

Министерство науки и образования Российской Федерации

Пензенский Государственный Университет

Кафедра «Системы автоматизированного проектирования»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

по курсу «Технологии разработки программного обеспечения»

на тему «Разработка компьютерной игры»

Выполнил:

Студент группы 18ВВЗ

Уразов Н.С.

Приняла:

К.т.н., доцент Эпп В.В.

Пенза 2022

РЕФЕРАТ

Пояснительная записка содержит 28 листов, 4 рисунка, 1 таблицу и 4 используемых источника.

С#, ООП, Объекты, Классы, Наследование, Инкапсуляция, Полиморфизм, Разработка, Игры.

Цель работы: разработать компьютерную 2D игру

Технологии разработки: Среда программирования Microsoft visual studio 2019, язык программирования С#, инструмент разработки – MonoGame.

Результаты работы: была проанализирована предметная область. В соответствии с этим спроектировано и разработано приложение.

И	Пис	№ докум	Подпи	Па					
Разраб		Учредител Н.С.			Разработка компьютерной игры	Пис	Пис	Пис	
Пис		Эксперт В.В.							
И	Конт								
Учред									
						ар. 18BB3			

Оглавление

Введение	5
1. Постановка задачи	6
2. Общие сведения	7
3. Выбор решения.....	9
3.1. Выбор жанра	9
3.2. Проектирование структуры приложения	9
4. Описание разработки приложения	10
4.1. Разработка объектов, спрайтов и анимации.....	10
5. Тестирование приложения	11
6. Технические и программные средства.....	13
Заключение.....	14
Используемая литература.	15
Приложение.....	16

					4	П.И.
Иг	П.И.	Но докум	Подпи	Па		

Введение

Программное обеспечение (ПО) – набор компонент, управляющих работой компьютера. Оно предназначено для обеспечения функционирования вычислительной машины, позволяет использовать ЭВМ для решения различных задач широкого профиля.

Выделяют несколько видов программного обеспечения, одним из которых является прикладное ПО. Целью его использования является выполнение разнообразных задач, таких как создание текстовых или графических документов, воспроизведение мультимедийной информации, хранение и обработка данных, выполнение сложных математических расчетов, проектирование в различных предметных областях и использование сети Интернет. Одним из самых распространенных видов прикладного программного обеспечения являются компьютерные игры.

Компьютерные игры стали самым популярным и известным видом программного обеспечения. Они пользуются популярностью у людей разного возраста и социального статуса, не зависимо от континента. Игры позволяют отдохнуть и получить приятные эмоции от их прохождения, что не может не привлекать внимание людей.

Игровая индустрия развивается все более быстрыми темпами. Множество компаний и миллионы разработчиков трудятся над созданием качественного продукта, которым будут пользоваться миллионы игроков по всему миру.

Компьютерные игры распространены не только на персональных компьютерах, но и на множестве других устройств, таких как игровые консоли, ноутбуки, планшетные компьютеры и даже мобильные устройства.

Таким образом, разработка компьютерных игр является перспективным направлением интеллектуальной и творческой деятельности не только для разработчиков, но и для дизайнеров, художников и даже сценаристов.

					5	П.И.
Иг	П.И.	Но докум	Подпи	Па		

1. Постановка задачи

В настоящей курсовой работе необходимо разработать 2D игру жанра аркада.

Идея игры заключается в том, что Игроку необходимо попасть во все треугольники.

Исходя из вышесказанного, необходимо проанализировать предметную область, спроектировать классы и их методы для соответствующих игровых объектов.

Для повышения качества разрабатываемой игры следует предусмотреть визуальные эффекты: разработать игровые спрайты, анимацию игры.

Среда разработки MonoGame

Язык программирования: C#

Операционная система: Windows 10.

Важно отметить, что язык C# является одним из самых мощных, быстро развивающихся и востребованных языков программирования. В C# предлагается расширенная реализация парадигмы объектной ориентации, которая включает инкапсуляцию, наследование и полиморфизм.

