

Filtr przedrostkowy (PK2 Projekt)

Wygenerowano przez Doxygen 1.8.17

1 Indeks klas	1
1.1 Lista klas	1
2 Indeks plików	3
2.1 Lista plików	3
3 Dokumentacja klas	5
3.1 Dokumentacja struktury BufferedFileReader	5
3.1.1 Opis szczegółowy	5
3.1.2 Dokumentacja atrybutów składowych	5
3.1.2.1 buf	5
3.1.2.2 file	6
3.1.2.3 readCount	6
3.2 Dokumentacja struktury LaunchConfig	6
3.2.1 Opis szczegółowy	6
3.2.2 Dokumentacja atrybutów składowych	6
3.2.2.1 inFilePath	6
3.2.2.2 outFilePath	7
3.2.2.3 prefixFilePath	7
3.3 Dokumentacja struktury PairOfStrings	7
3.3.1 Opis szczegółowy	7
3.3.2 Dokumentacja atrybutów składowych	7
3.3.2.1 left	7
3.3.2.2 right	8
3.4 Dokumentacja struktury PrefixTree	8
3.4.1 Opis szczegółowy	8
3.4.2 Dokumentacja atrybutów składowych	8
3.4.2.1 children	8
3.4.2.2 isLeaf	8
3.4.2.3 prefix	9
3.5 Dokumentacja struktury String	9
3.5.1 Opis szczegółowy	9
3.5.2 Dokumentacja atrybutów składowych	9
3.5.2.1 capacity	9
3.5.2.2 data	9
3.5.2.3 len	9
4 Dokumentacja plików	11
4.1 Dokumentacja pliku include/Filter/App.h	11
4.1.1 Dokumentacja funkcji	11
4.1.1.1 run()	11
4.2 Dokumentacja pliku include/Filter/BufferedReader.h	12
4.2.1 Dokumentacja definicji	12

4.2.1.1 BFR_BUF_SIZE	12
4.2.2 Dokumentacja definicji typów	12
4.2.2.1 BufferedFileReader	13
4.2.3 Dokumentacja funkcji	13
4.2.3.1 bfrClose()	13
4.2.3.2 bfrOpen()	13
4.2.3.3 bfrReadUntilWs()	13
4.3 Dokumentacja pliku include/Filter/Filter.h	14
4.3.1 Dokumentacja definicji typów	14
4.3.1.1 PrefixTree	14
4.3.2 Dokumentacja funkcji	14
4.3.2.1 filterInputFile()	14
4.4 Dokumentacja pliku include/Filter/ParseArgs.h	15
4.4.1 Dokumentacja definicji typów	15
4.4.1.1 LaunchConfig	15
4.4.2 Dokumentacja funkcji	15
4.4.2.1 parseArgs()	15
4.5 Dokumentacja pliku include/Filter/PrefixTree.h	16
4.5.1 Dokumentacja definicji	16
4.5.1.1 CHAR_COMBINATIONS	16
4.5.2 Dokumentacja definicji typów	17
4.5.2.1 PrefixTree	17
4.5.3 Dokumentacja funkcji	17
4.5.3.1 buildPrefixTree()	17
4.5.3.2 destroyPrefixTree()	17
4.5.3.3 prefixFilter()	18
4.6 Dokumentacja pliku include/Filter/String.h	18
4.6.1 Dokumentacja definicji typów	19
4.6.1.1 PairOfStrings	19
4.6.1.2 String	19
4.6.2 Dokumentacja funkcji	19
4.6.2.1 appendString()	19
4.6.2.2 appendStringRaw()	19
4.6.2.3 destroyString()	20
4.6.2.4 findNonWsInString()	20
4.6.2.5 findWsInString()	20
4.6.2.6 makeString()	21
4.6.2.7 makeStringWith()	21
4.6.2.8 reserveStringCapacity()	21
4.6.2.9 shrinkStringToFit()	23
4.6.2.10 splitString()	23
4.6.2.11 wrapWithString()	23

4.7 Dokumentacja pliku src/App.c	24
4.7.1 Dokumentacja funkcji	24
4.7.1.1 run()	24
4.8 Dokumentacja pliku src/BufferedReader.c	24
4.8.1 Dokumentacja funkcji	25
4.8.1.1 bfrClose()	25
4.8.1.2 bfrConsume()	25
4.8.1.3 bfrOpen()	26
4.8.1.4 bfrReadNext()	26
4.8.1.5 bfrReadUntilWs()	26
4.9 Dokumentacja pliku src/Filter.c	27
4.9.1 Dokumentacja definicji	27
4.9.1.1 OUTPUT_BUF_SIZE	27
4.9.2 Dokumentacja funkcji	27
4.9.2.1 filterInputFile()	27
4.10 Dokumentacja pliku src/Main.c	28
4.10.1 Dokumentacja funkcji	28
4.10.1.1 main()	28
4.11 Dokumentacja pliku src/ParseArgs.c	28
4.11.1 Dokumentacja funkcji	29
4.11.1.1 parseArgs()	29
4.12 Dokumentacja pliku src/PrefixTree.c	29
4.12.1 Dokumentacja funkcji	30
4.12.1.1 buildPrefixTree()	30
4.12.1.2 destroyPrefixTree()	30
4.12.1.3 initPrefixTreeNode()	30
4.12.1.4 initPrefixTreeNodeWith()	30
4.12.1.5 insertPrefixIntoTree()	31
4.12.1.6 makePrefixTree()	31
4.12.1.7 prefixFilter()	31
4.12.1.8 shouldContinueDown()	32
4.13 Dokumentacja pliku src/String.c	32
4.13.1 Dokumentacja funkcji	32
4.13.1.1 appendString()	32
4.13.1.2 appendStringRaw()	33
4.13.1.3 destroyString()	33
4.13.1.4 findNonWsInString()	33
4.13.1.5 findWsInString()	34
4.13.1.6 makeString()	34
4.13.1.7 makeStringWith()	34
4.13.1.8 reserveStringCapacity()	35
4.13.1.9 shrinkStringToFit()	35

