

Enum: T\_Err\_fsc\_file\_does\_not\_exist\_

Class: Err\_fsc\_file\_does\_not\_exist

AB: N

AD: N

**85. fsc file not present.**

Enum: T\_Err\_fsc\_cntl\_file\_not\_present\_

Class: Err\_fsc\_cntl\_file\_not\_present

AB: N

AD: N

**86. improper closing of rules construct.**

Enum: T\_Err\_improper\_closing\_of\_rules\_

Class: Err\_improper\_closing\_of\_rules

AB: N

AD: N

**87. improper directive.**

Enum: T\_Err\_improper\_directive\_

Class: Err\_improper\_directive

AB: N

AD: N

**88. invalid fsm-debug value.**

Enum: T\_Err\_fsm\_debug\_string\_not\_true\_or\_false\_

Class: Err\_fsm\_debug\_string\_not\_true\_or\_false

AB: N

AD: N

**89. invalid logical name.**

Enum: T\_Err\_invalid\_logical\_name\_

Class: Err\_invalid\_logical\_name

AB: N

AD: N

**90. invalid logical value.**

Enum: T\_Err\_invalid\_logical\_value\_

Class: Err\_invalid\_logical\_value

AB: N

AD: N

**91. la expression calculates an empty set.**

Enum: T\_Err\_la\_expr\_calc\_empty\_set\_

Class: Err\_la\_expr\_calc\_empty\_set

AB: N

AD: N

**92. linker's monolithic value not n or y.**

Enum: T\_Err\_monolithic\_value\_bad\_

Class: Err\_monolithic\_value\_bad AB: N AD: N

**93. linker's transitive value not n or y.**

Enum: T\_Err\_transitive\_value\_bad\_

Class: Err\_transitive\_value\_bad AB: N AD: N

**94. misplaced or missing T enumeration phrase.**

Enum: T\_ERR\_no\_T\_enum\_phrase\_

Class: ERR\_no\_T\_enum\_phrase AB: N AD: N

**95. misplaced or missing errors phrase.**

Enum: T\_ERR\_no\_errors\_phrase\_

Class: ERR\_no\_errors\_phrase AB: N AD: N

**96. misplaced or missing fsm phrase.**

Enum: T\_ERR\_no\_fsm\_phrase\_

Class: ERR\_no\_fsm\_phrase AB: N AD: N

**97. misplaced or missing lrk phrase.**

Enum: T\_ERR\_no\_lrk\_phrase\_

Class: ERR\_no\_lrk\_phrase AB: N AD: N

**98. misplaced or missing rc phrase.**

Enum: T\_ERR\_no\_rc\_phrase\_

Class: ERR\_no\_rc\_phrase AB: N AD: N

**99. misplaced or missing rules phrase.**

Enum: T\_ERR\_no\_rules\_phrase\_

Class: ERR\_no\_rules\_phrase AB: N AD: N

**100. misplaced or missing terminals phrase.**

Enum: T\_ERR\_no\_terminals\_phrase\_

Class: ERR\_no\_terminals\_phrase AB: N AD: N



**101. misplaced or misspelt Rule or T outside of Rules defs.**

Enum: T\_Err\_misplaced\_or\_misspelt\_Rule\_or\_T\_

Class: Err\_misplaced\_or\_misspelt\_Rule\_or\_T AB: N AD: N

Thrown by *pass3.lex* grammar. It is outside of a proper grammar's construct and does not have a symbol table entry. So it could be a misspelt T or an orphaned rule outside of its defining Rules construct.

**102. missing lrk-suffix kw.**

Enum: T\_Err\_missing\_lrk\_suffix\_kw\_

Class: Err\_missing\_lrk\_suffix\_kw AB: N AD: N

**103. missing terminals-refs kw.**

Enum: T\_Err\_missing\_terminals\_refs\_kw\_

Class: Err\_missing\_terminals\_refs\_kw AB: N AD: N

**104. missing terminals-suffix kw.**

Enum: T\_Err\_missing\_terminals\_suffix\_kw\_

Class: Err\_missing\_terminals\_suffix\_kw AB: N AD: N

**105. nested files exceeded.**

Enum: T\_Err\_nested\_files\_exceeded\_

Class: Err\_nested\_files\_exceeded AB: N AD: N

**106. nested files exceeded user-declaration directive.**

⟨ nested files exceeded user-declaration directive 106 ⟩ ≡

```
public: Err_nested_files_exceeded(yacco2 :: INT Nested_file_cnt, std :: string & File_name);
```

```
    yacco2 :: INT nested_cnt()
```

```
    {
        return nested_cnt;
    }
```

```
    ;
    std :: string * file_exceeded()
```

```
    {
        return &file_exceeded;
    }
```

```
    ;
```

```
private: yacco2 :: INT nested_cnt;
```

```
    std :: string file_exceeded;
```

**107. nested files exceeded user-implementation directive.**

```

⟨ nested files exceeded user-implementation directive 107 ⟩ ≡
  Err_nested_files_exceeded :: Err_nested_files_exceeded(yacco2 :: INT Nested_cnt,
    std :: string & File_name) T_CTOR("nested_files_exceeded",
    T_Enum :: T_Err_nested_files_exceeded_, 0, false, false)
  {
    nested_cnt_ = Nested_cnt;
    file_exceeded_ += File_name.c_str();
  }