						П...
Иг	П...	Но докум	Подпи	Па		

2. Общие сведения

2.1. Терминология

Игровая терминология складывается под сильным влиянием англоязычных названий. Часто в русском языке довольно сложно найти хорошую аналогию принятым английским терминам, поэтому в сфере игровой терминологии существует двойственность – довольно часто создатели игр пользуются англоязычными терминами в то время, как и русскоязычные названия так же находят применение.[4]

Sprite – этот термин часто заменяют русскоязычным неологизмом спрайт. В игровой индустрии синонимами слова спрайт являются такие слова, как "изображение", "картинка", иногда пользуются словом "текстура", однако обычно это понятие несет несколько иную смысловую нагрузку. Как правило спрайт – это двумерное изображение, причем, в узком смысле слова это – лишь изображение, а в широком – это игровой объект, который обладает гораздо более широким набором возможностей, нежели обычное изображение.

Texture – текстура. Обычно текстурами называют двумерные изображения, которые "накладывают" на трехмерные модели. В терминологии.

Background – фон. Так называется изображение, обычно – соответствующее размерам игрового поля, которое является фоном для других изображений. Фон может быть неподвижным и подвижным.

						П..
Иг	П..	Но докум	Подпи	Па		

2D-game – двумерная игра – игра, в которой использованы двумерные изображения.

Camera (камера) – так называют точку в игровом пространстве, с которой игрок видит игровой мир. С точки зрения положения камеры игры можно подразделить на игры от первого лица (камера расположена так, что реализует вид как бы "из глаз" персонажа), игры с видом слева-сверху (стратегии), игры с видом сзади – камера располагается обычно позади игрока и немного выше его. Существуют и игры, где камера может принимать различные позиции, например, реализующие вид из глаз персонажа и вид сзади.

						П...
Из	П...	Но докум	Полдп...	Па		

3. Выбор решения.

3.1. Выбор жанра

Было принято решение разработать двумерный однопользовательский платформер для персональных компьютеров. Данное решение было принято потому, что платформер обладает простой и понятной механикой, что означает легкость в разработке для начинающих игровых разработчиков. Так же двумерные игры проще в производстве, в отличие от трехмерных

3.2. Проектирование структуры приложения

Структура приложения зависит от используемой технологии разработки. В данном проекте использован язык программирования C# совместно с игровым движком MonoGame.

В начале игры выполняется загрузка всех объектов. При закрытии игры автоматически полностью выгружает игру из памяти компьютера, удаляет все объекты. Вышеописанные методы являются встроенными, и создаются по умолчанию.

При разработке программы использовалась концепция объектно-ориентированного программирования. Объектно-ориентированное программирование — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования. Такой подход называется восходящим программированием. Программа образуется путем объединения простых программных компонент для создания компонент более высокого уровня. Разрабатываемая игра должна иметь два уровня.

Поэтому необходим механизм классификации, известный как наследование. Наследование позволяет описать новый класс на основе уже существующего (родительского), при этом свойства и функциональность родительского класса заимствуются новым классом.

						Плн
Ис	Плн	Подпись	Подпись	Па		

4. Описание разработки приложения

4.1. Разработка объектов, спрайтов и анимации.

В разрабатываемом приложении был создан объект «triangle» и «circle».



Рисунок 4.1 – объект «triangle»



Рисунок 4.2 – объект и «circle».

При столкновении треугольника и круга, они исчезают.

						Пн
Иг	Пн	Но докум	Подпи	Па		

5. Тестирование приложения

При запуске игры, игрока встречает игровое поле.

