

## Projeto I - Processamento de XML com imagens binárias

Gerado por Doxygen 1.8.13



# Sumário

<b>1</b>	<b>Namespaces</b>	<b>1</b>
1.1	<a href="#">Lista de Namespaces</a>	1
<b>2</b>	<b>Índice dos Componentes</b>	<b>3</b>
2.1	<a href="#">Lista de Componentes</a>	3
<b>3</b>	<b>Índice dos Arquivos</b>	<b>5</b>
3.1	<a href="#">Lista de Arquivos</a>	5
<b>4</b>	<b>Namespaces</b>	<b>7</b>
4.1	<a href="#">Refência do Namespace structures</a>	7
4.1.1	<a href="#">Descrição Detalhada</a>	7
4.2	<a href="#">Refência do Namespace xml_processing</a>	7
4.2.1	<a href="#">Descrição Detalhada</a>	7
<b>5</b>	<b>Classes</b>	<b>9</b>
5.1	<a href="#">Referência da Template de Classe structures::LinkedList&lt; T &gt;</a>	9
5.1.1	<a href="#">Descrição Detalhada</a>	9
5.1.2	<a href="#">Construtores &amp; Destrutores</a>	10
5.1.2.1	<a href="#">LinkedList()</a>	10
5.1.2.2	<a href="#">~LinkedList()</a>	10
5.1.3	<a href="#">Métodos</a>	10
5.1.3.1	<a href="#">back()</a>	10
5.1.3.2	<a href="#">clear()</a>	10
5.1.3.3	<a href="#">dequeue()</a>	10

5.1.3.4	<code>empty()</code>	11
5.1.3.5	<code>enqueue()</code>	11
5.1.3.6	<code>front()</code>	11
5.1.3.7	<code>size()</code>	11
5.2	Referência da Template de Classe <code>structures::LinkedList&lt; T &gt;</code>	11
5.2.1	Descrição Detalhada	12
5.2.2	Construtores & Destrutores	12
5.2.2.1	<code>LinkedList()</code>	12
5.2.2.2	<code>~LinkedList()</code>	12
5.2.3	Métodos	13
5.2.3.1	<code>clear()</code>	13
5.2.3.2	<code>empty()</code>	13
5.2.3.3	<code>pop()</code>	13
5.2.3.4	<code>push()</code>	13
5.2.3.5	<code>size()</code>	13
5.2.3.6	<code>top()</code>	14
5.3	Referência da Classe <code>xml_processing::Matrix</code>	14
5.3.1	Descrição Detalhada	14
5.3.2	Construtores & Destrutores	14
5.3.2.1	<code>Matrix()</code>	14
5.3.2.2	<code>~Matrix()</code>	15
5.3.3	Métodos	15
5.3.3.1	<code>count_connected()</code>	15
5.3.3.2	<code>get_value()</code>	15
5.3.3.3	<code>set_value()</code>	16
5.4	Referência da Classe <code>xml_processing::XML</code>	16
5.4.1	Descrição Detalhada	16
5.4.2	Construtores & Destrutores	17
5.4.2.1	<code>XML()</code>	17
5.4.3	Métodos	17
5.4.3.1	<code>extract()</code>	17
5.4.3.2	<code>validation()</code>	18

<b>6 Arquivos</b>	<b>19</b>
6.1 Referência do Arquivo include/linked_queue.h . . . . .	19
6.1.1 Descrição Detalhada . . . . .	19
6.2 Referência do Arquivo include/linked_stack.h . . . . .	20
6.2.1 Descrição Detalhada . . . . .	20
6.3 Referência do Arquivo include/matrix.h . . . . .	20
6.3.1 Descrição Detalhada . . . . .	21
6.4 Referência do Arquivo include/xml.h . . . . .	21
6.4.1 Descrição Detalhada . . . . .	21
6.5 Referência do Arquivo main.cpp . . . . .	22
6.5.1 Descrição Detalhada . . . . .	22
6.5.2 Funções . . . . .	22
6.5.2.1 main() . . . . .	22
6.6 Referência do Arquivo src/matrix.cpp . . . . .	23
6.6.1 Descrição Detalhada . . . . .	23
6.7 Referência do Arquivo src/xml.cpp . . . . .	23
6.7.1 Descrição Detalhada . . . . .	23
<b>Índice</b>	<b>25</b>