```

**108. no \*\*\*.**

Enum: T\_Err\_no\_syntax\_code\_end\_present\_

Class: Err\_no\_syntax\_code\_end\_present

AB: N

AD: N

**109. no T in T-alphabet list.**

Enum: T\_Err\_no\_terminals\_in\_T\_alphabet\_list\_

Class: Err\_no\_terminals\_in\_T\_alphabet\_list

AB: N

AD: N

**110. no Ts in T-alphabet.**

Enum: T\_Err\_no\_terminals\_present\_in\_T\_alphabet\_

Class: Err\_no\_terminals\_present\_in\_T\_alphabet

AB: N

AD: N

**111. no close-brace.**

Enum: T\_Err\_no\_close\_brace\_

Class: Err\_no\_close\_brace

AB: N

AD: N

**112. no close-parenthesis.**

Enum: T\_Err\_no\_close\_parenthesis\_

Class: Err\_no\_close\_parenthesis

AB: N

AD: N

**113. no closing brace ending rules defs.**

Enum: T\_Err\_no\_close\_brace\_ending\_rules\_defs\_

Class: Err\_no\_close\_brace\_ending\_rules\_defs

AB: N

AD: N

**114. no cmd-lne-data.**

Enum: T\_Err\_no\_cmd\_lne\_data\_

Class: Err\_no\_cmd\_lne\_data

AB: N

AD: N

**115. no comma present.**

Enum: T\_Err\_no\_comma\_present\_

Class: Err\_no\_comma\_present

AB: N

AD: N

**116. no constant-defs keyword present.**

Enum: T\_Err\_no\_kdefs\_kw\_present\_

Class: Err\_no\_kdefs\_kw\_present

AB: N

AD: N

**117. no constant-defs-code present.**

Enum: T\_Err\_no\_kdefs\_code\_present\_

Class: Err\_no\_kdefs\_code\_present

AB: N

AD: N

**118. no constant-defs-directive present.**

Enum: T\_Err\_no\_constant\_defs\_present\_

Class: Err\_no\_constant\_defs\_present

AB: N

AD: N

**119. no directive present.**

Enum: T\_Err\_no\_directive\_present\_

Class: Err\_no\_directive\_present

AB: N

AD: N

**120. no end-T-alphabet present.**

Enum: T\_Err\_end\_T\_alphabet\_kw\_not\_present\_

Class: Err\_end\_T\_alphabet\_kw\_not\_present

AB: N

AD: N

**121. no end-list-of-native....**

Enum: T\_Err\_end\_list\_native\_T\_kw\_not\_present\_

Class: Err\_end\_list\_native\_T\_kw\_not\_present

AB: N

AD: N

**122. no end-list-of-trans....**

Enum: T\_Err\_end\_list\_of\_transitive\_threads\_kw\_not\_present\_

Class: Err\_end\_list\_of\_transitive\_threads\_kw\_not\_present

AB: N

AD: N

**123. no end-of-code.**

Enum: T\_Err\_no\_end\_of\_code\_

Class: Err\_no\_end\_of\_code

AB: N

AD: N

**124. no end-preamble present.**

Enum: T\_Err\_end\_preamble\_kw\_not\_present\_

Class: Err\_end\_preamble\_kw\_not\_present AB: N AD: N

**125. no file-name kw present.**

Enum: T\_Err\_filename\_kw\_not\_present\_

Class: Err\_filename\_kw\_not\_present AB: N AD: N

**126. no file-name present.**

Enum: T\_Err\_no\_filename\_present\_

Class: Err\_no\_filename\_present AB: N AD: N

**127. no file-name value.**

Enum: T\_Err\_filename\_value\_not\_present\_

Class: Err\_filename\_value\_not\_present AB: N AD: N

**128. no file-name-id present.**

Enum: T\_Err\_no\_filename\_id\_present\_

Class: Err\_no\_filename\_id\_present AB: N AD: N

**129. no file-of-T-alphabet.**

Enum: T\_Err\_file\_of\_T\_alphabet\_kw\_not\_present\_

Class: Err\_file\_of\_T\_alphabet\_kw\_not\_present AB: N AD: N

**130. no filename.**

Enum: T\_Err\_no\_filename\_

Class: Err\_no\_filename AB: N AD: N

**131. no fsm-class present.**

Enum: T\_Err\_no\_fsm\_class\_present\_

Class: Err\_no\_fsm\_class\_present AB: N AD: N

**132. no fsm-comments present.**

Enum: T\_Err\_no\_fsm\_comments\_present\_

Class: Err\_no\_fsm\_comments\_present AB: N AD: N

**133. no fsm-comments string present.**

Enum: T\_Err\_no\_fsm\_comments\_string\_

Class: Err\_no\_fsm\_comments\_string

AB: N

AD: N

**134. no fsm-date present.**

Enum: T\_Err\_no\_fsm\_date\_present\_

Class: Err\_no\_fsm\_date\_present

AB: N

AD: N

**135. no fsm-date string present.**

Enum: T\_Err\_no\_fsm\_date\_string\_

Class: Err\_no\_fsm\_date\_string

AB: N

AD: N

**136. no fsm-debug present.**

Enum: T\_Err\_no\_fsm\_debug\_present\_

Class: Err\_no\_fsm\_debug\_present

AB: N

AD: N

**137. no fsm-debug string present.**

Enum: T\_Err\_no\_fsm\_debug\_string\_

Class: Err\_no\_fsm\_debug\_string

AB: N

AD: N

**138. no fsm-filename id present.**

Enum: T\_Err\_no\_fsm\_filename\_id\_present\_

Class: Err\_no\_fsm\_filename\_id\_present

AB: N

AD: N

**139. no fsm-filename present.**

Enum: T\_Err\_no\_fsm\_filename\_present\_

Class: Err\_no\_fsm\_filename\_present

AB: N

AD: N

**140. no fsm-id-present.**

Enum: T\_Err\_no\_fsm\_id\_present\_

Class: Err\_no\_fsm\_id\_present

AB: N

AD: N

**141. no fsm-id-string present.**

Enum: T\_Err\_no\_fsm\_id\_string\_

Class: Err\_no\_fsm\_id\_string

AB: N

AD: N

**142. no fsm-namespace id present.**

Enum: T\_Err\_no\_fsm\_namespace\_id\_present\_

Class: Err\_no\_fsm\_namespace\_id\_present AB: N AD: N