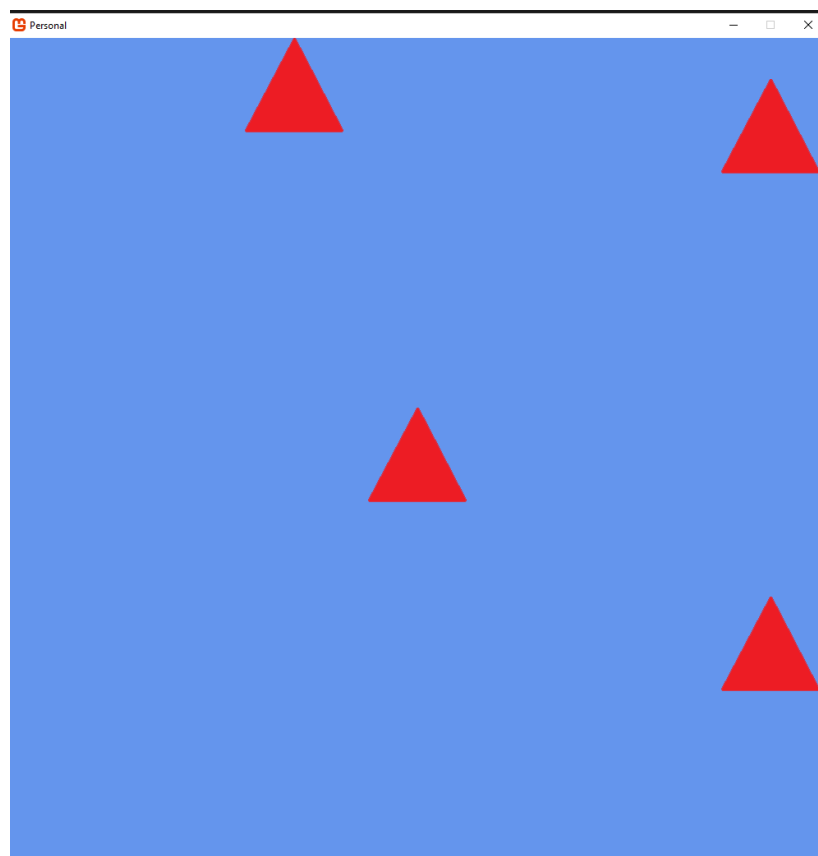


Рисунок 5.1 –Игровое поле.

						П...
Иг	П...	Но докум	Подпи	Па		

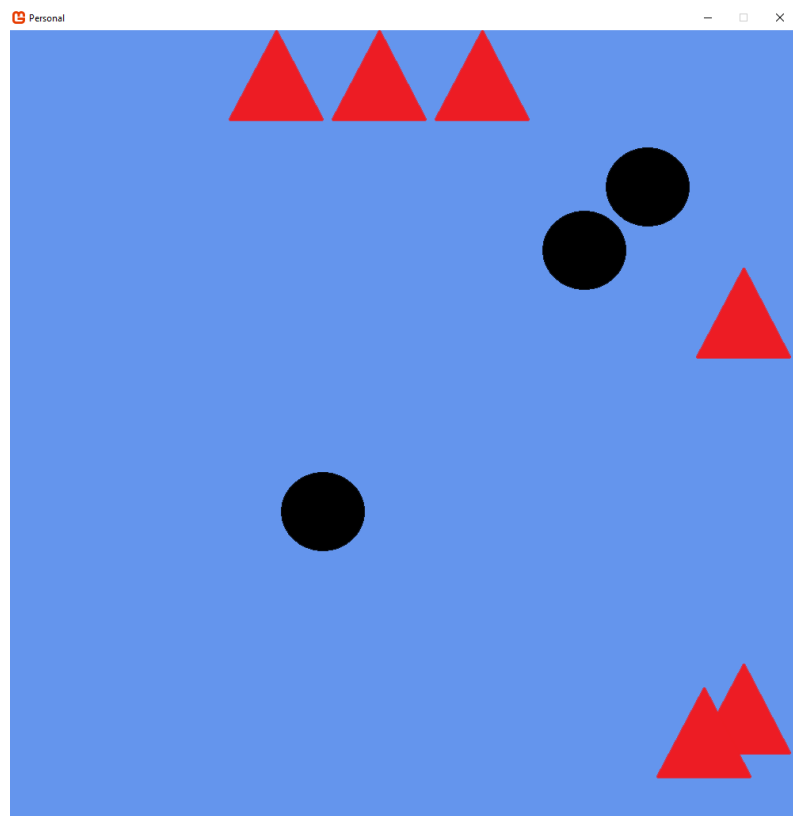


Рисунок 5.2 – Процесс игры.

