My Project

Generated by Doxygen 1.8.13

Contents

1 Main Page **1**

2 Class Index **3**

2.1 Class List . . . 3

3 File Index **5**

3.1 File List . . . 5

4 Class Documentation **7**

4.1 ConfigSettings Class Reference . . . 7

4.1.1 Detailed Description . . . 8

4.1.2 Constructor & Destructor Documentation . . . 8

4.1.2.1 ConfigSettings() . . . 8

4.1.3 Member Function Documentation . . . 8

4.1.3.1 checkFilePath() . . . 8

4.1.3.2 getData() . . . 8

4.1.3.3 loadSettings() . . . 9

4.2 MetaSettings::data Struct Reference . . . 9

4.3 Errors Struct Reference . . . 10

4.3.1 Detailed Description . . . 10

4.4 MetaSettings Class Reference . . . 10

4.4.1 Detailed Description . . . 11

4.4.2 Member Function Documentation . . . 11

4.4.2.1 checkFilePath() . . . 11

4.4.2.2 isEmpty() . . . 12

4.4.2.3 loadData() . . . 12
4.4.2.4 loadVec() . . . 12
4.5 PCB Class Reference . . . 12
4.5.1 Detailed Description . . . 13
4.5.2 Constructor & Destructor Documentation . . . 13
4.5.2.1 PCB() . . . 14
4.5.3 Member Function Documentation . . . 14
4.5.3.1 checkTimer() . . . 14
4.5.3.2 getStartLogString() . . . 14
4.5.3.3 loadProcess() . . . 15
4.5.3.4 logProcess() . . . 15
4.5.3.5 RRcount() . . . 15
4.5.3.6 runProcess() . . . 16
4.6 Timer Class Reference . . . 16
4.6.1 Constructor & Destructor Documentation . . . 16
4.6.1.1 Timer() . . . 16
4.6.2 Member Function Documentation . . . 17
4.6.2.1 getElapsedTime() . . . 17
4.6.2.2 start() . . . 17
4.6.2.3 stop() . . . 18

5 File Documentation **19**

5.1 Timer.cpp File Reference . . . 19
5.1.1 Detailed Description . . . 19

Index **21**

Chapter 1

Main Page

This file is the implementation of the Timer class

Chapter 2

Class Index

2.1 Class List

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

ConfigSettings
: Class for Configuration Data 7

MetaSettings::data 9

Errors
: Ouput for different possible errors 10

MetaSettings
: Class for meta data 10

PCB
: Process Control block 12

Timer 16

Chapter 3

File Index

3.1 File List

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

OSSim.hpp . . . . . **??**
Timer.cpp
This file is the implementation of the Timer class 19
Timer.hpp . . . . . **??**

Chapter 4

Class Documentation

4.1 ConfigSettings Class Reference

class for Configuration Data

`#include <OSSim.hpp>`

Public Member Functions

- ConfigSettings ()

  Constructs the ConfigSettings object.
- bool checkFilePath (const std::string &path)

  checks for proper file extension
- void loadSettings (const std::string &filePath) throw (std::logic_error)

  Loads settings.
- int getData (std::string &key) throw (std::logic_error)

  Gets the data.

Public Attributes

- std::string **metaFile**
- std::string **logFile**
- std::string **outputType**
- std::string **memType**
- std::string **type**
- int **memSize**
- int **blockSize**
- int **projQuan**
- int **hdQuan**
- int **quantum**
- double **version**

4.1.1 Detailed Description

class for Configuration Data

Abstract data type for collection and storing Configuartion Data information

4.1.2 Constructor & Destructor Documentation

4.1.2.1 ConfigSettings()

```
ConfigSettings::ConfigSettings ( )
```

Constructs the ConfigSettings object.

defines regular expressions for parising .config file, declares keys for map, initializes class variables

4.1.3 Member Function Documentation

4.1.3.1 checkFilePath()

```
bool ConfigSettings::checkFilePath (
            const std::string & path )
```

checks for proper file extension

Parameters

in	*path*	The path

Returns

return bool based on check

4.1.3.2 getData()

```
int ConfigSettings::getData (
            std::string & key ) throw std::logic_error)
```

Gets the data.

Parameters

key	The key

Returns

The data.

4.1.3.3 loadSettings()

```
void ConfigSettings::loadSettings (
            const std::string & filePath ) throw std::logic_error)
```

Loads settings.

