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 интерфейса приложения

5.1.1 Настроить форму:

- изменить цвет фона, текста, шрифт
- форма должна открываться по центру экрана и быть фиксированного размера
- заголовок формы: Арканоид

5.1.2 В режиме дизайнера добавить на форму:

- метку для отображения счета
- метку для отображения количества потерянных мячей
- панель-ракетку (Name: racket)
- панель-мяч (Name: ball)

Настроить внешний вид добавленных элементов.

5.1.3 На загрузку формы прописать код, программно размещающий на форме панели-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).

5.1.4 На загрузку формы прописать следующий код для изменения контура панели-мяча на круглую (высота и ширина панели должны быть одинаковыми):

```
System.Drawing.Drawing2D.GraphicsPath path =
    new System.Drawing.Drawing2D.GraphicsPath();
path.AddEllipse(ball.ClientRectangle);
ball.Region = new Region(path);
```

5.1.5 На загрузку формы прописать код, изменяющий начальное положение мяч (мяч должен появляться в случайном месте в нижней части игрового поля).

5.2 Реализация движения ракетки и мяча

5.2.1 Для того, чтобы ракетка двигалась за курсором мыши в пределах формы, создать у формы обработчик движения мыши и прописать в нем следующий код:

```
// определение центра ракетки, по нему будет находиться курсор мыши
int racketCenterX = racket.Width / 2;
// this.ClientSize - это прямоугольная клиентская область формы
// this.ClientSize.Width - это доступная (клиентская) ширина формы
if (e.X > racketCenterX && e.X < this.ClientSize.Width - racketCenterX)
{
    // изменение местоположения ракетки
    racket.Location = new Point(e.X - racketCenterX, racket.Top);
}
```

5.2.2 Для того, чтобы мяч двигался по таймеру, добавить на форму таймер и реализовать его запуск при загрузке формы. Создать обработчик срабатывания таймера, в котором изменять местоположение мяча. При старте игры мяч должен лететь вверх и влево или вправо. Для выполнения этого добавить в форму два целочисленных поля:

- ballSpeedX – скорость по координате X
- ballSpeedY – скорость по координате Y

Начальные значения полей генерировать при загрузке формы (не должны быть равны нулю).

5.3 Реализация запуска и остановки игры при нажатии пробела

5.3.1 Установить у формы свойство KeyPreview = true, чтобы форма перехватывала нажатия клавиш.

5.3.2 Добавить на форму обработчик нажатия клавиши KeyDown. В обработчике написать код, проверяющий, что нажата клавиша пробел (e.KeyCode == Keys.Space).

Если нажат пробел и игра запущена, то таймер должен останавливаться, на игровом поле должна появляться надпись «Пауза».

Если нажат пробел и игра остановлена, то таймер должен запускаться, 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 обработчике срабатывания таймера учесть следующее:

- при столкновении левого края мяча с левым краем формы или правого края мяча с правым краем формы 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 обработчике срабатывания таймера учесть следующее:

- при столкновении нижнего края мяча с верхним краем ракетки (если правый или левый край мяча касается ракетки) скорость движения мяча по координате Y должна меняться на противоположную по знаку.

5.4.3 Для появления мяча после потери в новой точке в обработчике срабатывания таймера при уходе мяча за нижний край формы следует:

- останавливать таймер,
- 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 таймер.

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:

- объект.**Bounds** – объект типа **Rectangle** (прямоугольник), возвращающий местоположение и размер объекта.
- прямоугольник1.**IntersectsWith**(прямоугольник2) – метод возвращает истину, если прямоугольники пересеклись.
- прямоугольник1.**Intersect**(прямоугольник2); – метод записывает в первый прямоугольник прямоугольник, находящийся на пересечении двух указанных прямоугольников.

5.5.1 Для определения столкновения панели-мяча с панелями-кирпичами и удаления тех, с которыми было столкновение, в обработчике срабатывания таймера после изменения местоположения мяча перебрать все элементы управления из коллекции this.Controls, используя следующий код:

```
// прямоугольная область мяча
Rectangle ballRectangle = ball.Bounds;
// проход по всем элементам управления формы
for (int i = this.Controls.Count - 1; i >= 0; i--)
{
    // item - текущий элемент управления
    Control item = this.Controls[i];
    // проверка, что item является панелью-кирпичом
    // (объект типа Panel, не мяч и не ракетка)
    if (item is Panel && item != ball && item != racket)
    {
        //проверка пересечения прямоугольных областей
        if (ballRectangle.IntersectsWith(item.Bounds))
        {
            //код, который должен быть выполнен при
столкновении
```

```
            }
        }
}
```

В случае пересечения следует удалять кирпич, с которым было пересечение, из коллекции this.Controls. При удалении последнего сообщить о победе.

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

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

```
ballRectangle.Intersect(item.Bounds);
```

Если у прямоугольника ballRectangle 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 Какие обработчики событий применяются для обработки событий клавиатуры?

8.2 Какие обработчики событий применяются для обработки событий мыши?

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

8.4 Какие методы используются для программного добавления и удаления элементов управления из коллекции элементов формы?