						Пл
Иг	Пл	Но докум	Подпи	Па		

6. Технические и программные средства

Игровые требования были получены на основе просмотра статистики из диспетчера задач, и монитора ресурсов.

1. Процессор, имеющий 32\64 разрядную архитектуру с тактовой частотой от 1 ГГц и выше.
2. Объём видео памяти примерно 400 мб
3. Размер оперативной памяти 220 Мб. (для 32х разрядного процессора) и 400МБ. (для 64-х разрядного процессора), и выше
4. Операционная система Windows 7 и старше.

						П..
Из	П..	По докум	Подпи	Па		

Заклучение

В ходе выполнения курсового проекта была разработана 2D. Разработаны игровые объекты и логика игры.

Программа написана на языке C# методом восходящего программирования с использованием возможностей Monogame.

						П.И.
Иг	П.И.	Но докум	Подпи	Па		

Используемая литература.

1. Джозеф Албахари, Бен Албахари С# 7.0. Справочник Полное описание языка. – М.: Питер, 2018 – 1026 с.
2. Эндрю Троелсен, Филипп Джепикс Язык программирования С#7 и платформы .NET и .NET Core 8-е издание– М.: Питер, 2018 – 1330 с.
3. Потапова Л.Е. Объектно-ориентированное программирование на языке с#/ Л.Е. Потапова. – Витебск, 2012 – 121с.
4. Эпп Виталина Викторовна Разработка 2d компьютерных игр Методические указания к курсовому проектированию по дисциплине «Технология разработки программного обеспечения» 32 стр. Пенза 2012.

						П..
Иэ	П..	Но докум	Подпи	Па		

Приложение

Приложение А Листинг кода.

Game1.cs

```
using System;
using System.Collections.Generic;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Content;
using Personal.Code;

namespace Personal
{
    public class Game1 : Game
    {
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;
        Rectangle scrBounds;
        Sprite fon;
        Sprite btn_play;
        GameState gameState = new GameState();
        Player player;
        Enemies enemy;

        Texture2D textureEnemies;
        Random randNum;
        MouseState mouse;
        int a = 0;
        private double click__timer = 0;
        private double c__timer = 0;
        public static int ScreenWidth;
        public static int ScreenHeight;
        int count = 20;
        int n = 0;
        int gg = 0;
        public bool IsWin { get; set; } = false;

        int EnemiesNumber { get; set; }

        int EnemiesSpeed { get; set; }

        bool wasPressed = false;

        public Game1(int enemiesNumber, int enemiesSpeed)
        {
            graphics = new GraphicsDeviceManager(this);
```

						П...
Иэ	П...	Но ёокум	Плопи	Па		

```

        Content.RootDirectory = "Content";
        graphics.PreferredBackBufferHeight = 1000;
        graphics.PreferredBackBufferWidth = 1000;
        IsMouseVisible = true;
        EnemiesNumber = enemiesNumber;
        EnemiesSpeed = enemiesSpeed;
    }
    int wp = 10;
    protected override void Initialize()
    {
        base.Initialize();
    }

    protected override void LoadContent()
    {
        spriteBatch = new SpriteBatch(GraphicsDevice);
        randNum = new Random();
        Services.AddService(typeof(SpriteBatch), spriteBatch);
        //textureEnemies = Content.Load<Texture2D>("circle");
        //CreateNewObject1();

    }
    protected void CreateNewObject()
    {
        if (n < EnemiesNumber) {
            /* if (c__timer <= 0)
            {*/
                enemy = new Enemies(this, Content.Load<Texture2D>("triangle"), new Vector2(0,
0), "triangle", this, EnemiesSpeed );
                Components.Add(enemy);
                n++;
                //wp = wp + 1;
                c__timer = 3f;
            }
        }
        protected void CreateNewObject1()
        {

            player = new Player(this, Content.Load<Texture2D>("circle"), new Vector2(0,880),
"player", this, 30);
            Components.Add(player);

        }

        protected override void UnloadContent()
        {
            //fon.spTexture.Dispose();
            // spriteBatch.Dispose(); ////////////????????????????????????????????????????
        }