Parameters

in	*filePath*	The file path

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

- OSSim.hpp
- OSSim.cpp

4.2 MetaSettings::data Struct Reference

Public Member Functions

- data & **operator=** (const data &other)

Public Attributes

- std::string **meta_data_code**
- std::string **meta_data_desc**
- int **numCycles**
- int **processNum**

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

- OSSim.hpp

4.3 Errors Struct Reference

ouput for different possible errors

`#include <OSSim.hpp>`

Public Member Functions

- std::string & **err** (std::string &str)

Public Attributes

- std::string **badConfExt** = "ERROR: The supplied configuration file does not use the required file extention. Please use '.conf' for all configuration files. The simulation will end now."
- std::string **badMetExt** = "ERROR: The supplied meta-data file does not use the required file extention. Please use '.mdf' for all meta-data files. The simulation will end now."
- std::string **badLogExt** = "ERROR: The supplied log file path does not use the required file extention. Please use '.lgf' for all log files. The simulation will end now."
- std::string **badConf** = "ERROR: The configuration file is missing or contains unexpected data. Please review the file content for errors and try again. The simulation will end now."
- std::string **badMeta** = "ERROR: The metadata file is missing or contains unexpected data. Please review the file content for errors and try again. The simulation will end now."
- std::string **badKey** = "ERROR: Requested data not found. This could be due to errors in the meta-data file. The simulation will end now."
- std::string **badOut** = "ERROR: the requested output configuration is not supported. The simulation will end now."

4.3.1 Detailed Description

ouput for different possible errors

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

- OSSim.hpp

4.4 MetaSettings Class Reference

class for meta data

`#include <OSSim.hpp>`

Classes

- struct data

Public Member Functions

- MetaSettings ()

  Constructs the Meta Settings object.
- bool checkFilePath (const std::string &path)

  check for proper file extention
- void loadData (std::string &line) throw (std::logic_error)

  Loads a data.
- void loadVec () throw (std::logic_error)

  Loads a vector.
- bool isEmpty () const

  Determines if empty.

Public Attributes

- int **processNum**
- int **numItems**
- std::string **filePath**
- std::string **alg**
- std::vector< std::queue< data > > **metaDataVec**
- std::vector< data > **metaData**

4.4.1 Detailed Description

class for meta data

Abstract data type for collection and storing Meta Data information

4.4.2 Member Function Documentation

4.4.2.1 checkFilePath()

```
bool MetaSettings::checkFilePath (
            const std::string & path )
```

check for proper file extention

Parameters

in	*path*	The path

Returns

bool based on check

4.4.2.2 isEmpty()

```
bool MetaSettings::isEmpty ( ) const
```

Determines if empty.

Returns

True if empty, False otherwise.

4.4.2.3 loadData()

```
void MetaSettings::loadData (
            std::string & metaFilePath ) throw std::logic_error)
```

Loads a data.

Parameters

metaFilePath	The meta file path

4.4.2.4 loadVec()

```
void MetaSettings::loadVec ( ) throw std::logic_error)
```

Loads a vector.

cretes a vector of queues and loads processes from the meta Data into it

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

- OSSim.hpp
- OSSim.cpp

4.5 PCB Class Reference

Process Control block.

```
#include <OSSim.hpp>
```

Public Member Functions

- PCB (MetaSettings *meta, ConfigSettings *config)
  Constructs Process Control Block object.
- void runSim () throw (std::logic_error)
  exectutes the simulation
- std::string getStartLogString (const std::string &desc, const std::string &code, int i) throw (std::logic_error)
  Gets the start log string.
- void logProcess (std::string line) throw (std::logic_error)
  Logs a process.

Static Public Member Functions

- static void * runProcess (void *p)
  pthread function for I/O processes
- static void * RRcount (void *q)
  timer for RR algorithm
- static void * loadProcess (void *settings)
  load thread
- static void readInTimer ()
  Timer for load process thread.
- static void checkTimer (int cdtime)
  Counts the number of down.

Public Attributes

- int **hdInCount**
- int **hdOutCount**
- int **projOutCount**
- int **processState**
- int **address**
- Timer * **program_timer**
- std::ostringstream **log**
- ConfigSettings * **pcbConfigurationData**
- MetaSettings * **pcbMetaData**

4.5.1 Detailed Description

Process Control block.

controls exectuition od simulation and manipulation of data

4.5.2 Constructor & Destructor Documentation

4.5.2.1 PCB()

```
PCB::PCB (
            MetaSettings * meta,
            ConfigSettings * config )
```

Constructs Process Control Block object.

defines integer values for process states, creates timer object, and intitalizes class variables

4.5.3 Member Function Documentation

4.5.3.1 checkTimer()

```
void PCB::checkTimer (
            int cdtime )  [static]
```

Counts the number of down.

