ИВАН ТРЕЩЕВ

ПРОГРАММИРОВАНИЕ ДЛЯ МОБИЛЬНЫХ ПЛАТФОРМ

WINDOWS PHONE

Иван Трещев

Программирование для мобильных платформ. Windows Phone

http://www.litres.ru/pages/biblio_book/?art=39145115
ISBN 9785449368690

Аннотация

Данная книга обобщает опыт работы лаборатории мобильных приложений на базе ФГБОУ ВО КнАГУ, где автор был ее руководителем. Приложения, разработанные в книге, были успешно выложены в магазин приложений. В книге вы найдете описание основных моментов для разработки приложений.

Содержание

Программирование для мобильных платформ Windows Phone

Иван Трещев

© Иван Трещев, 2018

ISBN 978-5-4493-6869-0
Создано в интеллектуальной издательской системе Ridero

Введение

Разработка мобильных приложений как написание электронных книг – автор может быстро получить результат, отклики, доход, известность. Современный рынок мобильных устройств полон различными аппаратами всевозможных форм-факторов. Программировать для платформ, которые легко могут уместиться в кармане весьма интересно и каждый может себя попробовать в этом амплуа.

Лаборатория которой руководил автор на протяжении 5 лет занималась разработкой различных приложений для самых популярных за последнее пятилетие операционных систем носимых устройств – Android, IOS, Windows Phone. Хотя сегодня платформа корпорации Microsoft уже мало используется, но возможность практически без дополнительных затрат со стороны программиста (в случае если Visual Studio уже установлено) разрабатывать эти самые приложения, выкладывать их в магазины при этом не неся затрат на тиражирование, продажу, экспозицию и другие накладные расходы вызывает неподдельный интерес среди любой среды, где собираются единомышленники по созданию кода.

Данная книга посвящена разработке приложений именно под платформу от Microsoft и является первой в цикле, которые автор намерен опубликовать.

У читателя предполагается опыт программирования

на объектно-ориентированном языке, желательно опыт на C#.

По мнению автора нет ничего более увлекательного для программиста, чем разработка игрового приложения – именно так можно заинтересовать аудиторию и постараться окунуть ее в «бездну программирования». Далее по тексту используется собирательное понятие игра, как отражение разрабатываемых мобильных приложений (соотвественно приложения и классы именуются Game).

Автор хотел бы выразить огромную благодарность Сыровацкой Е. С. и Вавиличеву А. В., которые проверяли тексты программ из книги, принимали непосредственное участие в тестировании приложений и внимание к работе.

Работа с XNA

Начальные теоретические сведения

Для разработки приложений и игр для платформы Windows Phone чаще всего используется язык программирования C#. Для написания основной логики используются:

```
//////////////////////////////////////////////////////////////////////////
if (условие1) // условие содержит логическое выражение
{
// Действия, которые надо выполнить, если условие1 выполняется
}
elseif (условие2) // Не обязательно
{
// Действия, которые надо выполнить, если условие1 не выполняется, но выполняется условие2
}
else// Не обязательно
{
// Действия, которые надо выполнить, если условия 1 и 2 не выполняются
}
//////////////////////////////////////////////////////////////////////////
```

switch (значение1)

{

case значение2:

// Действия, которые надо выполнить, если значения 1 и 2 равны

break;

case значение3:

// Действия, которые надо выполнить, если значения 1 и 3 равны

break;

<…>

}

//

тип [] mas1 = new **тип** [число элементов]; // Объявления одномерного массива

тип [,] mas2 = new **тип** [число строк, число столбцов]; // и двумерного массива

//

foreach (тип новая_переменная in mas1) // Цикл по всем элементам

{

// Действия для каждого объекта из mas1, где под объектом подразумевается новая_переменная

}

//

for (i = начальное_значение; i <= Конечное_значение; i+

+) // Цикл
{
// Тело цикла
}
//
While (условие) // Выполнять цикл пока условие = true
{
// Тело цикла
}
//
Random rand = new **Random** (); // Создает переменную rand для работы со случайными числами
// Присваивает переменной случайное значение от 0 до максимального значения:
имя_переменной = rand.Next (максимальное_значение);
//Присваивает переменной случайное значение от минимального значения до максимального значения:
имя_переменной = rand.Next (минимальное_значение, максимальное_значение);
//

Разработка игр с использованием XNA

В отличие от разработки приложений для Windows Phone, для создания игр целесообразно использовать набор инструментов XNA. При разработке игры на платформе Silverlight, что используется для создания приложений, может возникнуть трудность с отображением большого количества элементов на экране, а именно долгая отрисовка и как следствие подвисание самой игры, поэтому для создания игр для Windows Phone в SDK включена возможность использования XNA.