4.13.1.10 <code>splitString()</code>	35
4.13.1.11 <code>wrapWithString()</code>	36

Indeks	37
---------------	-----------

Rozdział 1

Indeks klas

1.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

BufferedReader	5
LaunchConfig	6
PairOfStrings	7
PrefixTree	8
String	9

Rozdział 2

Indeks plików

2.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

include/Filter/ App.h	11
include/Filter/ BufferedReader.h	12
include/Filter/ Filter.h	14
include/Filter/ ParseArgs.h	15
include/Filter/ PrefixTree.h	16
include/Filter/ String.h	18
src/ App.c	24
src/ BufferedReader.c	24
src/ Filter.c	27
src/ Main.c	28
src/ ParseArgs.c	28
src/ PrefixTree.c	29
src/ String.c	32

Rozdział 3

Dokumentacja klas

3.1 Dokumentacja struktury BufferedFileReader

```
#include <BufferedReader.h>
```

Atrybuty publiczne

- **FILE * file**
Uchwyt do pliku.
- **char buf [BFR_BUF_SIZE]**
Bufor do wczytywania danych.
- **size_t readCount**
Ilość dostępnych znaków w buforze.

3.1.1 Opis szczegółowy

Buforowany wczytywacz (reader) danych z pliku. Wczytuje dane z pliku porcjami w celu zwiększenia wydajności.

3.1.2 Dokumentacja atrybutów składowych

3.1.2.1 buf

```
char BufferedFileReader::buf[ BFR_BUF_SIZE]
```

Bufor do wczytywania danych.

3.1.2.2 file

```
FILE* BufferedFileReader::file
```

Uchwyt do pliku.

3.1.2.3 readCount

```
size_t BufferedFileReader::readCount
```

Ilość dostępnych znaków w buforze.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/Filter/ **BufferedReader.h**

3.2 Dokumentacja struktury LaunchConfig

```
#include <ParseArgs.h>
```

Atrybuty publiczne

- char **inFilePath** [FILENAME_MAX]
ścieżka do pliku wejściowego.
- char **prefixFilePath** [FILENAME_MAX]
ścieżka do pliku z przedrostkami.
- char **outFilePath** [FILENAME_MAX]
ścieżka do pliku wyjściowego.

3.2.1 Opis szczegółowy

Informacje o konfiguracji uruchomieniowej programu.

3.2.2 Dokumentacja atrybutów składowych

3.2.2.1 inFilePath

```
char LaunchConfig::inFilePath[FILENAME_MAX]
```

ścieżka do pliku wejściowego.

3.2.2.2 outFilePath

```
char LaunchConfig::outFilePath[FILENAME_MAX]
```

ścieżka do pliku wyjściowego.

3.2.2.3 prefixFilePath

```
char LaunchConfig::prefixFilePath[FILENAME_MAX]
```

ścieżka do pliku z przedrostkami.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/Filter/ **ParseArgs.h**

3.3 Dokumentacja struktury PairOfStrings

```
#include <String.h>
```

Atrybuty publiczne

- **String left**
- **String right**

3.3.1 Opis szczegółowy

Para napisów.

3.3.2 Dokumentacja atrybutów składowych

3.3.2.1 left

```
String PairOfStrings::left
```

3.3.2.2 right

```
String PairOfStrings::right
```

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/Filter/ **String.h**

3.4 Dokumentacja struktury PrefixTree

```
#include <PrefixTree.h>
```

Atrybuty publiczne

- struct **PrefixTree** * **children** [**CHAR_COMBINATIONS**]
Dzieci węzła.
- **String** **prefix**
Zawartość (część prefixu) węzła.
- bool **isLeaf**
Czy aktualny węzeł jest liściem drzewa?

3.4.1 Opis szczegółowy

Węzeł drzewa prefixowego.

3.4.2 Dokumentacja atrybutów składowych

3.4.2.1 children

```
struct PrefixTree* PrefixTree::children[ CHAR_COMBINATIONS]
```

Dzieci węzła.

3.4.2.2 isLeaf

```
bool PrefixTree::isLeaf
```

Czy aktualny węzeł jest liściem drzewa?

3.4.2.3 prefix

```
String PrefixTree::prefix
```

Zawartość (część prefixu) węzła.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/Filter/ **PrefixTree.h**

3.5 Dokumentacja struktury String

```
#include <String.h>
```

Atrybuty publiczne

- char * **data**
Wskaźnik na początek znaków w napisie.
- size_t **len**
Długość napisu.
- size_t **capacity**
Długość zarezerwowanej pamięci dla napisu (zawsze \geq len).

3.5.1 Opis szczegółowy

Struktura do zarządzania napisami

3.5.2 Dokumentacja atrybutów składowych

3.5.2.1 capacity

```
size_t String::capacity
```

Długość zarezerwowanej pamięci dla napisu (zawsze \geq len).

