Разработка приложения «Alien attack»

Отчет о проектной работе по курсу «Основы информатики и программирования»

Костин Данила

9 июня 2021

Описание игрового процесса

Как понятно из названия игры, мы будем иметь дело с некоторыми НЛО, которые будут просто летать по карте, стараясь сбить наш самолёт.

Описание игрового процесса

Рис. 1: Игровое окно

Описание игрового процесса

Пользователь управляет коричневым самолётом, его задача не врезаться в НЛО и сбить его путём пуска ракеты. Коротко перечислим функционал, доступный пользователю:

1. Самолёт летит в место клика
2. При нажатии клавиши «Space» самолёт выпускает ракету
3. В меню имеются кнопки начала новой игры или выхода
4. В меню выводится количество сбитых НЛО и секунды до перезарядки ракеты

Описание игрового процесса

Рис. 2: Пуск ракеты

Функции самолёта

Стоит также упомянуть, что при доходе самолёта до края экрана, он отскакивает он него, как в правиле угол падения равен углу отражения. На следующем слайде будет представлен список всех рабочих модулей проекта.

Описание модулей

Все модули были поделены на две группы:

1. Реализованы на Qml
    1.1 main
    1.2 Plane
    1.3 Enemy
    1.4 GameFuncs
    1.5 Rocket
2. Реализованы на C++
    2.1 helper

Модуль main

Данный модуль - основа всех модулей на qml. Он задаёт размеры окна, его фон, обрабатывает нажатие на область окна. При нажатии происходит вызов функции на языке C++, которая нормирует направляющий вектор в место клика мыши, чтобы самолёт полетел к месту клика без ускорения, а также функции поворота самолёта в направлении клика.

Часть кода обработки нажатия

```
if(mouse.button === Qt.LeftButton)
{

    xold = mouse.x - planer.x - 45
    yold = mouse.y - planer.y - 45
    vector()

    var p = helper.atang(xold, yold);

    planer.rotation = p[0];
}
```

Свойства объектов qml

Свойство	Назначение
x_s | y_s	Отражают скорость полёта самолёта
r_x | r_y	Отражают скорость полёта ракеты
xold | yold	Являются координатами точки клика
reload	Количество секунд до перезарядки
score	Счёт сбитых НЛО
ready_strike	Флаг готовности пуска ракеты

Таблица 1: Глобальные свойства

Модуль Plane

В данном модуле происходит прибавление к координатам самолёта координат некоторого вектора, вычисленного после клика мыши. Также здесь присутствует проверка того, чтобы самолёт не улетел за предел видимой части окна.

Часть кода перемещения самолёта

```
Timer{
    interval: 90;running: true; repeat: true;
    onTriggered:{
        if (plane.x > 1105 || plane.x < 0){

            x_s*=-1
            plane.x +=2*x_s
            var t = helper.reject(plane.rotation)
            plane.rotation = t[0] + 180
        }
        else
        {
            plane.x += x_s
        }
```

Модуль Enemy

Данный модуль содержит функции перемещения НЛО к некоторой точке в центре экрана по таймеру. Другая функция - это присваивание случайных координат НЛО, если он был сбит или улетел за пределы окна. Также в данном модуле происходит проверка того, что НЛО столкнулось с выпущенной ракетой. Если столкновение произошло, то ракета убирается из видимой части окна, счёт увеличивается, и происходит генерация нового положения НЛО.

Часть кода перемещения самолёта

```
Timer{
    interval: 40;running: true; repeat: true;

    onTriggered:{
        ele() //function of generating UFO position
        first.x += c_sx*3
        first.y += c_sy*3
        var t = helper.rast(first.x, first.y, rok.x, rok.y)
        if (t[0] === 1){
            score++
            rok.x = -3000
            rok.y= -3000
        }
        if (t[0] === 1 || first.x >= 1400 || first.y >= 900){
            enemy_flag = true
        }
    }
```

Модуль GameFuncs

Модуль основных функций меню снизу окна, а также отображения счёта, обработки перезарядки, вывода оставшихся секунд, а также содержит функцию открытия диалога с концом игры (при столкновении самолёта с НЛО).

Задаёт кнопки новой игры, при нажатии которой самолёт смещается в начальную точку расположения на взлётной полосе, и кнопку выхода.

