**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД**
**ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ**
**КАФЕДРА «ПРИКЛАДНА МАТЕМАТИКА ТА ІНФОРМАТИКА»**

# НАВЧАЛЬНИЙ МОДУЛЬ

## «Розробка ігрових додатків на базі движка Unity» дисципліни «Інструментальна підтримка розробки комп'ютерних ігрових додатків» підготовки студентів освітнього рівня «магістр» за спеціалізацією «Програмне забезпечення мультимедійних систем для ігрових додатків» спеціальності 121 «Інженерія програмного забезпечення»

**GAMEHUB: UNIVERSITY-ENTERPRISES COOPERATION IN GAME INDUSTRY IN UKRAINE"**

**GAMEHUB: СПІВРОБІТНИЦТВО МІЖ УНІВЕРСИТЕТАМИ ТА ПІДПРИЄМСТВАМИ В СФЕРІ ГРАЛЬНОЇ ІНДУСТРІЇ В УКРАЇНІ»**

**561728-EPP-1-2015-1- ES-EPPKA2-CBHE-JP**

*The handbooks were performed with support of the Erasmus+ Programme of the European Union (561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP). The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.*

**Покровськ, 2018**

УДК 004.032.6 : 004.92

Навчальний модуль «Розробка ігрових додатків на базі движка Unity» дисципліни «Інструментальна підтримка розробки комп'ютерних ігрових додатків» підготовки магістрів за спеціалізацією «Програмне забезпечення мультимедійних систем для ігрових додатків» спеціальності 121 «Інженерія програмного забезпечення» / Укладачі: Т.В. Скрипник, О.А. Тихонова. – Покровськ: ДонНТУ, 2018. – 87 с.

Навчальний модуль «Розробка ігрових додатків на базі движка Unity» дисципліни «Інструментальна підтримка розробки комп'ютерних ігрових додатків» входять до дисциплін підготовки студентів освітнього ступеня «магістр» спеціальності 121 «Програмна інженерія» за спеціалізацією «Програмне забезпечення мультимедійних систем для ігрових додатків», яка впроваджена в рамках виконання гранту Еразмус+ 561728-EPP-1-2015-1- ES-EPPKA2-CBHE-JP «GameHub: Співробітництво між університетами та підприємствами в сфері грального індустрії в Україні»

Укладачі:
**Т.В. Скрипник**, асистент кафедри ПМІ;
**О.А. Тихонова**, асистент кафедри ПМІ.

Рецензенти:
**Святний В.А.**, завідувач кафедри КІ, д.т.н., професор;
**Вовна О.В.**, завідувач кафедри ЕТ, д.т.н., доцент.

Розглянуто на засіданні кафедри «Прикладної математики і інформатики»
протокол № 11 від 15 травня 2018 року

Затверджено навчально-видавничим відділом ДонНТУ,
протокол № 14 від 12 червня 2018 року

Рекомендовано до друку Вченою радою ДонНТУ,
протокол № 10 від 21 червня 2018 року

**ЗМІСТ**

# ЗАГАЛЬНИЙ ВСТУП

В рамках виконання гранту Еразмус+ 561728-EPP-1-2015-1- ES-EPPKA2-CBHE-JP «GameHub: Співробітництво між університетами та підприємствами в сфері грального індустрії в Україні» в Донецькому національному технічному університеті впроваджена підготовка студентів спеціальності 121 «Програмна інженерія» за спеціалізацією «Програмне забезпечення мультимедійних систем для ігрових додатків».

Навчальний план підготовки студентів на рівні «магістр» передбачає опрацювання наступних навчальних модулів:

- «Методи теорії ігор в ігрових додатках» (Theory of Games in Game Applications), дисципліна «Математична теорія ігор»;
- «Розробка ігрових додатків на базі движка Unity» (Game Applications Development Based on Unity Engine), дисципліна «Інструментальна підтримка розробки комп'ютерних ігрових додатків»;
- «Розробка ігрових додатків для OS Android» (Game Applications Development for OS Android Platform), дисципліна «Інструментальна підтримка розробки комп'ютерних ігрових додатків»;
- «Розробка ігрових додатків на базі HTML-5 для WEB» (Game Applications development for WEB based on HTML-5) , дисципліна «Інструментальна підтримка розробки комп'ютерних ігрових додатків»;
- «3D графіка в ігрових додатках (на базі графічного редактора Blender)» (3D Graphics in Game Applications (Based on Blender Game Engine), дисципліна «Комп'ютерний синтез та обробка зображень»;
- «Місце ігрових додатків на ринку програмного забезпечення» (Place of Game Applications in Software Market), дисципліна «Сучасні засоби інформатики та комп'ютерний ринок».

