

Создание 2D игры на языке программирования C# с помощью библиотеки WinForms

Эрдман Александр Алексеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассмотрен процесс создания 2D игры без использования игровых движков по типу Unity3D, Unreal Engine 5 и подобных. Программа написана на языке программирования C# и системной библиотеки WinForms. Результатом исследования является оконное приложение и полное описание его работы.

Ключевые слова: C#, WinForms, разработка игр

Creating a 2D game in the C# programming language using the WinForms library

Erdman Alexander Alekseevich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article describes the process of creating a 2D game without using game engines such as Unity3D, Unreal Engine 5 and the like. The program is written in the C# programming language and the WinForms system library. The result of the research is a windowed application and a complete description of its operation.

Keywords: C#, WinForms, game development

1 Введение

1.1 Актуальность

Создание 2D игры на языке программирования C# с использованием библиотеки WinForms представляет собой процесс, который позволяет разработчикам глубже понять основы программирования и графического интерфейса. В отличие от использования специализированных игровых движков, таких как Unity или Unreal Engine, разработка игры с помощью WinForms требует более детального погружения в код и работу с графическими элементами напрямую. Этот подход не только развивает навыки программирования, но и предоставляет полный контроль над всеми аспектами игры, от графики до логики. В данном исследовании рассматриваются основные шаги и приемы, необходимые для создания простой 2D игры, начиная с настройки проекта и заканчивая реализацией игровой механики.

1.2 Обзор исследований

Д.К. Лопатин в своём труде изучил тему разработки простых интерфейсов с использованием библиотеки WinForms [1]. Д.А. Михайличенко рассмотрел ключевую ошибку проектирования программного продукта на этапе программирования пользовательского интерфейса [2]. Е.А. Пасюкова изучила вопрос о фреймворках для создания графических приложений и составила сравнительную таблицу по выбору более оптимального решения [3]. А.С. Маринчук в своей статье описал процесс создания популярной игры Flappy Bird с помощью языка программирования C# в Microsoft Visual Studio [4]. М.И. Половинченко и В.С. Елисеев рассмотрели самые распространенные движки «конструкторы» для создания 2D игр и провели сравнительный анализ между ними [5].

1.3 Цель исследования

Целью исследования является создание графического приложения на языке программирования C# в виде 2D игры на базе библиотеки WinForms.

2 Материалы и методы

Для создания приложения используется язык программирования C# и системная библиотека WinForms. В качестве среды разработки выступает Visual Studio 2019.

3 Результаты и обсуждения

Перед разработкой проекта проводится подготовка, которая заключается в поиске материалов. Из сторонних ресурсов выделяются изображения в формате .png, которые будут использоваться в проекте.

Следующим шагом является создание самого проекта в Visual Studio. Так как работа будет происходить непосредственно с библиотекой WinForms, то для этого выбирается соответствующий шаблон, содержащий в себе пустое графическое окно. После создания проекта производится масштабирование окна под размеры 785 на 611 пикселей ширины и высоты соответственно. Также добавляются элементы «pictureBox». Данные элементы представляют из себя графические программируемые объекты с заготовленными методами по умолчанию. Именно данные элементы будут представлять декорацию для игры, модель игрока, противников и подбираемого объекта в виде монеты для счёта очков. Помимо описанных объектов, также используется элемент «timer», который будет выполнять роль цикла для хода всей логики игры.

После первичной настройки проекта происходит добавление в «pictureBox» картинки игрока и противников в виде разных автомобилей. Далее загружается задний фон игры – изображение дороги. Данное изображение добавляется два раза в разные «pictureBox». Данное решение необходимо для того, чтобы сделать эффект движения автомобилей, то есть анимации. Такие же действия проделываются и для объекта «монетка», которая будет выступать счётчиком очков игрока. В последнюю очередь

добавляются элементы «button» и «label» для кнопки перезапуска и надписей соответственно.

После настройки графической составляющей проекта, будущее приложение будет выглядеть следующим образом (рис. 1).