3.5.2.2 data

```
char* String::data
```

Wskaźnik na początek znaków w napisie.

3.5.2.3 len

```
size_t String::len
```

Długość napisu.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/Filter/ **String.h**

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku include/Filter/App.h

```
#include <Filter/ParseArgs.h>
#include <stdbool.h>
```

Funkcje

- bool **run** (**LaunchConfig** *launchCfg_)

4.1.1 Dokumentacja funkcji

4.1.1.1 run()

```
bool run (
    LaunchConfig * launchCfg_ )
```

Uruchamia aplikację z odpowiednim ustawieniem.

Parametry

<i>launchCfg_</i>	ustawienie aplikacji
-------------------	----------------------

Zwraca

true jeśli wszystko przebiegło pomyślnie, w przeciwnym wypadku false

4.2 Dokumentacja pliku include/Filter/BufferedReader.h

```
#include <Filter/String.h>
#include <stdio.h>
#include <stdbool.h>
```

Komponenty

- struct **BufferedReader**

Definicje

- #define **BFR_BUF_SIZE** 1024 * 1024
*Rozmiar bloku danych wczytywanego przez **BufferedReader** (str. 5).*

Definicje typów

- typedef struct **BufferedReader** **BufferedReader**

Funkcje

- **BufferedReader** * **bfrOpen** (const char *path_)
- **String** **bfrReadUntilWs** (**BufferedReader** *reader_)
- void **bfrClose** (**BufferedReader** *reader_)

4.2.1 Dokumentacja definicji

4.2.1.1 BFR_BUF_SIZE

```
#define BFR_BUF_SIZE 1024 * 1024
```

Rozmiar bloku danych wczytywanego przez **BufferedReader** (str. 5).

4.2.2 Dokumentacja definicji typów

4.2.2.1 BufferedReader

```
typedef struct BufferedReader BufferedReader
```

Buforowany wczytywacz (reader) danych z pliku. Wczytuje dane z pliku porcjami w celu zwiększenia wydajności.

4.2.3 Dokumentacja funkcji

4.2.3.1 bfrClose()

```
void bfrClose (  
    BufferedReader * reader_ )
```

Usuwa z pamięci reader oraz zamyka plik.

Parametry

<i>reader</i> ↔	wskaźnik na reader
—	

4.2.3.2 bfrOpen()

```
BufferedReader* bfrOpen (  
    const char * path_ )
```

Otwiera buforowany reader wraz z danym plikiem.

Parametry

<i>path</i> ↔	ścieżka do pliku
—	

Zwraca

wskaźnik na dynamicznie stworzony reader.

4.2.3.3 bfrReadUntilWs()

```
String bfrReadUntilWs (  
    BufferedReader * reader_ )
```

Wczytuje dane do napotkania białego znaku.

Parametry

<i>reader</i> ↔	wskaźnik na reader
—	

Zwraca

Ciąg wczytanych znaków/danych, lub pusty **String** (str. 9), jeśli się nie udało wczytać.

4.3 Dokumentacja pliku include/Filter/Filter.h

```
#include <stdbool.h>
```

Definicje typów

- typedef struct **PrefixTree** **PrefixTree**

Funkcje

- bool **filterInputFile** (char const *inFilePath_, char const *outFilePath_, **PrefixTree** const *prefixTree_)

4.3.1 Dokumentacja definicji typów

4.3.1.1 PrefixTree

```
typedef struct PrefixTree PrefixTree
```

4.3.2 Dokumentacja funkcji

4.3.2.1 filterInputFile()

```
bool filterInputFile (  
    char const * inFilePath_,  
    char const * outFilePath_,  
    PrefixTree const * prefixTree_ )
```

Filtruje plik źródłowy po przedrostkach zawartych w drzewie przedrostkowym. Zapisuje przefiltrowane dane do osobnego pliku.

Parametry

<i>inFilePath</i> ↔ —	ścieżka do pliku wejściowego
<i>outFilePath</i> ↔ —	ścieżka do pliku wyjściowego
<i>prefixTree</i> ↔ —	wskaźnik na korzeń drzewa przedrostkowego

4.4 Dokumentacja pliku include/Filter/ParseArgs.h

```
#include <stdio.h>
#include <stdbool.h>
```

Komponenty

- struct **LaunchConfig**

Definicje typów

- typedef struct **LaunchConfig** **LaunchConfig**

Funkcje

- bool **parseArgs** (int argc_, char *argv_[], **LaunchConfig** *config_)

4.4.1 Dokumentacja definicji typów

4.4.1.1 LaunchConfig

```
typedef struct LaunchConfig LaunchConfig
```

Informacje o konfiguracji uruchomieniowej programu.

4.4.2 Dokumentacja funkcji

4.4.2.1 parseArgs()

```
bool parseArgs (
    int argc_,
    char * argv_[],
    LaunchConfig * config_ )
```

Wypełnia ustawienie programu na podstawie argumentów.

Parametry

<i>argc</i> ↔ —	ilość argumentów
<i>argv</i> ↔ —	zawartość argumentów
<i>config</i> ↔ —	wskaźnik na ustawienie programu, do wypełnienia.