# Capítulo 1

## Namespaces

### 1.1 Lista de Namespaces

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

<a href="#">structures</a>		
	Estruturas de dados . . . . .	7
<a href="#">xml_processing</a>		
	Código para processamento de xml . . . . .	7





## Capítulo 2

# Índice dos Componentes

### 2.1 Lista de Componentes

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

<a href="#">structures::LinkedList&lt; T &gt;</a>	
Fila Encadeada . . . . .	9
<a href="#">structures::LinkedList&lt; T &gt;</a>	
Pilha Encadeada . . . . .	11
<a href="#">xml_processing::Matrix</a>	
Matriz . . . . .	14
<a href="#">xml_processing::XML</a>	
Xml . . . . .	16



## Capítulo 3

# Índice dos Arquivos

### 3.1 Lista de Arquivos

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

<a href="#">main.cpp</a>	Código principal . . . . .	22
<a href="#">include/linked_queue.h</a>	Código da fila encadeada . . . . .	19
<a href="#">include/linked_stack.h</a>	Código da pilha encadeada . . . . .	20
<a href="#">include/matrix.h</a>	Declaração da classe Matrix . . . . .	20
<a href="#">include/xml.h</a>	Declaração das funções validação e extração de conteúdo de arquivos xml . . . . .	21
<a href="#">src/matrix.cpp</a>	Implementação da classe Matrix . . . . .	23
<a href="#">src/xml.cpp</a>	Implementação das funções validação e extração de arquivos XML . . . . .	23



## Capítulo 4

# Namespaces

### 4.1 Referência do Namespace structures

Estruturas de dados.

#### Componentes

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

#### 4.1.1 Descrição Detalhada

Estruturas de dados.

### 4.2 Referência do Namespace xml\_processing

Código para processamento de xml.

#### Componentes

- class [Matrix](#)  
*Matriz.*
- class [XML](#)  
*Xml.*

#### 4.2.1 Descrição Detalhada

Código para processamento de xml.



## Capítulo 5

# Classes

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

Fila Encadeada.

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

#### Métodos Públicos

- `LinkedList ()`  
*Construtor padrão.*
- `~LinkedList ()`  
*Destrutor.*
- `void clear ()`  
*Limpa fila.*
- `void enqueue (const T &data)`  
*Enfilerar.*
- `T dequeue ()`  
*Desenfilerar.*
- `T & front () const`  
*Primeiro dado da fila.*
- `T & back () const`  
*Último dado da fila.*
- `bool empty () const`  
*Verifica se a fila está vazia.*
- `std::size_t size () const`  
*Tamanho da fila.*

#### 5.1.1 Descrição Detalhada

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

Fila Encadeada.

## 5.1.2 Construtores & Destrutores

### 5.1.2.1 LinkedQueue()

```
template<typename T >
structures::LinkedQueue< T >::LinkedQueue ( )
```

Construtor padrão.

### 5.1.2.2 ~LinkedQueue()

```
template<typename T >
structures::LinkedQueue< T >::~~LinkedQueue ( )
```

Destrutor.

## 5.1.3 Métodos

### 5.1.3.1 back()

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

Último dado da fila.

### 5.1.3.2 clear()

```
template<typename T >
void structures::LinkedQueue< T >::clear ( )
```

Limpa fila.

### 5.1.3.3 dequeue()

```
template<typename T >
T structures::LinkedQueue< T >::dequeue ( )
```

Desenfileirar.



#### 5.1.3.4 `empty()`

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

Verifica se a fila está vazia.

#### 5.1.3.5 `enqueue()`

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

Enfilerar.

#### 5.1.3.6 `front()`

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

Primeiro dado da fila.

#### 5.1.3.7 `size()`

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

Tamanho da fila.

A documentação para esta classe foi gerada a partir do seguinte arquivo:

- `include/linked_queue.h`

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

Pilha Encadeada.

