**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Объектно-ориентированное программирование»**

Тема: Полиморфизм

|  |  |  |
| --- | --- | --- |
| Студент гр. 8381 |  | Почаев Н.А. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2020

## Цель работы.

Реализовать набор классов, для ведения логирования действий и состояний программы. Основные требования:

* Логирование действий пользователя;
* Логирование действий юнитов и базы.

## Задание.

* Выполнены основные требования к логированию
* Реализована возможность записи логов в файл
* Реализована возможность записи логов в терминал
* Взаимодействие с файлами должны быть по идиоме RAII
* Для логирования состояний перегружен оператор вывода в поток
* Переключение между разным логированием (логирование в файл, в терминал, без логирования) реализуется при помощи паттерна “Прокси”
* Реализован разный формат записи при помощи паттерна “Адаптер”

## Выполнение работы.

Написание работы производилось на базе операционной системы Windows 10 в среде разработки Qt Creator, для компиляции и отладки использовалась UNIX-подобная среда Cygwi и набор адаптированных инструментов MiniGW. Были задействованы пакеты GCC, CMake, а также GDB. Для компиляции текущей версии программы под Windows необходим MinGW 8.10 (для более полноценной поддержки C++17) и Qt версии 14.10 и выше.

## Реализованные классы

Классы, добавленные в программу в данной лабораторной работе и их функционал представлены в табл. 1. В ней приведено общее описание классов, отдельные моменты пояснены в комментариях к коду.

Таблица 1 – Основные добавленные классы

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| Time  (./Game/Logging) | Класс используется для работы со временем, вывода его в представлении “Y-m-d H:M:S” в приёмники логирования, а также хранения временной отметки начала работы.  В данном классе выполнена *перегрузка оператора вывода в поток,* для читабельного вывода в аналогично перегруженном операторе вывода последующего класса. |
| ILogger  (./Game/Logging) | Интерфейс, описывающий основные операции, для логгеров. |
| TermialLogger  (./Game/Logging/Loggers) | Класс логгере для записи в терминал. Содержит *перегрузку оператора вывода в поток*. |
| FileLogger  (./Game/Logging/Loggers) | Класс логгера для записи в файл. Содержит *перегрузку оператора вывода в поток*. Взаимодействие с файлом логирования LOG происходит по идиоме RAII – создаётся и инициализируется в конструкторе, а закрывается на запись в деструкторе. |
| ProxyLogger  (./Game/Logging/Loggers) | Класс, реализующий паттерн *“Прокси”* для возможности смены логгера “на лету”. |
| ILogAdapter  (./Game/Logging/Loggers) | Интерфейс, описывающий основные операции класса адаптера для логгеров. |
| ILogAdapterStringFormer (./Game/Logging/Loggers) | Интерфейс, описывающий основный операции формирования строк логирования для класса адаптера. |
| LogAdapter  (./Game/Logging/Loggers) | Класс, реализующий паттерн *“Адаптер”* для выбора различных режимов вывода логов: STANDART – обычный и ADVANCED – расширенный, с полной информацией, требуемой для выполнения операции. |

## Выводы.

В ходе выполнения лабораторной работы были написаны требуемые класс, а также реализовано логирование основных действий пользователя и событий игры.

# Приложение А Исходный код программы. MAIN.cPP

#include <iostream>

#include <QApplication>

#include <QGridLayout>

#include <QWidget>

#include <QLabel>

#include <QScreen>

#include "Tests/examples.h"

#include "Game/UIFacade.h"

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

{

std::shared\_ptr<UIFacade> game = std::make\_shared<UIFacade>(argc, argv);

game->start();

return 0;

}