```

						Π...
Με	Π...	Νο ὀγκύμ	Ποῦ...	Πα		

```

// TODO: Unload any non ContentManager content here

double timer = 0;
double time = 0;
/// <param name="gameTime">Provides a snapshot of timing values.</param>
protected override void Update(GameTime gameTime)
{
    if (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed ||
Keyboard.GetState().IsKeyDown(Keys.Escape)|| a == 15 || timer == 100000)
        Exit();

    graphics.ApplyChanges();

    click__timer -= click__timer > 0 ? gameTime.ElapsedGameTime.TotalSeconds : 0;
    c__timer -= c__timer > 0 ? gameTime.ElapsedGameTime.TotalSeconds : 0;

    var keyboardState = Keyboard.GetState();

    // CreateNewObject1();

    timer += gameTime.ElapsedGameTime.Milliseconds; // появляются квадраты через
3000

    if (timer > 500)
    {
        CreateNewObject();
        timer = 0;
    }
;

    if (keyboardState.IsKeyDown(Keys.Up) && !wasPressed)
    {
        CreateNewObject1();
        wasPressed = true;
    }
    else if (keyboardState.IsKeyUp(Keys.Up))
    {
        wasPressed = false;
    }
    //for (int i = 0; i < 1; i++) //// ВЫЛЕТАЮТ СНИЗУ 20 КРУГОВ БЕЗ НАЖАТИЯ
КЛАВИШИ
    //{
    //    if (c__timer <= 0)
    //    {
    //        enemy = new Enemies(this, Content.Load<Texture2D>("triangle"), new
Vector2(0, 0), "triangle", this, 15); // последний атрибут - скорость кругов

```

						П...
Иэ	П...	Но докум	Полпу	Па		

```

// Components.Add(enemy);

// c__timer = 3f; // промежуток между кругами
// }
//}

mouse = Mouse.GetState();

foreach (GameComponent spr in Components)
{
    if (spr.IsDead) //ВЗАИМОДЕЙСТВИЕ!!!!
    {
        Components.Remove(spr);
        // Components.Remove(enemy);
        count--;
        a++;
        /* if ((time > 50000) || (count == 0))
        {
            Exit();
        }
        */
        //gg++;

        if (a == EnemiesNumber)
        {
            IsWin = true;
            Exit();
        }

        return;
    }

    //gg++;
}

// TODO: Add your update logic here

base.Update(gameTime);
}

void screen()
{
    graphics.PreferredBackBufferHeight = ScreenHeight;
    graphics.PreferredBackBufferWidth = ScreenWidth;
}

```

						Πιι
Иэ	Πιι	Νο άηκνιμ	Πηθηι	Πε		

```

protected override void Draw(GameTime gameTime)
{
    GraphicsDevice.Clear(Color.CornflowerBlue);
    spriteBatch.Begin();

    ScreenHeight = 1000;
    ScreenWidth = 1000;

    screen();

    base.Draw(gameTime);
    spriteBatch.End();
}
}
}

```

Enemies.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Diagnostics;
using System.Text;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Content;

namespace Personal
{
    public partial class Enemies : GameComponent
    {
        MouseState mouse;
        Game1 game1;
        private Rectangle scrBounds;

        private int direction = 0;

        protected Random randNum;

        public Enemies(Game game, Texture2D newTexture, Vector2 position, String name,
            Game1 game1, int pix_per_step) : base(game, newTexture, position, name)
        {
            this.texture = newTexture;
            this.position = position;
            this.game1 = game1;
            randNum = new Random();
            // int speeds = randNum.Next(2, 50); // скорость квадрат (2,50)

```

						Π...
Иэ	Π...	Νο Ἀρκιμ	Ποῦπι	Πα		

```

this.pix_per_step = pix_per_step;
Rectangle = new Rectangle(0, 0, texture.Width, texture.Height);

this.Object_type = name;

scrBounds = new Rectangle(0, 0, 80, 72);