---

**143. no fsm-namespace present.**

Enum: T\_Err\_no\_fsm\_namespace\_present\_

Class: Err\_no\_fsm\_namespace\_present AB: N AD: N

---

**144. no fsm-version present.**

Enum: T\_Err\_no\_fsm\_version\_present\_

Class: Err\_no\_fsm\_version\_present AB: N AD: N

---

**145. no fsm-version string present.**

Enum: T\_Err\_no\_fsm\_version\_string\_

Class: Err\_no\_fsm\_version\_string AB: N AD: N

---

**146. no grammar-name present.**

Enum: T\_Err\_grammar\_name\_kw\_not\_present\_

Class: Err\_grammar\_name\_kw\_not\_present AB: N AD: N

---

**147. no grammar-name value.**

Enum: T\_Err\_grammar\_name\_value\_not\_present\_

Class: Err\_grammar\_name\_value\_not\_present AB: N AD: N

---

**148. no identifier present.**

Enum: T\_Err\_no\_identifier\_present\_

Class: Err\_no\_identifier\_present AB: N AD: N

---

**149. no int present.**

Enum: T\_Err\_no\_int\_present\_

Class: Err\_no\_int\_present AB: N AD: N

---

**150. no key-value present in definition.**

Enum: T\_Err\_no\_terminal\_key\_present\_

Class: Err\_no\_terminal\_key\_present AB: N AD: N

---

**151. no list-of-native-terminals.**

Enum: T\_Err\_list\_of\_terminals\_kw\_not\_present\_

Class: Err\_list\_of\_terminals\_kw\_not\_present

AB: N

AD: N

**152. no list-of-transit....**

Enum: T\_Err\_no\_list\_of\_trans\_threads\_kw\_

Class: Err\_no\_list\_of\_trans\_threads\_kw

AB: N

AD: N

**153. no monolithic present.**

Enum: T\_Err\_monolithic\_kw\_not\_present\_

Class: Err\_monolithic\_kw\_not\_present

AB: N

AD: N

**154. no name-space.**

Enum: T\_Err\_namespace\_kw\_not\_present\_

Class: Err\_namespace\_kw\_not\_present

AB: N

AD: N

**155. no name-space present.**

Enum: T\_Err\_no\_namespace\_present\_

Class: Err\_no\_namespace\_present

AB: N

AD: N

**156. no name-space value.**

Enum: T\_Err\_namespace\_value\_not\_present\_

Class: Err\_namespace\_value\_not\_present

AB: N

AD: N

**157. no name-space-id present.**

Enum: T\_Err\_no\_namespace\_id\_present\_

Class: Err\_no\_namespace\_id\_present

AB: N

AD: N

**158. no no-of-T present.**

Enum: T\_Err\_no\_of\_T\_kw\_not\_present\_

Class: Err\_no\_of\_T\_kw\_not\_present

AB: N

AD: N

**159. no open-brace.**

Enum: T\_Err\_no\_open\_brace\_

Class: Err\_no\_open\_brace

AB: N

AD: N

**160. no open-parenthesis.**

Enum: T\_Err\_no\_open\_parenthesis\_

Class: Err\_no\_open\_parenthesis AB: N AD: N

**161. no parallel thread function.**

Enum: T\_Err\_no\_pp\_funct\_id\_present\_

Class: Err\_no\_pp\_funct\_id\_present AB: N AD: N

**162. no parallel-code.**

Enum: T\_Err\_no\_pp\_code\_present\_

Class: Err\_no\_pp\_code\_present AB: N AD: N

**163. no parallel-code-syntax-code.**

Enum: T\_Err\_no\_pp\_code\_stc\_present\_

Class: Err\_no\_pp\_code\_stc\_present AB: N AD: N

**164. no parallel-control-monitor.**

Enum: T\_Err\_no\_pp\_ctrl\_mntor\_kw\_present\_

Class: Err\_no\_pp\_ctrl\_mntor\_kw\_present AB: N AD: N

**165. no parallel-la-bndary expr.**

Enum: T\_Err\_pp\_la\_boundary\_attribute\_not\_fnd\_

Class: Err\_pp\_la\_boundary\_attribute\_not\_fnd AB: N AD: N

**166. no parallel-la-boundary.**

Enum: T\_Err\_no\_pp\_bndry\_present\_

Class: Err\_no\_pp\_bndry\_present AB: N AD: N

**167. no parallel-la-boundary-expr.**

Enum: T\_Err\_no\_pp\_la\_bndary\_expr\_present\_

Class: Err\_no\_pp\_la\_bndary\_expr\_present AB: N AD: N

**168. no parallel-thread-function.**

Enum: T\_Err\_no\_pp\_thread\_function\_present\_

Class: Err\_no\_pp\_thread\_function\_present AB: N AD: N



**169. no preamble source code.**

Enum: T\_Err\_preamble\_srce\_code\_not\_present\_

Class: Err\_preamble\_srce\_code\_not\_present AB: N AD: N

**170. no rule name present.**

Enum: T\_Err\_no\_rule\_name\_present\_

Class: Err\_no\_rule\_name\_present AB: N AD: N

**171. no sub rule present.**

Enum: T\_Err\_no\_sub\_rule\_present\_

Class: Err\_no\_sub\_rule\_present AB: N AD: N

**172. no sym-class id present.**

Enum: T\_Err\_no\_sym\_class\_id\_present\_

Class: Err\_no\_sym\_class\_id\_present AB: N AD: N

**173. no sym-class present.**

Enum: T\_Err\_no\_sym\_class\_present\_

Class: Err\_no\_sym\_class\_present AB: N AD: N

**174. no symbol definition present.**

Enum: T\_Err\_no\_sym\_defs\_present\_

Class: Err\_no\_sym\_defs\_present AB: N AD: N

**175. no syntax-code present.**

Enum: T\_Err\_no\_syntax\_code\_present\_

Class: Err\_no\_syntax\_code\_present AB: N AD: N

**176. no terminal-def-code present.**

Enum: T\_Err\_no\_tdef\_code\_present\_

Class: Err\_no\_tdef\_code\_present AB: N AD: N

**177. no thread-name present.**

Enum: T\_Err\_threadname\_kw\_not\_present\_

