Shuvalova O. I.

DATABASES AS THE MAIN COMPONENT OF WEB-ORIENTED INFORMATION SYSTEM

Annotation. The aim of the article is to reveal the areas of application of Web-programming to highlight the essence of the database as the main component of Web-oriented information system in the course "Databases" for future teachers of computer science. To achieve this goal the following tasks are set:

• to reveal the features of software tools for the course "Databases" of future computer science teachers;

• demonstrate the features of the content and methods of teaching the course "Databases" of future teachers of computer science using Web-programming technologies;

• to analyze the results of the pedagogical experiment on the introduction of new methods.

Features of the methodical system of training Databases and construction of Web-oriented information systems are presented in the article. It is recommended to use MySQL in combination with Internet hosting. To build an information system, the starting software is recommended, which is based on the model of delimitation of the design part, databases, and software, MVC architecture. The content of the training covers the formation of the client and the administrator pages. Emphasis is placed on studying the dependence of the results of the Web-oriented information system on the results of SQL queries and the structure of the relational database. The analysis of the results of the pedagogical experiment proved the importance of the introduction of a new methodology for the formation of a systematic vision of the relationships of sections of the subject of Informatics and the place of relational databases in this system. The development of methods of teaching Web-programming in secondary school is considered promising.

Keywords: Methods of teaching computer science, databases, Web-programming, MySQL, PHP MyAdmin.

DOI 10.31392/NPU-nc.series 2.2020.22(29).21
УДК 004.9

Валерій Юрійович Габрусєв[1], Андрій Володимирович Вельгач[2], Олена Олегівна Кулянда[3]
Тернопільський національний педагогічний університет імені Володимира Гнатюка
[1]кандидат педагогічних наук, доцент кафедри інформатики та методики її навчання
ORCID ID 0000-0002-3392-6825
gabrusev@fizmat.tnpu.edu.ua
[2]кандидат фізико-математичних наук, викладач кафедри інформатики та методики її навчання
ORCID ID 0000-0001-9937-0244
velgandr@fizmat.tnpu.edu.ua
[3]кандидат медичних наук, доцент кафедри патологічної фізіології
ДВНЗ "Тернопільський державний медичний університет ім. І.Я. Горбачевського МОЗ України"
ORCID ID 0000-0001-6197-9046
kulyanda67@ukr.net

ДОСЛІДЖЕННЯ ФУНКЦІОНАЛЬНИХ ОСОБЛИВОСТЕЙ РУШІЯ UNITY 3D НА ПРИКЛАДІ РЕАЛІЗАЦІЇ 3D МІНІ-ГРИ

Анотація. У даний час розробка комп'ютерних ігор є актуальною, так як з розширенням ринку персональної електроніки, розширюється і ринок розваг. Загальнодоступність інструментальних засобів – ігрових рушіїв, доступ до різних інформативних навчальних і довідкових матеріалів дають можливість легко і швидко розробити прикладне програмне забезпечення, наприклад, комп’ютерну гру.

Існує немало публікацій та різних наукових досліджень на тему моделювання ігрових сцен. Даний етап розробки ігрових комп'ютерних додатків є досить обширним і включає у себе велику кількість завдань. Слід відзначити, що в такому випадку автори зачіпають тему не лише моделювання ігрових сцен, а також поверхнево охоплюють тему створення, моделювання, текстурування ігрових об'єктів, одночасно описуючи функціонал, за допомогою якого ці завдання виконуються.

У даній статті розглянуті найбільш відомі ігрові рушії, зроблено їх порівняльний аналіз. Виявлено їх переваги і недоліки, а також спектр завдань, для вирішення яких вони зручні у використанні. Під час порівняння різних ігрових рушіїв авторами зроблено вибір на користь Unity 3D з кількох причин: вигідна ліцензійна політика; сумісність з будь-якою платформою; відмінне співтовариство; велика кількість документації; легкість у використанні.

Метою даної статті є максимально повно описати процес моделювання сцени ігрового комп'ютерного додатка в середовищі Unity 3D. Для цього у статті розглянуто як приклад реалізацію

не складної 3d-гри. Зокрема, описуються функціональні можливості і інструменти для моделювання сцени в середовищі Unity, а також коротко дається опис основних об'єктів, які можуть бути додані у сцену. Разом з тим, дається короткий огляд вікна редактора та інспектора з усіма властивостями і методами ігрових об'єктів.

Автори стараються донести до читачів плюси і мінуси використання цього середовища для створення нескладних ігрових додатків, описують головне меню та інструментарій поверхнево, не заглиблюючись в тонкості всього процесу розробки.