4.5 Dokumentacja pliku include/Filter/PrefixTree.h

```
#include <Filter/String.h>
#include <stdio.h>
#include <stdbool.h>
```

Komponenty

- struct **PrefixTree**

Definicje

- #define **CHAR_COMBINATIONS** 256

Definicje typów

- typedef struct **PrefixTree** **PrefixTree**

Funkcje

- **PrefixTree** * **buildPrefixTree** (const char *prefixFilePath_)
- void **destroyPrefixTree** (**PrefixTree** *prefixTree_)
- bool **prefixFilter** (**PrefixTree** const *root_, **String** str_)

4.5.1 Dokumentacja definicji

4.5.1.1 CHAR_COMBINATIONS

```
#define CHAR_COMBINATIONS 256
```

4.5.2 Dokumentacja definicji typów

4.5.2.1 PrefixTree

```
typedef struct PrefixTree PrefixTree
```

Węzeł drzewa prefixowego.

4.5.3 Dokumentacja funkcji

4.5.3.1 buildPrefixTree()

```
PrefixTree* buildPrefixTree (
    const char * prefixFilePath_ )
```

Buduje drzewo przedrostkowe z wyrazów wczytanych z pliku.

Parametry

<i>prefixFile↔ Path_</i>	ścieżka do pliku z wyrazami
------------------------------	-----------------------------

Zwraca

Stworzone dynamicznie drzewo.

4.5.3.2 destroyPrefixTree()

```
void destroyPrefixTree (
    PrefixTree * prefixTree_ )
```

Zwalnia z pamięci zawartość drzewa. Nie zwalnia korzenia!

Parametry

<i>prefix↔ Tree_</i>	korzeń drzewa przedrostkowego.
--------------------------	--------------------------------

4.5.3.3 prefixFilter()

```
bool prefixFilter (
    PrefixTree const * root_,
    String str_ )
```

Sprawdza, czy wyraz powinien zostać zaakceptowany przez filtr przedrostkowy.

Parametry

<i>root</i> ↔ —	korzeń drzewa przedrostkowego
<i>str</i> ↔ —	wyraz do sprawdzenia

Zwraca

true jeśli wyraz posiada przedrostek znajdujący się w drzewie, w przeciwnym wypadku false.

4.6 Dokumentacja pliku include/Filter/String.h

```
#include <stdlib.h>
#include <inttypes.h>
```

Komponenty

- struct **String**
- struct **PairOfStrings**

Definicje typów

- typedef struct **String** **String**
- typedef struct **PairOfStrings** **PairOfStrings**

Funkcje

- **String** **makeString** ()
- **String** **makeStringWith** (char const *bytes_, size_t len_)
- **String** **wrapWithString** (char *bytes_, size_t len_)
- void **destroyString** (**String** *str_)
- int64_t **findWsInString** (**String** const *str_, size_t startPos_)
- int64_t **findNonWsInString** (**String** const *str_, size_t startPos_)
- **PairOfStrings** **splitString** (**String** *src_, size_t where_)
- void **appendString** (**String** *str_, **String** const *toAppend_)
- void **appendStringRaw** (**String** *str_, char const *bytes_, size_t len_)
- void **reserveStringCapacity** (**String** *str_, size_t newCapacity_)
- void **shrinkStringToFit** (**String** *str_)

4.6.1 Dokumentacja definicji typów

4.6.1.1 PairOfStrings

```
typedef struct PairOfStrings PairOfStrings
```

Para napisów.

4.6.1.2 String

```
typedef struct String String
```

Struktura do zarządzania napisami

4.6.2 Dokumentacja funkcji

4.6.2.1 appendString()

```
void appendString (
    String * str_,
    String const * toAppend_ )
```

Dopisuje napis na koniec drugiego.

Parametry

<i>str_</i>	napis, do którego zostanie dopisany inny
<i>toAppend_</i>	dopisywany napis
—	

4.6.2.2 appendStringRaw()

```
void appendStringRaw (
    String * str_,
    char const * bytes_,
    size_t len_ )
```

Dopisuje napis na koniec drugiego (wersja dla czystego C-stringa)

Parametry

<i>str_</i>	napis, do którego zostanie dopisany inny
<i>bytes_↔</i>	początek napisu do dopisania
<i>len_</i>	ilość znaków do dopisania

4.6.2.3 `destroyString()`

```
void destroyString (
    String * str_ )
```

Usuwa zawartość istniejącego obiektu **String** (str. 9). Nie usuwa przekazanego obiektu z pamięci!

Parametry

<i>str_↔</i>	String (str. 9), z którego zostanie zwolniona pamięć
<i>—</i>	

4.6.2.4 `findNonWsInString()`

```
int64_t findNonWsInString (
    String const * str_,
    size_t startPos_ )
```

Znajduje indeks pierwszego elementu nie będącego białym znakiem w napisie.

Parametry

<i>str_</i>	rozpatrywany napis
<i>start_↔</i> <i>Pos_</i>	pozycja, od której rozpoczynamy wyszukiwanie

Zwraca

pozycja znalezionego znaku, lub -1 gdy nie znaleziono.

4.6.2.5 `findWsInString()`

```
int64_t findWsInString (
    String const * str_,
    size_t startPos_ )
```

Znajduje indeks pierwszego białego znaku w napisie.

Parametry

<i>str_</i>	rozpatrywany napis
<i>start</i> ↔ <i>Pos_</i>	pozycja, od której rozpoczynamy wyszukiwanie

Zwraca

pozycja znalezionego białego znaku, lub -1 gdy nie znaleziono.

4.6.2.6 makeString()

```
String makeString ( )
```

Tworzy pusty **String** (str. 9).

Zwraca

pusty **String** (str. 9).

