Multiterminal messenger

Generated by Doxygen 1.8.13

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 message

Messenger buffer class.

Public Member Functions

- message ()

  Standard constructor.
- message (char *arg)

  The constructor which, when created, is oriented to the string arg, the buffer now points to it, and _data coincides with arg and _str points to the string from the second element.
- ~message ()

  Standard destructor.
- int add (const char *arg)

  Appends it to the current buffer line's string _str based on the value of _capacity.
- int capacity ()

  The function considers the new value as a private variable _capacity, based on changes to the _str buffer, and provides the new real value to the user.
- int mov (int len)

  Shifts a string to the beginning by a specified number of characters.
- message & operator= (const char *arg)

  The operator equals the string, that is, appends it to the end of the current line of the _str buffer, focusing on the _capacity value, if there is not enough space in the buffer, the first values in the buffer are erased, meaning _str.

Public Attributes

- char * _data

  pointer to full buffer.
- int _size

  line size or buffer size minus one.
- char * _str

  pointer to string in buffer.

Private Attributes

- int _capacity

  a pointer to the current character of the end of the string in the buffer, meaning str.

3.1.1 Detailed Description

Messenger buffer class.

This class stores a message buffer and controls all events occurring with the buffer.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 message() [1/2]

```
message ( )
```

Standard constructor.

3.1.2.2 message() [2/2]

```
message (
            char * arg )
```

The constructor which, when created, is oriented to the string arg, the buffer now points to it, and _data coincides with arg and _str points to the string from the second element.

Parameters

in	*arg*	pointer to the string, the size of the string must match SIZE_BUF_MAX.

3.1.2.3 ~message()

```
~message ( )
```

Standard destructor.

3.1.3 Member Function Documentation

3.1.3.1 add()

```
int add (
            const char * arg )
```

Appends it to the current buffer line's string _str based on the value of _capacity.

Parameters

in	*arg*	pointer to the string, the string must not exceed SIZE_BUF_MAX otherwise characters whose indices exceed SIZE_BUF_MAX will be ignored.

Returns

0.

Appends it to the current buffer line's string _str based on the value of _capacity.

Parameters

in	*arg*	pointer to the string, the string must not exceed SIZE_BUF_MAX otherwise characters whose indices exceed SIZE_BUF_MAX will be ignored.

Returns

0.

3.1.3.2 capacity()

```
int capacity ( )
```

The function considers the new value as a private variable _capacity, based on changes to the _str buffer, and provides the new real value to the user.

Returns

private variable value _capacity.

3.1.3.3 mov()

```
int mov (
            int len )
```

Shifts a string to the beginning by a specified number of characters.

Parameters

in	*len*	the length of the rubbed line.

Returns

0.

3.1.3.4 operator=()

```
message & operator= (
            const char * arg )
```

The operator equals the string, that is, appends it to the end of the current line of the _str buffer, focusing on the _capacity value, if there is not enough space in the buffer, the first values in the buffer are erased, meaning _str.

Parameters

in	*arg*	pointer to the string, the string must not exceed SIZE_BUF_MAX otherwise characters whose indices exceed SIZE_BUF_MAX will be ignored.

Returns

returns a link to this.

The operator equals the string, that is, appends it to the end of the current line of the _str buffer, focusing on the _capacity value, if there is not enough space in the buffer, the first values in the buffer are erased, meaning _str

Parameters

in	*arg*	pointer to the string, the string must not exceed SIZE_BUF_MAX otherwise characters whose indices exceed SIZE_BUF_MAX will be ignored.

Returns

returns a link to this.

3.1.4 Member Data Documentation

3.1.4.1 _capacity

```
int _capacity  [private]
```

a pointer to the current character of the end of the string in the buffer, meaning str.

3.1.4.2 _data

```
char* _data
```

pointer to full buffer.

3.1.4.3 _size

```
int _size
```

line size or buffer size minus one.

3.1.4.4 _str

```
char* _str
```

pointer to string in buffer.

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

- prog.cpp

Chapter 4

File Documentation

4.1 prog.cpp File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/shm.h>
```

Classes

- class message

  Messenger buffer class.

Functions

- int closeprog (message *buf)

  Prepares to close files or closes it before the end of the program.
- int ERROREn (message *buf)

  Function for an adequate program crash when a user enters an empty message.
- int main ()
- char * openfile (char *namefile)

  Opens or creates a file using functions ftok(), shmget().

Variables

- const int SIZE_BUF_MAX = 1024

  maximum allowed message buffer length.
- const int SIZE_STR_MAX = 256

  terminal messenger.

4.1.1 Function Documentation

4.1.1.1 closeprog()

```
int closeprog (
            message * buf )
```

Prepares to close files or closes it before the end of the program.

Parameters

out	*buf*	pointer to the message buffer to write.

Returns

0.

4.1.1.2 ERRORen()

```
int ERRORen (
            message * buf )
```

Function for an adequate program crash when a user enters an empty message.

Parameters

out	*buf*	pointer to the message buffer to write.

Returns

0.

4.1.1.3 main()

```
int main ( )
```

4.1.1.4 openfile()

```
char * openfile (
            char * namefile )
```

Opens or creates a file using functions ftok(), shmget().

Parameters

in	*namefile*	full file path.

Returns

pointer to string array.

4.1.2 Variable Documentation

4.1.2.1 SIZE_BUF_MAX

```
const int SIZE_BUF_MAX = 1024
```

maximum allowed message buffer length.

Constant of the maximum length of the buffer storing user messages.

4.1.2.2 SIZE_STR_MAX

```
const int SIZE_STR_MAX = 256
```

terminal messenger.

Terminal messenger launched in the console of a single computer.maximum allowable length of input string.

Constant of the maximum value of the string length, the message entered by the user.