Ключові слова: Unity 3D, JavaScript, ігровий додаток, середовище розробки, мова програмування.

Вступ. З розвитком технологій розробки програмного забезпечення пішли в минуле ті часи, коли розробка комп'ютерних ігор була доступна лише обраним. На даний час існують спеціалізовані програмні засоби, орієнтовані на розробку ігрових додатків, в яких містяться усі необхідні інструментальні засоби для забезпечення підтримки усього життєвого циклу розробки. На даний час ігрова індустрія пропонує значну кількість різноманітних ігрових фреймворків (інструментів) для розробки ігрового програмного забезпечення.

Ігровий фреймворк – центральний програмний компонент комп'ютерних та відеоігор або інших динамічних додатків з графікою, що опрацьовується у режимі реального часу. На його основі забезпечуються основні технології, спрощується розробка і часто забезпечується можливість виконувати гру на кількох платформах, таких як ігрові консолі та настільні операційні системи, наприклад Linux, Mac OS X і Microsoft Windows, Android тощо.

Основна функціональність зазвичай забезпечується через ігровий рушій, куди включається система рендерингу («візуалізатор»), модуль фізики, звук, систему сценаріїв, анімацію, штучний інтелект, мережевий взаємозв'язок, управління пам'яттю і багатопотоковість.

Часто на процесі розробки можна заощадити за рахунок повторного використання одного ігрового рушія для створення кількох різних ігор.

Сам термін «ігровий рушій» з'явився в середині 1990-х років – в цей час остаточно встановилося домінування IBM-сумісних комп'ютерів, а через швидкість процесорів і спеціальні технології програмування забезпечується 30 і більше кадрів в секунду в тривимірних іграх. Ігри Doom і Quake від ID Software виявилися настільки популярними, що інші розробники замість того, щоб працювати з чистого аркуша, ліцензували основні частини програмного забезпечення і створювали свою власну графіку, персонажів, зброю і рівні – «ігровий контент» або «ігрові ресурси». Рушій Quake був використаний в більш ніж десяти проектах і дав серйозний поштовх у розвитку ігрової індустрії.

Незважаючи на велику кількість багатофункціональних інструментів, під час розробки гри все одно потрібно виконати дуже значну кількість роботи, щоб створити справді цікавий продукт. Однозначною перевагою розробки комп'ютерної гри з використанням готового інструменту (ігрового рушія) є швидкість розробки програмної частини. Якщо раніше необхідно було писати значну кількість програмного коду, щоб використовувати просту можливість перевірки зіткнення між двома об'єктами, то тепер з новими інструментальними засобами такі обчислення робляться всього за однією командою.

Огляд ігрових рушіїв

Unreal Development Kit (UDK) – потужний набір інструментів для створення динамічних 3D-програм та ігор на базі ігрового рушія Unreal Engine 3. Придатний для створення ігор вищого класу з реалістичною графікою. Безкоштовний для некомерційного використання, також доступне комерційне ліцензування. Підтримуються платформи Windows і iOS.

Game Maker – досить простий набір інструментів для розробки ігрових додатків, що дозволяє створювати ігри для великої кількості платформ – Windows, Mac OS X, Ubuntu, Android, IOS, Tizen, Windows Phone, Windows 8, PlayStation 3, PS 4, PS Vita, Xbox One і HTML 5. Є підтримка Steamworks. У разі успіху гри можливе імпортування (перенесення) гри на іншу платформу без зайвих ускладнень.

Швидкість розробки навіть за скромних знань і мінімальній мотивації суб'єктивно швидше, ніж в процесі використання інших систем. Встановлення та налаштування для початківців максимально просте і не вимагає особливих знань. Компіляція під інші платформи не вимагає зміни коду гри і здійснюється одним натисненням на кнопку маніпулятора миша.

Godot – відкритий багатоплатформовий 2D та 3D ігровий рушій, який поширюється під ліцензією MIT, що розробляється співавторством Godot Engine Community. До публічного релізу у вигляді відкритого програмного засобу рушій використовувався в деяких компаніях. Середовище розробника використовується на платформах Windows, Linux, OS X, BSD і Haiku та передбачено можливість експортувати ігрові проекти на ПК, консолі, мобільні та веб платформи.

Unity 3D – система розробки 3D програм для Mac OS, Windows, Wii, Xbox 360, PlayStation 3, iOS і Android. Вбудований редактор коду, орієнтований на візуальний підхід до створення ігор. Доступна безкоштовна версія і професійна комерційна ліцензія.

