1МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №7**

**з дисципліни «Архітектура та проектування ПЗ»**

**на тему: «Файловий менеджер»**

Виконав: студент гр. ПЗ1911

Сіньков Г.О.

Прийняла: ас. каф. КІТ

Куроп'ятник О. С.

Дніпро, 2020

**Лабораторна робота №7**

**Тема**. Генерація коду.

**Мета**. Ознайомитися із взаємозв’язком діаграм та етапами генерації коду. Отримати практичні навички генерації коду для UML-моделі.

**Постановка задачі**

1.Індивідуальна постановка: предметної області – файловий менеджер.

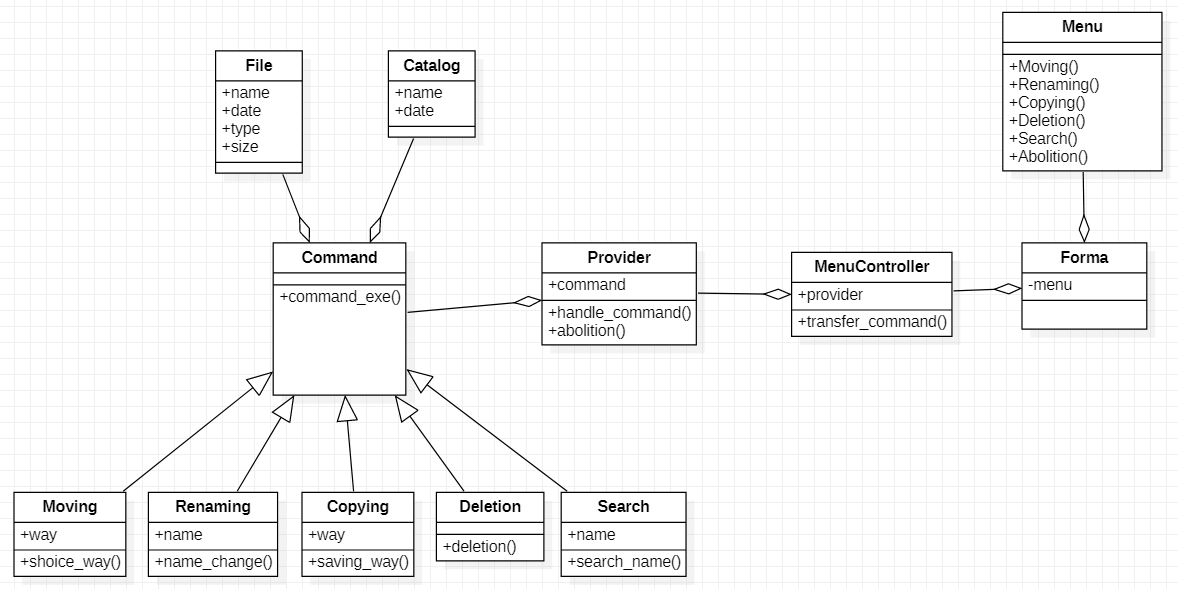


Рис.1 Діаграма класів

**Згенерований код**

Menu.h

/\*\*

\* Project Untitled

\*/

#ifndef \_MENU\_H

#define \_MENU\_H

class Menu {

public:

void Moving();

void Renaming();

void Copying();

void Deletion();

void Search();

void Abolition();

};

#endif //\_MENU\_H

Menu.cpp

/\*\*

\* Project Untitled

\*/

#include "Menu.h"

/\*\*

\* Menu implementation

\*/

void Menu::Moving() {

}

void Menu::Renaming() {

}

void Menu::Copying() {

}

void Menu::Deletion() {

}

void Menu::Search() {

}

void Menu::Abolition() {

}

Forma.h

/\*\*

\* Project Untitled

\*/

#ifndef \_FORMA\_H

#define \_FORMA\_H

class Forma {

private:

void menu;

};

#endif //\_FORMA\_H

Forma.cpp

/\*\*

\* Project Untitled

\*/

#include "Forma.h"

/\*\*

\* Forma implementation

\*/

MenuController.h

/\*\*

\* Project Untitled

\*/

#ifndef \_MENUCONTROLLER\_H

#define \_MENUCONTROLLER\_H

class MenuController {

public:

void provider;

void transfer\_command();

};