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

## Métodos Públicos

- `LinkedStack ()`  
*Construtor padrão.*
- `~LinkedStack ()`  
*Destrutor.*
- `void clear ()`  
*Limpar pilha.*
- `void push (const T &data)`  
*Empilha.*
- `T pop ()`  
*Desempilha.*
- `T & top () const`  
*Topo da pilha.*
- `bool empty () const`  
*Verifica se a pilha está vazia.*
- `std::size_t size () const`  
*Tamanho da pilha.*

### 5.2.1 Descrição Detalhada

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

Pilha Encadeada.

### 5.2.2 Construtores & Destrutores

#### 5.2.2.1 LinkedStack()

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

Construtor padrão.

#### 5.2.2.2 ~LinkedStack()

```
template<typename T >  
structures::LinkedStack< T >::~~LinkedStack ( )
```

Destrutor.

### 5.2.3 Métodos

#### 5.2.3.1 `clear()`

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

Limpar pilha.

#### 5.2.3.2 `empty()`

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

Verifica se a pilha está vazia.

#### 5.2.3.3 `pop()`

```
template<typename T >
T structures::LinkedList< T >::pop ( )
```

Desempilha.

#### 5.2.3.4 `push()`

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

Empilha.

#### 5.2.3.5 `size()`

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

Tamanho da pilha.

### 5.2.3.6 top()

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

Topo da pilha.

A documentação para esta classe foi gerada a partir do seguinte arquivo:

- include/linked\_stack.h

## 5.3 Referência da Classe xml\_processing::Matrix

Matriz.

```
#include <matrix.h>
```

### Métodos Públicos

- [Matrix](#) (int n\_lines, int n\_columns)  
*Constroi uma matriz.*
- [~Matrix](#) ()  
*Destrói a matriz.*
- int [count\\_connected](#) ()  
*Conta componentes conexos na matriz.*
- int [get\\_value](#) (int line, int column)  
*Pega uma posição da matriz.*
- void [set\\_value](#) (int line, int column, int value)  
*Coloca um inteiro na posição passada.*

### 5.3.1 Descrição Detalhada

Matriz.

### 5.3.2 Construtores & Destrutores

#### 5.3.2.1 Matrix()

```
xml_processing::Matrix::Matrix (
    int n_lines,
    int n_columns )
```

Constroi uma matriz.

Constroi uma matriz do tipo ponteiro para ponteiro de inteiros. Esta matriz será nula, ou seja, todos os elementos são zero.

**Parâmetros**

<i>n_lines</i>	Número de linhas da matriz.
<i>n_columns</i>	Número de colunas da matriz.

**5.3.2.2 `~Matrix()`**

```
xml_processing::Matrix::~~Matrix ( )
```

Destrói a matriz.

Libera todo o espaço de memória ocupado pela matriz.

**5.3.3 Métodos****5.3.3.1 `count_connected()`**

```
int xml_processing::Matrix::count_connected ( )
```

Conta componentes conexos na matriz.

A matriz neste caso é uma imagem binária. Cada pixel conexo encontrado na matriz é rotulado com um valor inteiro e enfileirado utilizando uma fila encadeada. Utilizamos vizinhança-4 para o cálculo de componentes conexos. A vizinhança-4 de um pixel são os pixels que estão em cima, em baixo, na direita e na esquerda.

**Retorna**

Inteiro com o número de componentes conexos na matriz.

**5.3.3.2 `get_value()`**

```
int xml_processing::Matrix::get_value (
    int line,
    int column )
```

Pega uma posição da matriz.

**Parâmetros**

<i>line</i>	Linha onde será definido o valor.
<i>column</i>	Coluna onde será definido o valor.

**Retorna**

Inteiro que está na posição passada.

**5.3.3.3 set\_value()**

```
void xml_processing::Matrix::set_value (
    int line,
    int column,
    int value )
```

Coloca um inteiro na posição passada.

**Parâmetros**

<i>line</i>	Linha onde será definido o valor.
<i>column</i>	Coluna onde será definido o valor.
<i>value</i>	Valor que será colocado na matriz.

