Федеральное государственное автономное образовательное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

Творческая работа

выполнил:

студент группы РИС-23-3Б

Богомягков Василий Александрович

проверила:

доцент кафедры ИТАС О. А. Полякова

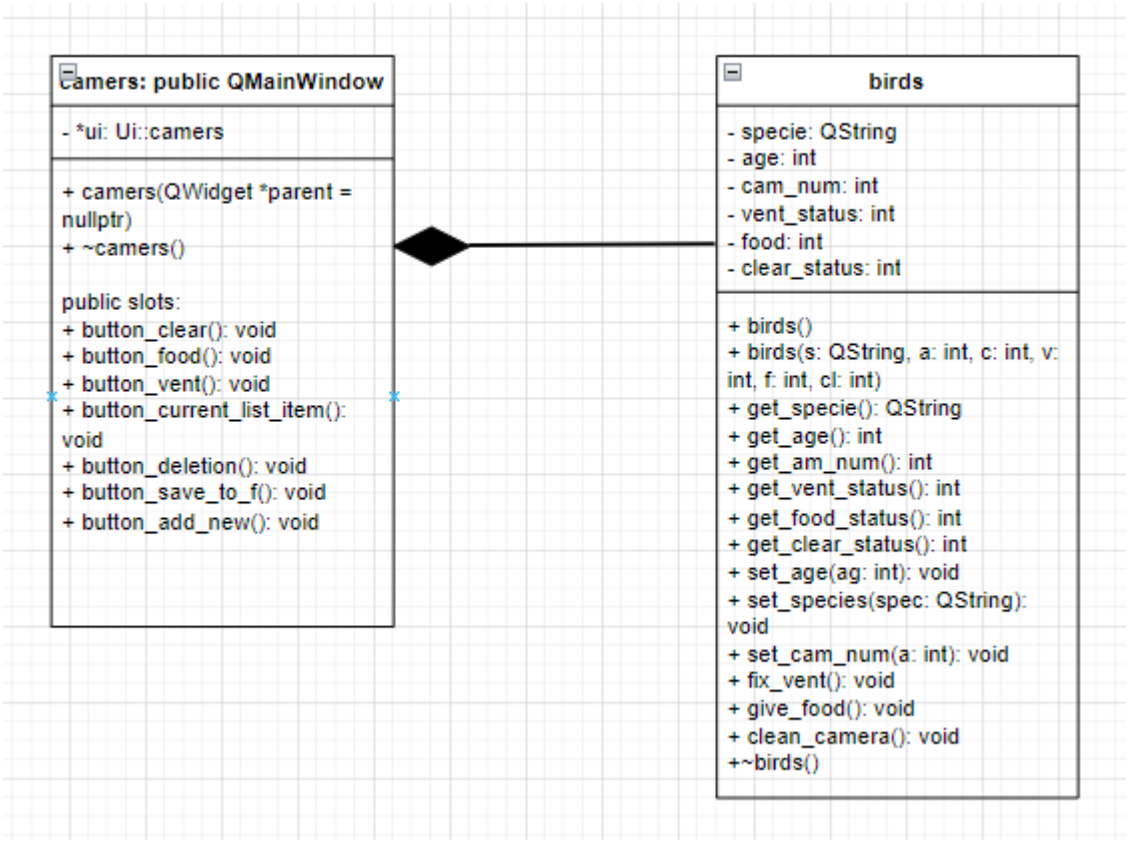
2024 г.

1. АРМ

АРМ специалиста орнитолога. Программа, которая позволяет следить за состоянием камер, в которых находятся наблюдаемые особи птиц. Можно узнать вид и возраст особи, состояние камеры, в которой эта особь находится (состояние работоспособности вентиляционной системы, количество оставшейся еды от обычной порции, то, насколько чистая камера), в процентах.

Для интерфейса был использован qt.

UML



Анализ

Вне функций хранится глобальный вектор vector<birds>, в котором будут храниться все объекты класса birds.

Класс birds:

- Поля – private

Методы:

- Конструктор
- Деструктор
- Методы, начинающиеся с “get_” – получение данных полей объекта

- Методы, начинающиеся с “set_”, fix_vent, clean_camera, give_food – установка данных полей объекта

Класс основного окна:

- При запуске программы считываются данные из файла bird_cams.txt в вектор vector<birds>, в список на экране выводятся некоторые данные об элементах (вид и номер камеры).