4.6.2.7 makeStringWith()

```
String makeStringWith (
    char const * bytes_,
    size_t len_ )
```

Tworzy **String** (str. 9) kopiując inny istniejący.

Parametry

<i>bytes</i> ↔ _	źródło do kopiowania danych
<i>len_</i>	ilość bajtów do skopiowania

Zwraca

nowo utworzony **String** (str. 9).

4.6.2.8 reserveStringCapacity()

```
void reserveStringCapacity (
    String * str_,
    size_t newCapacity_ )
```

Rezerwuje pamięć wewnątrz Stringa. Nigdy nie zmniejsza zaalokowanej pamięci.

Parametry

<i>str_</i>	napis, któremu rezerwujemy pamięć
<i>new↔ Capacity_</i>	nowa pojemność pamięci.

4.6.2.9 shrinkStringToFit()

```
void shrinkStringToFit (
    String * str_ )
```

Ucina zarezerwowana pamięć do ilości znaków faktycznie używanych przez Stringa.

Parametry

<i>str_↔ _</i>	modyfikowany napis
--------------------	--------------------

4.6.2.10 splitString()

```
PairOfStrings splitString (
    String * src_,
    size_t where_ )
```

Rozbija napis na dwa osobne napisy w następujący sposób: lewy zawiera indeksy [0, where_), prawy zawiera indeksy [where_, length). Nie zwalnia rozbijanego zapisu z pamięci!

Parametry

<i>src_</i>	rozbijany napis
<i>where_↔ _</i>	pozycja do rozbicia napisu

Zwraca

struktura zawierająca lewą i prawą część napisu

4.6.2.11 wrapWithString()

```
String wrapWithString (
    char * bytes_,
    size_t len_ )
```

Tworzy **String** (str. 9) opakowując istniejące dane.

Parametry

<i>bytes</i> ↔ —	źródło danych
<i>len</i> —	ilość bajtów

Zwraca

nowo utworzony **String** (str. 9) (opakowujący istniejące dane).

4.7 Dokumentacja pliku src/App.c

```
#include <Filter/App.h>
#include <Filter/PrefixTree.h>
#include <Filter/Filter.h>
#include <stdlib.h>
#include <assert.h>
```

Funkcje

- `bool run (LaunchConfig *launchCfg_)`

4.7.1 Dokumentacja funkcji**4.7.1.1 run()**

```
bool run (
    LaunchConfig * launchCfg_ )
```

Uruchamia aplikację z odpowiednim ustawieniem.

Parametry

<i>launch</i> ↔ <i>Cfg_</i>	ustawienie aplikacji
--------------------------------	----------------------

Zwraca

true jeśli wszystko przebiegło pomyślnie, w przeciwnym wypadku false

4.8 Dokumentacja pliku src/BufferedReader.c

```
#include <Filter/BufferedReader.h>
```

```
#include <stdlib.h>
#include <string.h>
#include <assert.h>
```

Funkcje

- bool **bfrReadNext** (**BufferedReader** *reader_)
- void **bfrConsume** (**BufferedReader** *reader_, size_t count_)
- **BufferedReader** * **bfrOpen** (const char *path_)
- String **bfrReadUntilWs** (**BufferedReader** *reader_)
- void **bfrClose** (**BufferedReader** *reader_)

4.8.1 Dokumentacja funkcji

4.8.1.1 bfrClose()

```
void bfrClose (
    BufferedReader * reader_ )
```

Usuwa z pamięci reader oraz zamyka plik.

Parametry

<i>reader</i> ↔ —	wskaźnik na reader
----------------------	--------------------

4.8.1.2 bfrConsume()

```
void bfrConsume (
    BufferedReader * reader_,
    size_t count_ )
```

Usuwa z bufora określoną liczbę znaków.

Parametry

<i>reader</i> ↔ —	wskaźnik na reader
<i>count</i> ↔ —	liczba znaków

4.8.1.3 bfrOpen()

```
BufferedReader* bfrOpen (
    const char * path_ )
```

Otwiera buforowany reader wraz z danym plikiem.

Parametry

<i>path_</i> ↔	ścieżka do pliku
—	

Zwraca

wskaźnik na dynamicznie stworzony reader.

4.8.1.4 bfrReadNext()

```
bool bfrReadNext (
    BufferedReader * reader_ )
```

Wczytuje następny blok danych. Rozmiar bloku jest określany makrem

Zobacz również

BFR_BUF_SIZE (str. 12)

Parametry

<i>reader_</i> ↔	wskaźnik na reader
—	

Zwraca

true jeśli udało się wczytać BFR_BUF_SIZE bajtów, w innym wypadku false.

4.8.1.5 bfrReadUntilWs()

```
String bfrReadUntilWs (
    BufferedReader * reader_ )
```

Wczytuje dane do napotkania białego znaku.

Parametry

<i>reader</i> ↔	wskaźnik na reader
—	

Zwraca

Ciąg wczytanych znaków/danych, lub pusty **String** (str. 9), jeśli się nie udało wczytać.

