

Service Reference Manual

1.0

Generated by Doxygen 1.3.2

Tue Jan 10 18:04:41 2006

Contents

1	Service Namespace Index	1
1.1	Service Namespace List	1
2	Service Hierarchical Index	3
2.1	Service Class Hierarchy	3
3	Service Compound Index	5
3.1	Service Compound List	5
4	Service File Index	7
4.1	Service File List	7
5	Service Namespace Documentation	9
5.1	Service Namespace Reference	9
5.2	std Namespace Reference	16
6	Service Class Documentation	17
6.1	Service::ClientSocket Class Reference	17
6.2	CTTNEException Class Reference	19
6.3	Service::ServerSocket Class Reference	21
6.4	Service::Socket Class Reference	23
6.5	yy_buffer_state Struct Reference	26
6.6	yyalloc Union Reference	27
6.7	YYSTYPE Union Reference	28
7	Service File Documentation	29
7.1	client.cpp File Reference	29
7.2	Constants.cpp File Reference	30
7.3	Constants.h File Reference	31
7.4	CTTNEException.cpp File Reference	33

7.5	CTTNEException.h File Reference	34
7.6	RPParser.cc File Reference	35
7.7	RPParser.h File Reference	45
7.8	RPParser.hh File Reference	46
7.9	RPParserTest.cpp File Reference	48
7.10	RPScanner.cc File Reference	49
7.11	server.cpp File Reference	59
7.12	Service.cpp File Reference	60
7.13	Service.h File Reference	61
7.14	ServiceTest.cpp File Reference	62
7.15	Socket.cpp File Reference	63
7.16	Socket.h File Reference	64

Chapter 1

Service Namespace Index

1.1 Service Namespace List

Here is a list of all namespaces with brief descriptions:

Service (Is the top class for server and client sockets)	9
std	16

Chapter 2

Service Hierarchical Index

2.1 Service Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

CTTNEException	19
Service::Socket	23
Service::ClientSocket	17
Service::ServerSocket	21
yy_buffer_state	26
yyalloc	27
YYSTYPE	28

Chapter 3

Service Compound Index

3.1 Service Compound List

Here are the classes, structs, unions and interfaces with brief descriptions:

Service::ClientSocket (ClientSocket specializes the Socket class to work as client)	17
CTTNEException	19
Service::ServerSocket (ServerSocket specializes the Socket class to work as server)	21
Service::Socket	23
yy_buffer_state	26
yyalloc	27
YYSTYPE	28

Chapter 4

Service File Index

4.1 Service File List

Here is a list of all files with brief descriptions:

client.cpp	29
Constants.cpp	30
Constants.h	31
CTTNEexception.cpp	33
CTTNEexception.h	34
RPParser.cc	35
RPParser.h	45
RPParser.hh	46
RPParserTest.cpp	48
RPScanner.cc	49
server.cpp	59
Service.cpp	60
Service.h	61
ServiceTest.cpp	62
Socket.cpp	63
Socket.h	64

Chapter 5

Service Namespace Documentation

5.1 Service Namespace Reference

is the top class for server and client sockets.

Compounds

- class **ClientSocket**
ClientSocket specializes the Socket class to work as client.
- class **ServerSocket**
ServerSocket specializes the Socket class to work as server.
- class **Socket**

Functions

- string **expandRepetitionPattern** (string compressed, vector< string > &result, const string &separator=",")
expands a "repetition pattern" into a vector of strings.
- string & **eraseWhitespace** (string &s)
erases all blanks, \n and \r from the string
- string & **eraseLeadingWhitespace** (string &s)
erases leading blanks, \n and \r from the string
- string & **eraseMultipleWhitespace** (string &s)
replaces multiple blanks, \n and \r by single blanks
- string & **eraseLineFollowings** (string &s)
replaces all \\n by a blank

- string & **eraseLinefeeds** (string &s)
replaces all $\backslash n$ by a blank
- string & **stripComment** (string &s, const string &cmt)
erases all characters after 'cmt'
- string & **stripComments** (string &s, const string &cmt)
erases in each line all characters after the 'cmt' character
- int **linebreaks** (const string &s)
returns the number of linebreaks in the string.
- string **intToString** (int i, int length)
converts the integer to a string and fills the head with blanks such that the string is 'length' characters long.
- string **withLineNumbers** (string s)
inserts line numbers into the string (for listing code)
- vector< string > **split** (const string &separator, string s, bool erasewithspace)
splits the string into substrings separated by 'separator' and erases all whitespaces if asked.
- string **readFile** (const string &file, const string &cmt)
reads the contents of the file and returns it as a string.
- **Co nDividable** (**Co n**, **Co m**, int k)
computes how many numbers between n and m are dividable by k.
- **Rt NullFct** (const **Co** &coordinate)
is a constant function which returns 0.
- int **moduloPos** (const **Co** &i, int n)
computes i modulo n and returns a positive number.
- void **remainder** (const **Co** &coordinate, int divisor, **Co** &blocks, int &rest)
*computes $c / divisor = blocks * divisor + rest$.*
- void **remainder** (const **Co** &coordinate, **FLT** divisor, **Co** &blocks, **FLT** &rest)
*computes $c / divisor = blocks * divisor + rest$.*
- **Rt roundX** (float x)
rounds x to a Rt value.
- **Fuzzy roundY** (float x)
rounds y to a 'Fuzzy' value
- bool **isPosInfinity** (const **Rt** &x)
A check for positive infinity.

- **bool isNegInfinity** (const **Rt** &x)
A check for negative infinity.
- **bool isInfinity** (const **Rt** &x)
A check for either positive or negative infinity.
- **template<class I> string xToString** (const I &i)
converts 'i' into a string.
- **template<class I> string join** (const string &separator, const vector< I > &items)
joins all items into a 'separator' separated string.
- **template<class I> string parseItem** (const string &strng, I &data, string &rest)
parses the complete string into data by using the >> operator.
- **template<class I> string parseItemVector** (const vector< string > &strng, vector< I > &vect)
parses strings of items into a vector.
- **template<class I> string parseItems** (const string &strng, vector< I > &vect, const string &separator="/"")
parses strings of items separated by 'separator' into a vector of items.
- **template<class I> string parseNestedItems** (const string &strng, vector< vector< I > > &pDates, const string &topseparator="'", const string &bottomseparator="/"")
parses nested strings of items separated by a top-separator and a bottom-separator.
- **template<class I> vector< FLT > vectorDifference** (const vector< I > &a, const vector< I > &b, bool clean=false)
computes the componentwise difference between the two vectors.
- **int toInt** (const **Co** &i)
converts i into an int.
- **float toFloat** (const **Rt** &x)
converts x into a float.
- **FLT toFLT** (const **Co** &x)
converts x into a FLT.
- **Rt toRt** (float i)
converts i to a Rt.

5.1.1 Detailed Description

is the top class for server and client sockets.

5.1.2 Function Documentation

5.1.2.1 `string& eraseLeadingWhitespace (string & s)`

erases leading blanks, `\n` and `\r` from the string

5.1.2.2 `string& eraseLinefeeds (string & s)`

replaces all `\n` by a blank

5.1.2.3 `string& eraseLineFollowings (string & s)`

replaces all `\\n` by a blank

5.1.2.4 `string& eraseMultipleWhitespace (string & s)`

replaces multiple blanks, `\n` and `\r` by single blanks

5.1.2.5 `string& eraseWhitespace (string & s)`

erases all blanks, `\n` and `\r` from the string

5.1.2.6 `string expandRepetitionPattern (string compressed, vector< string > & result, const string & separator = ",")`

expands a "repetition pattern" into a vector of strings.

A "repetition pattern" is either

- a token consisting of characters or numbers, e.g. "ab", "+2", "-22", "+2ab" or
- a 'separator' separated list of tokens or expressions, e.g. "ab,23, 45" or
- an expression * token, e.g. 2*ab or
- an expression *(expression), e.g. "2*(a,b)" or "2*(a,3*(b,c),4)".

The multiplicities are expanded and stored into 'result', e.g. "2*(-2, 2*(33,44a)) , 3*ff" yields "-2|33|44a|33|44a|-2|33|44a|33|44a|ff|ff|ff".

Returns:

an error string if a parse error occurred.

5.1.2.7 `string intToString (int i, int length)`

converts the integer to a string and fills the head with blanks such that the string is 'length' characters long.

5.1.2.8 `bool isInfinity (const Rt & x)`

A check for either positive or negative infinity.

5.1.2.9 bool isNegInfinity (const Rt & x)

A check for negative infinity.

5.1.2.10 bool isPosInfinity (const Rt & x)

A check for positive infinity.

5.1.2.11 template<class I> string join (const string & separator, const vector< I > & items)

joins all items into a 'separator' separated string.

works for all types which can be converted to a string with xToString

5.1.2.12 int linebreaks (const string & s)

returns the number of linebreaks in the string.

5.1.2.13 int moduloPos (const Co & i, int n)

computes i modulo n and returns a positive number.

5.1.2.14 Co nDividable (Co n, Co m, int k)

computes how many numbers between n and m are dividable by k.

5.1.2.15 Rt NullFct (const Co & coordinate)

is a constant function which returns 0.

5.1.2.16 template<class I> string parseItem (const string & strng, I & data, string & rest)

parses the complete string into data by using the >> operator.

If a parsing error occurs, an error string is returned and the rest of the string which could not be parsed is bound to 'rest'.

5.1.2.17 template<class I> string parseItems (const string & strng, vector< I > & vect, const string & separator = "/")

parses strings of items separated by 'separator' into a vector of items.

It is usable for every datatype with >> operator.

Returns:

an error string if not everything could be parsed properly.

5.1.2.18 `template<class I> string parseItemVector (const vector< string > & strng, vector< I > & vect)`

parses strings of items into a vector.

It is usable for every datatype with >> operator.

Returns:

an error string if not everything could be parsed properly.

5.1.2.19 `template<class I> string parseNestedItems (const string & strng, vector< vector< I > > & pDates, const string & topseparator = " ", const string & bottomseparator = "/")`

parses nested strings of items separated by a top-separator and a bottom-separator.

The function splits the string at the 'topseparator' and calls 'parseVector' for the split parts. The results are collected in a vector. It is usable for every datatype with >> operator. The function returns an error string if not everything could be parsed properly.

5.1.2.20 `string readFile (const string & file, const string & cmt)`

reads the contents of the file and returns it as a string.

Comments starting with 'cmt' in a line are deleted. Lines ending with \ are concatenated.

5.1.2.21 `void remainder (const Co & coordinate, FLT divisor, Co & blocks, FLT & rest)`

computes $c / \text{divisor} = \text{blocks} * \text{divisor} + \text{rest}$.

5.1.2.22 `void remainder (const Co & coordinate, int divisor, Co & blocks, int & rest)`

computes $c / \text{divisor} = \text{blocks} * \text{divisor} + \text{rest}$.