Unity – кросплатформовий інструмент для розробки тривимірних додатків, що функціонують зокрема під операційними системами Window, Linux OS X, Android. Всі версії Unity 3D вбудовано інтегрований редактор проектів, підтримується імпорт графічних та не графічних ресурсів (моделей, у тому числі анімованих, текстур, сценаріїв тощо), велика колекція вбудованих ландшафтів, шейдерну систему, за рахунок чого поєднується простота використання, гнучкість і продуктивність. Програмування графіки в Unity 3D здійснюється із використанням мов програмування JavaScript, Boo (діалект Phyton) та C # на основі платформи NET.

Серед основних особливостей Unity 3D необхідно зазначити вбудовану підтримку роботи у мережевому середовищі, використання фізичного рушія Ageia PhysX, змішування 3D-графіки в режимі реального часу з потоковим аудіо і відео. Через сервер ресурсів Unity забезпечується контроль версій в Unity 3D, підтримується широкий діапазон платформ: Windows, MacOs X, iPhone, iPod, iPad, Xperia PLAY, PS3, Flash 3D player.

Розробниками випускаються дві версії програмного продукту: звичайна версія і платна версія Unity 3D Pro. Перша відрізняється обмеженим функціоналом, використання другої дозволяє здійснити всі етапи графічного конвеєра, включаючи рендер в текстуру, ефекти пост-процесу, видалення з процесу рендерингу невидимих вершин і полігонів. На даний час розробниками проекту Unity видано 2018.4.1 (6.06.2019) версію Unity, з розширеним списком підтримуваних платформ: iOS, Android, XBox 360, PlayStation, Linux.

Основною концепцією розробки з використанням Unity 3D є розміщення на сцені легко керованих об'єктів, які, в свою чергу, складаються з багатьох компонентів. Створення окремих ігрових об'єктів і подальше розширення їх функціональності за допомогою додавання різних компонентів дозволяє нескінченно удосконалювати і ускладнювати проект.

Інструментарій Unity 3D використовується для розробки динамічних додатків з 2D та 3D графікою, опрацьовуваної у режимі реального часу, для реалізації концепції ігрового рушія (Game Engine).

Незважаючи на специфічність назви, ігрові рушії широко використовуються в інших типах динамічних додатків, що використовують 3D-графіку в режимі реального часу, таких, як демонстраційні рекламні ролики, архітектурні візуалізації, навчальні симулятори і середовища моделювання.

Ігровий рушій (Game Engine) – це центральний програмний компонент динамічних програм з тривимірною графікою, що опрацьовується в режимі реального часу, в тому числі комп'ютерних та відеоігор. Його використання забезпечує реалізацію основних технологій моделювання і 3D-візуалізації, спрощує процес розробки проектів, забезпечує можливість їх запуску на кількох платформах, таких як ігрові консолі та настільні операційні системи, наприклад, GNU Linux, Mac OS X і Microsoft Windows.

Через ігровий рушій забезпечується основна функціональність пакету Unity 3D. До нього включаються багаторазово використовувані програмні компоненти: графічний рушій (візуалізатор), фізичний рушій, звуковий рушій, систему сценаріїв, анімацію, штучний інтелект, набір бібліотек для підтримки мережевого середовища, управління пам'яттю і багатопотоковість.

На додаток до багаторазово використовуваних програмних компонентів, до ігрових рушіїв, як правило, додають набір візуальних інструментів для розробки проектів. На основі цих інструментів зазвичай складають інтегроване середовище для спрощеної, швидкої розробки динамічних додатків подібно до потокового виробництва, надаючи гнучку і багаторазово використовувану програмну платформу з усією необхідною функціональністю для розробки додатків, скорочуючи витрати, складність і час розробки.

Часто в ігрових рушіях передбачається компонентна архітектура, що дозволяє замінювати або розширювати деякі підсистеми рушія більш спеціалізованими (і часто більш дорогими) компонентами, наприклад, для симуляції фізичної природи дії (Havok), звуку (FMOD) або рендерингу (SpeedTree). Однак деякі ігрові рушії, такі як RenderWare, проектуються як набір мало пов’язаних компонентів, які можна вибірково комбінувати для створення власного рушія, замість більш традиційного підходу, який полягає в розширенні або налаштуванні гнучкого інтегрувального рішення.