Окно создания проекта представлено на рисунке 1.

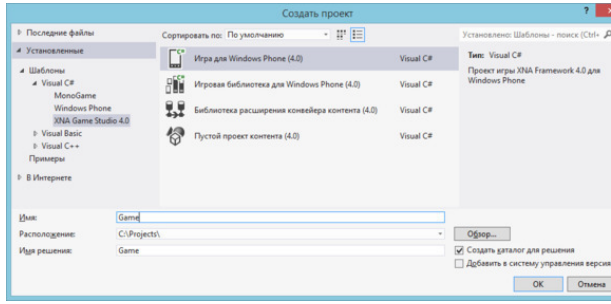

Рисунок 1 – Создание проекта игры для Windows Phone

Во вкладке «Обозреватель решений» показанны все фай-

лы, включенные в проект, работать предстоит с файлом Game1.cs, в нем располагается основная логика игры (рис.2).

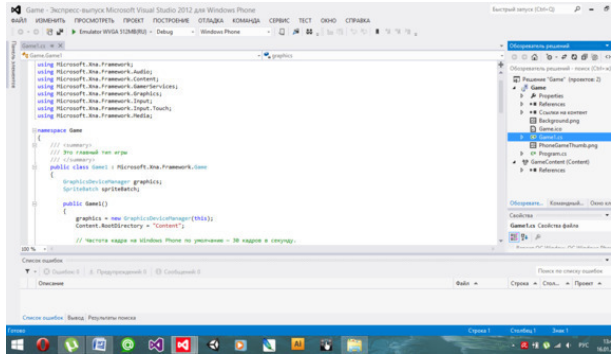

Рисунок 2 – Обозреватель решений

Background.png и PhoneGameThump.png являются иконками игры, которые отображаются в меню смартфона, их необходимо заменить на свои файлы с теми же названиями и размерами изображений.

В папке GameContent необходимо расположить весь контент игры: текстуры, звуки, шрифты и другое. Они помещаются в папку контента и добавляются в проект при нажатии правой копкой мыши в обозревателе решений по папке контента и выборе действия "Добавить существующий элемент" (рис 3):

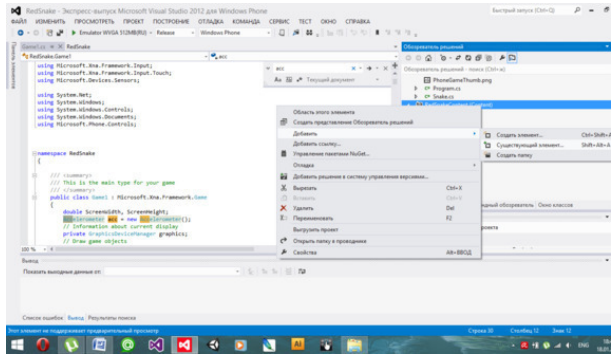

Рисунок 3 – Добавление контента

Изначально файл Game1.cs содержит несколько стандартных и необходимых методов:

public **Game1**()

{

// Здесь указываются ориентация экрана, частота обновления,

// разрешение и сенсорные жесты, которые будут использованы в игре

}

protected override void **LoadContent**()

{

// Здесь загружается весь контент, необходимый в игре

}

protected override void **Update**(GameTime gameTime)

{

```
    // Здесь располагается логика, выполняемая при обновлении экрана
    base.Update(gameTime);
    }
    protected override void Draw(GameTime gameTime)
    {
    // Здесь производится отрисовка графического контента
    base.Draw(gameTime);
    }
```