Методы:

- button_clear – кнопка очистки камеры, вызывает функцию clean_camera для текущего объекта
- button_vent – кнопка починки вентиляционной системы, вызывает функцию fix_vent для текущего объекта
- button_food – кнопка добавления порции еды, вызывает функцию give_food для текущего объекта
- button_deletion – удаляет выделенный в списке объект из списка и из вектора объектов, текущий объект – объект без данных
- button_save_to_f – основной файл открывается на запись, очищается с помощью QIODevice::Truncate, в него записываются данные из всех объектов вектора
- button_add_new – открытие файла и считывание из него данных о новом объекте
- button_current_list_item – берёт индекс выделенного объекта списка, текущий объект – объект из вектора с этим же индексом

Код

camers.cpp:

```
#include "camers.h"
#include "ui_camers.h"
std::vector<birds> v;
birds curren;
birds defaul;
camers::camers(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::camers)
{
    ui->setupUi(this);
    ui->clear_fixat->setEnabled(false);
    ui->vent_fixat->setEnabled(false);
    ui->food_fixat->setEnabled(false);
    ui->add_el_to_list->setEnabled(false);
    ui->sohran_v_file->setEnabled(false);
    ui->del_el_from_l->setEnabled(false);
    ui->age_output->setText(QString::number(defaul.get_age()));
    ui->species_output->setText(defaul.get_specie());
    ui->clear_output->setText(QString::number(defaul.get_clear_status()));
```

```
    ui->food_output->setText(QString::number(defaul.get_food_status()));
    ui->vent_output->setText(QString::number(defaul.get_vent_status()));
    QFile file("C:/data/bird_cams.txt");
    if (file.open(QIODevice::ReadOnly|QIODevice::Text))
    {
        QTextStream steram(&file);
        QString tem;
        tem = steram.readLine();
        int q = tem.toInt();
        for (int i=0; i<q; i++)
        {
            QString l1, l2, l3, l4, l5, l6;
            l1 = steram.readLine();
            l2 = steram.readLine();
            l3 = steram.readLine();
            l4 = steram.readLine();
            l5 = steram.readLine();
            l6 = steram.readLine();
            QString ltemp = l1+" ("+l3+")";
            birds temp(l1, l2.toInt(), l3.toInt(),
l4.toInt(),l5.toInt(),l6.toInt());
            v.push_back(temp);
            ui->list_cams->addItem(ltemp);
        }
        file.close();
    }
    connect(ui->choose_cur, &QPushButton::clicked, this,
&camers::button_current_list_item);
    connect(ui->clear_fixat, &QPushButton::clicked, this,
&camers::button_clear);
    connect(ui->food_fixat, &QPushButton::clicked, this,
&camers::button_food);
    connect(ui->vent_fixat, &QPushButton::clicked, this,
&camers::button_vent);
    connect(ui->del_el_from_l, &QPushButton::clicked, this,
&camers::button_deletion);
    connect(ui->add_el_to_list, &QPushButton::clicked, this,
&camers::button_add_new);
    connect(ui->sohran_v_file, &QPushButton::clicked, this,
&camers::button_save_to_f);
}
void camers::button_current_list_item()
{
    int ind = ui->list_cams->currentRow();
    curren = v.at(ind);
    ui->age_output->setText(QString::number(curren.get_age()));
    ui->species_output->setText(curren.get_specie());
    ui->clear_output->setText(QString::number(curren.get_clear_status()));
    ui->food_output->setText(QString::number(curren.get_food_status()));
    ui->vent_output->setText(QString::number(curren.get_vent_status()));
    if (curren.get_clear_status()<40)
    {
        ui->clear_fixat->setEnabled(true);
    }
    if (curren.get_food_status()<15)
        ui->food_fixat->setEnabled(true);
    if (curren.get_vent_status()<35)
        ui->vent_fixat->setEnabled(true);
    ui->add_el_to_list->setEnabled(true);
    ui->sohran_v_file->setEnabled(true);
    ui->del_el_from_l->setEnabled(true);
}
void camers::button_food()
{
```

```
    curren.give_food();
    ui->food_output->setText(QString::number(curren.get_food_status()));
    ui->food_fixat->setEnabled(false);
}
void camers::button_clear()
{
    curren.clean_camera();
    ui->clear_output->setText(QString::number(curren.get_clear_status()));
    ui->clear_fixat->setEnabled(false);
}
void camers::button_vent()
{
    curren.fix_vent();
    ui->vent_output->setText(QString::number(curren.get_vent_status()));
    ui->vent_fixat->setEnabled(false);
}
void camers::button_deletion()
{
    unsigned int ind = ui->list_cams->currentRow();
    delete ui->list_cams->takeItem(ind);
    for (unsigned int i=ind; i<v.size() - 1; i++)
    {
        v.at(i)=v.at(i+1);
    }
    v.pop_back();
    ui->age_output->setText(QString::number(defaul.get_age()));
    ui->species_output->setText(defaul.get_specie());
    ui->clear_output->setText(QString::number(defaul.get_clear_status()));
    ui->food_output->setText(QString::number(defaul.get_food_status()));
    ui->vent_output->setText(QString::number(defaul.get_vent_status()));
    ui->clear_fixat->setEnabled(false);
    ui->vent_fixat->setEnabled(false);
    ui->food_fixat->setEnabled(false);
    ui->add_el_to_list->setEnabled(false);
    ui->sohran_v_file->setEnabled(false);
    ui->del_el_from_l->setEnabled(false);
}
void camers::button_save_to_f()
{
    QFile file("C:/data/bird_cams.txt");
    if (file.open(QIODevice::ReadWrite|QIODevice::Truncate|QIODevice::Text))
    {
        QTextStream steram(&file);
        for (unsigned int i=0; i<v.size(); i++)
        {
            steram<<v[i].get_specie()<<endl;
            steram<<v[i].get_age()<<endl;
            steram<<v[i].get_cam_num()<<endl;
            steram<<v[i].get_vent_status()<<endl;
            steram<<v[i].get_food_status()<<endl;
            steram<<v[i].get_clear_status()<<endl;
        }
        file.close();
    }
}

void camers::button_add_new()
{
    QFile file("C:/data/new_el.txt");
    if (file.open(QIODevice::ReadOnly|QIODevice::Text))
    {
        QTextStream steram(&file);
        QString l1, l2, l3, l4, l5, l6;
        l1 = steram.readLine();
```

```
        l2 = steram.readLine();
        l3 = steram.readLine();
        l4 = steram.readLine();
        l5 = steram.readLine();
        l6 = steram.readLine();
        QString ltemp = l1+" ("+l3+")";
        birds temp(l1, l2.toInt(), l3.toInt(),
l4.toInt(),l5.toInt(),l6.toInt());
        v.push_back(temp);
        ui->list_cams->addItem(ltemp);
        file.close();
    }
}

camers::~camers()
{
    delete ui;
}
```

