

xml-shapes

v0

Gerado por Doxygen 1.8.15

1 Namespaces	1
1.1 Lista de Namespaces	1
2 Índice dos Componentes	3
2.1 Lista de Classes	3
3 Índice dos Arquivos	5
3.1 Lista de Arquivos	5
4 Namespace	7
4.1 Referência do Namespace math	7
4.1.1 Funções	7
4.1.1.1 count_shapes()	7
4.1.1.2 matrix_destroy()	8
4.1.1.3 matrix_init()	8
4.2 Referência do Namespace structures	8
4.3 Referência do Namespace xml	8
4.3.1 Funções	9
4.3.1.1 balanced()	9
4.3.1.2 extract() [1/2]	9
4.3.1.3 extract() [2/2]	9
5 Classes	11
5.1 Referência do Template da Classe structures::LinkedList< T >	11
5.1.1 Descrição detalhada	11
5.1.2 Construtores e Destrutores	12
5.1.2.1 ~LinkedList()	12
5.1.3 Funções membros	12
5.1.3.1 back()	12
5.1.3.2 clear()	12
5.1.3.3 dequeue()	12
5.1.3.4 empty()	13
5.1.3.5 enqueue()	13
5.1.3.6 front()	13
5.1.3.7 size()	13
5.2 Referência do Template da Classe structures::LinkedList< T >	14
5.2.1 Descrição detalhada	14
5.2.2 Construtores e Destrutores	14
5.2.2.1 ~LinkedList()	14
5.2.3 Funções membros	15
5.2.3.1 clear()	15
5.2.3.2 empty()	15
5.2.3.3 pop()	15
5.2.3.4 push()	15

5.2.3.5 size()	16
5.2.3.6 top()	16
6 Arquivos	17
6.1 Referência do Arquivo src/linked_queue.h	17
6.2 Referência do Arquivo src/linked_queue.inc	17
6.3 Referência do Arquivo src/linked_stack.h	17
6.4 Referência do Arquivo src/linked_stack.inc	18
6.5 Referência do Arquivo src/main.cpp	18
6.5.1 Funções	18
6.5.1.1 main()	18
6.5.1.2 matrix_init()	18
6.6 Referência do Arquivo src/matrix.cpp	19
6.7 Referência do Arquivo src/matrix.h	19
6.8 Referência do Arquivo src/xml.cpp	19
6.9 Referência do Arquivo src/xml.h	20
Índice Remissivo	21

Capítulo 1

Namespaces

1.1 Lista de Namespaces

Esta é a lista de todos os Namespaces com suas respectivas descrições:

math	7
structures	8
xml	8

Capítulo 2

Índice dos Componentes

2.1 Lista de Classes

Aqui estão as classes, estruturas, uniões e interfaces e suas respectivas descrições:

<code>structures::LinkedList< T ></code>	
Fila Encadeada	11
<code>structures::LinkedStack< T ></code>	
Pilha Encadeada	14

Capítulo 3

Índice dos Arquivos

3.1 Lista de Arquivos

Esta é a lista de todos os arquivos e suas respectivas descrições:

src/ linked_queue.h	17
src/ linked_queue.inc	17
src/ linked_stack.h	17
src/ linked_stack.inc	18
src/ main.cpp	18
src/ matrix.cpp	19
src/ matrix.h	19
src/ xml.cpp	19
src/ xml.h	20

Capítulo 4

Namespace

4.1 Referência do Namespace math

Funções

- int ** [matrix_init](#) (int height, int width)
Inicializa uma matriz como um array de height ponteiros para arrays com width ints. Deve ser destruído com [matrix_destroy\(\)](#) para liberar a memória.
- void [matrix_destroy](#) (int **M, int height)
Utilizado para liberar a memória alocada por [matrix_init\(\)](#).
- int [count_shapes](#) (int **E, int height, int width)
Calcula o numero de componentes conexos na matriz usando vizinhanca-4.

4.1.1 Funções

4.1.1.1 count_shapes()

```
int math::count_shapes (
    int ** E,
    int height,
    int width )
```

Calcula o numero de componentes conexos na matriz usando vizinhanca-4.

Definição na linha 29 do arquivo matrix.cpp.

Referenciado(a) por main().

4.1.1.2 `matrix_destroy()`

```
void math::matrix_destroy (
    int ** M,
    int height )
```

Utilizado para liberar a memoria alocada por [matrix_init\(\)](#).

Definição na linha 23 do arquivo matrix.cpp.

