

Лабораторна робота № 2: Права доступу, const, покажчики посилання

Мета

Отримати навички при передаванні об'єктів із застосуванням прав доступу та const-модифікаторів.

1.ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Розподілити в `swheel` права доступу `private`, `public`. Реалізувати клас `CGraphScreen` основна задача якого полягає у більш багатому відображенні даних `swheel` із застосуванням псевдографіки для наочного відображення пов'язаного об'єкта. `CGraphScreen` повинен містити поля згідно опису в індивідуальному завданні та наступні методи:

- `SetDataSource()` - для зміни об'єкта-джерела даних.
- `PrintData()` - виводитиме інформацію про отриманий об'єкт у якості аргументу. Оновити `CScreen` для збереження функціональності цього класу при роботі з оновленим `swheel`.

2.РОЗРОБКА ПРОГРАМИ

2.1 Засоби ООП

В ході розробки програми були використані такі засоби ООП:

- Абстракція - кожен об'єкт описує свою особливу сутність, яка визначається його полями.
- Інкапсуляція - поля об'єктів закриті для користувача, натомість ми даємо доступ до даних за допомогою геттерів та сеттерів, так користувач має можливість отримати готові дані, а не обробляти їх, для подальшого використання.

2.2 Ієрархія та структура класів

На рис 2.1 дивись ієрархію класів

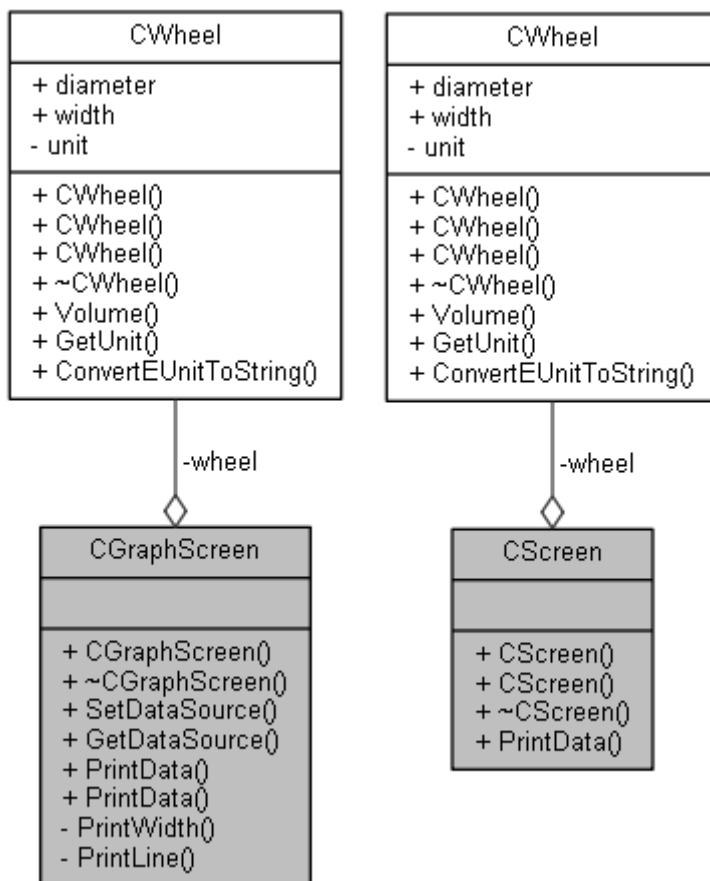


Рисунок 2.1 ієрархія класів

2.3 Опис програми

На рис 2.2 дивись структуру проекту.

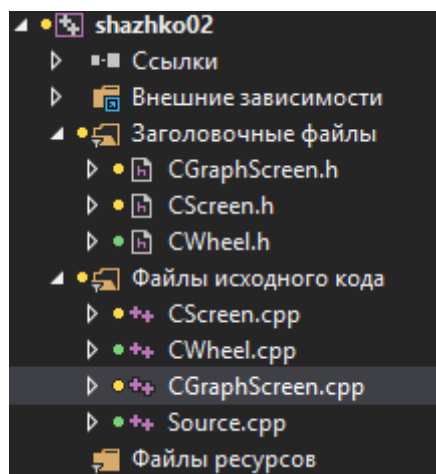


Рисунок 2.2 структура проекту

На рис 2.3 дивись призначення класів.

Класи, структури, об'єднання та інтерфейси з коротким описом.




