Trabalho de Implementação II - identificação de prefixos e indexação de dicionários

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 documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 structures::Parser Class Reference

Public Member Functions

- Parser (std::string fileName)

  Construtor do Parser.
- std::string getDicText ()

  Armazenamento do texto do arquivo do dicionário.
- void parsing (std::string dicText, structures::Trie trie)

  Extração das palavras, posições e comprimentos.

Public Attributes

- std::string fileName

3.1.1 Detailed Description

Classe para o Parser.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Parser()

```
structures::Parser::Parser (
            std::string fileName )  [inline]
```

Construtor do Parser.

Parameters

fileName	Nome do arquivo do dicionário.

3.1.3 Member Function Documentation

3.1.3.1 getDicText()

```
std::string structures::Parser::getDicText ( )  [inline]
```

Armazenamento do texto do arquivo do dicionário.

Returns

O texto do dicionário.

3.1.3.2 parsing()

```
void structures::Parser::parsing (
            std::string dicText,
            structures::Trie trie )  [inline]
```

Extração das palavras, posições e comprimentos.

Primeiramente ocorrerá a extração da palavra através da procura pelos caracteres ']' (begin) e '[' (endWord). Posteriormente, será buscada o índice do caractere de quebra de linha (end). O comprimento será a diferença entre o índice begin e o end, enquanto que a posição será o próprio begin. A atualização do índice begin é realizada em saltos conforme o comprimento das linhas do dicionário. Por fim, a palavra será inserida na Trie.

Parameters

dicText	O texto do dicionário.
trie	A árvore de indexação.

3.1.4 Member Data Documentation

3.1.4.1 fileName

```
std::string structures::Parser::fileName
```

Nome do arquivo que o Parser realizará a análise,

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

- parser.cpp

3.2 structures::Trie Class Reference

Classe da Trie. Estrutura de Dados responsável pela indexação das palavras.

Public Member Functions

- Trie ()

  Construtor. A Trie é construída com a raiz já instanciada, não pode existir nenhum caractere na raiz.
- ~Trie ()=default
- void insert (std::string word, unsigned long position, unsigned long length)

  Inserção na Trie.
- bool isPrefix (std::string word)

  Busca ma palavra na Trie e verifica se é um prefixo ou uma palavra.

3.2.1 Detailed Description

Classe da Trie. Estrutura de Dados responsável pela indexação das palavras.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 ~Trie()

```
structures::Trie::~Trie ( )  [default]
```

Destrutor

3.2.3 Member Function Documentation

3.2.3.1 insert()

```
void structures::Trie::insert (
            std::string word,
            unsigned long position,
            unsigned long length )  [inline]
```

Inserção na Trie.

A inserção será por caractere da palavra recebida no parâmetro, sendo que a cada iteração o ponteiro it receberá o filho correspondente ao caractere da palavra para inserir no próximo corretamente. O último caractere da palavra receberá a posição e o comprimento dela, garantindo que ele é o final de uma palavra.

Parameters

word	A palavra que será inserida.
position	A posição em que se encontra o primeiro caractere dessa palavra no dicionário.
length	O comprimento do texto correspondente da palavra no dicionário.

3.2.3.2 isPrefix()

```
bool structures::Trie::isPrefix (
            std::string word )  [inline]
```

Busca ma palavra na Trie e verifica se é um prefixo ou uma palavra.

A busca será caractere por caractere da palavra, caso o nó atual da árvore tenha um ponteiro nulo para a letra selecionada durante a iteração, retornará falso, ou seja, a palavra desejada não é um prefixo para nenhuma das palavras do dicionário e enviará ao terminal "is not prefix" Caso contrário, percorrerá os filhos dos nós até chegar na última letra e verificará se é o final de uma palavra, através do método isEndOfWord(). Se retornar verdadeiro, é uma palavra e enviará para o terminal aposição e o comrprimento da palavra, caso contrário, "is prefix".

Parameters

word	A palavra que será buscada.

Returns

Se é um prefixo.