5.1.2.23 `Rt roundX (float x)`

rounds x to a Rt value.

5.1.2.24 `Fuzzy roundY (float x)`

rounds y to a 'Fuzzy' value

5.1.2.25 `vector<string> split (const string & separator, string s, bool erasewithespace)`

splits the string into substrings separated by 'separator' and erases all whitespaces if asked.

5.1.2.26 `string& stripComment (string & s, const string & cmt)`

erases all characters after 'cmt'

5.1.2.27 `string& stripComments (string & s, const string & cmt)`

erases in each line all characters after the 'cmt' character

5.1.2.28 `float toFloat (const Rt & x) [inline]`

converts x into a float.

5.1.2.29 `FLT toFLT (const Co & x) [inline]`

converts x into a FLT.

5.1.2.30 `int toInt (const Co & i) [inline]`

converts i into an int.

5.1.2.31 `Rt toRt (float i) [inline]`

converts i to a Rt.

5.1.2.32 `template<class I> vector<FLT> vectorDifference (const vector< I > & a, const vector< I > & b, bool clean = false)`

computes the componentwise difference between the two vectors.

The items can be of any class which supports the '-' operator and which can be converted into a FLT. The vectors can be of different size. If 'clean' = true then trailing zeros are deleted.

5.1.2.33 `string withLineNumbers (string s)`

inserts line numbers into the string (for listing code)

5.1.2.34 `template<class I> string xToString (const I & i)`

converts 'i' into a string.

5.2 std Namespace Reference

Functions

- `template<class I> ostream & operator<< (ostream &s, const vector< I > &vec)`
a generic << operator for vectors.
- `template<class I> ostream & operator<< (ostream &s, const vector< I * > &vec)`
a generic << operator for vectors of pointers
- `template<class I> ostream & operator<< (ostream &s, const pair< string, vector< I * > * > &vec)`
a generic << operator for vectors of pairs of strings and a vector to pointers.
- `template<class I> ostream & operator<< (ostream &s, const pair< string, vector< I > * > &vec)`
a generic << operator for vectors of pairs of strings and pointers.

5.2.1 Function Documentation

5.2.1.1 `template<class I> ostream& operator<< (ostream & s, const pair< string, vector< I > * > & vec)`

a generic << operator for vectors of pairs of strings and pointers.

5.2.1.2 `template<class I> ostream& operator<< (ostream & s, const pair< string, vector< I * > * > & vec)`

a generic << operator for vectors of pairs of strings and a vector to pointers.

5.2.1.3 `template<class I> ostream& operator<< (ostream & s, const vector< I * > & vec)`

a generic << operator for vectors of pointers

5.2.1.4 `template<class I> ostream& operator<< (ostream & s, const vector< I > & vec)`

a generic << operator for vectors.

Chapter 6

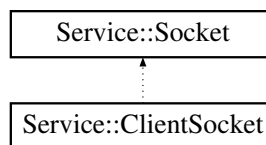
Service Class Documentation

6.1 Service::ClientSocket Class Reference

ClientSocket specializes the **Socket** class to work as client.

```
#include <Socket.h>
```

Inheritance diagram for Service::ClientSocket::



Public Member Functions

- **ClientSocket** (std::string host, int port)
creates a client socket for the given host and port.
- virtual ~**ClientSocket** ()
- const **ClientSocket** & **operator**<< (const std::string &) const
writes a string to the socket.
- const **ClientSocket** & **operator**>> (std::string &) const
reads a string from the socket.

Static Public Member Functions

- string **readUrl** (const string &url, const string &cmt)
reads a file from the given url and strips comments after 'cmt' string.

6.1.1 Detailed Description

`ClientSocket` specializes the `Socket` class to work as client.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 `Service::ClientSocket::ClientSocket (std::string host, int port)`

creates a client socket for the given host and port.

Exceptions:

if this fails.

6.1.2.2 `virtual Service::ClientSocket::~~ClientSocket () [inline, virtual]`

6.1.3 Member Function Documentation

6.1.3.1 `const ClientSocket & Service::ClientSocket::operator<< (const std::string &) const`

writes a string to the socket.

Exceptions:

if the write operation fails

6.1.3.2 `const ClientSocket & Service::ClientSocket::operator>> (std::string &) const`

reads a string from the socket.

Exceptions:

if the read operation fails

6.1.3.3 `string Service::ClientSocket::readUrl (const string & url, const string & cmt) [static]`

reads a file from the given url and strips comments after '*cmt*' string.

'url' is either a file name or a string starting with "http://"

Exceptions:

if the read operation fails

The documentation for this class was generated from the following files:

- `Socket.h`
- `Socket.cpp`

6.2 CTTNException Class Reference

```
#include <CTTNException.h>
```

Public Member Functions

- **CTTNException** ()
- **CTTNException** (bool **systemError**, const string &**module**, const string &**file**, const string &**method**, const string &**message**, const string &**hint**="")

Public Attributes

- string **module**
the module where the exception occurred.
- string **file**
the file where the exception occurred.
- string **method**
the method where the exception occurred.
- string **message**
an explanation
- string **recommendation**
a recommendation for the user.
- bool **systemError**
a flag indicating whether the error is a system error.

Friends

- ostream & **operator**<< (ostream &s, const **CTTNException** &e)

6.2.1 Constructor & Destructor Documentation

6.2.1.1 `CTTNEException::CTTNEException ()` [inline]

6.2.1.2 `CTTNEException::CTTNEException (bool systemError, const string & module, const string & file, const string & method, const string & message, const string & hint = "")` [inline]

6.2.2 Friends And Related Function Documentation

6.2.2.1 `ostream& operator<< (ostream & s, const CTTNEException & e)` [friend]

6.2.3 Member Data Documentation

6.2.3.1 `string CTTNEException::file`

the file where the exception occurred.

6.2.3.2 `string CTTNEException::message`

an explanation

6.2.3.3 `string CTTNEException::method`

the method where the exception occurred.

6.2.3.4 `string CTTNEException::module`

the module where the exception occurred.

6.2.3.5 `string CTTNEException::recommendation`

a recommendation for the user.

6.2.3.6 `bool CTTNEException::systemError`

a flag indicating whether the error is a system error.

The documentation for this class was generated from the following file:

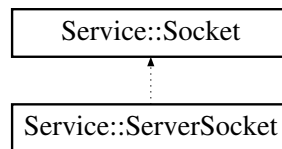
- `CTTNEException.h`

6.3 Service::ServerSocket Class Reference

ServerSocket specializes the **Socket** class to work as server.

```
#include <Socket.h>
```

Inheritance diagram for Service::ServerSocket:



Public Member Functions

- **ServerSocket** (int port)
creates a socket, binds it to the port and sets it into listening state.
- **ServerSocket** ()
- virtual **~ServerSocket** ()
- const **ServerSocket** & **operator**<< (const std::string &) const
writes a string to the socket.
- const **ServerSocket** & **operator**>> (std::string &) const
reads a string from the socket.
- void **accept** (**ServerSocket** &s)
accepts a connection from a client and binds it to s.

6.3.1 Detailed Description

ServerSocket specializes the **Socket** class to work as server.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 Service::ServerSocket::ServerSocket (int *port*)

creates a socket, binds it to the port and sets it into listening state.

Exceptions:

if one of the operations fails

6.3.2.2 `Service::ServerSocket::ServerSocket ()` [inline]

6.3.2.3 `Service::ServerSocket::~~ServerSocket ()` [virtual]

6.3.3 Member Function Documentation

6.3.3.1 `void Service::ServerSocket::accept (ServerSocket & s)`

accepts a connection from a client and binds it to s.

6.3.3.2 `const ServerSocket & Service::ServerSocket::operator<< (const std::string &) const`

writes a string to the socket.

Exceptions:

if the write operation fails

6.3.3.3 `const ServerSocket & Service::ServerSocket::operator>> (std::string &) const`

reads a string from the socket.

Exceptions:

if the read operation fails

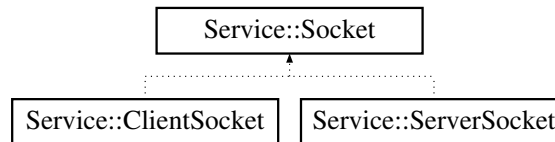
The documentation for this class was generated from the following files:

- `Socket.h`
- `Socket.cpp`

6.4 Service::Socket Class Reference

```
#include <Socket.h>
```

Inheritance diagram for Service::Socket::



Public Member Functions

- **Socket** ()
creates a socket, but does not initialise it.
- virtual **~Socket** ()
- bool **create** ()
creates a socket.
- bool **bind** (const int port)
binds the socket to a given port
- bool **listen** () const
activates the socket for listening to incoming connections
- bool **accept** (**Socket** &s) const
accepts a connection and binds to s.
- bool **connect** (const std::string host, const int port)
connects the (client) socket to a host and a port.
- bool **send** (const std::string) const
sends a string to the socket.
- int **recv** (std::string &) const
reads a string from the socket and binds it to s
- void **set_non_blocking** (const bool b)
sets the blocking status
- bool **is_valid** () const
returns true if the socket is operational.

Private Attributes

- int **m_sock**
- sockaddr_in **m_addr**

6.4.1 Constructor & Destructor Documentation

6.4.1.1 `Service::Socket::Socket ()`

creates a socket, but does not initialise it.

6.4.1.2 `Service::Socket::~~Socket ()` [virtual]

6.4.2 Member Function Documentation

6.4.2.1 `bool Service::Socket::accept (Socket & s) const`

accepts a connection and binds to *s*.

Returns:

true if it succeeded

6.4.2.2 `bool Service::Socket::bind (const int port)`

binds the socket to a given port

Returns:

true if the binding was successful.

6.4.2.3 `bool Service::Socket::connect (const std::string host, const int port)`

connects the (client) socket to a host and a port.

Returns:

true if it succeeded

6.4.2.4 `bool Service::Socket::create ()`

creates a socket.

Returns:

true if creation was successful.

6.4.2.5 `bool Service::Socket::is_valid () const` [inline]

returns true if the socket is operational.

6.4.2.6 `bool Service::Socket::listen () const`

activates the socket for listening to incoming connections

Returns:

true if the socket is now listening

6.4.2.7 int Service::Socket::recv (std::string &) const

reads a string from the socket and binds it to s

Returns:

a status value:

- -1: read error
- 0: nothing read
- 1: string read.

6.4.2.8 bool Service::Socket::send (const std::string) const

sends a string to the socket.