Class: Err\_threadname\_kw\_not\_present AB: N AD: N

**178. no thread-name value.**

Enum: T\_Err\_threadname\_value\_not\_present\_

Class: Err\_threadname\_value\_not\_present AB: N AD: N

**179. no transitive present.**

Enum: T\_Err\_transitive\_kw\_not\_present\_

Class: Err\_transitive\_kw\_not\_present AB: N AD: N

**180. no # in list-of-native-term....**

Enum: T\_Err\_no\_of\_terminals\_not\_present\_

Class: Err\_no\_of\_terminals\_not\_present AB: N AD: N

**181. no # in list-of-trans....**

Enum: T\_Err\_no\_of\_threads\_not\_present\_

Class: Err\_no\_of\_threads\_not\_present AB: N AD: N

Cross check against the  $O_2$  linker “fsc” info. The “list-of-transitive-threads” construct also states the number of entries in its list. Some how they don’t jive. i think thar’s something rotten in the fingers and diletants playing with the file?

**182. no-of-T value not present.**

Enum: T\_Err\_no\_of\_T\_value\_not\_present\_

Class: Err\_no\_of\_T\_value\_not\_present AB: N AD: N

**183. not :: in thread expr.**

Enum: T\_Err\_not\_dbl\_colon\_in\_th\_stmt\_

Class: Err\_not\_dbl\_colon\_in\_th\_stmt AB: N AD: N

**184. not a Rule in chained dispatcher expr.**

Enum: T\_Err\_not\_a\_Rule\_

Class: Err\_not\_a\_Rule AB: N AD: N

Thrown by *subrule\_def.lex* grammar supporting the new back-to-back thread call construct whereby the following procedure call is chained to the pushed “return T” of the 1st thread call. The term following the 1st thread call expression must be a rule whose contents use the *TRAs*hift operator.

**185. not a kw to start the top/down parse construct.**

Enum: T\_Err\_not\_kw\_defining\_grammar\_construct\_

Class: Err\_not\_kw\_defining\_grammar\_construct AB: N AD: N

Thrown by *pass3.lex* grammar. Caused by an identifier that has been misplaced. Could be a typo where the item should be within the defining Rules Vocabulary or a premature ending of the Rules construct by an extra close brace whereby it is trying to define a rule.

**186. not a lhs kw.**

Enum: T\_Err\_not\_a\_lhs\_kw\_

Class: Err\_not\_a\_lhs\_kw AB: N AD: N

**187. not a namespace id in thread expr.**

Enum: T\_Err\_not\_id\_for\_ns\_in\_th\_stmt\_

Class: Err\_not\_id\_for\_ns\_in\_th\_stmt AB: N AD: N

**188. not a terminal definition.**

Enum: T\_Err\_not\_a\_terminal\_definition\_

Class: Err\_not\_a\_terminal\_definition AB: N AD: N

**189. not a thread name id in expr.**

Enum: T\_Err\_not\_id\_for\_th\_name\_in\_th\_stmt\_

Class: Err\_not\_id\_for\_th\_name\_in\_th\_stmt AB: N AD: N

**190. not an arbitration-code keyword.**

Enum: T\_Err\_not\_arbitration\_code\_kw\_

Class: Err\_not\_arbitration\_code\_kw AB: N AD: N

**191. not an eosr in subrule expr.**

Enum: T\_Err\_not\_eos\_in\_subrule\_expr\_

Class: Err\_not\_eos\_in\_subrule\_expr AB: N AD: N

**192. preamble kw not present.**

Enum: T\_Err\_preamble\_kw\_not\_present\_

Class: Err\_preamble\_kw\_not\_present AB: N AD: N

**193. re-compile grammar: bad T-alphabet.**

Enum: T\_Err\_bad\_T\_alphabet\_

Class: Err\_bad\_T\_alphabet AB: N AD: N

Mismatch with the contents of the “T-alphabet” construct and the number of terminals stated in each individual grammar “fsc” file. Usually means that terminals of some ilk have changed the population and requires a recompilation per grammar to match the vocabulary numbers. That is regen grammars with the “-t” and “-err” switches. This gens the enumeration file that “o2linker” depends on.

**194. removal of term against empty set in la expr.**

Enum: T\_Err\_empty\_set\_removal\_in\_la\_expr\_

Class: Err\_empty\_set\_removal\_in\_la\_expr AB: N AD: N

Bad expression where a removal expression is against an empty set. For example, a - eolr - b is invalid. The eolr empties the set. a - eolr + b is a round about way of having just b in the set.

**195. report-card-ptr-0.**

Enum: T\_Err\_report\_card\_ptr\_0\_

Class: Err\_report\_card\_ptr\_0 AB: N AD: N

**196. rotten chr in T-alphabet.**

Enum: T\_Err\_rotten\_chr\_in\_T\_alphabet\_

Class: Err\_rotten\_chr\_in\_T\_alphabet AB: N AD: N

**197. rule already defined.**

Enum: T\_Err\_rule\_already\_defined\_

Class: Err\_rule\_already\_defined AB: N AD: N

**198. rule does not gen T strings - sick grammar.**

Enum: T\_ERR\_sick\_grammar\_

Class: ERR\_sick\_grammar AB: N AD: N

**199. rule not found in stbl.**

Enum: T\_Err\_rule\_not\_in\_stbl\_

Class: Err\_rule\_not\_in\_stbl AB: N AD: N

Post process the grammar’s vocabulary. A rule can be referenced before its definition. Just help the grammar writer of misselling or forgetfulness.