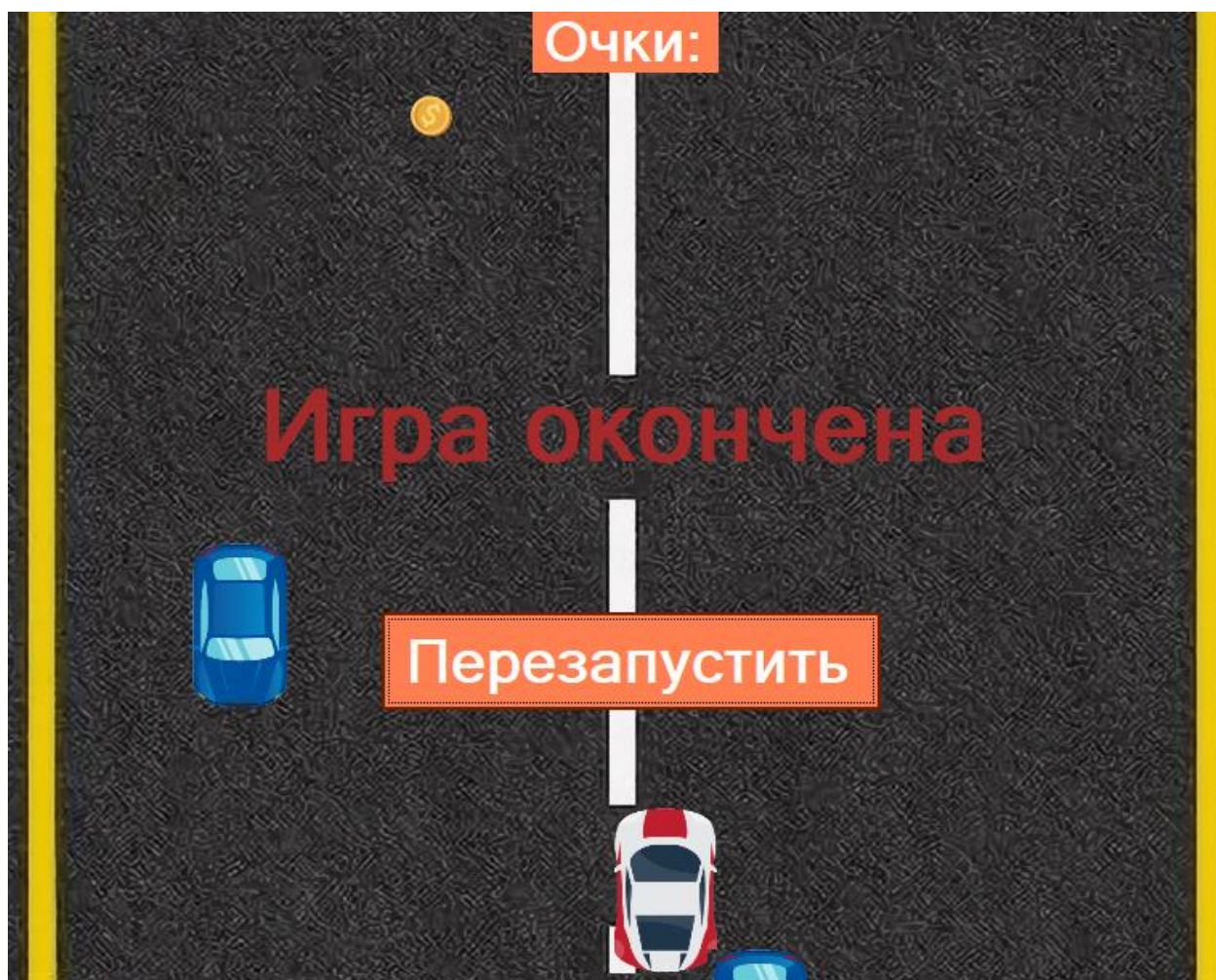


Рисунок 1. Внешний вид разрабатываемой игры

Геймплей игры следующий: игрок в виде автомобиля «движется» вперёд и попутно собирает монеты. Одновременно с этим в качестве препятствия двигаются автомобили противников на встречу игроку. При касании игрока и противника происходит остановка игры, где можно увидеть свой счёт и повторить заезд.

