УДК 004.42

РАЗРАБОТКА ИГРОВОГО ПРИЛОЖЕНИЯ С ИСПОЛЬЗОВАНИЕМ UNITY3D

Рижий С.Н., Терещенко Т.М.

DEVELOPMENT OF GAME APPLICATION USING UNITY3D

Rigiy S., Tereshchenko T.

В статье описан процесс создания уровня игрового приложения от разработки концепта до дизайна и конструирования уровня с помощью игрового движка Unity3D. Разработаны скрипты механик и головоломок на языке программирования C#. Разработанный уровень игры может быть использован в качестве прототипа.

Ключевые слова: *игровое приложение, скрипт, алгоритм, уровень, сцена, unity3d, C#, дизайн уровней, концепт документ.*

Введение. На данном этапе развития компьютерных технологий одним из набирающих популярность видов досуга стали компьютерные игры. В сфере компьютерной индустрии появилась ниша инди-игр – другими словами игр без бюджета или с низким бюджетом. Игры данной категории набирают популярность, поскольку в большинстве случаев их разработчики не боятся экспериментировать с игровым процессом. В результате этого процесса появился спрос на недорогие или бесплатные игровые движки (Game Engine). Unity3D является одним из таких движков, который выделяется из числа многих наличием большого количества функциональных возможностей, большого комьюнити, в том числе русскоязычного, а так же бесплатной версии без ограничений коммерческой деятельности.

Цель статьи – разработать уровень игры (ее прототип) с возможностью изменения в зависимости от конкретных требований технического задания.

Для создания игры необходимо выполнить несколько определенных шагов. На первом этапе создается концепт игры и выбирается ее жанр [1]. Было решено выбрать жанр First Person Puzzle с элементами хоррора. Данный жанр был выбран, поскольку он не имеет большого количества конкурентов на рынке, но при этом пользуется популярностью и оставляет место для экспериментов. Вид игры «от первого лица» был выбран, поскольку он увеличивает интеграцию игрока в игровой мир и позволяет ему сильнее ассоциировать главного героя игры с собой.

Концепт игры состоит в том, что игрока преследует противник, которого нельзя никак победить. Для того чтобы выжить необходимо убегать, но путь преграждают различные головоломки, которые необходимо быстро решать, чтобы продвигаться дальше. Решение осуществляется с помощью механики, которая заключается в том, что игрок может перемещаться между двумя практически идентичными мирами, в одном из которых проход может быть открыт, но для продвижения необходимо переместить некоторые объекты из другого мира.

После создания концепции игры необходимо создать алгоритмы, которые реализуют механику игры и написать скрипты механики [2].

Для представленного игрового приложения были реализованы следующие алгоритмы:

- Перенос объектов (рис. 1).
- Изменение видимости объектов на уровне.
- Применение эффекта «Оттенки серого» на объекты уровня.
- Отталкивание физических объектов при столкновении с персонажем.

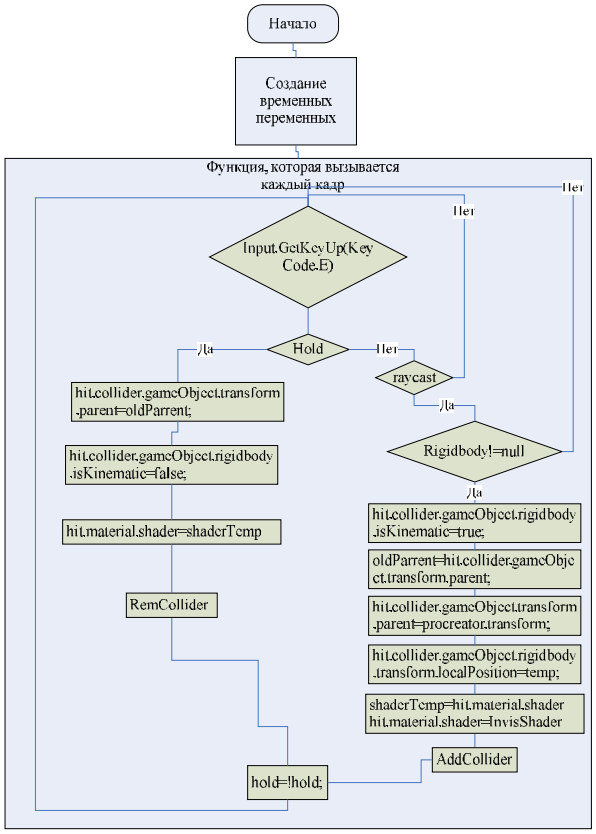

Рис. 1 Алгоритм переноса объектов

После создания всех алгоритмов необходимо написать код скриптов. Код был написан на языке C#, поскольку для данного движка он наиболее удобен и имеет высокую производительность.