4.9 Dokumentacja pliku src/Filter.c

```
#include <Filter/Filter.h>
#include <Filter/PrefixTree.h>
#include <Filter/BufferedReader.h>
#include <string.h>
#include <assert.h>
```

Definicje

- `#define OUTPUT_BUF_SIZE 1024 * 1024`

Funkcje

- `bool filterInputFile (char const *inFilePath_, char const *outFilePath_, PrefixTree const *prefixTree_)`

4.9.1 Dokumentacja definicji

4.9.1.1 OUTPUT_BUF_SIZE

```
#define OUTPUT_BUF_SIZE 1024 * 1024
```

Rozmiar bufora dla pliku wyjściowego

4.9.2 Dokumentacja funkcji

4.9.2.1 filterInputFile()

```
bool filterInputFile (
    char const * inFilePath_,
    char const * outFilePath_,
    PrefixTree const * prefixTree_ )
```

Filtruje plik źródłowy po przedrostkach zawartych w drzewie przedrostkowym. Zapisuje przefiltrowane dane do osobnego pliku.

Parametry

<i>inFilePath</i> ↔ —	ścieżka do pliku wejściowego
<i>outFilePath</i> ↔ —	ścieżka do pliku wyjściowego
<i>prefixTree</i> ↔ —	wskaźnik na korzeń drzewa przedrostkowego

4.10 Dokumentacja pliku src/Main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <Filter/ParseArgs.h>
#include <Filter/App.h>
```

Funkcje

- int **main** (int argc, char *argv[])

4.10.1 Dokumentacja funkcji

4.10.1.1 main()

```
int main (
    int argc,
    char * argv[] )
```

Funkcja główna programu. Parsuje argumenty i uruchamia właściwą logikę programu.

Parametry

<i>argc</i>	ilość argumentów
<i>argv</i>	zawartość argumentów

4.11 Dokumentacja pliku src/ParseArgs.c

```
#include <Filter/ParseArgs.h>
#include <assert.h>
#include <string.h>
```

Funkcje

- bool **parseArgs** (int argc, char *argv[], **LaunchConfig** *config_)

4.11.1 Dokumentacja funkcji

4.11.1.1 parseArgs()

```
bool parseArgs (
    int argc_,
    char * argv_[],
    LaunchConfig * config_ )
```

Wypełnia ustawienie programu na podstawie argumentów.

Parametry

<i>argc</i> ↔ —	ilość argumentów
<i>argv</i> ↔ —	zawartość argumentów
<i>config</i> ↔ —	wskaźnik na ustawienie programu, do wypełnienia.

4.12 Dokumentacja pliku src/PrefixTree.c

```
#include <Filter/PrefixTree.h>
#include <Filter/BufferedReader.h>
#include <stdlib.h>
#include <string.h>
#include <assert.h>
```

Funkcje

- **PrefixTree** * **makePrefixTree** ()
- void **insertPrefixIntoTree** (**PrefixTree** *root_, **String** prefix_)
- bool **shouldContinueDown** (**PrefixTree** *node_, **String** prefix_, size_t *startCharacter_)
- void **initPrefixTreeNode** (**PrefixTree** *node_)
- void **initPrefixTreeNodeWith** (**PrefixTree** *node_, **String** str_)
- **PrefixTree** * **buildPrefixTree** (const char *prefixFilePath)
- void **destroyPrefixTree** (**PrefixTree** *prefixTree_)
- bool **prefixFilter** (**PrefixTree** const *root_, **String** str_)

4.12.1 Dokumentacja funkcji

4.12.1.1 buildPrefixTree()

```
PrefixTree* buildPrefixTree (
    const char * prefixFilePath_ )
```

Buduje drzewo przedrostkowe z wyrazów wczytanych z pliku.

Parametry

<i>prefixFile↵ Path_</i>	ścieżka do pliku z wyrazami
------------------------------	-----------------------------

Zwraca

Stworzone dynamicznie drzewo.

4.12.1.2 destroyPrefixTree()

```
void destroyPrefixTree (
    PrefixTree * prefixTree_ )
```

Zwalnia z pamięci zawartość drzewa. Nie zwalnia korzenia!

Parametry

<i>prefix↵ Tree_</i>	korzeń drzewa przedrostkowego.
--------------------------	--------------------------------

4.12.1.3 initPrefixTreeNode()

```
void initPrefixTreeNode (
    PrefixTree * node_ )
```

4.12.1.4 initPrefixTreeNodeWith()

```
void initPrefixTreeNodeWith (
    PrefixTree * node_,
    String str_ )
```

4.12.1.5 insertPrefixIntoTree()

```
void insertPrefixIntoTree (
    PrefixTree * root_,
    String prefix_ )
```

Dodaje przedrostek do drzewa. Zarządza czasem życia `prefix_`!

Parametry

<i>root_</i> ↔ —	korzeń drzewa przedrostkowego
<i>prefix_</i> ↔ —	prefix do dodania

4.12.1.6 makePrefixTree()

```
PrefixTree * makePrefixTree ( )
```

Dynamicznie tworzy korzeń drzewa prefixowego.

Zwraca

Korzeń stworzonego drzewa.

4.12.1.7 prefixFilter()

```
bool prefixFilter (
    PrefixTree const * root_,
    String str_ )
```

Sprawdza, czy wyraz powinien zostać zaakceptowany przez filtr przedrostkowy.

Parametry

<i>root_</i> ↔ —	korzeń drzewa przedrostkowego
<i>str_</i> ↔ —	wyraz do sprawdzenia

Zwraca

true jeśli wyraz posiada przedrostek znajdujący się w drzewie, w przeciwnym wypadku false.

4.12.1.8 shouldContinueDown()