**200. rule used but undefined.**

Enum: T\_Err\_used\_rule\_but\_undefined\_

Class: Err\_used\_rule\_but\_undefined AB: N AD: N

**201. stbl char-pool full.**

Enum: T\_Err\_sym\_tbl\_char\_pool\_full\_

Class: Err\_sym\_tbl\_char\_pool\_full AB: N AD: N

**202. stbl context-buf-overflow.**

Enum: T\_Err\_sym\_tbl\_context\_buf\_overflow\_

Class: Err\_sym\_tbl\_context\_buf\_overflow AB: N AD: N

**203. stbl full.**

Enum: T\_Err\_sym\_tbl\_full\_

Class: Err\_sym\_tbl\_full AB: N AD: N

**204. stbl has entry but not a rule.**

Enum: T\_Err\_stble\_has\_entry\_but\_not\_a\_rule\_

Class: Err\_stble\_has\_entry\_but\_not\_a\_rule AB: N AD: N

**205. stbl scope-stk overflow.**

Enum: T\_Err\_sym\_tbl\_nested\_scope\_stk\_overflow\_

Class: Err\_sym\_tbl\_nested\_scope\_stk\_overflow AB: N AD: N

**206. stbl scope-stk underflow.**

Enum: T\_Err\_sym\_tbl\_nested\_scope\_stk\_underflow\_

Class: Err\_sym\_tbl\_nested\_scope\_stk\_underflow AB: N AD: N

**207. subrule overrun.**

Enum: T\_Err\_subrule\_overrun\_

Class: Err\_subrule\_overrun AB: N AD: N

**208. subscript out-of-range.**

Enum: T\_Err\_subscript\_out\_of\_range\_

Class: Err\_subscript\_out\_of\_range AB: N AD: N

**209. term not a lhs or parallel-control-monitor kw.**

Enum: T\_Err\_not\_lhs\_perl\_mntr\_

Class: Err\_not\_lhs\_perl\_mntr AB: N AD: N

**210. terminals-refs duplicate.**

Enum: T\_Err\_terminals\_refs\_dup\_def\_

Class: Err\_terminals\_refs\_dup\_def

AB: N

AD: N

**211. thread defined by another fsc file.**

Enum: T\_Err\_already\_defined\_in\_fsc\_file\_

Class: Err\_already\_defined\_in\_fsc\_file

AB: N

AD: N

Each individual thread's fsc file contains its bio. This error indicates that the grammar writer probably cloned off a grammar but did not edit properly the vital statistics. So go edit the erroneous grammar and correct its thread name, namespace etc.

**212. thread in stbl but subscript badly set.**

Enum: T\_Err\_bad\_thread\_subscript\_

Class: Err\_bad\_thread\_subscript

AB: N

AD: N

More of a sanity check in the symbol table. If this occurs i'm really in trouble.

**213. thread xrefed, but not defed.**

Enum: T\_Err\_thread\_refed\_but\_not\_defined\_

Class: Err\_thread\_refed\_but\_not\_defined

AB: N

AD: N

**214. thread xrefed, but not defed user-declaration directive.**

⟨thread xrefed, but not defed user-declaration directive 214⟩ ≡

**public:** *Err\_thread\_refed\_but\_not\_defined*(*CAbs\_lr1\_sym* \* *Th\_id*);

*CAbs\_lr1\_sym* \* *th\_id*();

**private:** *CAbs\_lr1\_sym* \* *th\_id*;

**215. thread xrefed, but not defed user-implementation directive.**

⟨thread xrefed, but not defed user-implementation directive 215⟩ ≡

*Err\_thread\_refed\_but\_not\_defined* :: *Err\_thread\_refed\_but\_not\_defined*(*CAbs\_lr1\_sym* \*

*Th\_id*)T\_CTOR("thread\_xrefed, but\_not\_defed",

*T\_Enum* :: *T\_Err\_thread\_refed\_but\_not\_defined*, 0, false, false)

```
{
  th_id = Th_id;
}
```

```
}
```

*yacco2* :: *CAbs\_lr1\_sym* \* *Err\_thread\_refed\_but\_not\_defined* :: *th\_id*()

```
{
```

```
  return th_id;
```

```
}
```

```
;
```

**216. token found in stbl but not a kw.**

Enum: T\_Err\_not\_a\_keyword\_

Class: Err\_not\_a\_keyword

AB: N

AD: N

The symbol is not a keyword. Could be an initialization problem with my  $O_2$  setup.

---

**217. transitive list thread not defined by fsc files.**

Enum: T\_Err\_bad\_th\_in\_list\_

Class: Err\_bad\_th\_in\_list

AB: N

AD: N

The hand coded “*???.fsc*” control file provides all the threads’ “fsc” files has thread entries missing. Typically the grammar writer created a new thread but forgot the include it in the global “fsc” file. So go add the missing thread entries in the global “fsc” file.

---

**218. undefined terminal in subrule.**

Enum: T\_Err\_subrule\_use\_undefined\_T\_

Class: Err\_subrule\_use\_undefined\_T

AB: N

AD: N

**219. unknown symbol type in stbl.**

Enum: T\_Err\_stbl\_entry\_unknown\_

Class: Err\_stbl\_entry\_unknown

AB: N

AD: N

**220. use of Non-terminal (rule) outside Rules’s construct.**

Enum: T\_Err\_use\_of\_N\_outside\_Rules\_construct\_

Class: Err\_use\_of\_N\_outside\_Rules\_construct

AB: N

AD: N

Thrown by *pass3.lex* grammar. The rule reference has already been seen by the parsed Rules construct and stored in the symbol table. The symbol table returns the found item as a rule. Cause is a reference to the rule outside of the Rules construct. Probably a typo by the grammar writer caused by the premature parsing of the Rules Vocabulary.

---

**221. use of T outside Rules’s construct.**

Enum: T\_Err\_use\_of\_T\_outside\_Rules\_construct\_

Class: Err\_use\_of\_T\_outside\_Rules\_construct

AB: N

AD: N

Thrown by *pass3.lex* grammar. The terminal definitions have already been parsed and stored in the symbol table. The symbol table returns the found item as a terminal rather than the unfound identifier. Cause is reference of a terminal outside of the Rules construct. Probably a typo by the grammar writer.