Referenciado(a) por `count_shapes()` e `main()`.

4.1.1.3 `matrix_init()`

```
int ** math::matrix_init (
    int height,
    int width )
```

Inicializa uma matriz como um array de height ponteiros para arrays com width ints. Deve ser destruido com [matrix_destroy\(\)](#) para liberar a memoria.

Definição na linha 12 do arquivo matrix.cpp.

Referenciado(a) por `count_shapes()` e `matrix_init()`.

4.2 Referência do Namespace structures

Componentes

- class [LinkedQueue](#)
Fila Encadeada.
- class [LinkedStack](#)
Pilha Encadeada.

4.3 Referência do Namespace xml

Funções

- bool [balanced](#) (const std::string &xml)
Confere a validade da estrutura do XML contido na string.
- std::string [extract](#) (const std::string &origin, const std::string &open, const std::string &close, std::size_t &from)
Extrai substring de uma string entre dois delimitadores encontrados a partir de uma dada posicao.
- std::string [extract](#) (const std::string &origin, const std::string &open, const std::string &close)
Extrai substring de uma string entre dois delimitadores.

4.3.1 Funções

4.3.1.1 `balanced()`

```
bool xml::balanced (
    const std::string & xml )
```

Confere a validade da estrutura do XML contido na string.

Definição na linha 12 do arquivo xml.cpp.

Referenciado(a) por `main()`.

4.3.1.2 `extract()` [1/2]

```
std::string xml::extract (
    const std::string & origin,
    const std::string & open,
    const std::string & close,
    std::size_t & from )
```

Extrai substring de uma string entre dois delimitadores encontrados a partir de uma dada posicao.

Retorna a substring extraída (sem delimitadores) e altera a posicao para o final do ultimo delimitador encontrado. Se nao encontrar nada, retorna string vazia e leva a posicao para npos.

Definição na linha 53 do arquivo xml.cpp.

Referenciado(a) por `extract()` e `main()`.

4.3.1.3 `extract()` [2/2]

```
std::string xml::extract (
    const std::string & origin,
    const std::string & open,
    const std::string & close )
```

Extrai substring de uma string entre dois delimitadores.

Retorna a substring extraída (sem delimitadores). Se nao encontrar nada, retorna string vazia.

Definição na linha 70 do arquivo xml.cpp.

Capítulo 5

Classes

5.1 Referência do `Template` da Classe `structures::LinkedList< T >`

Fila Encadeada.

```
#include <linked_queue.h>
```

Membros Públicos

- `~LinkedList ()`
Destrutor.
- `void clear ()`
Limpa a Fila.
- `void enqueue (const T &data)`
Enfileira.
- `T dequeue ()`
Desenfileira.
- `T & front () const`
Acessa a frente da Fila.
- `T & back () const`
Acessa o ultimo da Fila.
- `bool empty () const`
Confere se a Fila esta vazia.
- `std::size_t size () const`
Retorna o tamanho da Fila.

5.1.1 Descrição detalhada

```
template<typename T>  
class structures::LinkedList< T >
```

Fila Encadeada.

Definição na linha 12 do arquivo `linked_queue.h`.

5.1.2 Construtores e Destrutores

5.1.2.1 ~LinkedList()

```
template<typename T >
LinkedList::~~LinkedList ( )
```

Destrutor.

Definição na linha 5 do arquivo linked_queue.h.

5.1.3 Funções membros

5.1.3.1 back()

```
template<typename T >
T & LinkedList::back ( ) const
```

Acessa o ultimo da Fila.

Definição na linha 59 do arquivo linked_queue.h.

5.1.3.2 clear()

```
template<typename T >
void LinkedList::clear ( )
```

Limpa a Fila.

Definição na linha 10 do arquivo linked_queue.h.

5.1.3.3 dequeue()

```
template<typename T >
T LinkedList::dequeue ( )
```

Desenfileira.

Definição na linha 25 do arquivo linked_queue.h.

Referenciado(a) por math::count_shapes().

5.1.3.4 `empty()`

```
template<typename T >
bool LinkedList::empty ( ) const
```

Confere se a Fila esta vazia.

Definição na linha 67 do arquivo `linked_queue.h`.

Referenciado(a) por `math::count_shapes()`.

5.1.3.5 `enqueue()`

```
template<typename T >
void LinkedList::enqueue (
    const T & data )
```

Enfileira.

Definição na linha 16 do arquivo `linked_queue.h`.

Referenciado(a) por `math::count_shapes()`.