При разработке на Unity3d часто необходимо привязать выполнения своих функций к конкретному событию. В Unity3d таких событий достаточно большое количество, но делятся они на три большие группы:

1. События, вызываемые по событиям (загрузка сцены, выход пользователя).

   Данная группа событий выполняется на нерегулярной основе.
2. События , вызываемые при прорисовке кадра

   В этом случае все используемые скрипты вызываются в цикле прорисовки экрана, а значит, будут непосредственно влиять на FPS (частоту кадров в секунду). Поэтому здесь нужно очень аккуратно работать с функциями, которые требуют много времени на обработку.
3. События, вызываемые при расчете физики.

   Для расчета физики создается отдельная, независимая нить, события в которой вызываются с определенным интервалом времени. Размер этого интервала можно настроить в пункте меню: Edit - > Project Settings -> Time - > Fixed Timestep.

Зная эту разбивку разработчик уже может принимать решение о том, где какой код лучше расположить.

Однако, все вычисления, производимые как при расчете физики, так и при прорисовке, влияют на « отзывчивость» игры. Поэтому, при разработке программы, наиболее ресурсоемкие вычисления желательно оформлять в сопрограммы (Coroutine).

Согласно концепции игры необходимо сделать изменение между цветным и черно - белым изображением. Типичным решением этой задачи является применение шейдера в функции OnRenderImage, вызываемая после прорисовки сцены, для постобработки изображения на экране.

Шейдер — это программа для одной из ступеней графического конвейера, используемая для определения окончательных параметров объекта или изображения. Она может включать в себя произвольной сложности описание поглощения и рассеяния света, наложения текстуры, отражение и преломление, затемнение, смещение поверхности и эффекты пост-обработки.

Программируемые шейдеры гибки и эффективны. Сложные с виду поверхности могут быть визуализированы при помощи простых геометрических форм.

Поскольку функция постобработки изображения, как и несколько других доступны только в Pro-версии Unity, то было решено создать скрипт, который меняет шейдер материалов, используемых на объектах уровня. Нужно применить 2 шейдера - первый стандартный , а второй переводит изображение в оттенки серого. Для перевода цветов в оттенки серого была применена формула 1:

$$Y = 0.2121R + 0.7152\,G + 0.0722B \quad (1)$$

где Y - яркость цвета в оттенках серого;

R - значение красного цвета в RGB формате;

G - значение зеленого цвета в RGB формате;

B - значение голубого цвета в RGB формате.

После создания шейдера черно-белого эффекта необходимо создать скрипт, применяющий этот шейдер.

После создания всех скриптов механики игры создается уровень и головоломки, которые на этом уровне будут доступны [3]. Для этого используется редактор движка Unity3d.

Интерфейс редактора Unity разделен на следующие основные части: иерархия, проект, панель инструментов, сцена, инспектор.

Иерархия (Hierarchy) содержит все объекты (GameObject) открытой сцены. Некоторые из них - экземпляры ресурсов (3D-модели), другие - экземпляры префаб. Объекты добавляют на сцену или удаляют из нее, отражаются или перестают отображаться в иерархии.

Unity использует концепцию наследования. Каждый объект может быть дочерним по отношению к другому. Для этого достаточно перетащить

его на родительский объект в иерархии (Рис. 2). Дочерний объект будет двигаться и вращаться вместе с родительским.

Каждый проект содержит папку Assets. Содержимое этой папки представлено в Project View. Это ресурсы игры. Чтобы добавить ресурсы в проект, нужно перетащить файл из Проводника в Project View или через меню Assets → Import New Assets. Ресурс будет автоматически импортирован в проект и станет доступен для использования.

Сцены также же хранятся в папке Assets и отображаются в Project View. Они являются уровнями.

Рис. 2 Иерархия объектов

Элемент интерфейса «сцена» применяется для позиционирования объектов (окружение, персонажи, камеры, системы частиц и др.).