camers.h:

```
#ifndef CAMERS_H
#define CAMERS_H
#include <fstream>
#include <QFile>
#include <QMainWindow>
#include <QListWidget>
#include <QTextStream>
#include <vector>
QT_BEGIN_NAMESPACE
namespace Ui { class camers; }
QT_END_NAMESPACE

class birds
{
private:
    QString specie;
    int age;
    int cam_num;
    int vent_status;
    int food;
    int clear_status;
public:
    birds()
    {
        specie="";
        cam_num=0;
        vent_status=0;
        age=0;
        food=0;
        clear_status=0;
    }
    birds(QString s, int a, int c, int v, int f, int cl)
    {
        specie=s;
        cam_num=c;
        vent_status=v;
        age=a;
        food=f;
        clear_status=cl;
    }
    QString get_specie()
    {
        return specie;
    }
```

```
    int get_age()
    {
        return age;
    }
    int get_cam_num()
    {
        return cam_num;
    }
    int get_vent_status()
    {
        return vent_status;
    }
    int get_food_status()
    {
        return food;
    }
    int get_clear_status()
    {
        return clear_status;
    }
    void set_age(int ag)
    {
        age=ag;
    }
    void set_species(QString spec)
    {
        specie=spec;
    }
    void set_cam_num(int a)
    {
        cam_num=a;
    }
    ~birds(){}
    void fix_vent()
    {
        vent_status=100;
    }
    void give_food()
    {
        food=100;
    }
    void clean_camera()
    {
        clear_status=100;
    }
};

class camers : public QMainWindow
{
    Q_OBJECT

public:
    camers(QWidget *parent = nullptr);
    ~camers();
private:
    Ui::camers *ui;
public slots:
    void button_clear();
    void button_food();
    void button_vent();
    void button_current_list_item();
    void button_deletion();
    void button_save_to_f();
    void button_add_new();
```

```
};
#endif // CAMERS_H
```

main.cpp:

```
#include "camers.h"

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    camers w;
    w.show();
    return a.exec();
}
```

github

https://github.com/cirrew/arm_tvorch

2. Задача коммивояжера

[ещё не доделал]

UML

Анализ

Код

github