

My Project

Generated by Doxygen 1.9.8

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 EnglishRussianDictionary Class Reference	5
3.1.1 Constructor & Destructor Documentation	5
3.1.1.1 ~EnglishRussianDictionary()	5
3.1.2 Member Function Documentation	6
3.1.2.1 AddNewPairOfWords()	6
3.1.2.2 ChangeTranslation()	6
3.1.2.3 CountNumberOfWorks()	6
3.1.2.4 DeleteEnglishWord()	6
3.1.2.5 DeleteEnglishWordNoChildren()	6
3.1.2.6 DeleteEnglishWordOneChild()	6
3.1.2.7 DeleteEnglishWordTwoChildren()	7
3.1.2.8 FindPerantElement()	7
3.1.2.9 FindPerentForNew()	7
3.1.2.10 IsContain()	7
3.1.2.11 ReadDictionaryFromFile()	7
3.2 PairOfWords Struct Reference	7
3.2.1 Detailed Description	8
4 File Documentation	9
4.1 Sample-Test2/EnglishRussianDictionary.cpp File Reference	9
4.1.1 Detailed Description	9
4.2 Sample-Test2/EnglishRussianDictionary.h File Reference	9
4.2.1 Detailed Description	9
4.3 EnglishRussianDictionary.h	10
4.4 pch.h	10
4.5 Sample-Test2/test.cpp File Reference	11
Index	13

Chapter 1

Class Index

1.1 Class List

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

EnglishRussianDictionary	5
PairOfWords	
Struct, thet contain information about nodes of the binar tree	7

Chapter 2

File Index

2.1 File List

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

Sample-Test2/ EnglishRussianDictionary.cpp	9
Sample-Test2/ EnglishRussianDictionary.h	9
Sample-Test2/ pch.h	10
Sample-Test2/ test.cpp	11

Chapter 3

Class Documentation

3.1 EnglishRussianDictionary Class Reference

Public Member Functions

- `~EnglishRussianDictionary ()`
- `bool IsContain (std::string english_word)`
- `bool AddNewPairOfWords (std::string english_word, std::string russian_translation)`
- `int CountNumberOfWords ()`
- `bool DeleteEnglishWord (std::string english_word)`
- `bool ChangeTranslation (std::string english_word, std::string russian_translation)`
- `std::string operator[] (std::string english_word)`
- `bool ReadDictionaryFromFile (EnglishRussianDictionary &new_dictionary, std::string filename)`

Protected Member Functions

- `void DeleteAll (PairOfWords *root)`
- `PairOfWords * FindPerentForNew (std::string english_word, PairOfWords *root)`
- `bool DeleteEnglishWordNoChildren (PairOfWords *root)`
- `bool DeleteEnglishWordOneChild (PairOfWords *root)`
- `bool DeleteEnglishWordTwoChildren (PairOfWords *root)`
- `PairOfWords * FindPerantElement (std::string english_word, PairOfWords *root)`
- `PairOfWords * FindElement (std::string english_word, PairOfWords *root)`
- `int CountNumberOfWordsLogic (PairOfWords *root)`

Protected Attributes

- `PairOfWords * english_russian_dictionary_tree_root_`

3.1.1 Constructor & Destructor Documentation

3.1.1.1 ~EnglishRussianDictionary()

`EnglishRussianDictionary::~EnglishRussianDictionary ()`

Clears the memory

3.1.2 Member Function Documentation

3.1.2.1 AddNewPairOfWords()

```
bool EnglishRussianDictionary::AddNewPairOfWords (
    std::string english_word,
    std::string russian_translation )
```

Adds the pair of words. Also checks if the added element already in tree

3.1.2.2 ChangeTranslation()

```
bool EnglishRussianDictionary::ChangeTranslation (
    std::string english_word,
    std::string russian_translation )
```