---

**222. zero len symbol.**

Enum: T\_Err\_zero\_len\_sym\_

Class: Err\_zero\_len\_sym

AB: N

AD: N

---

**223. Index.**

*bad\_char*: [22](#), [23](#), [24](#).  
*Bad\_char*: [24](#).  
*bad\_char\_*: [22](#), [23](#), [24](#).  
*Bad\_univ\_seq*: [63](#).  
*bad\_univ\_seq*: [62](#), [63](#).  
*bad\_univ\_seq\_*: [62](#), [63](#).  
*c\_str*: [31](#), [34](#), [37](#), [40](#), [43](#), [46](#), [49](#), [52](#), [107](#).  
*CAbs\_lr1\_sym*: [23](#), [24](#), [214](#), [215](#).  
*cweb*: [2](#).  
*dtor\_Err\_bad\_char*: [24](#).  
*Err\_bad\_char*: [23](#), [24](#).  
*Err\_bad\_enum\_filename*: [42](#), [43](#).  
*Err\_bad\_errors\_hdrfilename*: [33](#), [34](#).  
*Err\_bad\_errors\_impfilename*: [36](#), [37](#).  
*Err\_bad\_filename*: [30](#), [31](#).  
*Err\_bad\_fsmcpp\_filename*: [39](#), [40](#).  
*Err\_bad\_fsmheader\_filename*: [45](#), [46](#).  
*Err\_bad\_fsmsym\_filename*: [48](#), [49](#).  
*Err\_bad\_fsmtbl\_filename*: [51](#), [52](#).  
*Err\_bad\_univ\_seq*: [62](#), [63](#).  
*Err\_char*: [23](#).  
*Err\_nested\_files\_exceeded*: [106](#), [107](#).  
*Err\_no\_of\_native\_Ts\_in\_list\_not\_equal*: [4](#), [5](#).  
*Err\_no\_of\_threads\_in\_list\_not\_equal*: [7](#), [8](#).  
*Err\_thread\_refed\_but\_not\_defined*: [214](#), [215](#).  
*false*: [5](#), [8](#), [24](#), [31](#), [34](#), [37](#), [40](#), [43](#), [46](#), [49](#), [52](#),  
[63](#), [107](#), [215](#).  
*file\_exceeded*: [106](#).  
*file\_exceeded\_*: [106](#), [107](#).  
*file\_name*: [30](#), [31](#), [33](#), [34](#), [36](#), [37](#), [39](#), [40](#), [42](#), [43](#),  
[45](#), [46](#), [48](#), [49](#), [51](#), [52](#).  
*File\_name*: [30](#), [31](#), [33](#), [34](#), [36](#), [37](#), [39](#), [40](#), [42](#), [43](#),  
[45](#), [46](#), [48](#), [49](#), [51](#), [52](#), [106](#), [107](#).  
*file\_name\_*: [30](#), [31](#), [33](#), [34](#), [36](#), [37](#), [39](#), [40](#), [42](#), [43](#),  
[45](#), [46](#), [48](#), [49](#), [51](#), [52](#).  
*INT*: [106](#), [107](#).  
*lex*: [101](#), [184](#), [185](#), [220](#), [221](#).  
*Nested\_cnt*: [107](#).  
*nested\_cnt*: [106](#).  
*nested\_cnt\_*: [106](#), [107](#).  
*Nested\_file\_cnt*: [106](#).  
*no*: [4](#), [5](#), [7](#), [8](#).  
*No*: [5](#), [7](#), [8](#).  
*no\_*: [4](#), [5](#), [7](#), [8](#).  
*pass3*: [101](#), [185](#), [220](#), [221](#).  
*std*: [30](#), [31](#), [33](#), [34](#), [36](#), [37](#), [39](#), [40](#), [42](#), [43](#), [45](#),  
[46](#), [48](#), [49](#), [51](#), [52](#), [106](#), [107](#).  
*string*: [30](#), [31](#), [33](#), [34](#), [36](#), [37](#), [39](#), [40](#), [42](#), [43](#), [45](#),  
[46](#), [48](#), [49](#), [51](#), [52](#), [106](#), [107](#).  
*subrule\_def*: [184](#).  
*T\_CTOR*: [5](#), [8](#), [24](#), [31](#), [34](#), [37](#), [40](#), [43](#), [46](#), [49](#),  
[52](#), [63](#), [107](#), [215](#).  
*T\_Enum*: [5](#), [8](#), [24](#), [31](#), [34](#), [37](#), [40](#), [43](#), [46](#), [49](#),  
[52](#), [63](#), [107](#), [215](#).  
*T\_Err\_bad\_char\_*: [24](#).  
*T\_Err\_bad\_enum\_filename\_*: [43](#).  
*T\_Err\_bad\_errors\_hdrfilename\_*: [34](#).  
*T\_Err\_bad\_errors\_impfilename\_*: [37](#).  
*T\_Err\_bad\_filename\_*: [31](#).  
*T\_Err\_bad\_fsmcpp\_filename\_*: [40](#).  
*T\_Err\_bad\_fsmheader\_filename\_*: [46](#).  
*T\_Err\_bad\_fsmsym\_filename\_*: [49](#).  
*T\_Err\_bad\_fsmtbl\_filename\_*: [52](#).  
*T\_Err\_bad\_univ\_seq\_*: [63](#).  
*T\_Err\_nested\_files\_exceeded\_*: [107](#).  
*T\_Err\_no\_of\_native\_Ts\_in\_list\_not\_equal\_*: [5](#).  
*T\_Err\_no\_of\_threads\_in\_list\_not\_equal\_*: [8](#).  
*T\_Err\_thread\_refed\_but\_not\_defined\_*: [215](#).  
*Th\_id*: [214](#), [215](#).  
*th\_id*: [214](#), [215](#).  
*th\_id\_*: [214](#), [215](#).  
*Value*: [4](#), [62](#).  
*yacco2*: [23](#), [24](#), [106](#), [107](#), [215](#).  
*zero\_out\_bad\_char*: [23](#), [24](#).