Returns:

true if it succeeded

6.4.2.9 void Service::Socket::set_non_blocking (const bool b)

sets the blocking status

- b == true: non-blocking
- b == false: blocking

6.4.3 Member Data Documentation

6.4.3.1 sockaddr_in Service::Socket::m_addr [private]

6.4.3.2 int Service::Socket::m_sock [private]

The documentation for this class was generated from the following files:

- Socket.h
- Socket.cpp

6.5 yy_buffer_state Struct Reference

Public Attributes

- FILE * yy_input_file
- char * yy_ch_buf
- char * yy_buf_pos
- yy_size_t yy_buf_size
- int yy_n_chars
- int yy_is_our_buffer
- int yy_is_interactive
- int yy_at_bol
- int yy_fill_buffer
- int yy_buffer_status

6.5.1 Member Data Documentation

6.5.1.1 int yy_buffer_state::yy_at_bol

6.5.1.2 char* yy_buffer_state::yy_buf_pos

6.5.1.3 yy_size_t yy_buffer_state::yy_buf_size

6.5.1.4 int yy_buffer_state::yy_buffer_status

6.5.1.5 char* yy_buffer_state::yy_ch_buf

6.5.1.6 int yy_buffer_state::yy_fill_buffer

6.5.1.7 FILE* yy_buffer_state::yy_input_file

6.5.1.8 int yy_buffer_state::yy_is_interactive

6.5.1.9 int yy_buffer_state::yy_is_our_buffer

6.5.1.10 int yy_buffer_state::yy_n_chars

The documentation for this struct was generated from the following file:

- RPScanner.cc

6.6 yyalloC Union Reference

Public Attributes

- short yyss
- YYSTYPE yyvs

6.6.1 Member Data Documentation

6.6.1.1 short yyalloC::yyss

6.6.1.2 YYSTYPE yyalloC::yyvs

The documentation for this union was generated from the following file:

- RPParser.cc

6.7 YYSTYPE Union Reference

```
#include <RPParser.hh>
```

Public Attributes

- void * **value**
- int **start**
- void * **value**

6.7.1 Member Data Documentation

6.7.1.1 int YYSTYPE::start

6.7.1.2 void* YYSTYPE::value

6.7.1.3 void* YYSTYPE::value

The documentation for this union was generated from the following files:

- **RPParser.cc**
- **RPParser.hh**

Chapter 7

Service File Documentation

7.1 client.cpp File Reference

```
#include "Socket.h"  
#include <iostream>  
#include <string>  
#include "CTNException.h"
```

Functions

- `int main (int argc, int argv[])`

7.1.1 Function Documentation

7.1.1.1 `int main (int argc, int argv[])`

7.2 Constants.cpp File Reference

```
#include "Constants.h"
```

7.3 Constants.h File Reference

Defines

- `#define FTOP 1000`
This is the top fuzzy integer value.
- `#define fFTOP 1000.0`
This is the top fuzzy integer value as float.
- `#define posINFINITY 9223372036854775807LL`
largest positive number
- `#define negINFINITY -9223372036854775800LL`
largest negative number
- `#define ZERO 0LL`
0
- `#define ONE 1LL`
1

Typedefs

- `typedef short int Fuzzy`
integer type for fuzzy values in FuTI
- `typedef float FLT`
can be changed to double
- `typedef long long int Co`
coordinates of partitions in PartLib
- `typedef long long int Rt`
reference time in PartLib and FuTI

7.3.1 Define Documentation

7.3.1.1 `#define fFTOP 1000.0`

This is the top fuzzy integer value as float.

7.3.1.2 `#define FTOP 1000`

This is the top fuzzy integer value.

7.3.1.3 `#define negINFINITY -9223372036854775800LL`

largest negative number

7.3.1.4 `#define ONE 1LL`

1

7.3.1.5 `#define posINFINITY 9223372036854775807LL`

largest positive number

7.3.1.6 `#define ZERO 0LL`

0

7.3.2 Typedef Documentation

7.3.2.1 `typedef long long int Co`

coordinates of partitions in PartLib

7.3.2.2 `typedef float FLT`

can be changed to double

7.3.2.3 `typedef short int Fuzzy`

integer type for fuzzy values in FuTI

7.3.2.4 `typedef long long int Rt`

reference time in PartLib and FuTI

7.4 CTTNException.cpp File Reference

```
#include "CTTNException.h"
```

Functions

- ostream & **operator**<< (ostream &s, const **CTTNException** &e)

7.4.1 Function Documentation

7.4.1.1 ostream& **operator**<< (ostream & s, const **CTTNException** & e)

7.5 CTTNException.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string>
#include <iostream>
```

Compounds

- class **CTTNException**

7.6 RPParser.cc File Reference

```
#include "Service.h"
#include "RPParser.h"
```

Compounds

- union **yyalloc**
- union **YYSTYPE**

Defines

- #define **YYBISON** 1
- #define **YYSKELETON_NAME** "yacc.c"
- #define **YYPURE** 0
- #define **YYLSP_NEEDED** 0
- #define **yyparse** RP_parse
- #define **yylex** RP_lex
- #define **yyerror** RP_error
- #define **yyval** RP_lval
- #define **yychar** RP_char
- #define **yydebug** RP_debug
- #define **ynerrs** RP_nerrs
- #define **TOKEN** 258
- #define **KOMMA** 259
- #define **TIMES** 260
- #define **OPENPARENTHESIS** 261
- #define **CLOSEPARENTHESIS** 262
- #define **ERROR** 263
- #define **YYDEBUG** 0
- #define **YYERROR_VERBOSE** 0
- #define **yystype** YYSTYPE
- #define **YYSTYPE_IS_DECLARED** 1
- #define **YYSTYPE_IS_TRIVIAL** 1
- #define **YYSTACK_ALLOC** malloc
- #define **YYSTACK_FREE** free
- #define **YYSTACK_GAP_MAXIMUM** (sizeof (union yyalloc) - 1)
- #define **YYSTACK_BYTES(N)**
- #define **YYCOPY**(To, From, Count)
- #define **YYSTACK_RELOCATE**(Stack)
- #define **YYFINAL** 8
- #define **YYLAST** 12
- #define **YYNTOKENS** 9
- #define **YYNNTS** 4
- #define **YYNRULES** 10
- #define **YYNSTATES** 17
- #define **YYUNDEFTOK** 2
- #define **YYMAXUTOK** 263

- `#define YYTRANSLATE(YYX) ((unsigned int) (YYX) <= YYMAXUTOK ? yytranslate[YYX] : YYUNDEFTOK)`
- `#define YYPACT_NINF -7`
- `#define YYTABLE_NINF -1`
- `#define YYSIZE_T unsigned int`
- `#define yyerrok (yerrstatus = 0)`
- `#define yyclearin (yychar = YYEMPTY)`
- `#define YYEMPTY (-2)`
- `#define YYEOF 0`
- `#define YYACCEPT goto yyacceptlab`
- `#define YYABORT goto yyabortlab`
- `#define YYERROR goto yyerrlab1`
- `#define YYFAIL goto yyerrlab`
- `#define YYRECOVERING() (!yerrstatus)`
- `#define YYBACKUP(Token, Value)`
- `#define YYTERROR 1`
- `#define YYERRCODE 256`
- `#define YYLOC_DEFAULT(Current, Rhs, N)`
- `#define YYLEX yylex ()`
- `#define YYDPRINTF(Args)`
- `#define YYDSYMPRINT(Args)`
- `#define YYDSYMPRINTF(Title, Token, Value, Location)`
- `#define YY_STACK_PRINT(Bottom, Top)`
- `#define YY_REDUCE_PRINT(Rule)`
- `#define YYINITDEPTH 200`
- `#define YYMAXDEPTH 10000`
- `#define YYPOPSTACK (yyvsp-, yyssp-)`

Typedefs

- `typedef YYSTYPE YYSTYPE`
- `typedef short yysigned_char`

Enumerations

- `enum yytokentype {`
`TOKEN = 258, KOMMA = 259, TIMES = 260, OPENPARENTHESIS = 261,`
`CLOSEPARENTHESIS = 262, ERROR = 263 }`

Functions

- `int RP_lex (void)`
- `void RP_Scan (const char *st)`
- `void RP_ClearBuffer ()`
- `void RP_append (int from, int to)`
- `int yyerror (char *s)`
- `void yydestruct (yytype, yyvaluep) int yytype`
- `int yyparse ()`

Variables

- vector< string > **RP_Strings**
- string **RP_Error** = ""
- const unsigned char **yytranslate** []
- const unsigned char **yyr1** []
- const unsigned char **yyr2** []
- const unsigned char **yydefact** []
- const **yysigned_char** **yydefgoto** []
- const **yysigned_char** **yypact** []
- const **yysigned_char** **yypgoto** []
- const unsigned char **yytable** []
- const unsigned char **yycheck** []
- const unsigned char **yystos** []
- **YYSTYPE** * **yyvaluep**
- int **yychar**
- **YYSTYPE** **yylval**
- int **yynerrs**

7.6.1 Define Documentation

7.6.1.1 **#define CLOSEPARENTHESIS 262**

7.6.1.2 **#define ERROR 263**

7.6.1.3 **#define KOMMA 259**

7.6.1.4 **#define OPENPARENTHESIS 261**

7.6.1.5 **#define TIMES 260**

7.6.1.6 **#define TOKEN 258**

7.6.1.7 **#define YY_REDUCE_PRINT(Rule)**

7.6.1.8 **#define YY_STACK_PRINT(Bottom, Top)**

7.6.1.9 **#define YYABORT goto yyabortlab**

7.6.1.10 **#define YYACCEPT goto yyacceptlab**

7.6.1.11 **#define YYBACKUP(Token, Value)**

Value:

```
do
  if (yychar == YYEMPTY && yylen == 1)
  {
    yychar = (Token);
    yylval = (Value);
    yytoken = YYTRANSLATE (yychar);
    YYPOPSTACK;
    goto yybackup;
  }
  \
  \
  \
  \
  \
  \
  \
```

```

    }
else
{
    yyerror ("syntax error: cannot back up");\
    YYERROR;\
}
while (0)

```

7.6.1.12 #define YYBISON 1

7.6.1.13 #define yychar RP_char

7.6.1.14 #define yyclearin (yychar = YYEMPTY)

7.6.1.15 #define YYCOPY(To, From, Count)

Value:

```

do
{
    register YYSIZE_T yyi;\
    for (yyi = 0; yyi < (Count); yyi++)\
        (To)[yyi] = (From)[yyi];\
}
while (0)

```

- 7.6.1.16 `#define YYDEBUG 0`
- 7.6.1.17 `#define yydebug RP_debug`
- 7.6.1.18 `#define YYDPRINTF(Args)`
- 7.6.1.19 `#define YYDSYMPRINT(Args)`
- 7.6.1.20 `#define YYDSYMPRINTF(Title, Token, Value, Location)`
- 7.6.1.21 `#define YYEMPTY (-2)`
- 7.6.1.22 `#define YYEOF 0`
- 7.6.1.23 `#define YYERRCODE 256`
- 7.6.1.24 `#define yyerrok (yyerrstatus = 0)`
- 7.6.1.25 `#define YYERROR goto yyerrlab1`
- 7.6.1.26 `#define yyerror RP_error`
- 7.6.1.27 `#define YYERROR_VERBOSE 0`
- 7.6.1.28 `#define YYFAIL goto yyerrlab`
- 7.6.1.29 `#define YYFINAL 8`
- 7.6.1.30 `#define YYINITDEPTH 200`
- 7.6.1.31 `#define YYLAST 12`
- 7.6.1.32 `#define YYLEX yylex ()`
- 7.6.1.33 `#define yylex RP_lex`
- 7.6.1.34 `#define YYLOC_DEFAULT(Current, Rhs, N)`

Value:

