Лабораторная работа №37
Разработка игрового приложения

1 Цель работы

1.1 Научиться разрабатывать интерактивные игровые приложения на C#.

2 Литература

2.1 Фленов М.Е. Библия C# / М.Е. Фленов. – Санкт-Петербург : БХВ-Петербург, 2016. – Режим доступа: https://ibooks.ru/reading.php?productid=353561, только для зарегистрированных пользователей. – Загл. с экрана. – гл.5, гл.13.

3 Подготовка к работе

3.1 Повторить теоретический материал (см. п.2).

3.2 Изучить описание лабораторной работы.

4 Основное оборудование

4.1 Персональный компьютер.

5 Задание

5.1 Настройка игры

5.1.1 В Обозревателе решений найдите Content.mgcb

5.1.2 Выберите Добавить → Новый элемент

5.1.3 Найдите и выберите "SpriteFont Description"

5.1.4 В поле имени введите "DefaultFont.spritefont" и нажмите "Добавить"

5.1.5 В папке Content появится файл DefaultFont.spritefont. Откройте его и укажите <Size>20</Size>. Сохраните файл (Ctrl+S)

5.1.6 Добавьте в игровой класс поля Rectangle для ракетки и шарика, поле для вектора скорости шарика, а так же списки Rectangle для блоков, которые будем разбивать

5.1.7 Добавьте переменные для хранения жизней, счета и других переменных игрового состояния.

5.1.8 Написать код, размещающий на экране кирпичи (4 ряда по 10 штук в каждом, у каждого ряда – свой цвет, ряд занимает всю строку от одной до другой стены, между каждым кирпичом должно быть расстояние в несколько пикселей).

5.1.9 Прописать начальное расположение ракетки и мяча, скорость мяча.

5.1.10 Создадим простую белую текстуру 1x1 пиксель для отрисовки всех элементов

```
_pixel = new Texture2D(GraphicsDevice, 1, 1);
_pixel.SetData(new Color[] { Color.White });
```

5.1.11 Добавим вспомогательные методы для отрисовки элементов и строк:

```
private void DrawRectangle(Rectangle rect, Color color)
{
```

```
    _spriteBatch.Draw(_pixel, rect, color);
}
private void DrawString(string text, Vector2 position,
Color color)
{
    // Загружаем шрифт (его мы создадим в конце)
    SpriteFont                font                =
Content.Load<SpriteFont>("DefaultFont");
    _spriteBatch.DrawString(font,    text,    position,
color);
}
```

5.1.12 Использовать данные методы для отрисовки игровых элементов в методе Draw.

5.2 Реализация движения ракетки и мяча

5.2.1 Для того, чтобы ракетка двигалась за курсором мыши в пределах окна, получать состояние мыши перемещать ракетку относительно горизонтальных координат курсора

5.2.2 Для того, чтобы мяч двигался, необходимо каждый кадр изменять местоположение мяча, в соответствии с установленной скоростью. При старте игры мяч должен лететь вверх и влево или вправо.

5.3 Реализация запуска и остановки игры при нажатии пробела

5.3.1 Добавить в метод Update код, проверяющий что нажата клавиша пробел.

Если нажат пробел и игра запущена, то выполнение Update должно прерываться, на игровом поле должна появляться надпись «Пауза».

Если нажат пробел и игра остановлена, то выполнение Update должно продолжаться, на игровом поле не должно быть надписи «Пауза».

5.4 Реализация отскакиваний мяча и появления нового мяча

5.4.1 Для того, чтобы мяч отскакивал от стен, в Update учесть следующее:

- при столкновении левого края мяча с левым краем экрана или правого края мяча с правым краем экрана скорость движения мяча по координате X должна меняться на противоположную по знаку,
- при столкновении верхнего края мяча с верхним краем формы скорость движения мяча по координате Y должна меняться на противоположную по знаку,

5.4.2 Для того, чтобы мяч отскакивал от ракетки, в Update учесть следующее:

- при столкновении нижнего края мяча с верхним краем ракетки _ball.Intersects(_paddle) скорость движения мяча по координате Y должна меняться на противоположную по знаку.

5.4.3 Для появления мяча после потери в новой точке в Update при уходе мяча за нижний край экрана следует:

- останавливать выполнение игры,
- увеличивать счетчик потерянных мячей, видимый пользователю,
- выводить окно с сообщением, что мяч потерян,

- изменять местоположение мяча аналогично тому, как это сделано при загрузке формы,
- запускать игру.

5.5 Реализация получения очков за выбитые кирпичи

При реализации потребуются следующие свойства и методы проверки пересечения прямоугольных областей, чтобы не проверять отдельно столкновения мяча с краями каждого кирпича: _ball.Intersects(_blocks[i])

В случае пересечения следует удалять кирпич, с которым было пересечение, из списка _blocks. При удалении последнего сообщить о победе.

5.5.1 При столкновении начислять определенное количество баллов (чем выше ряд, тем больше баллов) и выводить результат в метку Счет.

5.5.2 При столкновении для изменения вектора движения мяча следует до удаления кирпича из коллекции получить прямоугольник, находящийся на пересечении мяча и кирпича:

```
Rectangle intersection = Rectangle.Intersect(_ball, _blocks[i]);
```

Если у прямоугольника intersection высота меньше ширины, то было столкновение с нижним или верхним краем мяча, скорость движения мяча по координате Y должна меняться на противоположную по знаку. Иначе было столкновение с левым или правым краем мяча, скорость движения мяча по координате X должна меняться на противоположную по знаку.

6 Порядок выполнения работы

6.1 Используя Microsoft Visual Studio, создать проект C# и выполнить задания из п.5.

6.2 Ответить на контрольные вопросы.

7 Содержание отчета

7.1 Титульный лист

7.2 Цель работы

7.3 Ответы на контрольные вопросы

7.4 Вывод

8 Контрольные вопросы

8.1 Как получить состояние мыши?

8.2 Как получить состояние клавиатуры?

8.3 Для чего применяются методы Intersects() и Intersect()?