#endif //\_MENUCONTROLLER\_H

MenuController.cpp

/\*\*

\* Project Untitled

\*/

#include "MenuController.h"

/\*\*

\* MenuController implementation

\*/

void MenuController::transfer\_command() {

}

Provider.h

/\*\*

\* Project Untitled

\*/

#ifndef \_PROVIDER\_H

#define \_PROVIDER\_H

class Provider {

public:

void command;

void handle\_command();

void abolition();

};

#endif //\_PROVIDER\_H

Provider.cpp

/\*\*

\* Project Untitled

\*/

#include "Provider.h"

/\*\*

\* Provider implementation

\*/

void Provider::handle\_command() {

}

void Provider::abolition() {

}

Commande.h

/\*\*

\* Project Untitled

\*/

#ifndef \_COMMAND\_H

#define \_COMMAND\_H

class Command {

public:

void command\_exe();

};

#endif //\_COMMAND\_H

Commande.cpp

/\*\*

\* Project Untitled

\*/

#include "Command.h"

/\*\*

\* Command implementation

\*/

void Command::command\_exe() {

}

Moving.h

/\*\*

\* Project Untitled

\*/

#ifndef \_MOVING\_H

#define \_MOVING\_H

#include "Command.h"

class Moving: public Command {

public:

void way;

void shoice\_way();

};

#endif //\_MOVING\_H

Moving.cpp

/\*\*

\* Project Untitled

\*/

#include "Moving.h"

/\*\*

\* Moving implementation

\*/

void Moving::shoice\_way() {

}

Renaming.h

/\*\*

\* Project Untitled

\*/

#ifndef \_RENAMING\_H

#define \_RENAMING\_H

#include "Command.h"

class Renaming: public Command {

public:

void name;

void name\_change();

};

#endif //\_RENAMING\_H

Renaming.cpp

/\*\*

\* Project Untitled

\*/

#include "Renaming.h"

/\*\*

\* Renaming implementation

\*/

void Renaming::name\_change() {

}

Copying.h

/\*\*

\* Project Untitled

\*/

#ifndef \_COPYING\_H

#define \_COPYING\_H

#include "Command.h"

class Copying: public Command {

public:

void way;

void saving\_way();

};

#endif //\_COPYING\_H

Copying.cpp

/\*\*

\* Project Untitled

\*/

#include "Copying.h"

/\*\*

\* Copying implementation

\*/

void Copying::saving\_way() {

}

Deletion.h

/\*\*

\* Project Untitled

\*/

#ifndef \_DELETION\_H

#define \_DELETION\_H

#include "Command.h"

class Deletion: public Command {

public:

void deletion();

};

#endif //\_DELETION\_H

Deletion.cpp

/\*\*

\* Project Untitled

\*/

#include "Deletion.h"

/\*\*

\* Deletion implementation

\*/

void Deletion::deletion() {

}

Search.h

/\*\*

\* Project Untitled

\*/

#ifndef \_SEARCH\_H

#define \_SEARCH\_H

#include "Command.h"

class Search: public Command {

public:

void name;

void search\_name();

};

#endif //\_SEARCH\_H

Search.cpp

/\*\*

\* Project Untitled

\*/

#include "Search.h"

/\*\*

\* Search implementation

\*/

void Search::search\_name() {

}

File.h

/\*\*

\* Project Untitled

\*/

#ifndef \_FILE\_H

#define \_FILE\_H

class File {

public:

void name;

void date;

void type;

void size;

};

#endif //\_FILE\_H

File.cpp

/\*\*

\* Project Untitled

\*/

#include "File.h"

/\*\*

\* File implementation

\*/

Catalog.h

/\*\*

\* Project Untitled

\*/

#ifndef \_СATALOG\_H

#define \_СATALOG\_H

class Сatalog {

public:

void name;

void date;

};

#endif //\_СATALOG\_H

Catalog.cpp

/\*\*

\* Project Untitled

\*/

#include "Сatalog.h"

/\*\*

\* Сatalog implementation

\*/

Висновок: при виконані лабораторної роботи я зрозумів на скільки зручно створювати діаграму класів в UML і генерувати код по цій діаграмі, тому що не тратиш час на придумування назв класу, зв’язків, створення файлів, а витрачаєш час на створення функціоналу.