После разработки графического вида игры происходит программирование логики. В классе «Form1.cs», где происходит инициализация и загрузка графической формы, создаются четыре переменные: одна переменная типа «int» для хранения значения подобранных монет; вторая типа «Point» создаёт объект курсора, с помощью которого можно будет перемещать окно с игрой в удобное место на рабочем столе; третья типа «bool», которая нужна для отслеживания нажатия на окно приложения; последняя переменная с типом «bool» отвечает за обозначение прекращения игры.

После этого создаётся анимация движения автомобиля игрока. Фактически автомобиль игрока будет неподвижен относительно координаты Y, вместо этого будет двигаться сама дорога. Для реализации более плавного движения дороги и был создан второй «pictureBox», содержащий в себе копию изображения дороги. Для создания анимации используется метод «timer_Tick» элемента «timer». Данный метод позволяет выполнять все команды, находящиеся в нём. Внутри данной функции прописываются команды, которые равномерно изменяют положение дороги до указанной координаты, при достижении которой изображения дорог возвращаются в первоначальные координаты (рис. 2).

```
int speed = 10;
bg1.Top += speed;
bg2.Top += speed;

if (bg1.Top >= 650)
{
    bg1.Top = 0;
    bg2.Top = -600;
}
```

Рисунок 2. Команды анимации дороги

Аналогичным образом программируются противники с особенностью случайного появления в заданном диапазоне (рис. 3).

```
int enemySpeed = 7;
enemy1.Top += enemySpeed;
enemy2.Top += enemySpeed;

if (enemy1.Top >= 650)
{
    enemy1.Top = -200;
    Random rand = new Random();
    enemy1.Left = rand.Next(0, 300);
}

if (enemy2.Top >= 650)
{
    enemy2.Top = -500;
    Random rand = new Random();
    enemy2.Left = rand.Next(400, 750);
}
```

Рисунок 3. Движение противников

Здесь повторяется логика, как и с дорогой, только используются случайные координаты появления по оси X. В качестве генерации случайных чисел выступает объект класса «Random» с методом «Next()».

При старте игры в данном варианте получится так, что создаётся иллюзия движения машины игрока и полноценное движение машин противника.

Следующим шагом является скрытие элементов надписи и кнопки с помощью общего свойства «Visible = false». Это нужно для того, чтобы указанными элементы не появлялись при старте и самом геймплеи игры.

Реализации события завершения игры представляет из себя условную конструкцию с отслеживанием касания двух объектов – машины игрока и машины одного из противников (рис. 4).

```
if (player.Bounds.Intersects(enemy1.Bounds) || player.Bounds.Intersects(enemy2.Bounds))
{
    timer.Enabled = false;
    gameover.Visible = true;
    buttonRestart.Visible = true;
    lose = true;
}
```

Рисунок 4. Реализация остановки игры

При касании двух объектов с помощью метода «Bounds.Intersects()» происходит остановка таймера, что в свою очередь останавливает все команды, которые в нём находятся, а именно движение всех объектов. Также с помощью свойств «Visible = true» происходит появление надписи об окончании игры и кнопки перезапуска. Переменная «lose» принимает значение «true», что в свою очередь позволит останавливать управление игроком, которое будет описано далее.

Аналогичный подход используется для подбора «монеты» и считывания очков игрока (рис. 5)

```
if (player.Bounds.Intersects(coin.Bounds))
{
    coins++;
    score.Text = "Очки: " + coins;
    coin.Top = -200;
    Random rand = new Random();
    coin.Left = rand.Next(0, 750);
}
```

Рисунок 5. Реализация счётчика очков

В данной реализации происходит увеличение переменной «coins», а также вывод значения очков в текстовую надпись «label». После чего «монета» возвращается в случайное место по оси X.