```
Current.first_line  = Rhs[1].first_line;    \  
Current.first_column = Rhs[1].first_column; \  
Current.last_line   = Rhs[N].last_line;     \  
Current.last_column = Rhs[N].last_column;
```

```

7.6.1.35 #define YYLSP_NEEDED 0
7.6.1.36 #define yylval RP_lval
7.6.1.37 #define YYMAXDEPTH 10000
7.6.1.38 #define YYMAXUTOK 263
7.6.1.39 #define yynerrs RP_nerrs
7.6.1.40 #define YYNNTS 4
7.6.1.41 #define YYNRULES 10
7.6.1.42 #define YYNSTATES 17
7.6.1.43 #define YYNTOKENS 9
7.6.1.44 #define YYPACT_NINF -7
7.6.1.45 #define yyparse RP_parse
7.6.1.46 #define YYPOPSTACK (yyvsp-, yyssp-)
7.6.1.47 #define YYPURE 0
7.6.1.48 #define YYRECOVERING() (!!yyerrstatus)
7.6.1.49 #define YYSIZE_T unsigned int
7.6.1.50 #define YYSKELETON_NAME "yacc.c"
7.6.1.51 #define YYSTACK_ALLOC malloc
7.6.1.52 #define YYSTACK_BYTES(N)

```

Value:

```

((N) * (sizeof (short) + sizeof (YYSTYPE))          \
 + YYSTACK_GAP_MAXIMUM)

```

```

7.6.1.53 #define YYSTACK_FREE free
7.6.1.54 #define YYSTACK_GAP_MAXIMUM (sizeof (union yyalloc) - 1)
7.6.1.55 #define YYSTACK_RELOCATE(Stack)

```

Value:

```

do
{
  YYSIZE_T yynewbytes;
  YCOPY (&yyptr->Stack, Stack, yysize);

```

```
Stack = &yyptr->Stack; \
yynewbytes = yystacksize * sizeof (*Stack) + YYSTACK_GAP_MAXIMUM; \
yyptr += yynewbytes / sizeof (*yyptr); \
} \
while (0)
```

7.6.1.56 `#define yystype YYSTYPE`

7.6.1.57 `#define YYSTYPE_IS_DECLARED 1`

7.6.1.58 `#define YYSTYPE_IS_TRIVIAL 1`

7.6.1.59 `#define YYTABLE_NINF -1`

7.6.1.60 `#define YYTERROR 1`

7.6.1.61 `#define YYTRANSLATE(YYX) ((unsigned int) (YYX) <= YYMAXUTOK ? yytranslate[YYX] : YYUNDEFTOK)`

7.6.1.62 `#define YYUNDEFTOK 2`

7.6.2 Typedef Documentation

7.6.2.1 `typedef short yysigned_char`

7.6.2.2 `typedef union YYSTYPE YYSTYPE`

7.6.3 Enumeration Type Documentation

7.6.3.1 `enum yytokentype`

Enumeration values:

`TOKEN`

`KOMMA`

`TIMES`

`OPENPARENTHESIS`

`CLOSEPARENTHESIS`

`ERROR`

7.6.4 Function Documentation

- 7.6.4.1 void RP_append (int *from*, int *to*)
- 7.6.4.2 void RP_ClearBuffer ()
- 7.6.4.3 int RP_lex (void)
- 7.6.4.4 void RP_Scan (const char * *st*)
- 7.6.4.5 void yydestruct (yytype, yyvaluep) [static]
- 7.6.4.6 int yyerror (char * *s*)
- 7.6.4.7 int yyparse ()

7.6.5 Variable Documentation

- 7.6.5.1 string RP_Error = ""
- 7.6.5.2 vector<string> RP_Strings
- 7.6.5.3 int yychar
- 7.6.5.4 const unsigned char yycheck[] [static]

Initial value:

```
{
    3,   7,   4,   5,   3,   8,   8,   6,   9,  10,
    0,  12,   4
}
```

- 7.6.5.5 const unsigned char yydefact[] [static]

Initial value:

```
{
    0,   2,   7,   0,   5,   3,   0,   8,   1,   0,
    0,   9,   0,   6,   4,   0,  10
}
```

- 7.6.5.6 const ysigned_char yydefgoto[] [static]

Initial value:

```
{
    -1,   3,  10,   4
}
```

7.6.5.7 YYSTYPE yylval

7.6.5.8 int yynerrs

7.6.5.9 const yysigned_char yypact[] [static]

Initial value:

```
{
    -3,  -2,  -7,  10,   8,  -7,   1,  -7,  -7,  -3,
    -3,  -7,  -3,  -7,  -7,  -6,  -7
}
```

7.6.5.10 const yysigned_char yypgoto[] [static]

Initial value:

```
{
    -7,  -1,  -7,  -7
}
```

7.6.5.11 const unsigned char yyr1[] [static]

Initial value:

```
{
    0,   9,  10,  11,  10,  10,  10,  10,  10,  12,
   12
}
```

7.6.5.12 const unsigned char yyr2[] [static]

Initial value:

```
{
    0,   2,   1,   0,   4,   1,   3,   1,   2,   3,
    5
}
```

7.6.5.13 const unsigned char yystos[] [static]

Initial value:

```
{
    0,   3,   8,  10,  12,   4,   5,   8,   0,   4,
   11,   3,   6,  10,  10,  10,   7
}
```


7.7 RPParser.h File Reference

Namespaces

- namespace **Service**

7.8 RPParser.hh File Reference

Compounds

- union **YYSTYPE**

Defines

- #define **TOKEN** 258
- #define **KOMMA** 259
- #define **TIMES** 260
- #define **OPENPARENTHESIS** 261
- #define **CLOSEPARENTHESIS** 262
- #define **ERROR** 263
- #define **yystate** **YYSTYPE**
- #define **YYSTYPE_IS_DECLARED** 1
- #define **YYSTYPE_IS_TRIVIAL** 1

Typedefs

- typedef **YYSTYPE** **YYSTYPE**

Enumerations

- enum **y tokentype** {
TOKEN = 258, **KOMMA** = 259, **TIMES** = 260, **OPENPARENTHESIS** = 261,
CLOSEPARENTHESIS = 262, **ERROR** = 263 }

Variables

- **YYSTYPE** **RP_lval**

7.8.1 Define Documentation

7.8.1.1 `#define CLOSEPARENTHESIS 262`

7.8.1.2 `#define ERROR 263`

7.8.1.3 `#define KOMMA 259`

7.8.1.4 `#define OPENPARENTHESIS 261`

7.8.1.5 `#define TIMES 260`

7.8.1.6 `#define TOKEN 258`

7.8.1.7 `#define yystype YYSTYPE`

7.8.1.8 `#define YYSTYPE_IS_DECLARED 1`

7.8.1.9 `#define YYSTYPE_IS_TRIVIAL 1`

7.8.2 Typedef Documentation

7.8.2.1 `typedef union YYSTYPE YYSTYPE`

7.8.3 Enumeration Type Documentation

7.8.3.1 `enum ytokentype`

Enumeration values:

`TOKEN`

`KOMMA`

`TIMES`

`OPENPARENTHESIS`

`CLOSEPARENTHESIS`

`ERROR`

7.8.4 Variable Documentation

7.8.4.1 `YYSTYPE RP_lval`

7.9 RPParserTest.cpp File Reference

```
#include <functional>
#include "Service.h"
#include "RPParser.h"
```

Functions

- void **test** (string *testname*, string *result*, string *desired*)
- void **testE** (string *testname*)
- void **nTest** (string *s*)
- int **main** (int *argc*, char **argv*[])

7.9.1 Function Documentation

7.9.1.1 int **main** (int *argc*, char * *argv*[])

7.9.1.2 void **nTest** (string *s*)

7.9.1.3 void **test** (string *testname*, string *result*, string *desired*)

7.9.1.4 void **testE** (string *testname*)

7.10 RPS scanner.cc File Reference