Вывод текста и графики в XNA

Для того чтобы вывести текст на экран, необходимо добавить в папку с контектом файл с названием *.spritefont, где * – название шрифта, этот файл содержит следующие строки:

```
<?xml version="1.0" encoding="utf-8"?>
<XnaContent
xmlns:Graphics="Microsoft.Xna.Framework.Content.Pipeline.C
<Asset Type= " Graphics: FontDescription»>
<FontName>Segoe UI</FontName>
<Size>15</Size>
<Spacing>0</Spacing>
<UseKerning>true</UseKerning>
<Style>Bold</Style>
<CharacterRegions>
<CharacterRegion>
<Start> </Start>
<End>~</End>
</CharacterRegion>
<CharacterRegion>
<Start>A</Start>
<End>я</End>
</CharacterRegion>
</CharacterRegions>
```

</Asset>
</XnaContent>

FontName – в этих тегах заключено имя шрифта, Microsoft не рекомендует использовать какие-либо специфические шрифты, одни из рекомендуемых – шрифты Segoe разных разновидностей.

Size – определяет размер шрифта.

Style – определяет стиль шрифта (Regular, Bold, Italic).

CharacterRegion – определяет символы, которые можно использовать, в данном случае это русские и английские буквы, цифры и некоторые специальные символы.

После добавления этого файла в проект, его необходимо объявить в коде:

SpriteFont Название_переменной; // Создаем переменную в начале класса Game1

В методе LoadContent() прописываем следующее:

Название_переменной = Content.Load<SpriteFont>("*"); //где * – название файла

В методе Draw(GameTime gameTime) выводим текст на экран:

spriteBatch.DrawString(Название_переменной, "Текст, который необходимо вывести", new Vector2(координата_x, координата_y), Color.Цвет);

Вывод изображения аналогичен выводу текста:

Texture2D Название_переменной; // Создаем переменную

в начале класса Game1

В методе LoadContent() прописываем следующее:

название_переменной = Content.Load<Texture2D>("*"); // где * – название файла

В методе Draw(GameTime gameTime) рисуем текстуру:

spriteBatch.Draw(название_переменной, new Rectangle(координата_x, координата_y, ширина, высота), Color.White);

Обработка нажатий

Нажатия на экран

Современные мобильные телефоны в большинстве случаев оснащены большим сенсорным экраном и минимальным количеством аппаратных кнопок, поэтому и организация взаимодействия с играми построена на считывании жестов с экрана.

В методе Game1() прописываются жесты, что могут быть использованные в игре:

```
public Game1()
{
<…>
TouchPanel.EnabledGestures = GestureType.Tap | GestureType.FreeDrag;
<…>
}
```

Все возможные жесты можно посмотреть в подсказке, всплывающей при вводе" G*estureType*". Наиболее часто используемые из них, это Tap – нажатие, FreeDrag – перетягивание, Hold – долгое нажатие, DoubleTap – двойное нажатие. Так же можно обработать и действия при отсутствии жестов.

В логике игры размещается следующее:

```
while (TouchPanel.IsGestureAvailable)
{
// Считывание жеста
GestureSample gesture = TouchPanel.ReadGesture();
// Координаты касания и другие необходимые параметры
int tapY = (int)gesture.Position.Y;
int tapX = (int)gesture.Position.X;
<…>
switch (gesture.GestureType)
{
// Если жест является нажатием:
case GestureType.Tap:
<…>
break;
<…>
// Если жест является перетягиванием:
case GestureType.FreeDrag:
<…>
break;
}
}
```

Нажатия на аппаратные кнопки

Обработка нажатий на аппаратные кнопки тоже важна, однако в приложении запрещено использовать их для

нестандартных функций (например, аппаратную кнопку назад для установления паузы).

Одним из требований к приложениям и играм является то, что необходимо программировать действия для аппаратной кнопки назад таким образом, что после нажатия на нее показывается предыдущий модуль игры (не относится к игровым уровням), пример:

Меню – Список уровней – Уровень 1 – Уровень 2

Из любого уровня переход назад осуществляется в Список уровней, даже если новый уровень запускался после предыдущего, из Списка уровней соответственно в Меню, из Меню же происходит выход из приложения:

```
if (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed)
```

Конец ознакомительного фрагмента.

Текст предоставлен ООО «ЛитРес».

Прочитайте эту книгу целиком, купив полную легальную версию на ЛитРес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.