

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий
Кафедра прикладной математики

Отчет защищен с оценкой _____

Преподаватель _____ (подпись)
«__» _____ 2023 г.

Отчет
по лабораторной работе № 9
«Порождающие паттерны 1 - проектирование
(Factory method, Singleton) »по дисциплине «Объектно-
ориентированное программирование»

Студент гр. ПИ-02
Чередов Р.А.

Ассистент кафедры ПМ,
Рахманин Д. С.

Барнаул 2023

Тема: Порождающие паттерны 1 - проектирование
(Factory method, Singleton)

Отчет:

Диаграмма классов

Краткое описание назначения классов в системе

Постановка задачи конфигурирования системы

125	Чередов Роман Алексеевич	Системы анализа больших данных	Проектирование системы обработки потоков данных о местонахождении абонентов сети сотовой связи
-----	-----------------------------	-----------------------------------	--

Factory method

```
include <iostream>
using namespace std;
class Event {
protected:
    string Num;

public:

    Event(const string& Num) {
        this->Num = Num;
    }
    virtual void Display() { cout << "My name is " << Num << endl; }
    const string& GetNumber() { return Num; }
    virtual string& getInfo() = 0;
};

class SecurityEven : public Event {
private:
    string info;

public:

    SecurityEven(const string& Num, string inf) : Event(Num) {
        cout << "New security event (" << Num << "): " << inf << endl; info = inf;
    }
    virtual void Display() {
        cout << endl << Num << ": " << info;
    }
    virtual string& getInfo() {
        return info;
    }
};

class SearchEven : public Event {/
private:
    string info;

public:

    SearchEven(const string& Num, string inf) : Event(Num) {
        cout << "New search (" << Num << "): " << inf << endl;
        info = '|' + inf + '|';
    }
    virtual void Display() {
        cout << endl << Num << ": ";
        cout << "---->" << info << "|---<";
    }
    virtual string& getInfo() {
```

```

        return info;
    }
};

class Object { // Класс Factory method
public:
    Object() {}
    virtual Event* createEvent(const string Num, string info) = 0;
    Event* cloneEvent(Event* a) {
        return this->createEvent(a->GetNumber() + "(copy)", a->getInfo());
    }
};

class SecuirityObj : public Object {
public:
    SecuirityObj() : Object() {}
    virtual SecurityEven* createEvent(const string Num, string info) {
        return new SecurityEven(Num, info);
    }
};

class SearchObj : public Object {
public:
    SearchObj() : Object() {}
    virtual SearchEven* createEvent(const string Num, string info) {
        return new SearchEven(Num, info);
    }
};

int main() {

    Object* SecuirityChoice = new SecuirityObj();
    Event* a = SecuirityChoice->createEvent("1", "this is alarm");

    Event* b = SecuirityChoice->createEvent("2", "fire alarm");
    Event* c = SecuirityChoice->cloneEvent(a);
a -> Display();
b -> Display();
c -> Display();

    cout << endl << endl;
    Object* SearchChoice = new SearchObj();
    Event* e = SearchChoice->createEvent("2", "gps search");
    Event* f = SearchChoice->createEvent("1", "distance search");
    Event* g = SearchChoice->cloneEvent(e);

    e -> Display();
    f -> Display();
    g -> Display();

    delete SecuirityChoice, SearchChoice;
    delete a, b, c, e, f, g;
    return 0;
}

```

```
New security event (1): this is alarm
New security event (2): fire alarm
New security event (1(copy)): this is alarm

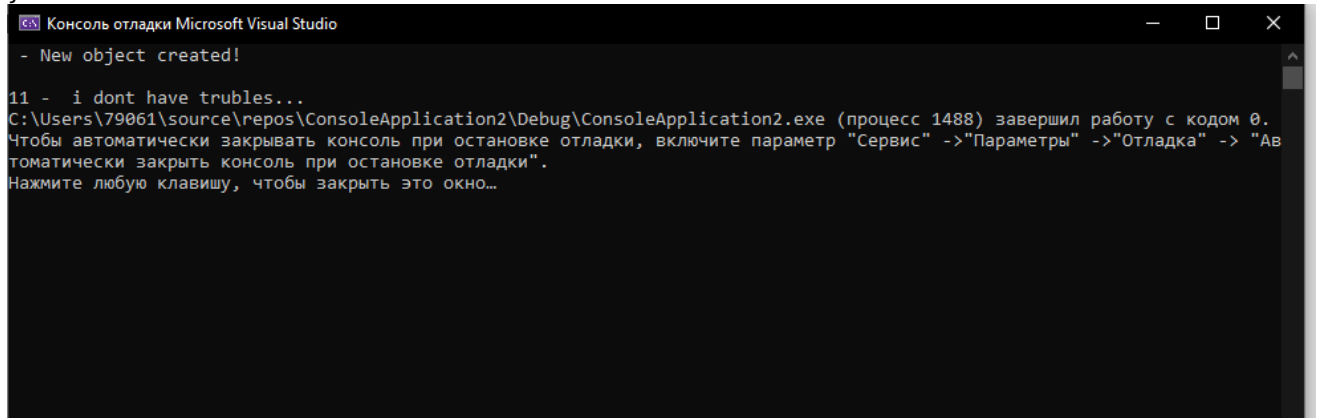
1: this is alarm
2: fire alarm
1(copy): this is alarm

New search (2): gps search
New search (1): distance search
New search (2(copy)): fire alarm

2: ---->|gps search||---<
1: ---->|distance search||---<
```

Singleton

```
#include <iostream>
using namespace std;
class ComplexControlSystem {
private:
int date;
static ComplexControlSystem* instance;
ComplexControlSystem(int data) { date = data; } //прячем конструкторы
public:
static ComplexControlSystem* getInstance(int data);
void diag() {
cout << endl << date << " - i dont have troubles...";
}
};
ComplexControlSystem* ComplexControlSystem::instance = 0; //для линкера
ComplexControlSystem* ComplexControlSystem::getInstance(int data) {
if (instance == 0) {
cout << " - New object created!" << endl;
instance = new ComplexControlSystem(data);
}
else
cout << " - Object already exists!" << endl;
return instance;
}
int main() {
ComplexControlSystem* a = ComplexControlSystem::getInstance(11);
a->diag();
delete a;
return 0;
}
```



Консоль отладки Microsoft Visual Studio

```
- New object created!
11 - i dont have troubles...
C:\Users\79061\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe (процесс 1488) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" -> "Параметры" -> "Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

