

ddt - RCI

Generated by Doxygen 1.8.9.1

Thu Mar 5 2015 22:55:25

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	args_parse_options Struct Reference	5
3.1.1	Detailed Description	5
3.1.2	Field Documentation	5
3.1.2.1	argc	5
3.1.2.2	argv	5
3.1.2.3	startup_data	5
3.2	startup_data Struct Reference	5
3.2.1	Field Documentation	6
3.2.1.1	dest_size	6
3.2.1.2	destination	6
3.2.1.3	family	6
3.2.1.4	ringport	6
3.3	transversal_data Struct Reference	6
3.3.1	Field Documentation	6
3.3.1.1	ring	6
3.3.1.2	startup_data	6
3.3.1.3	t	6
3.3.1.4	u	6
4	File Documentation	9
4.1	include/common.h File Reference	9
4.2	include/net_udp.h File Reference	9
4.2.1	Function Documentation	9
4.2.1.1	getsockaddr	9
4.3	include/options.h File Reference	10

4.3.1	Function Documentation	10
4.3.1.1	parse_options	10
4.4	src/main.c File Reference	10
4.4.1	Detailed Description	10
4.4.2	Function Documentation	10
4.4.2.1	main	10
4.5	src/net_udp.c File Reference	11
4.5.1	Detailed Description	11
4.5.2	Macro Definition Documentation	11
4.5.2.1	_XOPEN_SOURCE	11
4.5.3	Function Documentation	11
4.5.3.1	getsockaddr	11
4.6	src/options.c File Reference	11
4.6.1	Detailed Description	12
4.6.2	Macro Definition Documentation	12
4.6.2.1	_XOPEN_SOURCE	12
4.6.3	Function Documentation	12
4.6.3.1	parse_options	12
4.6.3.2	usage	12
Index		13

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

args_parse_options	Estrutura com os argumentos para a função parse_options	5
startup_data	Estrutura com os dados a ser recolhidos pela linha de comandos	5
transversal_data	Estrutura com as referências necessárias aos vários módulos do projecto	6

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

include/ common.h	Cabeçalho com declarações das estruturas de dados transversais aos vários módulos do projecto	9
include/ net_udp.h	Cabeçalhos e protótipos associados às funções relacionadas com o protocolo UDP	9
include/ options.h	Cabeçalhos e protótipos associados ao módulo de processamento das opções de linha de comandos	10
src/ main.c	Ficheiro principal do projecto	10
src/ net_udp.c	Ficheiro responsável pela interpretação dos argumentos passados via linha de comandos . .	11
src/ options.c	Ficheiro responsável pela interpretação dos argumentos passados via linha de comandos . .	11

Chapter 3

Data Structure Documentation

3.1 args_parse_options Struct Reference

Estrutura com os argumentos para a função [parse_options](#).

```
#include <options.h>
```

Data Fields

- `int * argc`
Ponteiro para o local em memória do contador de argumentos.
- `char *** argv`
Ponteiro para o local em memória dos argumentos.
- `struct startup_data * startup_data`
Ponteiro para uma das estruturas transversais ao projecto.

3.1.1 Detailed Description

Esta estrutura,

3.1.2 Field Documentation

3.1.2.1 `int* args_parse_options::argc`

3.1.2.2 `char*** args_parse_options::argv`

3.1.2.3 `struct startup_data* args_parse_options::startup_data`

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

- `include/options.h`

3.2 startup_data Struct Reference

Estrutura com os dados a ser recolhidos pela linha de comandos.

```
#include <common.h>
```

Data Fields

- int [ringport](#)
Porto do servidor TCP.
- int [family](#)
Família da socket.
- struct sockaddr * [destination](#)
Estrutura de destino para as mensagens UDP.
- socklen_t [dest_size](#)
Tamanho da estrutura de destino para as mensagens UDP.