Складові ігрових рушіїв

Графічний 3D-рушій (3D-graphics engine) – проміжне програмне забезпечення (ППО), основним призначенням якого є візуалізація (рендеринг) дво- або тривимірної комп'ютерної графіки. Може існувати як окремий продукт або у складі ігрового рушія і використовуватися для візуалізації

окремих зображень або комп'ютерного відео. Графічні рушії, що використовуються в програмах з опрацювання комп'ютерної графіки (таких, як 3Ds Max, Maya, Cinema 4D, Zbrush, Blender), зазвичай називаються «рендер» або «візуалізатор». Разом з тим основна і найважливіша відмінність «ігрових» графічних рушіїв від «не ігрових» полягає в тому, що перші повинні обов'язково працювати в режимі реального часу, тоді як другі можуть витрачати по кілька десятків годин на виведення одного зображення. Другою істотною відмінністю є те, що, починаючи приблизно з 1995-1997 року, графічні рушії генерують візуалізацію за допомогою графічних процесорів відеокарт (GPU). В програмних графічних рушіях використовуються тільки центральні процесори (CPU).

Фізичний рушій (physics engine) – програмний рушій, через який здійснюється моделювання фізичних законів реального світу у віртуальному просторі ігрової сцени з наперед заданою точністю апроксимації. Найчастіше фізичні рушії використовуються не як окремі самостійні програмні продукти, а як складові компоненти інших програм.

На практиці використання фізичних рушіїв дозволяє наповнити віртуальний ігровий простір статичними і динамічними об'єктами, вказати якісь загальні закони взаємозв’язків тіл і середовища в цьому просторі, які в тій чи іншій мірі наближені до фізичних, задаючи характер і міру взаємозв’язків. Власне розрахунок взаємозв’язків тіл здійснюється через рушій. Коли простого набору об'єктів, взаємопов’язаних за певними законами у віртуальному просторі, недостатньо, в силу неповного наближення фізичної моделі до реальної, можливо додавати до тіл зв'язки (joint). *Зв'язки є обмеженням фізики об'єктів, кожне з яких може накладатися на одне або два тіла.*

Розраховуючи взаємозв’язки тіл між собою та з середовищем, через фізичний рушій наближається фізична модель одержуваної системи до реальної, з передаванням уточнених геометричних даних в графічну систему. У Unity 3D для фізичного рушія використовується Nvidia's PhysX Engine.

Середовище Unity як інструмент розробки

Це мульти-платформовий інструмент для розробки дво- і тривимірних додатків та ігор, що функціонує під операційними системами Windows і OS X. Створені за допомогою Unity програми мають використовуватись під операційними системами Windows, OS X, Android, Apple iOS, Linux, а також на ігрових приставках Wii, PlayStation 3 і Xbox 360 (Рис. 1.).

Рис. 1. Головне вікно середовища Unity 3D.

Ресурси (Assets) проекту – це будівельні (складові) блоки всіх проектів Unity, в якості яких можуть бути використані файли зображень (текстури), 3D-моделі, звукові файли, які будуть використовуватися під час створення проекту.

Якщо ресурс (наприклад, геометрична 3D-модель) використовується в сцені гри, він стає в термінології Unity ігровим об'єктом (Game Object). До всіх цих об’єктів спочатку включається хоча б один компонент, що задає його положення в сцені і можливі перетворення (компонент Transform). Через змінні компонента Transform визначається положення (position), поворот (rotation) і масштаб

(scale) об'єкта в його локальній декартовій прямокутній системі координат X, Y, Z. Наявність змінних стосовно кожного компонента обумовлює можливість звернення до них з відповідної програми (сценарію).

Компоненти (components) в Unity 3D використовуються за різними призначенням: через них можна впливати на поведінку, зовнішній вигляд і багато інших функцій об'єктів, до яких прикріплюються (attaching). В Unity передбачено безліч компонентів різного призначення.

Для забезпечення динамічності різних 3D-додатків в Unity 3D використовуються сценарії, невеликі фрагменти програмного коду, які також є компонентами середовища.

Для розробки сценаріїв можна використовувати мови програмування UnityScript (діалект мови програмування JavaScript), C#, Boo (діалект мови програмування Python). Разом з тим JavaScipt набагато більш пристосований для розробки ігрового проекту завдяки своїй гнучкості, інтенсивному розвитку і загальній адаптованості для розробки Інтернет та прикладних додатків.

З метою дослідження можливостей використання та вивчення середовища Unity 3D було розроблено з навчальною метою платформонезалежну гру «Lemon» (Рис. 2). Завдання гри полягає в тому, щоб зібрати на ігровому полі за допомогою кульки якомога більше «золотих» монеток, ухиляючись від рухомих перешкод.