< # T in list not eq user-declaration directive 4 >  
< # T in list not eq user-implementation directive 5 >  
< # threads in list not eq user-declaration directive 7 >  
< # threads in list not eq user-implementation directive 8 >  
< bad char destructor directive 22 >  
< bad char user-declaration directive 23 >  
< bad char user-implementation directive 24 >  
< bad filename for Errors vocabulary header user-declaration directive 33 >  
< bad filename for Errors vocabulary header user-implementation directive 34 >  
< bad filename for Errors vocabulary implementation user-declaration directive 36 >  
< bad filename for Errors vocabulary implementation user-implementation directive 37 >  
< bad filename to output cpp user-declaration directive 39 >  
< bad filename to output cpp user-implementation directive 40 >  
< bad filename to output enumeration header user-declaration directive 42 >  
< bad filename to output enumeration header user-implementation directive 43 >  
< bad filename to output grammar header user-declaration directive 45 >  
< bad filename to output grammar header user-implementation directive 46 >  
< bad filename to output sym user-declaration directive 48 >  
< bad filename to output sym user-implementation directive 49 >  
< bad filename to output tbl user-declaration directive 51 >  
< bad filename to output tbl user-implementation directive 52 >  
< bad filename user-declaration directive 30 >  
< bad filename user-implementation directive 31 >  
< bad univ-seq user-declaration directive 62 >  
< bad univ-seq user-implementation directive 63 >  
< nested files exceeded user-declaration directive 106 >  
< nested files exceeded user-implementation directive 107 >  
< thread xrefed, but not defed user-declaration directive 214 >  
< thread xrefed, but not defed user-implementation directive 215 >

## Error Vocabulary

Date: January 2, 2015 at 16:29

File: yacco2\_err\_symbols Namespace: NS\_yacco2\_err\_symbols

Number of terminals: 191

<b>Error symbols vocabulary</b> .....	2	2
<b># T in list not eq</b> .....	3	2
# T in list not eq user-declaration directive .....	4	2
# T in list not eq user-implementation directive .....	5	2
<b># threads in list not eq</b> .....	6	2
# threads in list not eq user-declaration directive .....	7	2
# threads in list not eq user-implementation directive .....	8	3
<b>? ended subrule expr</b> .....	9	3
<b>Empty file no grammar constructs present</b> .....	10	3
<b>O2 epsilon badly gened: 0 items in fsc lists</b> .....	11	3
<b>O2 epsilon: T present, but no T list</b> .....	12	3
<b>T in list not defined in T-alphabet</b> .....	13	3
<b>T not found in stbl</b> .....	14	4
<b>T not returned from a thread</b> .....	15	4
<b>T-alphabet file does not exist</b> .....	16	4
<b>T-alphabet file not present</b> .....	17	4
<b>T-alphabet kw not present</b> .....	18	4
<b>already defined AB tag</b> .....	19	4
<b>already defined AD tag</b> .....	20	4
<b>bad char</b> .....	21	4
bad char destructor directive .....	22	4
bad char user-declaration directive .....	23	5
bad char user-implementation directive .....	24	5
<b>bad cmd-opt</b> .....	25	5
<b>bad directive</b> .....	26	5
<b>bad eos</b> .....	27	5
<b>bad esc</b> .....	28	5
<b>bad filename</b> .....	29	5
bad filename user-declaration directive .....	30	6
bad filename user-implementation directive .....	31	6
<b>bad filename for Errors vocabulary header</b> .....	32	6
bad filename for Errors vocabulary header user-declaration directive .....	33	6
bad filename for Errors vocabulary header user-implementation directive .....	34	7
<b>bad filename for Errors vocabulary implementation</b> .....	35	7
bad filename for Errors vocabulary implementation user-declaration directive .....	36	7
bad filename for Errors vocabulary implementation user-implementation directive .....	37	7
<b>bad filename to output cpp</b> .....	38	8
bad filename to output cpp user-declaration directive .....	39	8
bad filename to output cpp user-implementation directive .....	40	8
<b>bad filename to output enumeration header</b> .....	41	8
bad filename to output enumeration header user-declaration directive .....	42	8
bad filename to output enumeration header user-implementation directive .....	43	9
<b>bad filename to output grammar header</b> .....	44	9
bad filename to output grammar header user-declaration directive .....	45	9
bad filename to output grammar header user-implementation directive .....	46	9
<b>bad filename to output sym</b> .....	47	10
bad filename to output sym user-declaration directive .....	48	10
bad filename to output sym user-implementation directive .....	49	10
<b>bad filename to output tbl</b> .....	50	10
bad filename to output tbl user-declaration directive .....	51	10
bad filename to output tbl user-implementation directive .....	52	11
<b>bad int-no</b> .....	53	11

bad int-no range .....	54	11
bad operator la expr: not a + or - .....	55	11
bad pos of t def .....	56	11
bad pos of terminals-refs .....	57	11
bad term in la expr .....	58	12
bad term in subrule expr .....	59	12
bad thread expr .....	60	12
bad univ-seq .....	61	12
bad univ-seq user-declaration directive .....	62	12
bad univ-seq user-implementation directive .....	63	12
command line chaffe .....	64	12
comment-overflow .....	65	13
dup ? sufx directive .....	66	13
dup-entry in sym-table .....	67	13
duplicate T_enum phrase .....	68	13
duplicate directive .....	69	13
duplicate error phrase .....	70	13
duplicate fsm phrase .....	71	13
duplicate logical name .....	72	13
duplicate lrk phrase .....	73	13
duplicate parallel-parse phrase .....	74	14
duplicate raw character phrase .....	75	14
duplicate rule phrase .....	76	14
duplicate terminal phrase .....	77	14
duplicate-entry in alphabet .....	78	14
emitfile file does not exist .....	79	14
emitfile file not present .....	80	14
emitfile kw not present .....	81	14
file of T-alphabet not present .....	82	14
file-name file does not exist .....	83	15
fsc file does not exist .....	84	15
fsc file not present .....	85	15
improper closing of rules construct .....	86	15
improper directive .....	87	15
invalid fsm-debug value .....	88	15
invalid logical name .....	89	15
invalid logical value .....	90	15
la expression calculates an empty set .....	91	15
linker's monolithic value not n or y .....	92	16
linker's transitive value not n or y .....	93	16
misplaced or missing T enumeration phrase .....	94	16
misplaced or missing errors phrase .....	95	16
misplaced or missing fsm phrase .....	96	16
misplaced or missing lrk phrase .....	97	16
misplaced or missing rc phrase .....	98	16
misplaced or missing rules phrase .....	99	16
misplaced or missing terminals phrase .....	100	16
misplaced or misspelt Rule or T outside of Rules defs .....	101	17
missing lrk-sufx kw .....	102	17
missing terminals-refs kw .....	103	17
missing terminals-sufx kw .....	104	17
nested files exceeded .....	105	17