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

- [include/matrix.h](#)
- [src/matrix.cpp](#)

**5.4 Referência da Classe xml\_processing::XML**

Xml.

```
#include <xml.h>
```

**Métodos Públicos**

- [XML](#) (const std::string &xml\_string)  
*Construtor.*
- bool [validation](#) ()  
*Faz a validação do arquivo xml passado.*
- std::string [extract](#) (const std::string &tag\_begin, const std::string &tag\_end, std::size\_t &position)  
*Faz a extração do conteúdo entre as tags.*

**5.4.1 Descrição Detalhada**

Xml.

## 5.4.2 Construtores & Destrutores

### 5.4.2.1 XML()

```
xml_processing::XML::XML (
    const std::string & xml_string )
```

Construtor.

Recebe o arquivo xml no formato string para ser utilizado no restante dos metodos.

Parâmetros

<i>xml</i>	Arquivo xml no formato string.
------------	--------------------------------

## 5.4.3 Métodos

### 5.4.3.1 extract()

```
std::string xml_processing::XML::extract (
    const std::string & tag_begin,
    const std::string & tag_end,
    std::size_t & position )
```

Faz a extração do conteúdo entre as tags.

A extração copia o conteúdo entre as tags que estão a partir da posição passada para iniciar a busca no xml em formato de string.

Parâmetros

<i>tag_begin</i>	Indicador de abertura de uma tag.
<i>tag_end</i>	Indicador de fechamento de uma tag.
<i>position</i>	Posição inicial de busca no xml em formato string.

Retorna

String com o conteúdo entre as tags.

#### 5.4.3.2 validation()

```
bool xml_processing::XML::validation ( )
```

Faz a validação do arquivo xml passado.

A validação verifica se todas as tags abriram e fecharam corretamente. É utilizado uma pilha encadeada auxiliar neste processo.

##### Retorna

True se o arquivo for válido e false se for inválido.

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

- [include/xml.h](#)
- [src/xml.cpp](#)



## Capítulo 6

# Arquivos

### 6.1 Referência do Arquivo include/linked\_queue.h

Código da fila encadeada.

```
#include <cstdlib>
#include <stdexcept>
```

#### Componentes

- class `structures::LinkedQueue< T >`  
*Fila Encadeada.*

#### Namespaces

- `structures`  
*Estruturas de dados.*

#### 6.1.1 Descrição Detalhada

Código da fila encadeada.

#### Autores

Alisson Fabra da Silva e Eduardo Vinicius Betim.

#### Copyright

Copyright [2021] <Alisson Fabra da Silva, Eduardo Vinicius Betim>

## 6.2 Referência do Arquivo include/linked\_stack.h

Código da pilha encadeada.

```
#include <stdint>
#include <stdexcept>
```

### Componentes

- class `structures::LinkedStack< T >`  
*Pilha Encadeada.*

### Namespaces

- `structures`  
*Estruturas de dados.*

### 6.2.1 Descrição Detalhada

Código da pilha encadeada.

#### Autores

Alisson Fabra da Silva e Eduardo Vinicius Betim.

#### Copyright

Copyright [2021] <Alisson Fabra da Silva, Eduardo Vinicius Betim>

## 6.3 Referência do Arquivo include/matrix.h

Declaração da classe Matrix.

### Componentes

- class `xml_processing::Matrix`  
*Matriz.*

### Namespaces

- `xml_processing`  
*Código para processamento de xml.*

### 6.3.1 Descrição Detalhada

Declaração da classe Matrix.

#### Autores

Alisson Fabra da Silva e Eduardo Vinicius Betim.

#### Copyright

Copyright [2021] <Alisson Fabra da Silva, Eduardo Vinicius Betim>

## 6.4 Referência do Arquivo include/xml.h

Declaração das funções validação e extração de conteúdo de arquivos xml.

```
#include <string>
#include <cstdlib>
```

### Componentes

- class `xml_processing::XML`  
*Xml.*

### Namespaces

- `xml_processing`  
*Código para processamento de xml.*

### 6.4.1 Descrição Detalhada

Declaração das funções validação e extração de conteúdo de arquivos xml.

#### Autores

Alisson Fabra da Silva e Eduardo Vinicius Betim.