Навчальний план підготовки студентів на рівні «бакалавр» передбачає опрацювання наступних навчальних модулів:

- «Архітектура ігрових додатків» (Game Applications Architecture), дисципліна «Архітектура та проектування програмного забезпечення»;
- «Основи створення ігрових додатків (на базі GameMaker)» (Basics of Game Applications Development with GameMaker Studio), дисципліна «Комп'ютерна графіка»;

- «Особливості тестування ігрових додатків» (Features of Game Applications Testing), дисципліна «Якість ПЗ та тестування»;
- «Командна розробка ігрових додатків» (Team Development of Game Applications), дисципліна «Групова динаміка і комунікації».

Для кожного модуля розроблено: довідник модуля, опорні конспекти (презентації) лекцій та методичні вказівки виконання лабораторних робіт і практичних занять за модулем. Методичні матеріали всіх модулів, доступні за посиланням http://89.185.3.253:9090/login , log: guest, pass: gamehub.

Дане видання містить опис навчального модуля «Розробка ігрових додатків на базі движка Unity» дисципліни «Інструментальна підтримка розробки комп'ютерних ігрових додатків» входять до дисциплін підготовки студентів освітнього ступеня «магістр» спеціальності 121 «Програмна інженерія».

# Д О В І Д Н И К  М О Д У Л Я

**«Розробка ігрових додатків на базі движка Unity»**

**дисципліни «Інструментальна підтримка розробки комп'ютерних ігрових додатків»**

**підготовки студентів освітнього рівня «магістр» за спеціалізацією «Програмне забезпечення мультимедійних систем для ігрових додатків» спеціальності 121 «Інженерія програмного забезпечення»**

# 1 Вступ

Метою викладання навчальної дисципліни “Інструментальна підтримка розробки комп’ютерних ігрових додатків” є формування знань та вмінь студента в області розробки і використання програмного забезпечення засобами сучасних середовищ підтримки та розробки ігрових додатків.

Основними завданнями вивчення дисципліни “Інструментальна підтримка розробки комп’ютерних ігрових додатків ” є:

- знайомство із основними інструментальними засобами створення сучасних ігрових додатків для різних апаратних та програмних платформ, порівняльний аналіз особливостей, переваг і недоліків цих платформ, оцінка перспектив їх використання;
- огляд сучасних движків для створення ігрових додатків, що дозволяють оптимізувати та прискорити процес розробки за допомогою ефективної реалізації модульної розробки додатку та ефективної підтримки кросплатформності;
- оволодіння повним циклом розробки типового ігрового додатку, організація та контроль процесів планування, розробки, просування та підтримки ігрового додатку, вміння спланувати роботу команди розробників відповідно до спеціалізації кожного з них;
- реалізація типових елементів ігрових додатків у відповідності до платформи реалізації, оцінка ефективності організації ігрової механіки з урахуванням типових сценаріїв використання інтерфейсу користувача та різних способів взаємодії з платформою;
- урахування особливостей платформи реалізації ігрового додатку, вміння використати ці особливості належним чином, оптимізація програмного коду відповідно до можливостей апаратних платформ.

**Змістовний модуль** «Розробка ігрових додатків на базі движка Unity» дисципліни «Інструментальна підтримка розробки комп’ютерних ігрових додатків» орієнтовано на оволодіння студентами практичних навичок використання ігрового движка Unity для побудови ігрових додатків..

# 2 Опис модуля

Галузь знань: 12 “Інформаційні технології”.

Напрям підготовки: 121 “Інженерія програмного забезпечення”.

Рівень: магістр.