5.1.3.6 `front()`

```
template<typename T >
T & LinkedList::front ( ) const
```

Acessa a frente da Fila.

Definição na linha 51 do arquivo `linked_queue.h`.

5.1.3.7 `size()`

```
template<typename T >
std::size_t LinkedList::size ( ) const
```

Retorna o tamanho da Fila.

Definição na linha 72 do arquivo `linked_queue.h`.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [src/linked_queue.h](#)
- [src/linked_queue.inc](#)

5.2 Referência do `Template` da Classe `structures::LinkedList< T >`

Pilha Encadeada.

```
#include <linked_stack.h>
```

Membros Públicos

- `~LinkedList()`
Destrutor.
- `void push(const T &data)`
Empilha.
- `T pop()`
Desempilha.
- `T & top() const`
Acessa o topo da Pilha.
- `bool empty() const`
Confere se a Pilha esta vazia.
- `std::size_t size() const`
Retorna o tamanho da Pilha.
- `void clear()`
Limpa a Pilha.

5.2.1 Descrição detalhada

```
template<typename T>  
class structures::LinkedList< T >
```

Pilha Encadeada.

Definição na linha 12 do arquivo `linked_stack.h`.

5.2.2 Construtores e Destrutores

5.2.2.1 `~LinkedList()`

```
template<typename T >  
LinkedList::~~LinkedList ( )
```

Destrutor.

Definição na linha 5 do arquivo `linked_stack.h`.

5.2.3 Funções membros

5.2.3.1 `clear()`

```
template<typename T >
void LinkedStack::clear ( )
```

Limpa a Pilha.

Definição na linha 10 do arquivo `linked_stack.h`.

5.2.3.2 `empty()`

```
template<typename T >
bool LinkedStack::empty ( ) const
```

Confere se a Pilha esta vazia.

Definição na linha 49 do arquivo `linked_stack.h`.

Referenciado(a) por `xml::balanced()`.

5.2.3.3 `pop()`

```
template<typename T >
T LinkedStack::pop ( )
```

Desempilha.

Definição na linha 22 do arquivo `linked_stack.h`.

Referenciado(a) por `xml::balanced()`.

5.2.3.4 `push()`

```
template<typename T >
void LinkedStack::push (
    const T & data )
```

Empilha.

Definição na linha 16 do arquivo `linked_stack.h`.

Referenciado(a) por `xml::balanced()`.

5.2.3.5 size()

```
template<typename T >  
std::size_t LinkedStack::size ( ) const
```

Retorna o tamanho da Pilha.

Definição na linha 54 do arquivo linked_stack.h.

5.2.3.6 top()

```
template<typename T >  
T & LinkedStack::top ( ) const
```

Acessa o topo da Pilha.

Definição na linha 41 do arquivo linked_stack.h.

Referenciado(a) por xml::balanced().

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [src/linked_stack.h](#)
- [src/linked_stack.inc](#)

Capítulo 6

Arquivos

6.1 Referência do Arquivo src/linked_queue.h

```
#include <cstdlib>
#include <stdexcept>
#include "linked_queue.inc"
```

Componentes

- class `structures::LinkedList< T >`
Fila Encadeada.

`Namespaces`

- `structures`

6.2 Referência do Arquivo src/linked_queue.inc

6.3 Referência do Arquivo src/linked_stack.h

```
#include <cstdlib>
#include <stdexcept>
#include "linked_stack.inc"
```

Componentes

- class `structures::LinkedList< T >`
Pilha Encadeada.

Namespaces

- [structures](#)

6.4 Referência do Arquivo src/linked_stack.inc

6.5 Referência do Arquivo src/main.cpp

```
#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <cctype>
#include "xml.h"
#include "linked_queue.h"
#include "matrix.h"
```

Funções

- static int ** [matrix_init](#) (int height, int width, const std::string &data)
Inicializa uma matriz a partir da string que representa seus valores. Deve ser destruído com [matrix_destroy\(\)](#) para liberar a memória.
- int [main](#) ()

6.5.1 Funções

6.5.1.1 main()

```
int main ( )
```

Definição na linha 19 do arquivo main.cpp.

6.5.1.2 matrix_init()

```
static int ** matrix_init (
    int height,
    int width,
    const std::string & data ) [static]
```

Inicializa uma matriz a partir da string que representa seus valores. Deve ser destruído com [matrix_destroy\(\)](#) para liberar a memória.

Definição na linha 70 do arquivo main.cpp.

Referenciado(a) por main().