#### Copyright

Copyright [2021] <Alisson Fabra da Silva, Eduardo Vinicius Betim>

## 6.5 Referência do Arquivo main.cpp

Código principal.

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

### Funções

- `int main ()`

*Função main, ponto de partida para a execução do programa.*

#### 6.5.1 Descrição Detalhada

Código principal.

#### Autores

Alisson Fabra da Silva e Eduardo Vinicius Betim.

#### Copyright

Copyright [2021] <Alisson Fabra da Silva, Eduardo Vinicius Betim>

#### 6.5.2 Funções

##### 6.5.2.1 main()

```
int main ( )
```

Função main, ponto de partida para a execução do programa.

Recebe um arquivo xml com uma imagem binária no input, valida o arquivo e caso seja válido processa o imagem.

#### Retorna

Retona 0 caso não ocorra erro, -1 caso o xml não abra, -2 caso o xml seja inválido e -3 caso a imagem do xml tenha dimensão inválida.

## 6.6 Referência do Arquivo src/matrix.cpp

Implementação da classe Matrix.

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

### 6.6.1 Descrição Detalhada

Implementação da classe Matrix.

#### Autores

Alisson Fabra da Silva e Eduardo Vinicius Betim.

#### Copyright

Copyright [2021] <Alisson Fabra da Silva, Eduardo Vinicius Betim>

## 6.7 Referência do Arquivo src/xml.cpp

Implementação das funções validação e extração de arquivos XML.

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

### 6.7.1 Descrição Detalhada

Implementação das funções validação e extração de arquivos XML.

#### Autores

Alisson Fabra da Silva e Eduardo Vinicius Betim.

#### Copyright

Copyright [2021] <Alisson Fabra da Silva, Eduardo Vinicius Betim>



# Índice Remissivo

- ~LinkedList
  - structures::LinkedList, 10
- ~LinkedList
  - structures::LinkedList, 12
- ~Matrix
  - xml\_processing::Matrix, 15
- back
  - structures::LinkedList, 10
- clear
  - structures::LinkedList, 10
  - structures::LinkedList, 13
- count\_connected
  - xml\_processing::Matrix, 15
- dequeue
  - structures::LinkedList, 10
- empty
  - structures::LinkedList, 10
  - structures::LinkedList, 13
- enqueue
  - structures::LinkedList, 11
- extract
  - xml\_processing::XML, 17
- front
  - structures::LinkedList, 11
- get\_value
  - xml\_processing::Matrix, 15
- include/linked\_queue.h, 19
- include/linked\_stack.h, 20
- include/matrix.h, 20
- include/xml.h, 21
- LinkedList
  - structures::LinkedList, 10
- LinkedList
  - structures::LinkedList, 12
- main
  - main.cpp, 22
- main.cpp, 22
  - main, 22
- Matrix
  - xml\_processing::Matrix, 14
- pop
  - structures::LinkedList, 13
- push
  - structures::LinkedList, 13
- set\_value
  - xml\_processing::Matrix, 16
- size
  - structures::LinkedList, 11
  - structures::LinkedList, 13
- src/matrix.cpp, 23
- src/xml.cpp, 23
- structures, 7
- structures::LinkedList
  - ~LinkedList, 10
  - back, 10
  - clear, 10
  - dequeue, 10
  - empty, 10
  - enqueue, 11
  - front, 11
  - LinkedList, 10
  - size, 11
- structures::LinkedList< T >, 9
- structures::LinkedList
  - ~LinkedList, 12
  - clear, 13
  - empty, 13
  - LinkedList, 12
  - pop, 13
  - push, 13
  - size, 13
  - top, 13
- structures::LinkedList< T >, 11
- top
  - structures::LinkedList, 13
- validation
  - xml\_processing::XML, 17
- XML
  - xml\_processing::XML, 17
- xml\_processing, 7
- xml\_processing::Matrix, 14
  - ~Matrix, 15
  - count\_connected, 15
  - get\_value, 15
  - Matrix, 14
  - set\_value, 16
- xml\_processing::XML, 16

extract, [17](#)  
validation, [17](#)  
XML, [17](#)