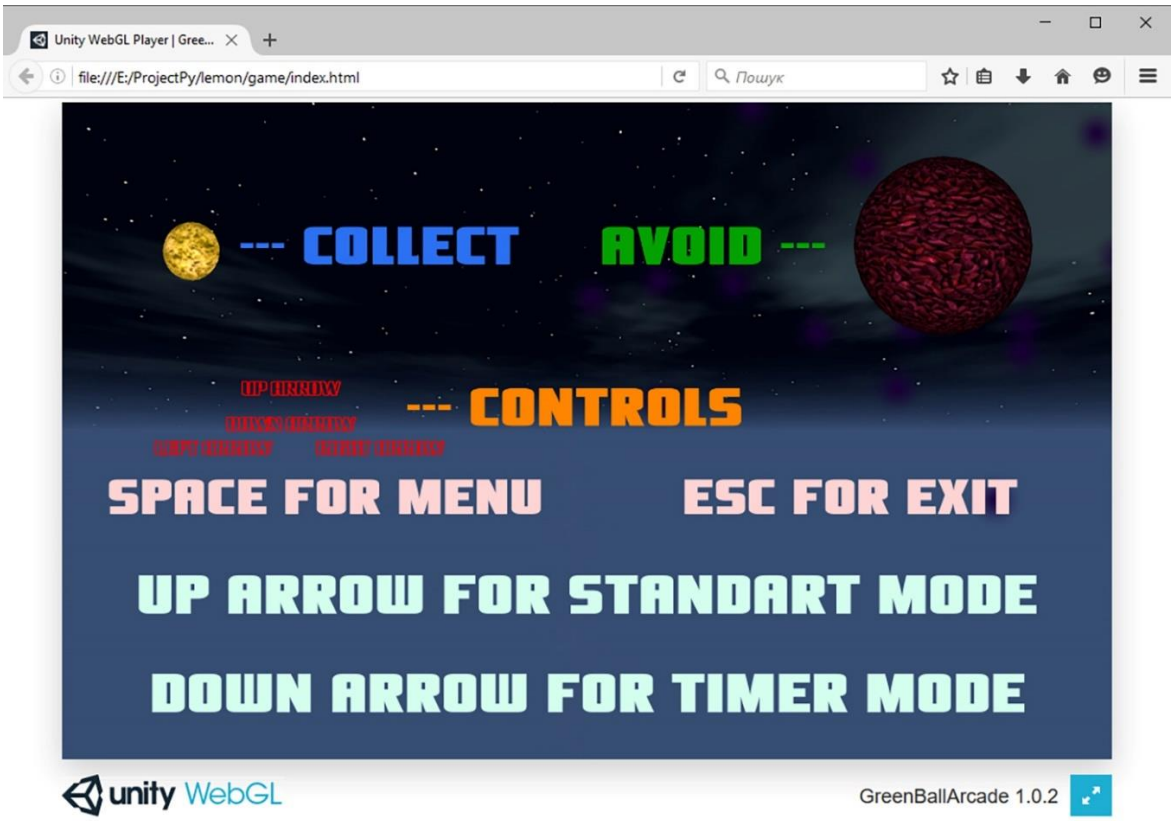

Рис. 2. Стартова сторінка і головне меню web-версії розробленої гри «Lemon»

Основні функції: *під час розробки гри необхідно дотримуватися наступних вимог.*

1. У розроблюваній грі передбачається два режими:
   a. стандартна гра – зібрати максимальну кількість монеток;
   b. гра на час – зібрати максимальну кількість монеток за відведений час – 10 сек.
2. Після підбору монетки кількість перешкод збільшується на 1 у стандартній грі, а в грі на час додається додатковий час 10 сек.
3. Управління кулькою здійснюється клавішами управління курсору, у версії для платформи Android за допомогою спеціальних датчиків руху (гіроскоп, акселерометр).
4. Клавіша ESC використовується для завершення гри.
5. Після наїзду кульки до межі ігрового поля відбувається перемикання камери для огляду ігрової сцени, поки кулька знову не вийде на поле.
6. Перешкоди у формі куль рухаються по ігровому полю самостійно і випадковими траєкторіями.
7. Під час підбору монетки на екрані виникає ефект ареолу.
8. Після «наїзду на перешкоду» гра припиняється і лічильник обнуляється.
9. Гра містить кілька рівнів. Кожен рівень відрізняється кількістю перешкод.

10. На екрані відображається кількість підібраних монеток, рівень, час до завершення гри у режимі гри на час.
11. Після підбирання 3 монеток рівень гри збільшується на 1.

