My Project

Generated by Doxygen 1.15.0

Chapter 1

Dokumentacja Projektu Listy Dwukierunkowej

1.1 Wprowadzenie

Niniejsza dokumentacja techniczna opisuje implementację **Listy Dwukierunkowej** (Double-Linked List) w języku C++. Jest to podstawowa struktura danych, w której każdy element (węzeł) zawiera wskaźniki do poprzedniego i następnego elementu, co umożliwia efektywne operacje dwukierunkowe i implementację wzorców projektowych.

Struktura jest zaimplementowana z użyciem klas Node (pojedynczy element) oraz List (zarządzająca lista).

1.1.1 Zaimplementowane funkcje (Lista Dwukierunkowa)

Zaimplementowano wszystkie podstawowe metody manipulacji listą:

- Dodawanie na początek i koniec: List::addFront, List::addBack
- Dodawanie pod indeks: List::insertAt
- Usuwanie z przodu, z tyłu i pod indeksem: List::removeFront, List::removeBack, List::removeAt
- Wyświetlanie listy: List::display (od początku), List::displayReverse (od końca)
- Czyszczenie listy: List::clear

1.1.2 Braki i Wymagane Uzupełnienia

W celu pełnej realizacji wymagań projektu (List.h), konieczne jest jeszcze:

- Wprowadzenie wzorców projektowych **Factory** oraz **Iterator**.
- Metody `displayNext()` i `displayPrevious()` są przewidziane do integracji z wzorcem Iteratora.

See also

List

Node

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Class Documentation

4.1 List Class Reference

Klasa implementujaca liste dwukierunkowa.

`#include <list.h>`

Public Member Functions

- **List** ()

  Konstruktor listy. Inicjalizuje glowe i ogon na nullptr.
- ~**List** ()

  Destruktor listy. Usuwa wszystkie wezly, zwalniajac pamiec.
- void addFront (int val)

  Dodaje element na poczatek listy.
- void addBack (int val)

  Dodaje element na koniec listy.
- void insertAt (int val, int index)

  Dodaje element pod wskazany indeks.
- void **removeFront** ()

  Usuwa element z poczatku listy.
- void **display** ()

  Wyswietla cala liste od poczatku.
- void **displayReverse** ()

  Wyswietla cala liste od konca.
- void **removeBack** ()

  Usuwa element z konca listy.
- void removeAt (int index)

  Usuwa element spod wskazanego indeksu.
- void **clear** ()

  Usuwa wszystkie elementy z listy, zwalniajac pamiec.

4.1.1 Detailed Description

Klasa implementujaca liste dwukierunkowa.

- Lista jest zarzadzana na stercie (uzycie operatora new/delete wewnatrz). W przyszlosci powinna zostac zaimplementowana z uzyciem wzorcow Factory i Iterator.

4.1.2 Member Function Documentation

4.1.2.1 addBack()

```
void List::addBack (
            int val)
```

Dodaje element na koniec listy.

Parameters

val	Wartosc do dodania.

4.1.2.2 addFront()

```
void List::addFront (
            int val)
```

Dodaje element na poczatek listy.

Parameters

val	Wartosc do dodania.

4.1.2.3 insertAt()

```
void List::insertAt (
            int val,
            int index)
```

Dodaje element pod wskazany indeks.

Parameters

val	Wartosc do dodania.
index	Indeks, pod ktory ma byc dodany element.

4.1.2.4 removeAt()

```
void List::removeAt (
            int index)
```

Usuwa element spod wskazanego indeksu.

Parameters

index	Indeks elementu do usuniecia.

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

- list.h
- list.cpp

4.2 Node Class Reference

Klasa reprezentujaca pojedynczy wezel (element) listy dwukierunkowej.

`#include <node.h>`

Public Member Functions

- Node (int val)
  Konstruktor wezla.
- ∼**Node** ()
  Destruktor wezla.

Public Attributes

- int **data**
  Przechowywana wartosc liczbowa.
- Node ∗ **next**
  Wskaznik na nastepny wezel w liscie.
- Node ∗ **prev**
  Wskaznik na poprzedni wezel w liscie.

4.2.1 Detailed Description

Klasa reprezentujaca pojedynczy wezel (element) listy dwukierunkowej.

Wezel jest alokowany na stercie i zawiera dane, a takze wskazniki do poprzedniego i nastepnego elementu.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Node()

```
Node::Node (
            int val)
```

Konstruktor wezla.

Parameters

val	Wartosc, ktora ma byc przechowywana w wezle.

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

- node.h
- node.cpp

Chapter 5

File Documentation

5.1 list.h

```
00001 #pragma once
00002 #include "Node.h"
00003
00009 class List {
00010 private:
00011     Node* head;
00012     Node* tail;
00013
00014     // Brakuje tu jeszcze klasy Factory i Iterator, ktore sa wymagane w projekcie.
00015
00016 public:
00020     List();
00021
00025     ~List();
00026
00027     // --- Metody Funkcjonalne Wymagane w Projekcie ---
00028
00033     void addFront(int val);
00034
00039     void addBack(int val);
00040
00046     void insertAt(int val, int index);
00047
00051     void removeFront();
00052
00056     void display();
00057
00061     void displayReverse();
00062
00066     void removeBack();
00067
00072     void removeAt(int index);
00073
00077     void clear();
00078
00079     // Brakuje metod:
00080     // - displayNext()
00081     // - displayPrevious()
00082     // - Factory i Iterator
00083 };
```

5.2 node.h

```
00001 #pragma once
00002
00009 class Node {
00010 public:
00011     int data;
00012     Node* next;
00013     Node* prev;
00014
00019     Node(int val);
00020
00024     ~Node();
00025 };
```

Index