Changes the translation

3.1.2.3 CountNumberOfWords()

```
int EnglishRussianDictionary::CountNumberOfWords ( )
```

Counts number of words in dictionary

3.1.2.4 DeleteEnglishWord()

```
bool EnglishRussianDictionary::DeleteEnglishWord (
    std::string english_word )
```

Searches and deletes element

3.1.2.5 DeleteEnglishWordNoChildren()

```
bool EnglishRussianDictionary::DeleteEnglishWordNoChildren (
    PairOfWords * root ) [protected]
```

algorithm, that deletes element without children

3.1.2.6 DeleteEnglishWordOneChild()

```
bool EnglishRussianDictionary::DeleteEnglishWordOneChild (
    PairOfWords * root ) [protected]
```

algorithm, that deletes element with one child

3.1.2.7 DeleteEnglishWordTwoChildren()

```
bool EnglishRussianDictionary::DeleteEnglishWordTwoChildren (
    PairOfWords * root ) [protected]
```

algorithm, that deletes element with two children

3.1.2.8 FindPerantElement()

```
PairOfWords * EnglishRussianDictionary::FindPerantElement (
    std::string english_word,
    PairOfWords * root ) [protected]
```

searches for elemen, that is perent

3.1.2.9 FindPerentForNew()

```
PairOfWords * EnglishRussianDictionary::FindPerentForNew (
    std::string english_word,
    PairOfWords * root ) [protected]
```

Searches for plsce, were element could be instolen

3.1.2.10 IsContain()

```
bool EnglishRussianDictionary::IsContain (
    std::string english_word )
```

Searches for an element

3.1.2.11 ReadDictionaryFromFile()

```
bool EnglishRussianDictionary::ReadDictionaryFromFile (
    EnglishRussianDictionary & new_dictionary,
    std::string filename )
```

Reads information from file

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

- Sample-Test2/ **EnglishRussianDictionary.h**
- Sample-Test2/ **EnglishRussianDictionary.cpp**

3.2 PairOfWords Struct Reference

Struct, that contain information about nodes of the binar tree.

```
#include <EnglishRussianDictionary.h>
```

Public Attributes

- `std::string english_word_`
It's english word, that is used like key.
- `std::string russian_tranlation_`
It's the tranlsition of the english word.
- `PairOfWords * right_child_ = NULL`
It's the pointer to one of 2 children.
- `PairOfWords * left_child_ = NULL`
It's the pointer to one of 2 children.

3.2.1 Detailed Description

Struct, that contain information about nodes of the binar tree.

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

- Sample-Test2/ **EnglishRussianDictionary.h**

Chapter 4

File Documentation

4.1 Sample-Test2/EnglishRussianDictionary.cpp File Reference

```
#include "EnglishRussianDictionary.h"  
#include <string>
```

4.1.1 Detailed Description

File, that contains all information about realization of main class **EnglishRussianDictionary** (p. 5)

4.2 Sample-Test2/EnglishRussianDictionary.h File Reference

```
#include <string>  
#include <fstream>  
#include "gtest/gtest.h"
```

Classes

- struct **PairOfWords**
Struct, that contains information about nodes of the binary tree.
- class **EnglishRussianDictionary**