```
bool shouldContinueDown (
    PrefixTree * node_,
    String prefix_,
    size_t * startCharacter_ )
```

Sprawdza czy dodając prefix powinniśmy iść dalej w dół drzewa, czy rozbić aktualny węzeł i zrobić gałąź.

Parametry

<i>node_</i>	aktualnie rozpatrywany węzeł
<i>prefix_</i>	dodawany prefix
<i>startCharacter_</i>	aktualna pozycja do porównywania prefixu

4.13 Dokumentacja pliku src/String.c

```
#include <Filter/String.h>
#include <stdlib.h>
#include <ctype.h>
#include <assert.h>
```

Funkcje

- **String** makeString ()
- **String** makeStringWith (char const *bytes_, size_t len_)
- **String** wrapWithString (char *bytes_, size_t len_)
- void destroyString (**String** *str_)
- int64_t findWsInString (**String** const *str_, size_t startPos_)
- int64_t findNonWsInString (**String** const *str_, size_t startPos_)
- **PairOfStrings** splitString (**String** *src_, size_t where_)
- void appendString (**String** *str_, **String** const *toAppend_)
- void appendStringRaw (**String** *str_, char const *bytes_, size_t len_)
- void reserveStringCapacity (**String** *str_, size_t newCapacity_)
- void shrinkStringToFit (**String** *str_)

4.13.1 Dokumentacja funkcji

4.13.1.1 appendString()

```
void appendString (
    String * str_,
    String const * toAppend_ )
```

Dopisuje napis na koniec drugiego.

Parametry

<i>str_</i>	napis, do którego zostanie dopisany inny
<i>to</i> ↔ <i>Append</i> ↔ —	dopisywany napis

4.13.1.2 appendStringRaw()

```
void appendStringRaw (
    String * str_,
    char const * bytes_,
    size_t len_ )
```

Dopisuje napis na koniec drugiego (wersja dla czystego C-stringa)

Parametry

<i>str_</i>	napis, do którego zostanie dopisany inny
<i>bytes</i> ↔ —	początek napisu do dopisania
<i>len_</i>	ilość znaków do dopisania

4.13.1.3 destroyString()

```
void destroyString (
    String * str_ )
```

Usuwa zawartość istniejącego obiektu **String** (str. 9). Nie usuwa przekazanego obiektu z pamięci!

Parametry

<i>str</i> ↔ —	String (str. 9), z którego zostanie zwolniona pamięć
-------------------	---

4.13.1.4 findNonWsInString()

```
int64_t findNonWsInString (
    String const * str_,
    size_t startPos_ )
```

Znajduje indeks pierwszego elementu nie będącego białym znakiem w napisie.

Parametry

<i>str_</i>	rozpatrywany napis
<i>start</i> ↔ <i>Pos_</i>	pozycja, od której rozpoczynamy wyszukiwanie

Zwraca

pozycja znalezionego znaku, lub -1 gdy nie znaleziono.

4.13.1.5 findWsInString()

```
int64_t findWsInString (
    String const * str_,
    size_t startPos_ )
```

Znajduje indeks pierwszego białego znaku w napisie.

Parametry

<i>str_</i>	rozpatrywany napis
<i>start</i> ↔ <i>Pos_</i>	pozycja, od której rozpoczynamy wyszukiwanie

Zwraca

pozycja znalezionego białego znaku, lub -1 gdy nie znaleziono.

4.13.1.6 makeString()

```
String makeString ( )
```

Tworzy pusty **String** (str. 9).

Zwraca

pusty **String** (str. 9).

4.13.1.7 makeStringWith()

```
String makeStringWith (
    char const * bytes_,
    size_t len_ )
```

Tworzy **String** (str. 9) kopiując inny istniejący.

Parametry

<i>bytes</i> ↔ —	źródło do kopiowania danych
<i>len</i> —	ilość bajtów do skopiowania

Zwraca

nowo utworzony **String** (str. 9).

4.13.1.8 reserveStringCapacity()

```
void reserveStringCapacity (
    String * str_,
    size_t newCapacity_ )
```

Rezerwuje pamięć wewnątrz Stringa. Nigdy nie zmniejsza zaalokowanej pamięci.

Parametry

<i>str</i> —	napis, któremu rezerwujemy pamięć
<i>new</i> ↔ <i>Capacity</i> —	nowa pojemność pamięci.

4.13.1.9 shrinkStringToFit()

```
void shrinkStringToFit (
    String * str_ )
```

Ucina zarezerwowana pamięć do ilości znaków faktycznie używanych przez Stringa.

Parametry

<i>str</i> ↔ —	modyfikowany napis
-------------------	--------------------

4.13.1.10 splitString()

```
PairOfStrings splitString (
    String * src_,
    size_t where_ )
```

Rozbija napis na dwa osobne napisy w następujący sposób: lewy zawiera indeksy [0, where_), prawy zawiera indeksy [where_, length). Nie zwalnia rozbijanego zapisu z pamięci!

Parametry

<i>src_</i>	rozbijany napis
<i>where_↔</i> _	pozycja do rozbicia napisu

Zwraca

struktura zawierająca lewą i prawą część napisu

4.13.1.11 wrapWithString()