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

- trie.cpp

Chapter 4

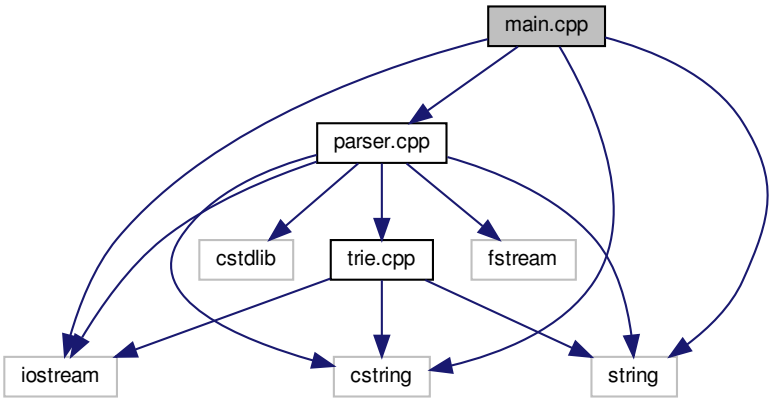
File Documentation

4.1 main.cpp File Reference

Código do main.

```
#include <iostream>
#include <string>
#include <cstring>
#include "parser.cpp"
```

Include dependency graph for main.cpp:



Functions

- int main ()

  Recebe o nome do arquivo como entrada, inicializa a trie e o parser e chama os métodos necessários para a realização correta do trabalho.

4.1.1 Detailed Description

Código do main.

Propósito: Indexação e recuperação eficiente de palavras em grandes arquivos de dicionários (mantidos em memória secundária).

Author

Maria Eduarda de Melo Hang

Date

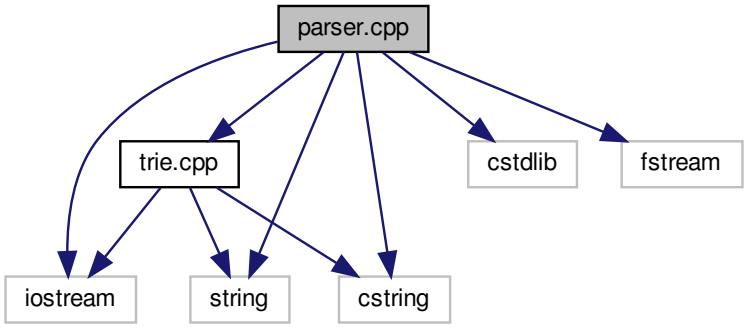
15 November 2018

4.2 parser.cpp File Reference

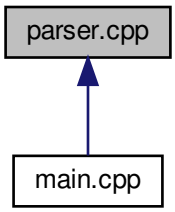
Parser dos arquivos dos dicionários.

```
#include <iostream>
#include <string>
#include <cstring>
#include <cstdlib>
#include <fstream>
#include "trie.cpp"
```

Include dependency graph for parser.cpp:



This graph shows which files directly or indirectly include this file:



Classes

- class structures::Parser

4.2.1 Detailed Description

Parser dos arquivos dos dicionários.

Author

Maria Eduarda de Melo Hang

Date

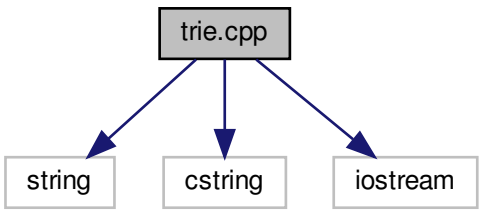
15 November 2018

4.3 trie.cpp File Reference

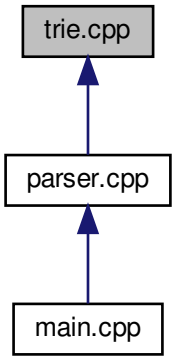
Árvore de Prefixos que suporta os caracteres de a até z como nodos. Não suporta acentos ou caracteres especiais para a inserção.

```
#include <string>
#include <cstring>
#include <iostream>
```

Include dependency graph for trie.cpp:



This graph shows which files directly or indirectly include this file:



Classes

- class structures::Trie

  Classe da Trie. Estrutura de Dados responsável pela indexação das palavras.

Macros

- #define **STRUCTURES_TRIE_H**
- #define K 26

4.3.1 Detailed Description

Árvore de Prefixos que suporta os caracteres de a até z como nodos. Não suporta acentos ou caracteres especiais para a inserção.

Author

Maria Eduarda de Melo Hang.

Date

15 November 2018.

4.3.2 Macro Definition Documentation

4.3.2.1 K

```
#define K 26
```

constante para guardar a quantidade de caracteres do alfabeto.

Index