4.2.1 Detailed Description

This file contains prototypes of main Class

4.3 EnglishRussianDictionary.h

Go to the documentation of this file.

```

00001
00007 #pragma once
00008 #include <string>
00009 #include <fstream>
00010 #include "gtest/gtest.h"
00011 // #include
00012 // #include
00013 "D:\LW1\packages\Microsoft.googletest.v140.windesktop.msvcstl.static.rt-dyn.1.8.1.6\build\native\include\gtest\gtest.h"
00012 // #include
00013 "D:\LW1\packages\Microsoft.googletest.v140.windesktop.msvcstl.static.rt-dyn.1.8.1.6\build\native\include\gtest\gtest.h"
00013
00015 struct PairOfWords
00016 {
00017     std::string english_word_;
00018     std::string russian_tranlation_;
00019     PairOfWords* right_child_ = NULL;
00020     PairOfWords* left_child_ = NULL;
00021 };
00022
00023
00024 class EnglishRussianDictionary
00025 {
00026 protected:
00027     PairOfWords* english_russian_dictionary_tree_root_;
00028
00029     void DeleteAll(PairOfWords* root); //done
00030
00031     PairOfWords* FindPerentForNew(std::string english_word, PairOfWords* root); //done
00032
00033     bool DeleteEnglishWordNoChildren(PairOfWords* root);
00034
00035     bool DeleteEnglishWordOneChild(PairOfWords* root);
00036
00037     bool DeleteEnglishWordTwoChildren(PairOfWords* root);
00038
00039     PairOfWords* FindPerantElement(std::string english_word, PairOfWords* root);
00040
00041     PairOfWords* FindElement(std::string english_word, PairOfWords* root); //done
00042
00043     int CountNamberOfWordsLogic(PairOfWords* root); //done
00044
00045 public:
00046     EnglishRussianDictionary(); //done
00047
00048     ~EnglishRussianDictionary(); //done
00049
00050     bool IsContain(std::string english_word); //done
00051
00052     bool AddNewPairOfWords(std::string english_word, std::string russian_translation); //done
00053
00054     int CountNamberOfWords(); //done
00055
00056     bool DeleteEnglishWord(std::string english_word);
00057
00058     bool ChangeTranslation(std::string english_word, std::string russian_translation); //done
00059
00060     std::string operator[](std::string english_word); //done
00061
00062     bool ReadDictionaryFromFile(EnglishRussianDictionary& new_dictionary, std::string filename); //done
00063 };
00064
00065

```

4.4 pch.h

```

00001 //
00002 // pch.h
00003 //
00004
00005 #pragma once
00006
00007 #include "gtest/gtest.h"

```

4.5 Sample-Test2/test.cpp File Reference

```
#include "pch.h"  
#include "EnglishRussianDictionary.h"
```

Functions

- **TEST** (TestCaseName, TestName)
- **TEST** (DictionaryTest, FindElementTest)
- **TEST** (DictionaryTest, DeleteElementTest)
- **TEST** (DictionaryTest, AddNewTest)
- **TEST** (DictionaryTest, CountNamberTest)
- **TEST** (DictionaryTest, ReadDictionaryFromFileTest)

Index

- ~EnglishRussianDictionary
 - EnglishRussianDictionary, 5
- AddNewPairOfWords
 - EnglishRussianDictionary, 6
- ChangeTranslation
 - EnglishRussianDictionary, 6
- CountNumberOfWords
 - EnglishRussianDictionary, 6
- DeleteEnglishWord
 - EnglishRussianDictionary, 6
- DeleteEnglishWordNoChildren
 - EnglishRussianDictionary, 6
- DeleteEnglishWordOneChild
 - EnglishRussianDictionary, 6
- DeleteEnglishWordTwoChildren
 - EnglishRussianDictionary, 6
- EnglishRussianDictionary, 5
 - ~EnglishRussianDictionary, 5
 - AddNewPairOfWords, 6
 - ChangeTranslation, 6
 - CountNumberOfWords, 6
 - DeleteEnglishWord, 6
 - DeleteEnglishWordNoChildren, 6
 - DeleteEnglishWordOneChild, 6
 - DeleteEnglishWordTwoChildren, 6
 - FindPerantElement, 7
 - FindPerentForNew, 7
 - IsContain, 7
 - ReadDictionaryFromFile, 7
- FindPerantElement
 - EnglishRussianDictionary, 7
- FindPerentForNew
 - EnglishRussianDictionary, 7
- IsContain
 - EnglishRussianDictionary, 7
- PairOfWords, 7
- ReadDictionaryFromFile
 - EnglishRussianDictionary, 7
- Sample-Test2/EnglishRussianDictionary.cpp, 9
- Sample-Test2/EnglishRussianDictionary.h, 9, 10
- Sample-Test2/pch.h, 10
- Sample-Test2/test.cpp, 11