Назва дисципліни: Інструментальна підтримка розробки комп’ютерних ігрових додатків

Назва змістовного модуля: Розробка ігрових додатків на базі движка Unity

Семестр: **2**

Кількість кредитних одиниць: дисципліна - 6,0, модуль -2,0

Орієнтовна кількість часів: дисципліна - 180, модуль - 60

Викладачі : асистент Скрипник Т.В.;

асистент Тихонова О.А.

# 3 Мета та передбачувані результати вивчання модуля

## 3.1 Мета модуля

Мета модуля – сформувати у студентів поглиблені знання в галузі розробки ігрових додатків з використанням середовища Unity, сформувати необхідні навички та уміння для практичного використання інструментів движка Unity в процесі створення ігрового додатку.

### 3.1.1 Знання та їх використання

У разі успішного оволодіння матеріалами модуля студент буде вміти:

- обґрунтувати використання відповідних об’єктів движка Unity для вирішення задач створення геймплею;
- аналізувати та використовувати скрипти Unity для організації геймплею;
- використовувати теоретичні та практичні знання мови C# для управління об’єктами Unity;
- проводити якісну оцінку користувальницького інтерфейсу та засобів його створення в Unity;
- проводити адаптацію користувальницького інтерфейсу з урахуванням програмного та апаратного оточення.

### 3.1.2 **Дослідницькі навички**

У разі успішного вивчення модуля студент буде вміти

- досліджувати особливості ігрових додатків з метою вибору методів, технологій та інструментів розробки;
- досліджувати об’єкти Unity та засоби їх використання при створенні ігрових додатків різних типів та жанрів;
- на основі базових знань мов програмування системно аналізувати програмні засоби, що можуть бути використані для створення геймплею;
- формулювати власні висновки й узагальнення стосовно використання засобів движка Unity та графічних редакторів, файли яких сумісні з Unity.

### 3.1.3 **Спеціальні вміння**

У разі успішного вивчення модуля студент буде вміти:

- аналізувати особливості ігрового додатку з метою вибору оптимальних засобів реалізації геймплея в середовищі Unity;
- визначати шляхи створення інтерактивного геймплея з використанням стандартних та власних ігрових персонажів;
- визначати засоби організації взаємодії об’єктів в ігровому середовищі;
- приймати участь у розробці ігрових додатків різних жанрів.

### 3.1.4 **Соціальні вміння**

У разі успішного вивчення модуля студент буде вміти: дотримуватися регламентних рамок, виконувати строки проведення та виконання робіт, критично ставитися до результатів особистої роботи, вміти проводити самооцінку виконаних робіт, проводити якісний аналіз та давати оцінку роботі колег в проектній групі, обґрунтовувати використання сучасних інструментальних засобів розробки ігрових додатків в процесі колективної роботи над проектом, приймати участь в групових обговореннях проекту ігрового додатку, презентувати та аргументувати свої рішення.

### 3.1.5 **Особисті якості**

У разі успішного вивчення модуля студент буде вміти:

- самостійно вирішувати питання, що стосуються вибору засобів реалізації ігрового додатку в середовищі Unity;

- аналізувати сучасну науково-технічну літературу із розробки ігрових додатків засобами движка Unity та визначати прогресивні шляхи створення ігрових додатків.

## 4 Місце модуля в структурі дисципліни

| Номер | Змістовний модуль | Тиждень вивчання |
|---|---|---|
| 1 | Розробка ігрових додатків на базі движка Unity | 1-6 |
| 2 | Розробка ігрових додатків на для OS Android | 7-11 |
| 3 | Розробка ігрових додатків на для OS iOS. | 12-16 |

## 5 Інформаційне наповнення змістовного модуля 3

| Номер тижня | Зміст |
|---|---|
| 1 | Огляд сучасних інструментальних засобів розробки ігрових додатків. Основи роботи в середовищі Unity. Створення простого ігрового середовища засобами Unity. |
| 2-3 | Організація геймплея за допомогою скриптів в Unity. |
| 4-5 | Динамічне створення об’єктів. Створення персонажу та керування ним. Особливості використання тригерів. Створення інтерактивного геймплея з персонажем за допомогою засобів движка Unity. |
| 6 | Організація користувальницького інтерфейсу засобами Unity. Збірка та налаштування готового проекту. |

## 6 Форми навчання

Навчальне навантаження модуля складається з аудиторної та самостійної роботи. Аудиторна робота включає 6 лекцій, 3 лабораторні роботи (наскрізна структура завдань) та заключний практичний семінар із обговорення отриманих результатів.