Структура проекту гри «Lemon»

Після створення нового проекту було створено перший елемент – меню. Наступний етап – заповнення сцени ігровими об’єктами: камера, через яку відбувається рендеринг рівня, GUI текст (game user interface), джерело світла, а також монетка і фіолетова куля.

Для того, щоб створити новий об’єкт, треба натиснути відповідну кнопку на верхній панелі завдань і з контекстного меню обрати відповідну опцію. Після того, як об’єкт створено, необхідно надати йому певних властивостей, наприклад, текстове меню, позицію на екрані. Для цього слугує панель «Inspector», де можна маніпулювати з об’єктом, змінювати та надавати йому нових атрибутів (Рис. 3).

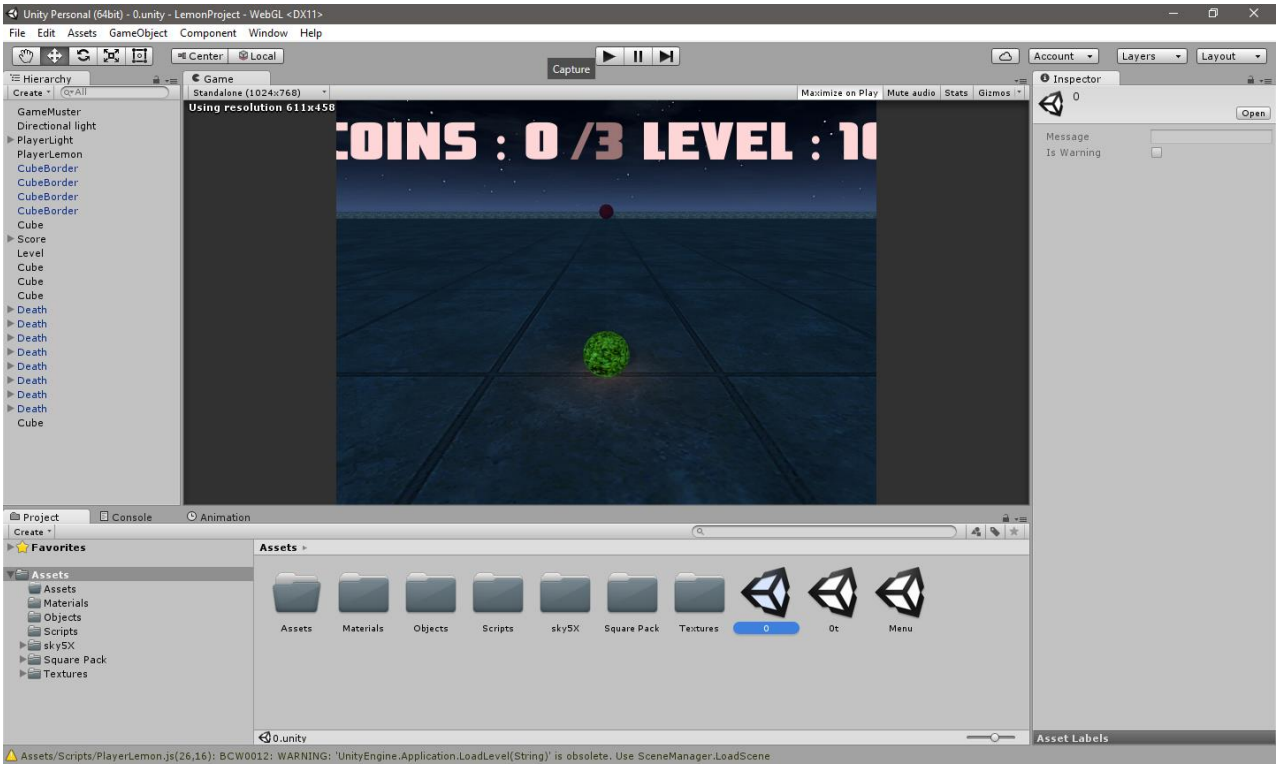

Рис. 3. Головне вікно Unity 3D з розроблюваним проектом гри «Lemon».