6.6 Referência do Arquivo src/matrix.cpp

```
#include "matrix.h"
#include <cassert>
#include <utility>
#include "linked_queue.h"
```

Namespaces

- [math](#)

Funções

- `int ** math::matrix_init (int height, int width)`
Inicializa uma matriz como um array de height ponteiros para arrays com width ints. Deve ser destruído com [matrix_destroy\(\)](#) para liberar a memória.
- `void math::matrix_destroy (int **M, int height)`
Utilizado para liberar a memória alocada por [matrix_init\(\)](#).
- `int math::count_shapes (int **E, int height, int width)`
Calcula o número de componentes conexos na matriz usando vizinhança-4.

6.7 Referência do Arquivo src/matrix.h

Namespaces

- [math](#)

Funções

- `int ** math::matrix_init (int height, int width)`
Inicializa uma matriz como um array de height ponteiros para arrays com width ints. Deve ser destruído com [matrix_destroy\(\)](#) para liberar a memória.
- `void math::matrix_destroy (int **M, int height)`
Utilizado para liberar a memória alocada por [matrix_init\(\)](#).
- `int math::count_shapes (int **E, int height, int width)`
Calcula o número de componentes conexos na matriz usando vizinhança-4.

6.8 Referência do Arquivo src/xml.cpp

```
#include "xml.h"
#include <string>
#include <cstddef>
#include "linked_stack.h"
```

Namespaces

- [xml](#)

Funções

- bool [xml::balanced](#) (const std::string &xml)
Confere a validade da estrutura do XML contido na string.
- std::string [xml::extract](#) (const std::string &origin, const std::string &open, const std::string &close, std::size_t &from)
Extraí substring de uma string entre dois delimitadores encontrados a partir de uma dada posição.
- std::string [xml::extract](#) (const std::string &origin, const std::string &open, const std::string &close)
Extraí substring de uma string entre dois delimitadores.

6.9 Referência do Arquivo src/xml.h

```
#include <string>
#include <cstdint>
```

Namespaces

- [xml](#)

Funções

- bool [xml::balanced](#) (const std::string &xml)
Confere a validade da estrutura do XML contido na string.
- std::string [xml::extract](#) (const std::string &origin, const std::string &open, const std::string &close, std::size_t &from)
Extraí substring de uma string entre dois delimitadores encontrados a partir de uma dada posição.
- std::string [xml::extract](#) (const std::string &origin, const std::string &open, const std::string &close)
Extraí substring de uma string entre dois delimitadores.

Índice Remissivo

- ~LinkedList
 - structures::LinkedList< T >, 12
- ~LinkedList
 - structures::LinkedList< T >, 14
- back
 - structures::LinkedList< T >, 12
- balanced
 - xml, 9
- clear
 - structures::LinkedList< T >, 12
 - structures::LinkedList< T >, 15
- count_shapes
 - math, 7
- dequeue
 - structures::LinkedList< T >, 12
- empty
 - structures::LinkedList< T >, 12
 - structures::LinkedList< T >, 15
- enqueue
 - structures::LinkedList< T >, 13
- extract
 - xml, 9
- front
 - structures::LinkedList< T >, 13
- main
 - main.cpp, 18
- main.cpp
 - main, 18
 - matrix_init, 18
- math, 7
 - count_shapes, 7
 - matrix_destroy, 7
 - matrix_init, 8
- matrix_destroy
 - math, 7
- matrix_init
 - main.cpp, 18
 - math, 8
- pop
 - structures::LinkedList< T >, 15
- push
 - structures::LinkedList< T >, 15
- size
 - structures::LinkedList< T >, 13
 - structures::LinkedList< T >, 15
- src/linked_queue.h, 17
- src/linked_queue.inc, 17
- src/linked_stack.h, 17
- src/linked_stack.inc, 18
- src/main.cpp, 18
- src/matrix.cpp, 19
- src/matrix.h, 19
- src/xml.cpp, 19
- src/xml.h, 20
- structures, 8
- structures::LinkedList< T >, 11
 - ~LinkedList, 12
 - back, 12
 - clear, 12
 - dequeue, 12
 - empty, 12
 - enqueue, 13
 - front, 13
 - size, 13
- structures::LinkedList< T >, 14
 - ~LinkedList, 14
 - clear, 15
 - empty, 15
 - pop, 15
 - push, 15
 - size, 15
 - top, 16
- top
 - structures::LinkedList< T >, 16
- xml, 8
 - balanced, 9
 - extract, 9