```
#include <stdio.h>
#include <iostream>
#include "RPParser.hh"
#include <string>
```

Compounds

- struct `yy_buffer_state`

Defines

- #define `yy_create_buffer` `RP__create_buffer`
- #define `yy_delete_buffer` `RP__delete_buffer`
- #define `yy_scan_buffer` `RP__scan_buffer`
- #define `yy_scan_string` `RP__scan_string`
- #define `yy_scan_bytes` `RP__scan_bytes`
- #define `yy_flex_debug` `RP__flex_debug`
- #define `yy_init_buffer` `RP__init_buffer`
- #define `yy_flush_buffer` `RP__flush_buffer`
- #define `yy_load_buffer_state` `RP__load_buffer_state`
- #define `yy_switch_to_buffer` `RP__switch_to_buffer`
- #define `yyin` `RP_in`
- #define `yylen` `RP_leng`
- #define `yylex` `RP_lex`
- #define `yyout` `RP_out`
- #define `yyrestart` `RP_restart`
- #define `yytext` `RP_text`
- #define `FLEX_SCANNER`
- #define `YY_FLEX_MAJOR_VERSION` 2
- #define `YY_FLEX_MINOR_VERSION` 5
- #define `yyconst`
- #define `YY_PROTO(proto)` `()`
- #define `YY_NULL` 0
- #define `YY_SC_TO_UI(c)` `((unsigned int) (unsigned char) c)`
- #define `BEGIN yy_start` `= 1 + 2 *`
- #define `YY_START` `((yy_start - 1) / 2)`
- #define `YYSTATE` `YY_START`
- #define `YY_STATE_EOF(state)` `(YY_END_OF_BUFFER + state + 1)`
- #define `YY_NEW_FILE` `yyrestart(yyin)`
- #define `YY_END_OF_BUFFER_CHAR` 0
- #define `YY_BUF_SIZE` 16384
- #define `EOB_ACT_CONTINUE_SCAN` 0
- #define `EOB_ACT_END_OF_FILE` 1
- #define `EOB_ACT_LAST_MATCH` 2
- #define `yyless(n)`
- #define `unput(c)` `yyunput(c, yytext_ptr)`

- #define **YY_BUFFER_NEW** 0
- #define **YY_BUFFER_NORMAL** 1
- #define **YY_BUFFER_EOF_PENDING** 2
- #define **YY_CURRENT_BUFFER** yy_current_buffer
- #define **YY_FLUSH_BUFFER** yy_flush_buffer(yy_current_buffer)
- #define **yy_new_buffer** yy_create_buffer
- #define **yy_set_interactive**(is_interactive)
- #define **yy_set_bol**(at_bol)
- #define **YY_AT_BOL**() (yy_current_buffer → yy_at_bol)
- #define **yywrap**() 1
- #define **YY_SKIP_YYWRAP**
- #define **yytext_ptr** yytext
- #define **YY_DO_BEFORE_ACTION**
- #define **YY_NUM_RULES** 9
- #define **YY_END_OF_BUFFER** 10
- #define **REJECT** reject_used_but_not_detected
- #define **yymore**() yymore_used_but_not_detected
- #define **YY_MORE_ADJ** 0
- #define **YY_RESTORE_YY_MORE_OFFSET**
- #define **INITIAL** 0
- #define **YY_NO_PUSH_STATE** 1
- #define **YY_NO_POP_STATE** 1
- #define **YY_NO_TOP_STATE** 1
- #define **YY_READ_BUF_SIZE** 8192
- #define **ECHO** (void) fwrite(yytext, yyleng, 1, yyout)
- #define **YY_INPUT**(buf, result, max_size)
- #define **yyterminate**() return YY_NULL
- #define **YY_START_STACK_INCR** 25
- #define **YY_FATAL_ERROR**(msg) yy_fatal_error(msg)
- #define **YY_DECL** int yylex YY_PROTO((void))
- #define **YY_BREAK** break;
- #define **YY_RULE_SETUP** YY_USER_ACTION
- #define **YY_EXIT_FAILURE** 2
- #define **yyless**(n)

Typedefs

- typedef **yy_buffer_state** * **YY_BUFFER_STATE**
- typedef unsigned int **yy_size_t**
- typedef unsigned char **YY_CHAR**
- typedef int **yy_state_type**

Functions

- void **yyrestart** **YY_PROTO** ((FILE *input_file))
- void **yy_switch_to_buffer** **YY_PROTO** ((**YY_BUFFER_STATE** new_buffer))
- void **yy_load_buffer_state** **YY_PROTO** ((void))
- **YY_BUFFER_STATE** **yy_create_buffer** **YY_PROTO** ((FILE *file, int size))
- void **yy_delete_buffer** **YY_PROTO** ((**YY_BUFFER_STATE** b))

- void yy_init_buffer **YY_PROTO** ((**YY_BUFFER_STATE** b, FILE *file))
- **YY_BUFFER_STATE** yy_scan_buffer **YY_PROTO** ((char *base, **yy_size_t** size))
- **YY_BUFFER_STATE** yy_scan_string **YY_PROTO** ((yyconst char *yy_str))
- **YY_BUFFER_STATE** yy_scan_bytes **YY_PROTO** ((yyconst char *bytes, int len))
- void *yy_flex_alloc **YY_PROTO** ((**yy_size_t**)
- void *yy_flex_realloc **YY_PROTO** ((void *, **yy_size_t**)
- void yy_flex_free **YY_PROTO** ((void *))
- **yy_state_type** yy_try_NUL_trans **YY_PROTO** ((**yy_state_type** current_state))
- void yy_fatal_error **YY_PROTO** ((yyconst char msg[]))
- void yyunput **YY_PROTO** ((int c, char *buf_ptr))
- void yy_init_buffer (b, file) **YY_BUFFER_STATE** b

Variables

- int **yyleng**
- FILE * **yyin** = (FILE *) 0
- FILE * **yyout** = (FILE *) 0
- **YY_BUFFER_STATE** **yy_current_buffer** = 0
- char **yy_hold_char**
- int **yy_n_chars**
- char * **yy_c_buf_p** = (char *) 0
- int **yy_init** = 1
- int **yy_start** = 0
- int **yy_did_buffer_switch_on_eof**
- char * **yytext**
- yyconst short int **yy_accept** [21]
- yyconst int **yy_ec** [256]
- yyconst int **yy_meta** [13]
- yyconst short int **yy_base** [23]
- yyconst short int **yy_def** [23]
- yyconst short int **yy_nxt** [37]
- yyconst short int **yy_chk** [37]
- **yy_state_type** **yy_last_accepting_state**
- char * **yy_last_accepting_cpos**
- **YY_BUFFER_STATE** **BUFFER**
- register char * **yy_bp**
- int **size**
- FILE * **file**
- int **len**

7.10.1 Define Documentation

- 7.10.1.1 `#define BEGIN yy_start = 1 + 2 *`
- 7.10.1.2 `#define ECHO (void) fwrite(yytext, yyleng, 1, yyout)`
- 7.10.1.3 `#define EOB_ACT_CONTINUE_SCAN 0`
- 7.10.1.4 `#define EOB_ACT_END_OF_FILE 1`
- 7.10.1.5 `#define EOB_ACT_LAST_MATCH 2`
- 7.10.1.6 `#define FLEX_SCANNER`
- 7.10.1.7 `#define INITIAL 0`
- 7.10.1.8 `#define REJECT reject_used_but_not_detected`
- 7.10.1.9 `#define unput(c) yyunput(c, yytext_ptr)`
- 7.10.1.10 `#define YY_AT_BOL() (yy_current_buffer → yy_at_bol)`
- 7.10.1.11 `#define YY_BREAK break;`
- 7.10.1.12 `#define YY_BUF_SIZE 16384`
- 7.10.1.13 `#define YY_BUFFER_EOF_PENDING 2`
- 7.10.1.14 `#define YY_BUFFER_NEW 0`
- 7.10.1.15 `#define YY_BUFFER_NORMAL 1`
- 7.10.1.16 `#define yy_create_buffer RP_create_buffer`
- 7.10.1.17 `#define YY_CURRENT_BUFFER yy_current_buffer`
- 7.10.1.18 `#define YY_DECL int yylex YY_PROTO((void))`
- 7.10.1.19 `#define yy_delete_buffer RP_delete_buffer`
- 7.10.1.20 `#define YY_DO_BEFORE_ACTION`

Value:

```
yytext_ptr = yy_bp; \  
  yyleng = (int) (yy_cp - yy_bp); \  
  yy_hold_char = *yy_cp; \  
  *yy_cp = '\0'; \  
  yy_c_buf_p = yy_cp;
```

- 7.10.1.21 `#define YY_END_OF_BUFFER 10`
- 7.10.1.22 `#define YY_END_OF_BUFFER_CHAR 0`
- 7.10.1.23 `#define YY_EXIT_FAILURE 2`
- 7.10.1.24 `#define YY_FATAL_ERROR(msg) yy_fatal_error(msg)`
- 7.10.1.25 `#define yy_flex_debug RP_flex_debug`
- 7.10.1.26 `#define YY_FLEX_MAJOR_VERSION 2`
- 7.10.1.27 `#define YY_FLEX_MINOR_VERSION 5`
- 7.10.1.28 `#define YY_FLUSH_BUFFER yy_flush_buffer(yy_current_buffer)`
- 7.10.1.29 `#define yy_flush_buffer RP_flush_buffer`
- 7.10.1.30 `#define yy_init_buffer RP_init_buffer`
- 7.10.1.31 `#define YY_INPUT(buf, result, max_size)`

Value:

```
if ( yy_current_buffer->yy_is_interactive ) \
    { \
        int c = '*', n; \
        for ( n = 0; n < max_size && \
              (c = getc( yyin )) != EOF && c != '\n'; ++n ) \
            buf[n] = (char) c; \
        if ( c == '\n' ) \
            buf[n++] = (char) c; \
        if ( c == EOF && ferror( yyin ) ) \
            YY_FATAL_ERROR( "input in flex scanner failed" ); \
        result = n; \
    } \
else if ( ((result = fread( buf, 1, max_size, yyin )) == 0) \
          && ferror( yyin ) ) \
    YY_FATAL_ERROR( "input in flex scanner failed" );
```

```

7.10.1.32 #define yy_load_buffer_state RP__load_buffer_state
7.10.1.33 #define YY_MORE_ADJ 0
7.10.1.34 #define yy_new_buffer yy_create_buffer
7.10.1.35 #define YY_NEW_FILE yyrestart( yyin )
7.10.1.36 #define YY_NO_POP_STATE 1
7.10.1.37 #define YY_NO_PUSH_STATE 1
7.10.1.38 #define YY_NO_TOP_STATE 1
7.10.1.39 #define YY_NULL 0
7.10.1.40 #define YY_NUM_RULES 9
7.10.1.41 #define YY_PROTO(proto) ()
7.10.1.42 #define YY_READ_BUF_SIZE 8192
7.10.1.43 #define YY_RESTORE_YY_MORE_OFFSET
7.10.1.44 #define YY_RULE_SETUP YY_USER_ACTION
7.10.1.45 #define YY_SC_TO_UI(c) ((unsigned int) (unsigned char) c)
7.10.1.46 #define yy_scan_buffer RP__scan_buffer
7.10.1.47 #define yy_scan_bytes RP__scan_bytes
7.10.1.48 #define yy_scan_string RP__scan_string
7.10.1.49 #define yy_set_bol(at_bol)

```

Value:

```

{ \
    if ( ! yy_current_buffer ) \
        yy_current_buffer = yy_create_buffer( yyin, YY_BUF_SIZE ); \
    yy_current_buffer->yy_at_bol = at_bol; \
}

```

```

7.10.1.50 #define yy_set_interactive(is_interactive)

```

Value:

```

{ \
    if ( ! yy_current_buffer ) \
        yy_current_buffer = yy_create_buffer( yyin, YY_BUF_SIZE ); \
    yy_current_buffer->yy_is_interactive = is_interactive; \
}

```

- 7.10.1.51 `#define YY_SKIP_YYWRAP`
- 7.10.1.52 `#define YY_START ((yy_start - 1) / 2)`
- 7.10.1.53 `#define YY_START_STACK_INCR 25`
- 7.10.1.54 `#define YY_STATE_EOF(state) (YY_END_OF_BUFFER + state + 1)`
- 7.10.1.55 `#define yy_switch_to_buffer RP__switch_to_buffer`
- 7.10.1.56 `#define yyconst`
- 7.10.1.57 `FILE * yyin RP_in`
- 7.10.1.58 `int yyleng RP_leng`
- 7.10.1.59 `#define yyles(n)`

Value:

```
do \
    { \
        \
        yytext[yyleng] = yy_hold_char; \
        yy_c_buf_p = yytext + n; \
        yy_hold_char = *yy_c_buf_p; \
        *yy_c_buf_p = '\0'; \
        yyleng = n; \
    } \
while ( 0 )
```

- 7.10.1.60 `#define yyles(n)`

Value:

