

## My Project

Generated by Doxygen 1.13.2



---

<b>1 Class Index</b>	<b>1</b>
1.1 Class List . . . . .	1
<b>2 File Index</b>	<b>3</b>
2.1 File List . . . . .	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 RESPProtocol Class Reference . . . . .	5
3.2 StorageEngine Class Reference . . . . .	5
3.3 TCPServer Class Reference . . . . .	5
<b>4 File Documentation</b>	<b>7</b>
4.1 resp_protocol.h . . . . .	7
4.2 storage_engine.h . . . . .	7
4.3 tcp_server.h . . . . .	7
<b>Index</b>	<b>9</b>



# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">RESPProtocol</a>	.....	5
<a href="#">StorageEngine</a>	.....	5
<a href="#">TCPServer</a>	.....	5



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

src/ <a href="#">resp_protocol.h</a> . . . . .	7
src/ <a href="#">storage_engine.h</a> . . . . .	7
src/ <a href="#">tcp_server.h</a> . . . . .	7





## Chapter 3

# Class Documentation

### 3.1 RESPProtocol Class Reference

#### Static Public Member Functions

- static std::string **encodeSimpleString** (const std::string &str)
- static std::string **encodeBulkString** (const std::string &str)
- static std::string **encodeError** (const std::string &msg)
- static std::string **encodeInteger** (int num)
- static std::vector< std::string > **decodeArray** (const std::string &resp)

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

- src/resp\_protocol.h
- src/resp\_protocol.cpp

### 3.2 StorageEngine Class Reference

#### Public Member Functions

- **StorageEngine** (size\_t cap=1000)
- void **set** (const std::string &key, const std::string &value)
- std::string **get** (const std::string &key)
- void **del** (const std::string &key)
- void **flush** ()

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

- src/storage\_engine.h
- src/storage\_engine.cpp

### 3.3 TCPServer Class Reference

#### Public Member Functions

- **TCPServer** (int port, size\_t storage\_capacity=1000)
- void **run** ()

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

- src/tcp\_server.h
- src/tcp\_server.cpp



# Chapter 4

## File Documentation

### 4.1 resp\_protocol.h

```
00001 #ifndef RESP_PROTOCOL_H
00002 #define RESP_PROTOCOL_H
00003
00004 #include <string>
00005 #include <vector>
00006
00007 class RESPProtocol {
00008 public:
00009     static std::string encodeSimpleString(const std::string& str);
00010     static std::string encodeBulkString(const std::string& str);
00011     static std::string encodeError(const std::string& msg);
00012     static std::string encodeInteger(int num);
00013     static std::vector<std::string> decodeArray(const std::string& resp);
00014 };
00015
00016 #endif
```

### 4.2 storage\_engine.h

```
00001 #ifndef STORAGE_ENGINE_H
00002 #define STORAGE_ENGINE_H
00003
00004 #include <unordered_map>
00005 #include <list>
00006 #include <string>
00007
00008 class StorageEngine {
00009 private:
00010     std::unordered_map<std::string, std::string> kv_store;
00011     std::list<std::string> lru_list;
00012     std::unordered_map<std::string, std::list<std::string>::iterator> lru_map;
00013     size_t capacity;
00014
00015     void evictIfNeeded();
00016
00017 public:
00018     StorageEngine(size_t cap = 1000);
00019     void set(const std::string& key, const std::string& value);
00020     std::string get(const std::string& key);
00021     void del(const std::string& key);
00022     void flush();
00023 };
00024
00025 #endif
```

### 4.3 tcp\_server.h

```
00001 #ifndef TCP_SERVER_H
```

```
00002 #define TCP_SERVER_H
00003
00004 #include <netinet/in.h>
00005 #include <string>
00006 #include "storage_engine.h"
00007
00008 #ifdef __linux__
00009     #include <sys/epoll.h>
00010 #elif __APPLE__
00011     #include <sys/event.h>
00012 #endif
00013
00014 class TCPServer {
00015 private:
00016     int server_fd;
00017     struct sockaddr_in address;
00018     StorageEngine storage;
00019
00020 #ifdef __linux__
00021     int epoll_fd;
00022 #elif __APPLE__
00023     int kqueue_fd;
00024 #endif
00025
00026     void add_to_event_loop(int fd);
00027     void accept_new_connection();
00028     void handle_client(int client_fd);
00029     std::string process_command(const std::string &request);
00030
00031 public:
00032     explicit TCPServer(int port, size_t storage_capacity = 1000);
00033     ~TCPServer();
00034     void run();
00035 };
00036
00037 #endif
```

# Index

RESPProtocol, [5](#)

src/resp\_protocol.h, [7](#)

src/storage\_engine.h, [7](#)

src/tcp\_server.h, [7](#)

StorageEngine, [5](#)

TCPServer, [5](#)