}

public virtual void Move()
{
    // sprPosition += speed;
}

public override void Update(GameTime gameTime)
{

    mouse = Mouse.GetState();
    // движение квадратов

    // position.Y += pix_per_step;
    // position.Y += pix_per_step;

    /* if (direction == 1) // справа налево (внизу)
    {
        if (position.Y + texture.Height + pix_per_step > Game1.ScreenHeight - 200 &&
position.X - pix_per_step < 0)
            direction = 2;
        else
            position.X -= pix_per_step;
    }

    if (direction == 2) //снизу вверх (слева)
    {
        if (position.X - pix_per_step < 0 && position.Y - pix_per_step < 0)
            direction = 0;
        else
            position.Y -= pix_per_step;
    }
    */

```

						П...
Иэ	П...	Но днккм	Плрдпи	Пэ		

```

if (direction == 0) // Движение справа налево слева направо и тд
{
    if (position.Y + texture.Height + pix_per_step < Game1.ScreenHeight && position.X
+ texture.Width + pix_per_step > Game1.ScreenWidth)

        {
            direction = 1;
        }
    else
    {
        position.X += pix_per_step;
        // position.Y += pix_per_step;
    }
}

/*if (direction == 1) //

{

    if (position.Y - pix_per_step < 0 || position.X - pix_per_step < 0)
    {
        direction = 0;
    }
    else
    {
        position.X -= pix_per_step;
    }

}
*/

if (direction == 1) //

{

    if (position.Y + texture.Height + pix_per_step > 1000)
    {
        direction = 2;
    }
    else
    {
        position.Y += pix_per_step;
    }

}

if (direction == 2) // Движение справа налево слева направо и тд
{
    if (position.Y - pix_per_step < 0)

```

						Пн
Ис	Пн	По докум	Полн	Па		

```

        {
            direction = 0;
        }
        else
        {
            position.X -= pix_per_step;
            position.Y -= pix_per_step;
        }
    }

    Move();

    base.Update(gameTime);
}
}
}

```

Program.cs

```

using System;

namespace Personal
{
    public static class Program
    {
        [STAThread]
        static void Main()
        {
            //using (var game = new Game1(5, 10))
            var level1 = new Game1(5,10);
            var level2 = new Game1(10, 20);
            level1.Run();

            if (level1.IsWin)
                level2.Run();
        }
    }
}

```

						Π...
Με	Π...	Νο Ἀρκυμ	Ποῦ...	Πα		

GameComponent.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Diagnostics;
using System.Text;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Content;

namespace Personal
{
    public partial class GameComponent : DrawableGameComponent
    {
        protected int pix_per_step;
        protected Vector2 Velocity;
        protected Vector2 position;
        protected Texture2D texture;

        public GameComponent(Game game, Texture2D texture, Vector2 position, string
Object_type) : base(game) { }

        public Vector2 Velo { get { return Velocity; } set => Velocity = value; }
        public string Object_type { get; set; }
        public bool IsDead { get; set; } = false;
        public Vector2 Position { get { return position; } set => position = value; }
        public Rectangle Rectangle { get; protected set; }
        public Rectangle Properties => new Rectangle((int)Position.X - Rectangle.Width / 2,
(int)Position.Y - Rectangle.Height / 2, Rectangle.Width, Rectangle.Height);

        public override void Draw(GameTime gameTime)
        {
            SpriteBatch sprBatch = (SpriteBatch)Game.Services.GetService(typeof(SpriteBatch));
            //if (Game1.gameState.Menu == GameState.PLAY)
            //{
                sprBatch.Draw(texture, Position, Color.White);
            //}
            base.Draw(gameTime);
        }
    }
}
```

Player.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
```

						Π...
Μα	Π...	Νο Ἀρκυμ	Ποῦ...	Πα		