3.2.1 Field Documentation

3.2.1.1 socklen_t startup_data::dest_size

3.2.1.2 struct sockaddr* startup_data::destination

3.2.1.3 int startup_data::family

3.2.1.4 int startup_data::ringport

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

- include/[common.h](#)

3.3 transversal_data Struct Reference

Estrutura com as referências necessárias aos vários módulos do projecto.

```
#include <common.h>
```

Data Fields

- int [u](#)
Socket UDP.
- int [t](#)
Socket TCP.
- int [ring](#)
Anel a ser utilizado.
- struct [startup_data](#) startup_data
Parâmetros da linha de comandos.

3.3.1 Field Documentation

3.3.1.1 int transversal_data::ring

3.3.1.2 struct startup_data transversal_data::startup_data

3.3.1.3 int transversal_data::t

3.3.1.4 int transversal_data::u

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

- [include/common.h](#)

Chapter 4

File Documentation

4.1 include/common.h File Reference

Cabeçalho com declarações das estruturas de dados transversais aos vários módulos do projecto.

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip.h>
```

Data Structures

- struct [startup_data](#)
Estrutura com os dados a ser recolhidos pela linha de comandos.
- struct [transversal_data](#)
Estrutura com as referências necessárias aos vários módulos do projecto.

4.2 include/net_udp.h File Reference

Cabeçalhos e protótipos associados às funções relacionadas com o protocolo UDP.

Functions

- void [getsockaddr](#) (char *node, char *service, struct [startup_data](#) *startup_data)
Função de leitura do endereço e do porto e criação da estrutura de destino.

4.2.1 Function Documentation

4.2.1.1 void getsockaddr (char * node, char * service, struct startup_data * startup_data)

Esta função recebe os parâmetros conseguidos via linha de comandos e determina qual a estrutura de destino (relativa ao servidor UDP).

A função `getaddrinfo()` é utilizada em vez da função `gethostbyname()`, por uma questão de preferência pessoal e também pelo facto da função `getaddrinfo()` estar especificada como thread-safe (a função `gethostbyname()` só é thread-safe em algumas implementações). Os detalhes da função `getaddrinfo()` podem ser consultados em [RFC 2553](#).

Na estrutura comum é alocada memória para a struct `sockaddr` de destino.

4.3 include/options.h File Reference

Cabeçalhos e protótipos associados ao módulo de processamento das opções de linha de comandos.

```
#include "common.h"
```

Data Structures

- struct [args_parse_options](#)
Estrutura com os argumentos para a função [parse_options](#).

Functions

- void * [parse_options](#) (struct [args_parse_options](#) *params)
Função responsável pela interpretação dos argumentos passados via linha de comandos.

4.3.1 Function Documentation

4.3.1.1 void* parse_options (struct args_parse_options * params)

Esta função recorre à função getopt() de maneira a obter os diversos parâmetros passados via linha de comandos. Os argumentos são passados via ponteiro para estrutura de maneira a que a função seja compatível com threads.

4.4 src/main.c File Reference

Ficheiro principal do projecto.

```
#include <stdlib.h>
#include <stdio.h>
#include "common.h"
#include "options.h"
```

Functions

- int [main](#) (int argc, char **argv)
Função principal do programa ddt.

4.4.1 Detailed Description

Este ficheiro serve como chave de abóbada para os vários módulos desenvolvidos e responsáveis pelas diversas funcionalidades do projecto.

4.4.2 Function Documentation

4.4.2.1 int main (int argc, char ** argv)

Esta função serve-se dos módulos criados nos restantes ficheiros-fonte para implementar as funcionalidades necessárias ao projecto, como especificadas no enunciado.

4.5 src/net_udp.c File Reference

Ficheiro responsável pela interpretação dos argumentos passados via linha de comandos.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <string.h>
#include "common.h"
#include "net_udp.h"
```

Macros

- `#define _XOPEN_SOURCE`

Functions

- void [getsockaddr](#) (char *node, char *service, struct [startup_data](#) *startup_data)

Função de leitura do endereço e do porto e criação da estrutura de destino.

4.5.1 Detailed Description

As funções implementadas neste módulo correspondem à interpretação dos argumentos passados via linha de comandos, e correspondente transferência para a estrutura de dados transversal aos vários módulos do projecto.