```
do \
    { \
        \
        *yy_cp = yy_hold_char; \
        YY_RESTORE_YY_MORE_OFFSET \
        yy_c_buf_p = yy_cp = yy_bp + n - YY_MORE_ADJ; \
        YY_DO_BEFORE_ACTION; \
    } \
while ( 0 )
```

- 7.10.1.61 `#define yylex RP_lex`
- 7.10.1.62 `#define yymore() yymore_used_but_not_detected`
- 7.10.1.63 `FILE * yyout RP_out`
- 7.10.1.64 `#define yyrestart RP_restart`
- 7.10.1.65 `#define YYSTATE YY_START`
- 7.10.1.66 `#define yyterminate() return YY_NULL`
- 7.10.1.67 `char * yytext RP_text`
- 7.10.1.68 `#define yytext_ptr yytext`
- 7.10.1.69 `#define yywrap() 1`

7.10.2 Typedef Documentation

- 7.10.2.1 `typedef struct yy_buffer_state* YY_BUFFER_STATE`
- 7.10.2.2 `typedef unsigned char YY_CHAR`
- 7.10.2.3 `typedef unsigned int yy_size_t`
- 7.10.2.4 `typedef int yy_state_type`

7.10.3 Function Documentation

- 7.10.3.1 `void yy_init_buffer (b, file)`
- 7.10.3.2 `void yyunput YY_PROTO ((int c, char *buf_ptr)) [static]`
- 7.10.3.3 `void yy_fatal_error YY_PROTO ((yyconst char msg[])) [static]`
- 7.10.3.4 `yy_state_type yy_try_NUL_trans YY_PROTO ((yy_state_type current_state)) [static]`
- 7.10.3.5 `void yy_flex_free YY_PROTO ((void *)) [static]`
- 7.10.3.6 `void* yy_flex_realloc YY_PROTO ((void *, yy_size_t)) [static]`
- 7.10.3.7 `void* yy_flex_alloc YY_PROTO ((yy_size_t)) [static]`
- 7.10.3.8 `YY_BUFFER_STATE yy_scan_bytes YY_PROTO ((yyconst char *bytes, int len))`
- 7.10.3.9 `YY_BUFFER_STATE yy_scan_string YY_PROTO ((yyconst char *yy_str))`
- 7.10.3.10 `YY_BUFFER_STATE yy_scan_buffer YY_PROTO ((char *base, yy_size_t size))`
- 7.10.3.11 `void yy_init_buffer YY_PROTO ((YY_BUFFER_STATE b, FILE *file))`
- 7.10.3.12 `void yy_flush_buffer YY_PROTO ((YY_BUFFER_STATE b))`
- 7.10.3.13 `YY_BUFFER_STATE yy_create_buffer YY_PROTO ((FILE *file, int size))`
- 7.10.3.14 `int input YY_PROTO ((void))`

```
{ 0,
  0, 0, 10, 8, 7, 7, 8, 8, 3, 4,
  1, 8, 5, 7, 0, 0, 5, 6, 2, 0
}
```

7.10.4.6 yyconst short int yy_base[23] [static]

Initial value:

```
{ 0,
  0, 0, 23, 24, 0, 0, 0, 16, 24, 24,
  24, 9, 8, 0, 14, 12, 5, 24, 24, 24,
  11, 12
}
```

7.10.4.7 register char* yy_bp

7.10.4.8 char* yy_c_buf_p = (char *) 0 [static]

7.10.4.9 yyconst short int yy_chk[37] [static]

Initial value:

```
{ 0,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
  1, 1, 21, 21, 22, 22, 17, 16, 15, 13,
  12, 8, 3, 20, 20, 20, 20, 20, 20, 20,
  20, 20, 20, 20, 20, 20
}
```

7.10.4.10 YY_BUFFER_STATE yy_current_buffer = 0 [static]

7.10.4.11 yyconst short int yy_def[23] [static]

Initial value:

```
{ 0,
  20, 1, 20, 20, 21, 21, 22, 20, 20, 20,
  20, 20, 20, 21, 22, 20, 20, 20, 20, 0,
  20, 20
}
```

7.10.4.12 `int yy_did_buffer_switch_on_eof` [static]

7.10.4.13 `yyconst int yy_ec[256]` [static]

7.10.4.14 `char yy_hold_char` [static]

7.10.4.15 `int yy_init = 1` [static]

7.10.4.16 `char* yy_last_accepting_cpos` [static]

7.10.4.17 `yy_state_type yy_last_accepting_state` [static]

7.10.4.18 `yyconst int yy_meta[13]` [static]

Initial value:

```
{ 0,
  1, 2, 2, 3, 1, 1, 1, 1, 1, 1,
  4, 4
}
```

7.10.4.19 `int yy_n_chars` [static]

7.10.4.20 `yyconst short int yy_nxt[37]` [static]

Initial value:

```
{ 0,
  4, 5, 6, 5, 7, 8, 9, 10, 11, 12,
  4, 13, 14, 14, 15, 15, 17, 19, 18, 17,
  17, 16, 20, 3, 20, 20, 20, 20, 20, 20,
  20, 20, 20, 20, 20, 20
}
```

7.10.4.21 `int yy_start = 0` [static]

7.10.4.22 `FILE* yyin = (FILE *) 0`

7.10.4.23 `int yyleng`

7.10.4.24 `FILE * yyout = (FILE *) 0`

7.10.4.25 `char* yytext`

7.11 server.cpp File Reference

```
#include <string>
#include <iostream>
#include <stdio.h>
#include "Socket.h"
#include "CTTNEException.h"
```

Functions

- `int main (int argc, int argv[])`

Variables

- `int PORT = 1952`

7.11.1 Function Documentation

7.11.1.1 `int main (int argc, int argv[])`

7.11.2 Variable Documentation

7.11.2.1 `int PORT = 1952`

7.12 Service.cpp File Reference

```
#include <sstream>
#include <fstream>
#include <cstdarg>
#include <stdio.h>
#include <limits.h>
#include <dirent.h>
#include "Service.h"
#include "CTNException.h"
```

Namespaces

- namespace **Service**

7.13 Service.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string>
#include <sstream>
#include <iostream>
#include <vector>
#include <map>
#include "Constants.h"
```

Namespaces

- namespace **Service**
- namespace **std**

7.14 ServiceTest.cpp File Reference

```
#include <functional>
#include "Service.h"
```

Functions

- void **test** (string *testname*, string *result*, string *desired*)
- void **testE** (string *testname*)
- void **nTest** (string *s*)
- int **main** (int *argc*, char **argv*[])

7.14.1 Function Documentation

7.14.1.1 int **main** (int *argc*, char * *argv*[])

7.14.1.2 void **nTest** (string *s*)

7.14.1.3 void **test** (string *testname*, string *result*, string *desired*)

7.14.1.4 void **testE** (string *testname*)

7.15 Socket.cpp File Reference

```
#include <string.h>
#include <errno.h>
#include <fcntl.h>
#include <iostream>
#include <netdb.h>
#include "Socket.h"
#include "CTNException.h"
```