Лабораторні роботи 1-3 модуля орієнтовані на послідовне (step by step) виконання наскрізного завдання та мають на меті відпрацювання навичок і вмінь розробки ігрових додатків у програмному середовищі

Unity. Кожен студент відпрацьовує навички на сценах ігрового додатку, що відповідають його індивідуальному завданню. Розроблені сцени складають частини загального проекту та повинні наприкінці курсу бути об'єднані в один працездатний ігровий додаток.

Практичний семінар – заняття, на якому студенти презентують сцени ігрового додатку, що були розроблені згідно наскрізного завдання. Кожен студент в команді повинен оцінити результати роботи колег, виступаючи у ролі тестувальників та перших гравців. За результатами обговорення на семінарі формулюються зауваження щодо функціональних та нефункціональних вимог до гри, що сприяли б підвищенню якості геймплея та інтересу гравців.

Заліковий звіт – надання докладного письмового звіту та презентація в групі гри, розробленої в програмному середовищі Unity, при послідовному виконанні наскрізного завдання лабораторних робіт 1-3.

Самостійна робота студентів передбачає підготовку до аудиторних занять і лабораторних робіт, а також підготовку до тесту та оформлення залікового звіту та презентації за результатами виконання лабораторних робіт.

Підготовка до поточних аудиторних занять є аналіз літератури, Інтернет-матеріалів по темам лекцій і лабораторних робот, підготовка до тестів.

Контактні години передбачають індивідуальні консультації та контроль студентів в он-лайн режимі.

## 7 Порядок проведення атестації

Загальний принцип оцінювання підсумкових знань студента з курсу " Інструментальна підтримка розробки комп'ютерних ігрових додатків " полягає в тестуванні студентів на лекціях, оцінці поточної практичної роботи студента у навчальному семестрі на лабораторних роботах та оцінки контрольного заходу у формі іспиту, у результаті яких студент має сумарну оцінку в балах.

За результатами вивчення кожного змістовного модуля передбачена оцінка виконання залікового завдання. Оцінка включає: результати тестування студентів на лекціях, результати виконання лабораторних робіт модуля, обговорення результатів (самооцінка та взаємооцінка), оцінку

залікового звіту. Сумарно оцінка кожного змістовного модуля не може бути більш ніж 20 балів. Максимальна оцінка іспиту 40 балів.

Оцінка результатів вивчення дисципліни в цілому

| **Поточне тестування** | **Балів** |
|---|---|
| Змістовний модуль 1 | до 20 |
| Змістовний модуль 2 | до 20 |
| Змістовний модуль 3 | до 20 |
| Іспит | до 40 |
| Разом | До 100 |

Графік проведення поточного оцінювання модуля 1

| **Номер тижня** | **Оцінювання** |
|---|---|
| 1 | Оцінювання виконання лабораторної роботи 1 |
| 3 | Оцінювання виконання лабораторної роботи 2 |
| 5 | Оцінювання виконання лабораторної роботи 3 |
| 6 | Оцінка за результатами представлення та обговорення отриманих результатів на заключному практичному семінарі<br>Оцінювання залікового звіту |

Методики оцінки виконання лабораторних робіт, презентації та обговорення результатів на заключному практичному семінарі, та оцінювання заключного звіту наведені в методичних вказівках до виконання лабораторних робіт.

Інструментарій оцінювання рівня сформованості соціальних компетенцій передбачає оцінку викладачем і взаємну оцінку студентами навичок: планування виконання наскрізного завдання, контролю ходу виконання завдання, логічно вірного, аргументованого і чіткого уявлення та викладання результатів виконання наскрізного завдання, адекватного реагування на критику і зауваження інших учасників заключного семінару.