```

using System.Diagnostics;
using System.Text;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Content;

namespace Personal.Code
{
    public partial class Player : GameComponent
    {
        private int direction = 0;
        private Rectangle scrBounds;
        public int height { get; }
        public int width { get; }
        Game1 game1;

        bool is_MouseLB_Pressed = false;
        Vector2 new_position = new Vector2();

        public Player(Game game, Texture2D sprTexture, Vector2 position, String name, Game1
game1, int pix_per_step) : base(game, sprTexture, position, name)
        {
            this.position = position;
            this.Object_type = name;

            this.texture = sprTexture;
            this.game1 = game1;
            this.pix_per_step = pix_per_step;

            Rectangle = new Rectangle(0, 0, texture.Width, texture.Height);

            // height = 80;
            // width = 72;
            scrBounds = new Rectangle(0, 0, 80, 72);
        }

        public override void Update(GameTime gameTime)
        {
            KeyboardState kbState = Keyboard.GetState();

            /*if (direction == 0) // сверху вниз (справа)
            {
                if (position.Y + texture.Height + pix_per_step > Game1.ScreenWidth && position.Y
+ texture.Height + pix_per_step > Game1.ScreenHeight - 200)
                {
                    direction = 1;

```

						П...
Иэ	П...	Но днхм	Плхп...	Пэ		

```

    }
    else
    {
        position.Y += pix_per_step;
    }

}

if (direction == 1)
{
    if (position.Y + texture.Height + pix_per_step > Game1.ScreenHeight - 200 &&
position.X - pix_per_step < 0)
        direction = 2;
    else
    {
        position.X -= pix_per_step;
    }
    // position.X -= pix_per_step;
}

if (direction == 2) //снизу вверх (слева)
{
    if (position.Y - pix_per_step < 0 && position.Y - pix_per_step < 0)
        direction = 0;
    else
        position.X += pix_per_step;
    position.Y -= pix_per_step; // по диагонали
}

*/

// position.X += pix_per_step; // слева направо
//if (kbState.IsKeyDown(Keys.Space))
//{

//    position.Y += pix_per_step; // по диагонали
//    position.X -= pix_per_step; //
//}

foreach (GameComponent spr1 in game1.Components) // ВЗАИМОДЕЙСТВИЕ
{
    //Если объект - это круг и границы данного спрайта накладываются на область
    круга
    if (spr1.Object_type == "triangle" && Properties.Intersects(spr1.Properties))
    {
        //game1.Kill();
        spr1.IsDead = true;
    }
}

```

						П...
Иг	П...	Но докум	Полн...	Па		

```

        // Exit(0);
    }
}
Move();
// Check();
base.Update(gameTime);

}

void Move()
{
    /*
    KeyboardState kbState = Keyboard.GetState();
    if (kbState.IsKeyDown(Keys.Up))
        position.Y -= 5;
    if (kbState.IsKeyDown(Keys.Down))
        position.Y += 5;
    if (kbState.IsKeyDown(Keys.Left))
        position.X -= 5;
    if (kbState.IsKeyDown(Keys.Right))
        position.X += 5;
    */

    position.X += 10;
    position.Y -= 10;

}

void Check()
{
    if (position.X < scrBounds.Left)
    {
        position.X = scrBounds.Left;
    }
    if (position.X > scrBounds.Width - width)
    {
        position.X = scrBounds.Width - width;
    }
    if (position.Y < scrBounds.Top)
    {
        position.Y = scrBounds.Top;
    }
    if (position.Y > scrBounds.Height - height)
    {
        position.Y = scrBounds.Height - height;
    }
}

```

						Π11
112	Π11	Νο 2024/25	Προβλεπ	Π2		

```

    }
}
}

```

Sprite.cs

```

using System;
using System.Collections.Generic;
using System.Text;
using System.Threading.Tasks;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Content;

namespace Personal.Code
{
    class Sprite
    {
        public Texture2D spTexture;
        public Vector2 spPosition;

        public Sprite(Texture2D spTexture, Vector2 spPosition)
        {
            this.spTexture = spTexture;
            this.spPosition = spPosition;
        }
    }
}

```

						Π11
Μ2	Π11	Νο 2024/25	Προβλεπ.	Π2		