Namespaces

- namespace **Service**

7.16 Socket.h File Reference

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <unistd.h>
#include <string>
#include <arpa/inet.h>
#include "Service.h"
```

Namespaces

- namespace **Service**

Variables

- const int **MAXHOSTNAME** = 200
- const int **MAXCONNECTIONS** = 5
- const int **MAXRECV** = 500

7.16.1 Variable Documentation

7.16.1.1 const int **MAXCONNECTIONS** = 5

7.16.1.2 const int **MAXHOSTNAME** = 200

7.16.1.3 const int **MAXRECV** = 500

Index

- ~ClientSocket
 - Service::ClientSocket, 18
- ~ServerSocket
 - Service::ServerSocket, 22
- ~Socket
 - Service::Socket, 24
- accept
 - Service::ServerSocket, 22
 - Service::Socket, 24
- BEGIN
 - RPSscanner.cc, 52
- bind
 - Service::Socket, 24
- BUFFER
 - RPSscanner.cc, 56
- client.cpp, 29
 - main, 29
- ClientSocket
 - Service::ClientSocket, 18
- CLOSEPARENTHESIS
 - RPParser.cc, 37, 41
 - RPParser.hh, 47
- Co
 - Constants.h, 32
- connect
 - Service::Socket, 24
- Constants.cpp, 30
- Constants.h, 31
 - Co, 32
 - fFTOP, 31
 - FLT, 32
 - FTOP, 31
 - Fuzzy, 32
 - negINFINITY, 31
 - ONE, 32
 - posINFINITY, 32
 - Rt, 32
 - ZERO, 32
- create
 - Service::Socket, 24
- CTTNEException, 19
 - CTTNEException, 20
 - file, 20
 - message, 20
 - method, 20
 - module, 20
 - operator<<, 20
 - recommendation, 20
 - systemError, 20
- CTTNEException.cpp, 33
 - operator<<, 33
- CTTNEException.h, 34
- ECHO
 - RPSscanner.cc, 52
- EOB_ACT_CONTINUE_SCAN
 - RPSscanner.cc, 52
- EOB_ACT_END_OF_FILE
 - RPSscanner.cc, 52
- EOB_ACT_LAST_MATCH
 - RPSscanner.cc, 52
- eraseLeadingWhitespace
 - Service, 12
- eraseLinefeeds
 - Service, 12
- eraseLineFollowings
 - Service, 12
- eraseMultipleWhitespace
 - Service, 12
- eraseWhitespace
 - Service, 12
- ERROR
 - RPParser.cc, 37, 41
 - RPParser.hh, 47
- expandRepetitionPattern
 - Service, 12
- fFTOP
 - Constants.h, 31
- file
 - CTTNEException, 20
 - RPSscanner.cc, 56
- FLEX_SCANNER
 - RPSscanner.cc, 52
- FLT
 - Constants.h, 32
- FTOP

- Constants.h, 31
- Fuzzy
 - Constants.h, 32
- INITIAL
 - RPScanner.cc, 52
- intToString
 - Service, 12
- is_valid
 - Service::Socket, 24
- isInfinity
 - Service, 12
- isNegInfinity
 - Service, 12
- isPosInfinity
 - Service, 13
- join
 - Service, 13
- KOMMA
 - RPParser.cc, 37, 41
 - RPParser.hh, 47
- len
 - RPScanner.cc, 56
- linebreaks
 - Service, 13
- listen
 - Service::Socket, 24
- m_addr
 - Service::Socket, 25
- m_sock
 - Service::Socket, 25
- main
 - client.cpp, 29
 - RPParserTest.cpp, 48
 - server.cpp, 59
 - ServiceTest.cpp, 62
- MAXCONNECTIONS
 - Socket.h, 64
- MAXHOSTNAME
 - Socket.h, 64
- MAXRECV
 - Socket.h, 64
- message
 - CTTNEException, 20
- method
 - CTTNEException, 20
- module
 - CTTNEException, 20
- moduloPos
 - Service, 13
- nDividable
 - Service, 13
- negINFINITY
 - Constants.h, 31
- nTest
 - RPParserTest.cpp, 48
 - ServiceTest.cpp, 62
- NullFct
 - Service, 13
- ONE
 - Constants.h, 32
- OPENPARENTHESIS
 - RPParser.cc, 37, 41
 - RPParser.hh, 47
- operator<<
 - CTTNEException, 20
 - CTTNEException.cpp, 33
 - Service::ClientSocket, 18
 - Service::ServerSocket, 22
 - std, 16
- operator>>
 - Service::ClientSocket, 18
 - Service::ServerSocket, 22
- parseItem
 - Service, 13
- parseItems
 - Service, 13
- parseItemVector
 - Service, 13
- parseNestedItems
 - Service, 14
- PORT
 - server.cpp, 59
- posINFINITY
 - Constants.h, 32
- readFile
 - Service, 14
- readUrl
 - Service::ClientSocket, 18
- recommendation
 - CTTNEException, 20
- recv
 - Service::Socket, 24
- REJECT
 - RPScanner.cc, 52
- remainder
 - Service, 14
- roundX
 - Service, 14
- roundY
 - Service, 14

- RP_append
 - RPParser.cc, 42
- RP_ClearBuffer
 - RPParser.cc, 42
- RP_Error
 - RPParser.cc, 42
- RP_lex
 - RPParser.cc, 42
- RP_lval
 - RPParser.hh, 47
- RP_Scan
 - RPParser.cc, 42
- RP_Strings
 - RPParser.cc, 42
- RPParser.cc, 35
 - CLOSEPARENTHESIS, 37, 41
 - ERROR, 37, 41
 - KOMMA, 37, 41
 - OPENPARENTHESIS, 37, 41
 - RP_append, 42
 - RP_ClearBuffer, 42
 - RP_Error, 42
 - RP_lex, 42
 - RP_Scan, 42
 - RP_Strings, 42
 - TIMES, 37, 41
 - TOKEN, 37, 41
 - YY_REDUCE_PRINT, 37
 - YY_STACK_PRINT, 37
 - YYABORT, 37
 - YYACCEPT, 37
 - YYBACKUP, 37
 - YYBISON, 38
 - yychar, 38, 42
 - yycheck, 42
 - yyclearin, 38
 - YYCOPY, 38
 - YYDEBUG, 38
 - yydebug, 39
 - yydefact, 42
 - yydefgoto, 42
 - yydestruct, 42
 - YYDPRINTF, 39
 - YYDSYMPRINT, 39
 - YYDSYMPRINTF, 39
 - YYEMPTY, 39
 - YYEOF, 39
 - YYERRCODE, 39
 - yyerok, 39
 - YYERROR, 39
 - yyerror, 39, 42
 - YYERROR_VERBOSE, 39
 - YYFAIL, 39
 - YYFINAL, 39
 - YYINITDEPTH, 39
 - YYLAST, 39
 - YYLEX, 39
 - yylex, 39
 - YYLOC_DEFAULT, 39
 - YYLSP_NEEDED, 39
 - yylval, 40, 42
 - YYMAXDEPTH, 40
 - YYMAXUTOK, 40
 - yyneris, 40, 43
 - YYNNTS, 40
 - YYNRULES, 40
 - YYNSTATES, 40
 - YYNTOKENS, 40
 - ypact, 43
 - YYPACT_NINF, 40
 - yparse, 40, 42
 - ypgoto, 43
 - YYPOPSTACK, 40
 - YYPURE, 40
 - yyr1, 43
 - yyr2, 43
 - YYRECOVERING, 40
 - yysigned_char, 41
 - YYSIZE_T, 40
 - YYSKELETON_NAME, 40
 - YYSTACK_ALLOC, 40
 - YYSTACK_BYTES, 40
 - YYSTACK_FREE, 40
 - YYSTACK_GAP_MAXIMUM, 40
 - YYSTACK_RELOCATE, 40
 - yystos, 43
 - YYSTYPE, 41
 - yystate, 41
 - YYSTYPE_IS_DECLARED, 41
 - YYSTYPE_IS_TRIVIAL, 41
 - yytable, 43
 - YYTABLE_NINF, 41
 - YYTERROR, 41
 - yytokentype, 41
 - YYTRANSLATE, 41
 - yytranslate, 44
 - YYUNDEFTOK, 41
 - yyvaluep, 44
- RPParser.h, 45
- RPParser.hh, 46
 - CLOSEPARENTHESIS, 47
 - ERROR, 47
 - KOMMA, 47
 - OPENPARENTHESIS, 47
 - RP_lval, 47
 - TIMES, 47
 - TOKEN, 47
 - YYSTYPE, 47

- yystype, 47
- YYSTYPE_IS_DECLARED, 47
- YYSTYPE_IS_TRIVIAL, 47
- yytokentype, 47
- RPParserTest.cpp, 48
- RPParserTest.cpp
 - main, 48
 - nTest, 48
 - test, 48
 - testE, 48
- RPScanner.cc, 49
 - BEGIN, 52
 - BUFFER, 56
 - ECHO, 52
 - EOB_ACT_CONTINUE_SCAN, 52
 - EOB_ACT_END_OF_FILE, 52
 - EOB_ACT_LAST_MATCH, 52
 - file, 56
 - FLEX_SCANNER, 52
 - INITIAL, 52
 - len, 56
 - REJECT, 52
 - size, 56
 - unput, 52
 - yy_accept, 56
 - YY_AT_BOL, 52
 - yy_base, 57
 - yy_bp, 57
 - YY_BREAK, 52
 - YY_BUF_SIZE, 52
 - YY_BUFFER_EOF_PENDING, 52
 - YY_BUFFER_NEW, 52
 - YY_BUFFER_NORMAL, 52
 - YY_BUFFER_STATE, 56
 - yy_c_buf_p, 57
 - YY_CHAR, 56
 - yy_chk, 57
 - yy_create_buffer, 52
 - YY_CURRENT_BUFFER, 52
 - yy_current_buffer, 57
 - YY_DECL, 52
 - yy_def, 57
 - yy_delete_buffer, 52
 - yy_did_buffer_switch_on_eof, 57
 - YY_DO_BEFORE_ACTION, 52
 - yy_ec, 58
 - YY_END_OF_BUFFER, 52
 - YY_END_OF_BUFFER_CHAR, 53
 - YY_EXIT_FAILURE, 53
 - YY_FATAL_ERROR, 53
 - yy_flex_debug, 53
 - YY_FLEX_MAJOR_VERSION, 53
 - YY_FLEX_MINOR_VERSION, 53
 - YY_FLUSH_BUFFER, 53
 - yy_flush_buffer, 53
 - yy_hold_char, 58
 - yy_init, 58
 - yy_init_buffer, 53, 56
 - YY_INPUT, 53
 - yy_last_accepting_cpos, 58
 - yy_last_accepting_state, 58
 - yy_load_buffer_state, 53
 - yy_meta, 58
 - YY_MORE_ADJ, 54
 - yy_n_chars, 58
 - yy_new_buffer, 54
 - YY_NEW_FILE, 54
 - YY_NO_POP_STATE, 54
 - YY_NO_PUSH_STATE, 54
 - YY_NO_TOP_STATE, 54
 - YY_NULL, 54
 - YY_NUM_RULES, 54
 - yy_nxt, 58
 - YY_PROTO, 54, 56
 - YY_READ_BUF_SIZE, 54
 - YY_RESTORE_YY_MORE_OFFSET, 54
 - YY_RULE_SETUP, 54
 - YY_SC_TO_UI, 54
 - yy_scan_buffer, 54
 - yy_scan_bytes, 54
 - yy_scan_string, 54
 - yy_set_bol, 54
 - yy_set_interactive, 54
 - yy_size_t, 56
 - YY_SKIP_YYWRAP, 54
 - YY_START, 55
 - yy_start, 58
 - YY_START_STACK_INCR, 55
 - YY_STATE_EOF, 55
 - yy_state_type, 56
 - yy_switch_to_buffer, 55
 - yyconst, 55
 - yyin, 55, 58
 - yylen, 55, 58
 - yyless, 55
 - yylex, 55
 - yymore, 56
 - yyout, 56, 58
 - yyrestart, 56
 - YYSTATE, 56
 - yyterminate, 56
 - yytext, 56, 58
 - yytext_ptr, 56
 - yywrap, 56
- Rt
 - Constants.h, 32

- send
 - Service::Socket, 25
- server.cpp, 59
 - main, 59
 - PORT, 59
- ServerSocket
 - Service::ServerSocket, 21
- Service, 9
 - eraseLeadingWhitespace, 12
 - eraseLinefeeds, 12
 - eraseLineFollowings, 12
 - eraseMultipleWhitespace, 12
 - eraseWhitespace, 12
 - expandRepetitionPattern, 12
 - intToString, 12