Позицію і розташування об’єкта можна змінювати у “Inspector” або у вікні з сценою рівня. Після створення всіх необхідних ігрових об’єктів для меню, необхідно написати функцію переходу до ігрового рівня. Тобто перебуваючи в меню під час натискання на клавішу “вгору” буде завантажено рівень з назвою “0” (стандартний), а під час натискання на клавішу “вниз” – рівень з назвою “0t” (на час).

```
// Приклад сценарію Menu.js
function Update () {
if (Input.GetKeyUp ("up"))
 {
 Application.LoadLevel ("0");
 }
 if (Input.GetKeyUp ("down"))
 {
 Application.LoadLevel ("0t");
 }
 if (Input.GetKeyUp ("escape"))
 {
 Application.Quit ();
 }
}
```

В меню використовується один сценарій – “Menu”. Через нього забезпечується перехід від меню до ігрового рівня, а також під час переходу від ігрового рівня до меню за цим сценарієм обнуляється рахунок, час та рівень гравця, для того щоб нова гра розпочалася з нуля.

У проекті використовується 3 рівні – меню, стандартний і таймер. Кожному рівню відповідає 3D сцена та GameObjects, для яких можна задавати певні властивості (фізичні властивості, форму, розмір, текст, текстуру, звуки, сценарії тощо). Налаштування параметрів здійснюється за допомогою панелі «Inspector» головного вікна середовища Unity 3D.

Взаємозв'язок між GameObjects залежить від їх налаштувань та прикріплених сценаріїв. Щоб GameObjects почали виконуватися їх необхідно розмістити на сцені. Сценарій також буде виконуватися тільки тоді, коли GameObject, до якого він прикріплений, розміщений на сцені.

Переходи і взаємозв'язок між рівнями здійснюється також із використанням сценаріїв. Тобто, можна задати будь-який взаємозв'язок. Розробка сценаріїв здійснювалася мовою програмування UnityScript.

Сценарій завжди додається до одного з ігрових об'єктів. Щоб додати сценарій до об'єкту, необхідно обрати об'єкти за допомогою панелі "Inspector", додати новий компонент за допомогою відповідної кнопки з контекстного меню, обравши Scripts, після чого на екрані буде виведено перелік сценаріїв, які вже є в матеріалах проекту, а також з яких можна обирати необхідний сценарій, і він автоматично буде доданий до об'єкту. До одного об'єкту можна додати кілька сценаріїв.

Розроблюваний проект гри складається з наступних об'єктів:

- **Materials** – матеріали, які використовуються під час маніпулювання з ігрових об'єктів, матеріалами фактично є модифіковані текстури разом із звуками;
- **Objects** – містить безпосередньо ігрові об'єкти, які розміщуються на сцені;
- **Scripts** – містить розроблені сценарії.

Перелік ресурсів гри «Lemon»:

- **Sky5x** – це є набір текстур для скайбоксів, спеціальний тип текстур для неба і папка з матеріалами для системи частинок (ареол, який з'являється навколо монеток після їх підбирання);
- **Square Pack** – папка з використовуваними шрифтами;
- **Textures** – текстури, які використовуються в матеріях для GameObjects.

Перелік стандартних об'єктів, що використовуються під час розробки гри:

- **Камера** – рендер зображення;
- **Світло** – використовується три типи освітлення: освітлення поля гри, освітлення кульки, освітлення монетки;
- **Об'єкт «Паралелепіпед»** – створення ігрового поля;
- **Об'єкт «Куля»** – створення кульок перешкод, основної кульки, монетки, великих кульок, які є однаковими для двох режимів гри.

Перелік розроблених сценаріїв та їх призначення в грі «Lemon»:

- **Camra** – наближається камера, коли кулька біля бордюру, і віддаляється, коли не біля бордюру;
- **Coin** – «поведінка» монетки в звичайному режимі, обертання, зникання після дотику і додавання «+1» до рахунку;
- **CoinTimer** – поведінка монетки в звичайному режимі, обертання, зникання після дотику, до таймера додає час +10 с.;
- **DeathOBJ** – поведінка великих кульок через які програєш, вони рухаються випадково;
- **DeathTrig** – тригер визначення кінця гри, якщо відбувається дотик до великої кульки;
- **GM** – основний сценарій з правилами гри для основного режиму, управляє взаємозв'язком між об'єктами, рівнів і розташовує монетки і великі кульки;
- **GMt** – основний сценарій з правилами гри для режиму гри на час, управляє взаємозв'язком між об'єктами, рівнів і розташовує монетки і великі кульки;
- **Level** – текст з лічильником рівнів і встановлює текст "Рівень + 1";
- **LevelUpText** – вилучає текст з екрану "Рівень + 1";
- **Menu** – сценарій меню, управляє кнопками і обертає монетку в меню;
- **PlayerLem** – сценарій для зеленої кульки вперед-назад взаємозв'язком між об'єктами;
- **PlayerLight** – сценарій управління освітленням зеленої кульки, обертання камери і кульки в ліво або вправо, залежно від положення камери (вони пов'язані);
- **Score** – текст з рахунком;
- **Timer** – текст з таймером і сам таймер.