nested files exceeded user-declaration directive .....	106	17
nested files exceeded user-implementation directive .....	107	18
no *** .....	108	18
no T in T-alphabet list .....	109	18
no Ts in T-alphabet .....	110	18
no close-brace .....	111	18
no close-parenthesis .....	112	18
no closing brace ending rules defs .....	113	18
no cmd-lne-data .....	114	18
no comma present .....	115	19
no constant-defs keyword present .....	116	19
no constant-defs-code present .....	117	19
no constant-defs-directive present .....	118	19
no directive present .....	119	19
no end-T-alphabet present .....	120	19
no end-list-of-native... .....	121	19
no end-list-of-trans... .....	122	19
no end-of-code .....	123	19
no end-preamble present .....	124	20
no file-name kw present .....	125	20
no file-name present .....	126	20
no file-name value .....	127	20
no file-name-id present .....	128	20
no file-of-T-alphabet .....	129	20
no filename .....	130	20
no fsm-class present .....	131	20
no fsm-comments present .....	132	20
no fsm-comments string present .....	133	21
no fsm-date present .....	134	21
no fsm-date string present .....	135	21
no fsm-debug present .....	136	21
no fsm-debug string present .....	137	21
no fsm-filename id present .....	138	21
no fsm-filename present .....	139	21
no fsm-id-present .....	140	21
no fsm-id-string present .....	141	21
no fsm-namespace id present .....	142	22
no fsm-namespace present .....	143	22
no fsm-version present .....	144	22
no fsm-version string present .....	145	22
no grammar-name present .....	146	22
no grammar-name value .....	147	22
no identifier present .....	148	22
no int present .....	149	22
no key-value present in definition .....	150	22
no list-of-native-terminals .....	151	23
no list-of-transit... .....	152	23
no monolithic present .....	153	23
no name-space .....	154	23
no name-space present .....	155	23
no name-space value .....	156	23
no name-space-id present .....	157	23

no no-of-T present	158	23
no open-brace	159	23
no open-parenthesis	160	24
no parallel thread function	161	24
no parallel-code	162	24
no parallel-code-syntax-code	163	24
no parallel-control-monitor	164	24
no parallel-la-bndary expr	165	24
no parallel-la-boundary	166	24
no parallel-la-boundary-expr	167	24
no parallel-thread-function	168	24
no preamble source code	169	25
no rule name present	170	25
no sub rule present	171	25
no sym-class id present	172	25
no sym-class present	173	25
no symbol definition present	174	25
no syntax-code present	175	25
no terminal-def-code present	176	25
no thread-name present	177	25
no thread-name value	178	26
no transitive present	179	26
no # in list-of-native-term...	180	26
no # in list-of-trans...	181	26
no-of-T value not present	182	26
not :: in thread expr	183	26
not a Rule in chained dispatcher expr	184	26
not a kw to start the top/down parse construct	185	27
not a lhs kw	186	27
not a namespace id in thread expr	187	27
not a terminal definition	188	27
not a thread name id in expr	189	27
not an arbitration-code keyword	190	27
not an eosr in subrule expr	191	27
preamble kw not present	192	27
re-compile grammar: bad T-alphabet	193	28
removal of term against empty set in la expr	194	28
report-card-ptr-0	195	28
rotten chr in T-alphabet	196	28
rule already defined	197	28
rule does not gen T strings - sick grammar	198	28
rule not found in stbl	199	28
rule used but undefined	200	28
stbl char-pool full	201	29
stbl context-buf-overflow	202	29
stbl full	203	29
stbl has entry but not a rule	204	29
stbl scope-stk overflow	205	29
stbl scope-stk underflow	206	29
subrule overrun	207	29
subscript out-of-range	208	29
term not a lhs or parallel-control-monitor kw	209	29

<b>terminals-refs duplicate</b> .....	<b>210</b>	<b>30</b>
<b>thread defined by another fsc file</b> .....	<b>211</b>	<b>30</b>
<b>thread in stbl but subscript badly set</b> .....	<b>212</b>	<b>30</b>
<b>thread xrefed, but not defed</b> .....	<b>213</b>	<b>30</b>
thread xrefed, but not defed user-declaration directive .....	<b>214</b>	<b>30</b>
thread xrefed, but not defed user-implementation directive .....	<b>215</b>	<b>30</b>
<b>token found in stbl but not a kw</b> .....	<b>216</b>	<b>31</b>
<b>transitive list thread not defined by fsc files</b> .....	<b>217</b>	<b>31</b>
<b>undefined terminal in subrule</b> .....	<b>218</b>	<b>31</b>
<b>unknown symbol type in stbl</b> .....	<b>219</b>	<b>31</b>
<b>use of Non-terminal (rule) outside Rules's construct</b> .....	<b>220</b>	<b>31</b>
<b>use of T outside Rules's construct</b> .....	<b>221</b>	<b>31</b>
<b>zero len symbol</b> .....	<b>222</b>	<b>31</b>
<b>Index</b> .....	<b>223</b>	<b>32</b>