```
String wrapWithString (
    char * bytes_,
    size_t len_ )
```

Tworzy **String** (str. 9) opakowując istniejące dane.

Parametry

<i>bytes_↔</i> _	źródło danych
<i>len_</i>	ilość bajtów

Zwraca

nowo utworzony **String** (str. 9) (opakowujący istniejące dane).

Indeks

App.c
 run, 24
App.h
 run, 11
appendString
 String.c, 32
 String.h, 19
appendStringRaw
 String.c, 33
 String.h, 19

BFR_BUF_SIZE
 BufferedReader.h, 12
bfrClose
 BufferedReader.c, 25
 BufferedReader.h, 13
bfrConsume
 BufferedReader.c, 25
bfrOpen
 BufferedReader.c, 25
 BufferedReader.h, 13
bfrReadNext
 BufferedReader.c, 26
bfrReadUntilWs
 BufferedReader.c, 26
 BufferedReader.h, 13
buf
 BufferedReader, 5
BufferedReader, 5
 buf, 5
 BufferedReader.h, 12
 file, 5
 readCount, 6
BufferedReader.c
 bfrClose, 25
 bfrConsume, 25
 bfrOpen, 25
 bfrReadNext, 26
 bfrReadUntilWs, 26
BufferedReader.h
 BFR_BUF_SIZE, 12
 bfrClose, 13
 bfrOpen, 13
 bfrReadUntilWs, 13
 BufferedReader, 12
buildPrefixTree
 PrefixTree.c, 30
 PrefixTree.h, 17

capacity

String, 9
CHAR_COMBINATIONS
 PrefixTree.h, 16
children
 PrefixTree, 8

data
 String, 9
destroyPrefixTree
 PrefixTree.c, 30
 PrefixTree.h, 17
destroyString
 String.c, 33
 String.h, 20

file
 BufferedReader, 5
Filter.c
 filterInputFile, 27
 OUTPUT_BUF_SIZE, 27
Filter.h
 filterInputFile, 14
 PrefixTree, 14
filterInputFile
 Filter.c, 27
 Filter.h, 14
findNonWsInString
 String.c, 33
 String.h, 20
findWsInString
 String.c, 34
 String.h, 20

include/Filter/App.h, 11
include/Filter/BufferedReader.h, 12
include/Filter/Filter.h, 14
include/Filter/ParseArgs.h, 15
include/Filter/PrefixTree.h, 16
include/Filter/String.h, 18
inFilePath
 LaunchConfig, 6
initPrefixTreeNode
 PrefixTree.c, 30
initPrefixTreeNodeWith
 PrefixTree.c, 30
insertPrefixIntoTree
 PrefixTree.c, 30
isLeaf
 PrefixTree, 8

LaunchConfig, 6

- inFilePath, 6
- outFilePath, 6
- ParseArgs.h, 15
- prefixFilePath, 7
- left
 - PairOfStrings, 7
- len
 - String, 9
- main
 - Main.c, 28
- Main.c
 - main, 28
- makePrefixTree
 - PrefixTree.c, 31
- makeString
 - String.c, 34
 - String.h, 21
- makeStringWith
 - String.c, 34
 - String.h, 21
- outFilePath
 - LaunchConfig, 6
- OUTPUT_BUF_SIZE
 - Filter.c, 27
- PairOfStrings, 7
 - left, 7
 - right, 7
 - String.h, 19
- parseArgs
 - ParseArgs.c, 29
 - ParseArgs.h, 15
- ParseArgs.c
 - parseArgs, 29
- ParseArgs.h
 - LaunchConfig, 15
 - parseArgs, 15
- prefix
 - PrefixTree, 8
- prefixFilePath
 - LaunchConfig, 7
- prefixFilter
 - PrefixTree.c, 31
 - PrefixTree.h, 17
- PrefixTree, 8
 - children, 8
 - Filter.h, 14
 - isLeaf, 8
 - prefix, 8
 - PrefixTree.h, 17
- PrefixTree.c
 - buildPrefixTree, 30
 - destroyPrefixTree, 30
 - initPrefixTreeNode, 30
 - initPrefixTreeNodeWith, 30
 - insertPrefixIntoTree, 30
 - makePrefixTree, 31
 - prefixFilter, 31
 - shouldContinueDown, 31
- PrefixTree.h
 - buildPrefixTree, 17
 - CHAR_COMBINATIONS, 16
 - destroyPrefixTree, 17
 - prefixFilter, 17
 - PrefixTree, 17
- readCount
 - BufferedReader, 6
- reserveStringCapacity
 - String.c, 35
 - String.h, 21
- right
 - PairOfStrings, 7
- run
 - App.c, 24
 - App.h, 11
- shouldContinueDown
 - PrefixTree.c, 31
- shrinkStringToFit
 - String.c, 35
 - String.h, 23
- splitString
 - String.c, 35
 - String.h, 23
- src/App.c, 24
- src/BufferedReader.c, 24
- src/Filter.c, 27
- src/Main.c, 28
- src/ParseArgs.c, 28
- src/PrefixTree.c, 29
- src/String.c, 32
- String, 9
 - capacity, 9
 - data, 9
 - len, 9
 - String.h, 19
- String.c
 - appendString, 32
 - appendStringRaw, 33
 - destroyString, 33
 - findNonWsInString, 33
 - findWsInString, 34
 - makeString, 34
 - makeStringWith, 34
 - reserveStringCapacity, 35
 - shrinkStringToFit, 35
 - splitString, 35
 - wrapWithString, 36
- String.h
 - appendString, 19
 - appendStringRaw, 19
 - destroyString, 20
 - findNonWsInString, 20
 - findWsInString, 20
 - makeString, 21

- makeStringWith, 21
- PairOfStrings, 19
- reserveStringCapacity, 21
- shrinkStringToFit, 23
- splitString, 23
- String, 19
- wrapWithString, 23

wrapWithString

- String.c, 36
- String.h, 23