Для створення меню використаний набір шрифтів "Square Pack", текстури SkyBox для створення фону. Всі матеріали є безкоштовними, і завантаженні з онлайн магазину Unity Store. Функції всіх ігрових об'єктів описані у різних невеликих сценаріях, що значно полегшує налагодження програми.

Висновки. Unity 3D – засіб для розробки віртуальних ігрових програм, віртуальних середовищ, є одним з лідерів стосовно зручності та швидкості розробки 3D ігор та інших програмних продуктів у порівнянні з іншими розглянутими у роботі засобами. Unity має цілу низку переваг порівняно зі своїми найближчими аналогами:

1) використання для написання сценаріїв таких поширених мов C#, JavaScript, Phyton (Boo), що дозволяє програмісту легко увійти в розробку гри без вивчення додаткових мов програмування. Це важливий момент, тому що на відміну від інших рушіїв, де використовується мова C++, у використовуваних мовах є багато елементів і засобів, які вже реалізовані, і програмісту потрібно тільки скористатися ними.

2) платформонезалежність, тобто один і той же код, написаний для Unity 3D, з мінімальними змінами може бути перенесений на різні платформи (PC, Mac, Android, iOS, Web, ігрові консолі). Це скорочує час на розробку гри в кілька разів.

3) Unity Store, де розміщено велику кількість різних плагінів і ресурсів для створення гри. Зрозуміло, що є безкоштовні та платні, але всі вони зібрані в одному місці зі зручним пошуком та можливістю завантажити, інтегрувати і отримати відразу робочий функціонал.

Готові ігрові рушії значно спростили життєвий цикл розробки та супроводу ігрових додатків. Однак, створення сценаріїв це тільки одна сторона розробки комп'ютерної гри. Створення якісного ігрового продукту буде так само включати в себе створення великого обсягу графічного матеріалу.

Отже, розробкою відео ігор може займається розробник у одній особі або авторський колектив. Для розробки ігор використовують різні інструментальні засоби, одним з яких є конструктор ігор.

Список використаних джерел

[1] Торн А. Искусство создания сценариев в unity. Пер. с англ. ДМК Пресс, 2015. – 368 с.
[2] Хокинг Д. Unity в действии. Мультиплатформенная разработка на C#. Пер. с англ. Издательский дом Питер, 2016. 336 с.
[3] Unity Real-Time Development Platform | 3D, 2D VR & AR Engine/ URL: http://www.unity3d.com

References

[1] Thorne A. (2015) The art of scripting in unity. Translation from English. DMK Press, 368 p.
[2] Hawking D. (2016) Unity in action. Multiplatform development in C #. Translation from English. Piter Publishing House, 336 p.
[3] Unity Real-Time Development Platform | 3D, 2D VR & AR Engine/ URL: http://www.unity3d.com

V. Habrusiev, A. Velhach, O. Kulianda

RESEARCH ON THE FUNCTIONAL FEATURES OF UNITY 3D MOVEMENT ON THE CASE OF 3D MINI-GAME REALIZATION

Abstract. Currently, the development of computer games is relevant, as with the expansion of the market of personal electronics, the entertainment market is expanding. General availability of tools – game engines, access to various informative educational and reference materials make it possible to develop application software easily, and quickly, such as a computer game.

There are many publications and various scientific studies on the topic of modeling game scenes. This stage of development of gaming computer applications is quite extensive and includes many tasks. It should be noted that in this case, the authors touch on the topic not only of modeling game scenes, but also superficially cover the topic of creating, modeling, texturing game objects, while describing the functionality with which these tasks are performed.

This article considers the most known game engines, their comparative analysis is made. Their advantages and disadvantages are revealed, as well as a range of tasks for which they are easy to use. When comparing different game engines, the authors chose Unity 3D for several reasons: favorable licensing policy; compatibility with any platform; great community; a large amount of documentation; ease of use.

The purpose of this article is to fully describe the process of modeling the scene of a gaming computer application in Unity 3D. To do this, the article considers as an example the implementation of a simple 3D game. In particular, the functionality and tools for modeling a scene in the Unity environment are described, as well as a brief description of the main objects that can be added to the scene. However, a brief overview of the editor and inspector window with all the properties and methods of game objects is given.

The authors try to convey to readers the pros and cons of using this environment to create simple game applications, describe the main menu and tools superficially, without delving into the intricacies of the entire development process.

Keywords: Unity 3D, JavaScript, game application, development environment, programming language.