Движение автомобилем игрока осуществляется с помощью встроенной функции формы «KeyDown» (рис. 6).

```
private void Form1_KeyDown(object sender, KeyEventArgs e)
{
    if (lose)
    {
        return;
    }
    int speed = 10;
    if ((e.KeyCode == Keys.Left || e.KeyCode == Keys.A) && player.Left > 50)
    {
        player.Left -= speed;
    }
    else if ((e.KeyCode == Keys.Right || e.KeyCode == Keys.D) && player.Right < 750)
    {
        player.Left += speed;
    }
}
```

Рисунок 6. Функция управления автомобилем игрока

В данном коде присутствует конструкция для проверки значения переменной «lose» и в зависимости от её значения запускаются конструкции управления или функция возвращает пустой результат. Скорость движения по оси X задается переменной «speed». Сдвиг влево или вправо осуществляется тогда, когда нажаты клавиши «Стрелка влево» и «А» для движения влево и «Стрелка вправо» и «D» для движения вправо. Данный алгоритм реализуется с помощью простого сравнения нажатых клавиш с заданными.

Далее следует создание обработчика события нажатия на кнопку «Перезапустить» (рис. 7).

```
private void buttonRestart_Click(object sender, EventArgs e)
{
    enemy1.Top = -200;
    enemy2.Top = -500;
    gameover.Visible = false;
    buttonRestart.Visible = false;
    timer.Enabled = true;
    lose = false;
    coins = 0;
    coin.Top = -200;
    score.Text = "Очки: " + coins;
}
```

Рисунок 7. Обработчик события нажатия на кнопку «Перезапустить»

Команды здесь схожи из кода касания автомобилей только имеется дополнительное обнуление счёта и возврат «монеты» на своё место старта.

Основной функционал игры сделан. Дополнительно создаётся функция «KeyPress», с помощью которой происходит закрытие игры по нажатию на клавишу «Escape». Также создаются методы для возможности перемещения окна с игрой в удобное место на рабочем столе (рис. 8).

```
InitializeComponent();

bg1.MouseDown += MouseClickDown;
bg1.MouseUp += MouseClickUp;
bg1.MouseMove += MouseClickMove;
bg2.MouseDown += MouseClickDown;
bg2.MouseUp += MouseClickUp;
bg2.MouseMove += MouseClickMove;
gameover.Visible = false;
buttonRestart.Visible = false;
KeyPreview = true;
}

2 references
private void MouseClickDown(object sender, MouseEventArgs e)
{
    dragging = true;
    pos.X = e.X;
    pos.Y = e.Y;
}

2 references
private void MouseClickUp(object sender, MouseEventArgs e)
{
    dragging = false;
}

2 references
private void MouseClickMove(object sender, MouseEventArgs e)
{
    if (dragging)
    {
        Point currPoint = PointToScreen(new Point(e.X, e.Y));
        this.Location = new Point(currPoint.X - pos.X, currPoint.Y - pos.Y + bg1.Top);
    }
}

1 reference
private void Form1_KeyPress(object sender, KeyPressEventArgs e)
{
    if(e.KeyChar == (char)Keys.Escape)
        this.Close();
}
```

Рисунок 8. Методы перемещения окна по рабочему столу и метод закрытия приложения по нажатию на клавишу

Так как приложение не имеет рамки, то для перемещения окна используются «pictureBox», в которые помещены изображения дороги. При зажатой кнопки мыши на них можно перемещать окно с игрой в любое место на экране.

В результате исследование была создана 2D игра на базе системной библиотеки WinForms языка программирования C#. Был описан ход разработки приложение и объяснение всей логики игры.

Библиографический список

1. Лопатин Д.К. Разработка простых интерфейсов с использованием библиотеки WinForms // Журнал научных публикаций аспирантов и докторантов. 2014. № 3 (93). С. 265-267.
2. Михайличенко Д.А. Ключевая ошибка проектирования программного продукта в WinForm Microsoft Visual Studio // Труды Ростовского государственного университета путей сообщения. 2016. № 2. С. 38-40.
3. Пасюкова Е.А. Обзор технических средств разработки графического интерфейса на языке программирования C# // Постулат. 2021. № 2 (64).
4. Маринчук А.С. Создание игры Flappy Bird в Microsoft Visual Studio на языке программирования C# // Постулат. 2021. № 1 (63).
5. Половинченко М.И., Елисеев В.С. Анализ движков для создания 2D игр // Мехатроника, автоматика и робототехника. 2020. № 6. С. 73-75.