При выстреле ракеты задаётся флаг (ready_strike) того, что начата перезарядка, и пока количество секунд не дойдет до 0, функция check каждую секунду будет уменьшать время перезарядки.

Функция проверки возможности стрелять

```
function check(){

    if (ready_strike)
        tex.text = "Ready"

    if (!ready_strike || remem == 0)
        {
            tex.text = remem
            if (remem === 0){
                ready_strike = true
                remem = reload
                return
            }
            remem--
        }
    }
```

Функция проигрыша

```
function lose(){
        var t = helper.rast(pl.x+45, pl.y+45, f.x+100, f.y+70)
        if (t[0] == true){
            planer.x = 2000
            planer.y = 2000
            losing.open()
        }
    }
```

Функции самолёта

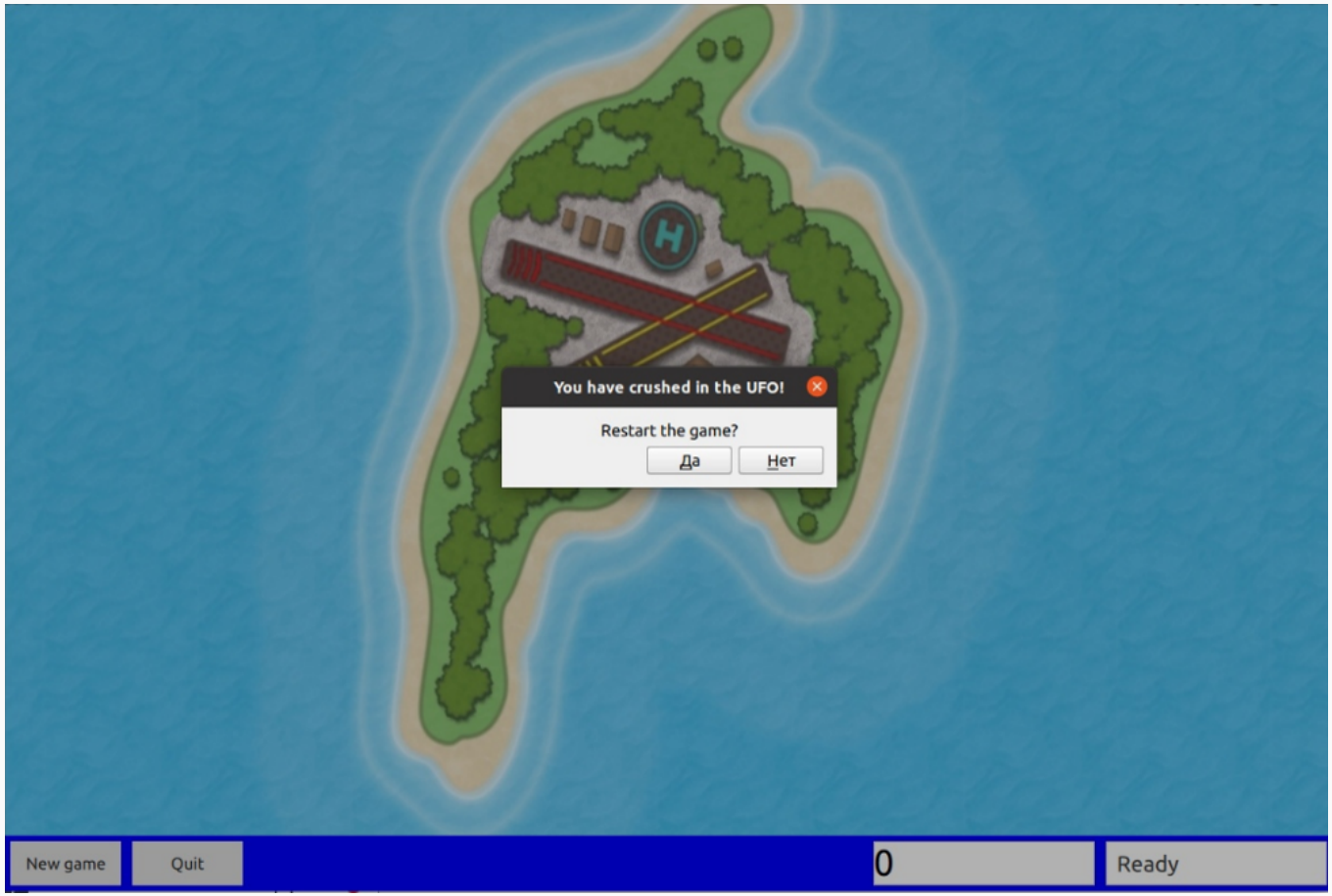

Рис. 3: Диалог о проигрыше

Функции модуля helper

Далее будут перечислены все функции модуля helper:

1. rast
2. rand
3. atang
4. reject
5. vector

Функция rast

Данная функция находит расстояние между двумя точками по следующей формуле:

$$length = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2};$$

Она возвращает true, если расстояние между точками меньше 70 пикселей (это расстояние было выбрано экспериментально, оно удовлетворяет размерам объектов)

Функция вычисление расстояния

```
QVector<double> Helper::rast(double x, double y,
double x1, double y1){

    bool f = false;
    double dist = sqrt((x-x1+70)*(x-x1+70)+(y-y1+50)*(y-y1+50));

    if (dist < 70)
        f = true;
    QVector <double> p;
    p.push_back(f);
    return p;
}
```

Функция rand

Данная функция случайным образом генерирует число, получает от него остаток от деления на 4. Полученное число - есть сторона, с которой прилетит НЛО. Так, если это число будет 0, то НЛО прилетит сверху, 1 - справа, и так далее по часовой стрелке.

Функция генерации стороны

```
QVector<double>Helper::rand(){
    QVector<double> copy;

    QTime midnight(0,0,0);
    qsrand(midnight.secsTo(QTime::currentTime()));
    double x2 = 0;
    double y2 = 0;
    int koef = qrand() % 4;

    if (koef == 0){
        y2 -=400;
        x2 = qrand() % 1200;
    }
    ...
```

Функция atang

Данная функция получает координаты вектора, направляющего полёт самолёта, получает из них тангенс, и через формулу арктангенса выражает угол поворота самолёта в градусах:

$$rotation = \frac{\text{arctg tg } \frac{y}{x} * 180}{\pi};$$

Функия поворота в градусах

```
QVector<double>Helper::atang(double x, double y){
    QVector<double> copy;
    double tan = (y/x);
    double atan = qAtan(tan)*180/M_PI-90;
    if ((x > 0 && y < 0) || (x>0 && y>0))
        atan +=180;
    QVector<double> cop;
    cop.push_back(atan);
    return cop;
}
```

Функция vector

Данная функция получает координаты вектора и нормирует их - то есть делит каждую координату на длину вектора:

$$norma_x = \frac{x}{\sqrt{x^2 + y^2}}; norma_y = \frac{y}{\sqrt{x^2 + y^2}};$$

Функция нормирования вектора

```
QVector<double> Helper::vector(double x, double y){
    QVector<double> copy;
    double x2 = x/(sqrt(x*x+y*y));
    double y2 = y/(sqrt(x*x+y*y));

    copy.push_back(x2);
    copy.push_back(y2);
    return copy;
}
```

Функция vector

Данная функция получает поворот самолёта в градусах, и чтобы найти угол, который нужно будет присвоить самолёту при его столкновении с границей окна, из 180 вычитается текущий поворот. Таким образом будет найден угол - разность 180 и угла падения, он же совпадает с углом отражения.

Функия поворота в градусах

```
QVector<double> Helper::reject(double x){
    QVector<double> coqy;
    double grad;
    if (x > 0)
    {
        grad = 180 - x;
    }
    if (x < 0)
    {
        grad = -180 - x;
    }
    if (x==0 || x == 90 || x ==-90)
        grad = 180;
    coqy.push_back(grad);
    return coqy;
}
```

Заключения

В заключение можно сказать, что была создано функционирующее приложение с графическим интерфейсом. Данное приложение получилось достаточно объемным в связи с большим количеством взаимодействий объектов таких, как ракета и НЛО, самолёт и НЛО, самолёт и граница окна и так далее. В проекте использовался как язык qml, так и C++, без которого не возможна была бы реализация большинства функций приложения.

Спасибо за внимание !