При оцінюванні результатів вивчення модуля 1 необхідно враховувати наступне. При умові виконання кожної окремої лабораторної

роботи єдиний заключний письмовий звіт за модулем надається студентом на 6 тижні семестру. Подовження терміну можливе лише при наявності поважної причини, передбаченої порядком навчання студентів у вищій школі. При цьому:

- електронна версія єдиного звіту надається викладачеві через систему Інтернет на початку 6 тижня;
- при обговоренні на заключному практичному семінарі на 6 тижні студент демонструє закінчений варіант сцен, інтегрованих у загальну гру з використанням об’єктів та скриптів, як створених в Unity, так й імпортованих до ігрового додатку з графічних редакторів і зовнішніх файлів;
- за кожний день прострочки представлення та здачі єдиного звіту по модулю 1 знімається 1 бал (не більш ніж 5 днів – 5 балів).

Метод оцінки змістовного модуля

Кількість балів в загальній оцінці змістовного модулю відповідає наступному:

| | |
|---|---|
| Виконання лабораторної роботи 1 | максимально 4 бали. |
| Виконання лабораторної роботи 2 | максимально 4 бали. |
| Виконання лабораторної роботи 3 | максимально 4 бали. |
| Практичний семінар | максимально 4 бали. |
| Оцінка залікового звіту | максимально 2 бали. |
| Підсумкове тестування по модулю | максимально 2 бали. |

Усі набрані бали підсумовуються (максимально 20 балів), штрафні бали за запізнення в виконанні лабораторних робіт та представленні єдиного звіту (максимально мінус 5 балів) віднімаються. Сумарна оцінка (від 0 до 20 балів) є індивідуальна оцінка студента освоєння змістовного модуля 1.

Метод оцінки дисципліни в цілому

Оцінки студентів за результатами вивчення змістовних модулів 1 – 4 підсумовуються. Оцінка іспиту (максимально 40 балів) додається. Таким чином розраховується сумарна оцінка студента в балах за дисципліною.

Сумарна оцінка в балах, переводиться за нижченаведеною шкалою оцінювання в національну оцінку:

| **Сума балів за всі види навчальної діяльності** | **Оцінка за національною шкалою** |
|---|---|
| 90 – 100 | Відмінно |
| 74 -89 | Добре |
| 60-73 | Задовільно |
| 0-59 | Не зараховано |

## 8 Зворотній зв’язок

Інформація щодо результатів тестування, виконання лабораторних робіт та загальна оцінка змістовного модулю надається кожному студенту як індивідуально, так і всієї групи в цілому.

Інформація щодо результатів тестування надається студентам на 13 тижні навчання.

Інформація щодо оцінки виконання лабораторної роботи надається студентові під час заняття.

Інформація щодо оцінки змістовного модуля в цілому надається студентам на 13 тижні навчання.

Зворотній зв'язок передбачає горизонтальні комунікації, які полягають у обміні інформацією в процесі виконання завдань і у взаємній оцінці виконаної роботи на заключному семінарі. При цьому важливим є те, що студенти добре розуміють специфіку роботи інших студентів, знають про проблеми, що виникають в процесі виконання лабораторних робіт. Такі комунікації сприяють формуванню організаційної культури, впливають на групові норми, цінності, моделі поведінки, стиль спілкування.

Контактні дані для on-line допомоги та консультування:

Викладачі : асистент Скрипник Т.В., tetiana.skrypnyk@donntu.edu.ua, асистент Тихонова О.А., oksana.tykhonova@donntu.edu.ua.

# 9 Викладацький склад та допоміжні джерела

Обов'язки викладачів.

- подача матеріалів модуля згідно з програмою
- оцінка результатів тестування та виконання лабораторних робіт

Обов'язки координатора дисципліни:

- планування та внесення змін до модуль;
- координація і управління професорсько-викладацьким складом;
- координація проведення тестування, лабораторних робіт та іспиту.

Обов'язки допоміжного персоналу.

Допоміжний персонал здійснює підготовку комп’ютерної техніки до виконання лабораторних робіт студентами та надає технічну підтримку студентів під час виконання лабораторних робіт.

Контактні дані викладачів:
асистент Скрипник Т.В., tetiana.skrypnyk@donntu.edu.ua,
асистент Тихонова О.А., oksana.tykhonova@donntu.edu.ua.
Контактні дані куратора:
професор, д.т.н. Башков Є.О., eabashkov@i.ua .

# 10 Навчальна програма і матеріали

## 10.1 Тема 1 Огляд сучасних інструментальних засобів розробки ігрових додатків. Основи роботи в середовищі Unity. Створення простого ігрового середовища засобами Unity.