4.5.2 Macro Definition Documentation

4.5.2.1 `#define _XOPEN_SOURCE`

4.5.3 Function Documentation

4.5.3.1 void [getsockaddr](#) (char * node, char * service, struct [startup_data](#) * startup_data)

Esta função recebe os parâmetros conseguidos via linha de comandos e determina qual a estrutura de destino (relativa ao servidor UDP).

A função [getaddrinfo](#)() é utilizada em vez da função [gethostbyname](#)(), por uma questão de preferência pessoal e também pelo facto da função [getaddrinfo](#)() estar especificada como thread-safe (a função [gethostbyname](#)() só é thread-safe em algumas implementações). Os detalhes da função [getaddrinfo](#)() podem ser consultados em [RFC 2553](#).

Na estrutura comum é alocada memória para a struct [sockaddr](#) de destino.

4.6 src/options.c File Reference

Ficheiro responsável pela interpretação dos argumentos passados via linha de comandos.

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "common.h"
#include "options.h"
#include "net_udp.h"
```

Macros

- `#define _XOPEN_SOURCE`

Functions

- void `usage` (char *appname)
Função que mostra a mensagem Usage.
- void * `parse_options` (struct `args_parse_options` *params)
Função responsável pela interpretação dos argumentos passados via linha de comandos.

4.6.1 Detailed Description

As funções implementadas neste módulo correspondem à interpretação dos argumentos passados via linha de comandos, e correspondente transferência para a estrutura de dados transversal aos vários módulos do projecto.

4.6.2 Macro Definition Documentation

4.6.2.1 `#define _XOPEN_SOURCE`

4.6.3 Function Documentation

4.6.3.1 void* `parse_options` (struct `args_parse_options` * *params*)

Esta função recorre à função `getopt()` de maneira a obter os diversos parâmetros passados via linha de comandos. Os argumentos são passados via ponteiro para estrutura de maneira a que a função seja compatível com threads.

4.6.3.2 void `usage` (char * *appname*)

Index

- `_XOPEN_SOURCE`
 - `net_udp.c`, [11](#)
 - `options.c`, [12](#)
- `argc`
 - `args_parse_options`, [5](#)
- `args_parse_options`, [5](#)
 - `argc`, [5](#)
 - `argv`, [5](#)
 - `startup_data`, [5](#)
- `argv`
 - `args_parse_options`, [5](#)
- `dest_size`
 - `startup_data`, [6](#)
- `destination`
 - `startup_data`, [6](#)
- `family`
 - `startup_data`, [6](#)
- `getsockaddr`
 - `net_udp.c`, [11](#)
 - `net_udp.h`, [9](#)
- `include/common.h`, [9](#)
- `include/net_udp.h`, [9](#)
- `include/options.h`, [10](#)
- `main`
 - `main.c`, [10](#)
- `main.c`
 - `main`, [10](#)
- `net_udp.c`
 - `_XOPEN_SOURCE`, [11](#)
 - `getsockaddr`, [11](#)
- `net_udp.h`
 - `getsockaddr`, [9](#)
- `options.c`
 - `_XOPEN_SOURCE`, [12](#)
 - `parse_options`, [12](#)
 - `usage`, [12](#)
- `options.h`
 - `parse_options`, [10](#)
- `parse_options`
 - `options.c`, [12](#)
 - `options.h`, [10](#)
- `ring`
 - `transversal_data`, [6](#)
- `ringport`
 - `startup_data`, [6](#)
- `src/main.c`, [10](#)
- `src/net_udp.c`, [11](#)
- `src/options.c`, [11](#)
- `startup_data`, [5](#)
 - `args_parse_options`, [5](#)
 - `dest_size`, [6](#)
 - `destination`, [6](#)
 - `family`, [6](#)
 - `ringport`, [6](#)
 - `transversal_data`, [6](#)
- `t`
 - `transversal_data`, [6](#)
- `transversal_data`, [6](#)
 - `ring`, [6](#)
 - `startup_data`, [6](#)
 - `t`, [6](#)
 - `u`, [6](#)
- `u`
 - `transversal_data`, [6](#)
- `usage`
 - `options.c`, [12](#)