Parameters

in	*cdtime*	The cdtime

4.5.3.2 getStartLogString()

```
std::string PCB::getStartLogString (
            const std::string & desc,
            const std::string & code,
            int i ) throw std::logic_error)
```

Gets the start log string.

Parameters

in	*desc*	The description
in	*code*	The code
in	*i*	process number

Returns

The start log string.

4.5.3.3 loadProcess()

```
void * PCB::loadProcess (
            void * settings )  [static]
```

load thread

thread to itteralivley load the meta data 10 times every 100ms

Parameters

settings	Meta Data object

Returns

retruns a void pointer

4.5.3.4 logProcess()

```
void PCB::logProcess (
            std::string line ) throw std::logic_error)
```

Logs a process.

Parameters

in	*line*	The line

4.5.3.5 RRcount()

```
void * PCB::RRcount (
            void * q )  [static]
```

timer for RR algorithm

Parameters

q	void pointer to quantum for RR algorithm

Returns

void pointer

4.5.3.6 runProcess()

```
void * PCB::runProcess (
            void * p )  [static]
```

pthread function for I/O processes

Parameters

p	cycle times

Returns

void pointer

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

- OSSim.hpp
- OSSim.cpp

4.6 Timer Class Reference

Public Member Functions

- Timer ()
- void start () throw (runtime_error)
- void stop () throw (logic_error)
- double getElapsedTime () const throw (logic_error)
- void **lap** () throw (logic_error)

Public Attributes

- struct timeval beginTime **duration**

4.6.1 Constructor & Destructor Documentation

4.6.1.1 Timer()

```
Timer::Timer ( )
```

This is a constructor for the Timer class.

Postcondition

1. duration is initialized to 0
2. timerWasStarted is initialized to false

4.6.2 Member Function Documentation

4.6.2.1 getElapsedTime()

```
double Timer::getElapsedTime ( ) const throw logic_error)
```

getElapsedTime() returns the value of duration. exception is thrown if duration is 0 or timer was never started

Precondition

1. duration has been calculated
2. timerWasStarted is set to false

Postcondition

1. duration is returned

Exceptions

logic_error	is thrown if timer was not stopped
logic_error	is thrown if duration was not calculated

4.6.2.2 start()

```
void Timer::start ( ) throw runtime_error)
```

start() records the current time. will throw an exception if recorded time is negative

Precondition

1. timeWasStarted is false
2. gettimerofday() works correctly
3. durration is 0

Postcondition

1. beginTime struct is initialized to current time
2. timWasStarted set to true

Exceptions

runtime_error	if time returned is negative, "gettimeofday() returned -1. Serious problem. Can't run timer."

4.6.2.3 stop()

```
void Timer::stop ( ) throw logic_error)
```

stop() records the current time then calculates duation. Will throw an exception if timer was never started.

Precondition

1. timerWasStarted set to true

Postcondition

1. duration struct initialized to current time
2. value stored in beginTime is multiplied by 1000000 then the product is added to the origial value ad sotred in t2
3. value stored in endTime is nultiplied by 1000000 then the product is added to the original value and stored in t1
4. t2 is subracted from t1 and stored in duration

Exceptions

logic_error	is thrown if timer was never started

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

- Timer.hpp
- Timer.cpp

Chapter 5

File Documentation

5.1 Timer.cpp File Reference

this file is the implementation of the Timer class

```
#include "Timer.hpp"
```

5.1.1 Detailed Description

this file is the implementation of the Timer class

Version

Original Code 1.0.0 (09/24/2017) - Nathan Smith

Index

checkFilePath
    ConfigSettings, 8
    MetaSettings, 11
checkTimer
    PCB, 14
ConfigSettings, 7
    checkFilePath, 8
    ConfigSettings, 8
    getData, 8
    loadSettings, 9

Errors, 10

getData
    ConfigSettings, 8
getElapsedTime
    Timer, 17
getStartLogString
    PCB, 14

isEmpty
    MetaSettings, 11

loadData
    MetaSettings, 12
loadProcess
    PCB, 14
loadSettings
    ConfigSettings, 9
loadVec
    MetaSettings, 12
logProcess
    PCB, 15

MetaSettings, 10
    checkFilePath, 11
    isEmpty, 11
    loadData, 12
    loadVec, 12
MetaSettings::data, 9

PCB, 12
    checkTimer, 14
    getStartLogString, 14
    loadProcess, 14
    logProcess, 15
    PCB, 13
    RRcount, 15
    runProcess, 15

RRcount
    PCB, 15
runProcess
    PCB, 15

start
    Timer, 17
stop
    Timer, 18

Timer, 16
    getElapsedTime, 17
    start, 17
    stop, 18
    Timer, 16
Timer.cpp, 19