 CGraphScreen	Класс описывающий расширенное отображение объектов класса CWheel
 CScreen	Класс описывающий отображение объектов класса CWheel
 CWheel	Класс описывающий колесо

Рисунок 2.3 призначення класів

З попередньої лабораторної роботи збереглися класи `CWheel` та `CScreen`. У `Window` змінено модифікатори доступу, додано 2 методи, які виводять ширину та висоту вікна. Методи виводу класу `CScreen` змінено, для коректної роботи з `private` та `public` полями. Додано клас `CGraphScreen` у якому завдяки псевдографіки ми виводимо наш `CWheel`.

2.4 Важливі фрагменти програми

У програмі слід зауважити увагу на таких моментах:

Клас `CGraphScreen` функція виведення даних за допомогою псевдографіки:

```
void CGraphScreen::PrintData(const CWheel& data) {
    SHORT diameter = (SHORT)data.diameter;
    SHORT radius = ((SHORT)diameter) >> 1;
    CONSOLE_SCREEN_BUFFER_INFO cursorinfo;
    PrintLine(data.width, (SHORT)data.diameter);
    if (!GetConsoleScreenBufferInfo(GetStdHandle(STD_OUTPUT_HANDLE), &cursorinfo)){ std::cout <<
"\nERROR\n"; return; }
    cursorinfo.dwCursorPosition.X = radius;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), {
cursorinfo.dwCursorPosition.X, cursorinfo.dwCursorPosition.Y });

    for (SHORT i = 0; i <= radius; i++)std::cout << "*";
    this->PrintWidth(data.width, { diameter << 1, cursorinfo.dwCursorPosition.Y });
    for (SHORT i = 1; i < diameter; i++) {
        cursorinfo.dwCursorPosition.Y++;
        SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), { (SHORT)abs(radius -
i), cursorinfo.dwCursorPosition.Y });
        std::cout << "*";
        if (i <= radius) SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), { (
diameter + i), cursorinfo.dwCursorPosition.Y });
        else
            SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), { (diameter +
(radius-(i-radius))), cursorinfo.dwCursorPosition.Y });
        std::cout << "*";
        this->PrintWidth(data.width, { diameter << 1, cursorinfo.dwCursorPosition.Y });
    }
    cursorinfo.dwCursorPosition.Y++;
    cursorinfo.dwCursorPosition.X = radius;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), {
cursorinfo.dwCursorPosition.X, cursorinfo.dwCursorPosition.Y });
    for (SHORT i = 0; i <= radius; i++)std::cout << "*";
    this->PrintWidth(data.width, { diameter << 1, cursorinfo.dwCursorPosition.Y });
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), {
(SHORT)0, cursorinfo.dwCursorPosition.Y+1 });

    PrintLine(data.width, (SHORT)data.diameter);
}
```

Демонстрація роботи програми:

```
int main() {
    CWheel *iWheel1 = new CWheel(6, 5, EUNIT_CENTIMETERS);
    CWheel *iWheel2 = new CWheel(8, 3, EUNIT_CENTIMETERS);

    std::cout << "===== CScreen =====\n";
    CScreen *screen1 = new CScreen(*iWheel1);
```

```

screen1->PrintData();
CScreen *screen2 = new CScreen(*iWheel2);
screen2->PrintData();

std::cout << "\n\n===== CGraphScreen =====\n";

CGraphScreen graphScreen;
graphScreen.PrintData(iWheel1);
std::cout << "\n";
graphScreen.SetDataSource(iWheel2);
graphScreen.PrintData();

delete screen1;

delete screen2;
delete iWheel1;
delete iWheel2;
return 0;
}

```

3.РЕЗУЛЬТАТИ РОБОТИ

Результат роботи показано на рис 3.1.

```

===== CScreen =====
    Diameter: 6
    Width: 5
    Units: CENTIMETERS
Volume: 141.372
    Diameter: 8
    Width: 3
    Units: CENTIMETERS
Volume: 150.796

===== CGraphScreen =====
-----
****      *****
*          *****
*          *****
*          *****
*          *****
*          *****
****      *****
-----

*****    ***
*          ***
*          ***
*          ***
*          ***
*          ***
*          ***
*          ***
*****    ***
-----

```

Рисунок 3.1 результат роботи

ВИСНОВКИ

В результаті лабораторної роботи було розроблено програму з використанням прав доступу та const методів. Були придбані навички роботи з цими технологіями.