 - isInfinity, 12
 - isNegInfinity, 12
 - isPosInfinity, 13
 - join, 13
 - linebreaks, 13
 - moduloPos, 13
 - nDividable, 13
 - NullFct, 13
 - parseItem, 13
 - parseItems, 13
 - parseItemVector, 13
 - parseNestedItems, 14
 - readFile, 14
 - remainder, 14
 - roundX, 14
 - roundY, 14
 - split, 14
 - stripComment, 14
 - stripComments, 15
 - toFloat, 15
 - toFLT, 15
 - toInt, 15
 - toRt, 15
 - vectorDifference, 15
 - withLineNumbers, 15
 - xToString, 15
- Service.cpp, 60
- Service.h, 61
- Service::ClientSocket, 17
- Service::ClientSocket
 - ~ClientSocket, 18
 - ClientSocket, 18
 - operator<<, 18
 - operator>>, 18
 - readUrl, 18
- Service::ServerSocket, 21
- Service::ServerSocket
 - ~ServerSocket, 22
 - accept, 22
 - operator<<, 22
 - operator>>, 22
 - ServerSocket, 21
- Service::Socket, 23
 - ~Socket, 24
 - accept, 24
 - bind, 24
 - connect, 24
 - create, 24
 - is_valid, 24
 - listen, 24
 - m_addr, 25
 - m_sock, 25
 - recv, 24
 - send, 25
 - set_non_blocking, 25
 - Socket, 24
- ServiceTest.cpp, 62
- ServiceTest.cpp
 - main, 62
 - nTest, 62
 - test, 62
 - testE, 62
- set_non_blocking
 - Service::Socket, 25
- size
 - RPSscanner.cc, 56
- Socket
 - Service::Socket, 24
- Socket.cpp, 63
- Socket.h, 64
 - MAXCONNECTIONS, 64
 - MAXHOSTNAME, 64
 - MAXRECV, 64
- split
 - Service, 14
- start
 - YYSTYPE, 28
- std, 16
 - operator<<, 16
- stripComment
 - Service, 14
- stripComments
 - Service, 15
- systemError
 - CTTNEException, 20
- test
 - RPParserTest.cpp, 48
 - ServiceTest.cpp, 62
- testE
 - RPParserTest.cpp, 48
 - ServiceTest.cpp, 62
- TIMES

- RPParser.cc, 37, 41
- RPParser.hh, 47
- toFloat
 - Service, 15
- toFLT
 - Service, 15
- toInt
 - Service, 15
- TOKEN
 - RPParser.cc, 37, 41
 - RPParser.hh, 47
- toRt
 - Service, 15
- unput
 - RPScanner.cc, 52
- value
 - YYSTYPE, 28
- vectorDifference
 - Service, 15
- withLineNumbers
 - Service, 15
- xToString
 - Service, 15
- yy_accept
 - RPScanner.cc, 56
- YY_AT_BOL
 - RPScanner.cc, 52
- yy_at_bol
 - yy_buffer_state, 26
- yy_base
 - RPScanner.cc, 57
- yy_bp
 - RPScanner.cc, 57
- YY_BREAK
 - RPScanner.cc, 52
- yy_buf_pos
 - yy_buffer_state, 26
- YY_BUF_SIZE
 - RPScanner.cc, 52
- yy_buf_size
 - yy_buffer_state, 26
- YY_BUFFER_EOF_PENDING
 - RPScanner.cc, 52
- YY_BUFFER_NEW
 - RPScanner.cc, 52
- YY_BUFFER_NORMAL
 - RPScanner.cc, 52
- YY_BUFFER_STATE
 - RPScanner.cc, 56
- yy_buffer_state, 26
 - yy_at_bol, 26
 - yy_buf_pos, 26
 - yy_buf_size, 26
 - yy_buffer_status, 26
 - yy_ch_buf, 26
 - yy_fill_buffer, 26
 - yy_input_file, 26
 - yy_is_interactive, 26
 - yy_is_our_buffer, 26
 - yy_n_chars, 26
- yy_buffer_status
 - yy_buffer_state, 26
- yy_c_buf_p
 - RPScanner.cc, 57
- yy_ch_buf
 - yy_buffer_state, 26
- YY_CHAR
 - RPScanner.cc, 56
- yy_chk
 - RPScanner.cc, 57
- yy_create_buffer
 - RPScanner.cc, 52
- YY_CURRENT_BUFFER
 - RPScanner.cc, 52
- yy_current_buffer
 - RPScanner.cc, 57
- YY_DECL
 - RPScanner.cc, 52
- yy_def
 - RPScanner.cc, 57
- yy_delete_buffer
 - RPScanner.cc, 52
- yy_did_buffer_switch_on_eof
 - RPScanner.cc, 57
- YY_DO_BEFORE_ACTION
 - RPScanner.cc, 52
- yy_ec
 - RPScanner.cc, 58
- YY_END_OF_BUFFER
 - RPScanner.cc, 52
- YY_END_OF_BUFFER_CHAR
 - RPScanner.cc, 53
- YY_EXIT_FAILURE
 - RPScanner.cc, 53
- YY_FATAL_ERROR
 - RPScanner.cc, 53
- yy_fill_buffer
 - yy_buffer_state, 26
- yy_flex_debug
 - RPScanner.cc, 53
- YY_FLEX_MAJOR_VERSION
 - RPScanner.cc, 53
- YY_FLEX_MINOR_VERSION
 - RPScanner.cc, 53

- YY_FLUSH_BUFFER
 - RPSscanner.cc, 53
- yy_flush_buffer
 - RPSscanner.cc, 53
- yy_hold_char
 - RPSscanner.cc, 58
- yy_init
 - RPSscanner.cc, 58
- yy_init_buffer
 - RPSscanner.cc, 53, 56
- YY_INPUT
 - RPSscanner.cc, 53
- yy_input_file
 - yy_buffer_state, 26
- yy_is_interactive
 - yy_buffer_state, 26
- yy_is_our_buffer
 - yy_buffer_state, 26
- yy_last_accepting_cpos
 - RPSscanner.cc, 58
- yy_last_accepting_state
 - RPSscanner.cc, 58
- yy_load_buffer_state
 - RPSscanner.cc, 53
- yy_meta
 - RPSscanner.cc, 58
- YY_MORE_ADJ
 - RPSscanner.cc, 54
- yy_n_chars
 - RPSscanner.cc, 58
 - yy_buffer_state, 26
- yy_new_buffer
 - RPSscanner.cc, 54
- YY_NEW_FILE
 - RPSscanner.cc, 54
- YY_NO_POP_STATE
 - RPSscanner.cc, 54
- YY_NO_PUSH_STATE
 - RPSscanner.cc, 54
- YY_NO_TOP_STATE
 - RPSscanner.cc, 54
- YY_NULL
 - RPSscanner.cc, 54
- YY_NUM_RULES
 - RPSscanner.cc, 54
- yy_nxt
 - RPSscanner.cc, 58
- YY_PROTO
 - RPSscanner.cc, 54, 56
- YY_READ_BUF_SIZE
 - RPSscanner.cc, 54
- YY_REDUCE_PRINT
 - RPParser.cc, 37
- YY_RESTORE_YY_MORE_OFFSET
 - RPSscanner.cc, 54
- YY_RULE_SETUP
 - RPSscanner.cc, 54
- YY_SC_TO_UI
 - RPSscanner.cc, 54
- yy_scan_buffer
 - RPSscanner.cc, 54
- yy_scan_bytes
 - RPSscanner.cc, 54
- yy_scan_string
 - RPSscanner.cc, 54
- yy_set_bol
 - RPSscanner.cc, 54
- yy_set_interactive
 - RPSscanner.cc, 54
- yy_size_t
 - RPSscanner.cc, 56
- YY_SKIP_YYWRAP
 - RPSscanner.cc, 54
- YY_STACK_PRINT
 - RPParser.cc, 37
- YY_START
 - RPSscanner.cc, 55
- yy_start
 - RPSscanner.cc, 58
- YY_START_STACK_INCR
 - RPSscanner.cc, 55
- YY_STATE_EOF
 - RPSscanner.cc, 55
- yy_state_type
 - RPSscanner.cc, 56
- yy_switch_to_buffer
 - RPSscanner.cc, 55
- YYABORT
 - RPParser.cc, 37
- YYACCEPT
 - RPParser.cc, 37
- yyalloc, 27
 - yyss, 27
 - yyvs, 27
- YYBACKUP
 - RPParser.cc, 37
- YYBISON
 - RPParser.cc, 38
- yychar
 - RPParser.cc, 38, 42
- yycheck
 - RPParser.cc, 42
- yyclearin
 - RPParser.cc, 38
- yyconst
 - RPSscanner.cc, 55
- YYCOPY
 - RPParser.cc, 38

- YYDEBUG
 - RPParser.cc, 38
- yydebug
 - RPParser.cc, 39
- yydefact
 - RPParser.cc, 42
- yydefgoto
 - RPParser.cc, 42
- yydestruct
 - RPParser.cc, 42
- YYDPRINTF
 - RPParser.cc, 39
- YYDSYMPRINT
 - RPParser.cc, 39
- YYDSYMPRINTF
 - RPParser.cc, 39
- YYEMPTY
 - RPParser.cc, 39
- YYEOF
 - RPParser.cc, 39
- YYERRCODE
 - RPParser.cc, 39
- yyerrok
 - RPParser.cc, 39
- YYERROR
 - RPParser.cc, 39
- yyerror
 - RPParser.cc, 39, 42
- YYERROR_VERBOSE
 - RPParser.cc, 39
- YYFAIL
 - RPParser.cc, 39
- YYFINAL
 - RPParser.cc, 39
- yyin
 - RPScanner.cc, 55, 58
- YYINITDEPTH
 - RPParser.cc, 39
- YYLAST
 - RPParser.cc, 39
- yyleng
 - RPScanner.cc, 55, 58
- yyless
 - RPScanner.cc, 55
- YYLEX
 - RPParser.cc, 39
- yylex
 - RPParser.cc, 39
 - RPScanner.cc, 55
- YYLLOC_DEFAULT
 - RPParser.cc, 39
- YYLSP_NEEDED
 - RPParser.cc, 39
- yylval
 - RPParser.cc, 40, 42
- YYMAXDEPTH
 - RPParser.cc, 40
- YYMAXUTOK
 - RPParser.cc, 40
- yymore
 - RPScanner.cc, 56
- yyneres
 - RPParser.cc, 40, 43
- YYNNTS
 - RPParser.cc, 40
- YYNRULES
 - RPParser.cc, 40
- YYNSTATES
 - RPParser.cc, 40
- YYNTOKENS
 - RPParser.cc, 40
- yyout
 - RPScanner.cc, 56, 58
- yyract
 - RPParser.cc, 43
- YYRACT_NINF
 - RPParser.cc, 40
- yyrarse
 - RPParser.cc, 40, 42
- yyrgoto
 - RPParser.cc, 43
- YYPOPSTACK
 - RPParser.cc, 40
- YYPURE
 - RPParser.cc, 40
- yyr1
 - RPParser.cc, 43
- yyr2
 - RPParser.cc, 43
- YYRECOVERING
 - RPParser.cc, 40
- yyrestart
 - RPScanner.cc, 56
- yysigned_char
 - RPParser.cc, 41
- YYSIZE_T
 - RPParser.cc, 40
- YYSKELETON_NAME
 - RPParser.cc, 40
- yyss
 - yyalloc, 27
- YYSTACK_ALLOC
 - RPParser.cc, 40
- YYSTACK_BYTES
 - RPParser.cc, 40
- YYSTACK_FREE
 - RPParser.cc, 40
- YYSTACK_GAP_MAXIMUM

- RPParser.cc, 40
- YYSTACK_RELOCATE
 - RPParser.cc, 40
- YYSTATE
 - RPScanner.cc, 56
- yystos
 - RPParser.cc, 43
- YYSTYPE, 28
 - RPParser.cc, 41
 - RPParser.hh, 47
 - start, 28
 - value, 28
- yystate
 - RPParser.cc, 41
 - RPParser.hh, 47
- YYSTYPE_IS_DECLARED
 - RPParser.cc, 41
 - RPParser.hh, 47
- YYSTYPE_IS_TRIVIAL
 - RPParser.cc, 41
 - RPParser.hh, 47
- yytable
 - RPParser.cc, 43
- YYTABLE_NINF
 - RPParser.cc, 41
- yyterminate
 - RPScanner.cc, 56
- YYTERROR
 - RPParser.cc, 41
- yytext
 - RPScanner.cc, 56, 58
- yytext_ptr
 - RPScanner.cc, 56
- yytokentype
 - RPParser.cc, 41
 - RPParser.hh, 47
- YYTRANSLATE
 - RPParser.cc, 41
- yytranslate
 - RPParser.cc, 44
- YYUNDEFTOK
 - RPParser.cc, 41
- yyvaluep
 - RPParser.cc, 44
- yyvs
 - yyalloc, 27
- yywrap
 - RPScanner.cc, 56
- ZERO
 - Constants.h, 32