Создаем в редакторе Unity «префабы» - тип ресурсов, предназначенный для многократного использования. Каждый объект на сцене является копией оригинального префаба этого объекта. При изменении оригинального префаба изменяются все его копии в уровнях, что является удобным для быстрого редактирования. Далее расставляем объекты на уровне в нужном порядке и прикрепляем скрипты. После этого расставляем освещение в необходимых местах, добавляем звуковые эффекты и настраиваем скрипты.

В результате был получен готовый рабочий уровень (Рис. 3).

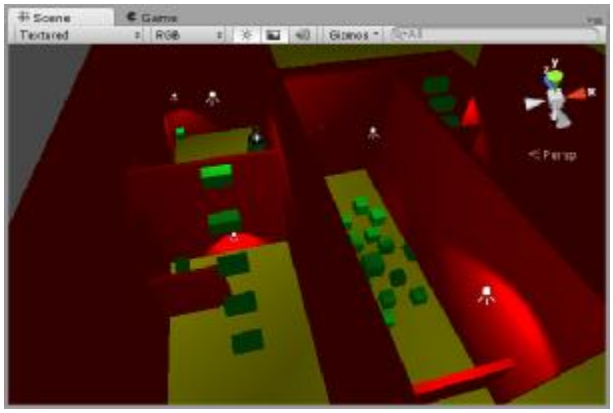

Рис. 3 Готовый уровень

Данный уровень является прототипом игры, поскольку он создан для изображения механики игры и в нем отсутствуют многие черты готовой игры – такие как текстуры, большое количество головоломок, а все объекты изображены простыми фигурами.

Выводы. В результате проведённых исследований был разработан уровень игры или ее прототип. Предусмотрена возможность изменения в зависимости от конкретных требований технического задания.

Для усовершенствования игры необходимо создать большее количество головоломок и расширить механику и возможности действий. Так же необходимо использовать текстуры и создать 3d объекты для уровня в специализированных программах, таких как 3d max, zbrush, blender или maya.

Литература

1. Компьютерные игры: как это делается // Выбор жанра игры: URL: http://blitz-3d.narod.ru/teoria/zalc/glava06/index.htm
2. Статья о скриптах Unity3D URL: http://habrahabr.ru/post/128711/
3. Компьютерные игры: как это делается // Разработка компьютерных игр: URL: http://blitz-3d.narod.ru/teoria/zalc/glava06/index.htm
4. Уроки Unity3D: URL: http://www.unity3dstudent.com/
5. Библиотека функций Unity3D: URL: http://docs.unity3d.com/Documentation/ScriptReference/

References

1. Komp'juternye igry: kak jeto delaetsja // Vybor zhanra igry: URL: http://blitz-3d.narod.ru/teoria/zalc/glava06/index.htm
2. Stat'ja o skriptah Unity3D URL: http://habrahabr.ru/post/128711/
3. Komp'juternye igry: kak jeto delaetsja // Razrabotka komp'juternyh igr: URL: http://blitz-3d.narod.ru/teoria/zalc/glava06/index.htm
4. Uroki Unity3D: URL: http://www.unity3dstudent.com/

5. Biblioteka funkcij Unity3D: URL: http://docs.unity3d.com/Documentation/ScriptReference

Рижий С.М., Терещенко Т.М. Створення ігрової програми за допомогою Unity3D

У статті описано процес створення рівня ігрової програми від розробки концепту до дизайну та конструювання рівня за допомогою ігрового движка Unity3D. Розроблено скрипти механік і головоломок на мові програмування C #. Розроблений рівень гри може бути використаний як прототип.

Ключові слова: Ігрова програма, скрипт, алгоритм, рівень, сцена, unity3d, C#, дизайн рівнів, концепт документ.

Rigiy S., Tereshchenko T. Development of game application using Unity3d

This article describes development of a game program from concept formation to making design and construction level using the Unity3D game engine. The scripts of mechanic and puzzles were developed on C#. Designed game level can be used as a prototype.

Keywords: game software, scripts, algorithm level, scene, unity3d, C #, level design, concept document.

Рижий Сергій Миколайович. Студент п'ятого курсу кафедри комп'ютерних наук Східноукраїнського національного університету імені Володимира Даля. Email: k.o.y.o.t@mail.ru

Терещенко Тетяна Михайлівна. Доцент, к.т.н. кафедри комп'ютерних наук Східноукраїнського національного університету імені Володимира Даля. Email: whiteopal@yandex.ru

Рецензент: Заслужений діяч науки і техніки України, д.т.н., проф. Ульшин В.О.

Стаття подана 22.04.2014