### 10.1.1 Мета та очікувані результати

Ознайомити студентів із основними інструментальними засобами створення сучасних ігрових додатків для різних апаратних та програмних платформ.

Ознайомити студентів з елементами інтерфейсу Unity та базовими об’єктами, що використовуються для створення 2D та 3D ігор. Ознайомити студентів з основними підходами до побудови ігрових додатків засобами Unity. Ознайомити студентів з особливостями роботи з об’єктами різних типів в середовищі Unity.

Сформувати у студентів вміння, навички та основні прийоми роботи з інтерфейсом движка Unity. Сформувати у студентів практичні навички в створенні простого ігрового середовища засобами движка Unity.

### 10.1.2 Лекція

Лекція знайомить студентів із основними інструментальними засобами створення сучасних ігрових додатків для різних апаратних та програмних платформ. Надається порівняльний аналіз особливостей, переваг і недоліків цих платформ. Проводиться огляд сучасних движків створення ігрових додатків, що дозволяють оптимізувати та прискорити процес розробки за допомогою ефективної реалізації модульної розробки додатку та ефективної підтримки кросплатформності.

Студенти ознайомлюються з основними об’єктами Unity та елементами інтерфейсу середовища. Дається класифікація стандартних об’єктів, наводиться опис їх основних властивостей. Розглядаються особливості використання матеріалів та текстур, створення власних префабів та матеріалів, експорт текстур та об’єктів. Розглядаються особливості використання ассетів.

### 10.1.3 Лабораторна робота №1. Створення простого ігрового середовища засобами Unity.

Лабораторна робота орієнтована на ознайомлення з середовищем Unity та отримання практичних навичок в створенні простого ігрового середовища засобами движка Unity.

Мета лабораторної роботи:

- Ознайомити студентів з основними елементами інтерфейсу движка Unity; пояснити призначення пунктів меню, панелей інструментів та робочих вікон;
- Навчити студентів створювати проект гри;
- Навчити студентів створювати об’єкти ігрового середовища та редагувати їх властивості;
- Навчити студентів створювати працювати зі стандартними ассетами, префабами, текстурами та матеріалами;
- Навчити студентів створювати власні матеріали та текстури; вміння експортувати двовимірні та тривимірні об’єкти, створені в різних графічних редакторах.

Відповідно до наскрізного завдання студент створює сцену в проекті гри та визначає необхідні елементи для подальшого розвитку проекту в цілому та конкретної сцени відповідно до індивідуального завдання.

У разі успішного виконання лабораторної роботи студент буде вміти запускати додаток Unity, створювати новий проект гри Unity, зберігати та завантажувати його; створювати та редагувати нові сцени гри в Unity; інтегрувати створені сцени в існуючий ігровий додаток; буде знати та вміти застосовувати засоби Unity для створення ігрового середовища.

Успішне засвоєння матеріалів та виконання лабораторної роботи сприяє формування у студента наступних соціальних навичок та вмінь:

- вміння реалізувати свою діяльність творчо, проявляти пізнавальну активність;
- вміння, необхідні для виконання групової та командної роботи;
- вміння критично ставитися до результатів групової та особистої роботи;
- вміння проводити самооцінку та групове оцінювання виконаних робіт
- вміння дотримуватися регламентних рамок, оцінювати строки проведення та виконання роботи,

- вміння обґрунтувати той чи інший спосіб діяльності при виконанні практичних завдань;
- комунікаційна ефективність;

вміння застосовувати раніше отримані знання як основу для засвоєння нових знань.

### 10.1.4 **Методичні матеріали та вказівки**

Методичні матеріали та вказівки доступні за посиланням http://89.185.3.253:9090/login , log: **guest**, pass: **gamehub ,** директорія **02_M_Game Applications Development Based on Unity Engine**.

Опорний конспект лекції № 1 дивись

02_M_Lec_1_Part_1_Game Applications Development Based on Unity Engine.pdf

Методичні вказівки щодо виконання лабораторної роботи дивись

02_M_Lab_ Game Applications Development Based on Unity Engine.pdf.

### 10.1.5 **Питання, що виносяться на іспит.**

- Охарактеризуйте призначення основних елементів інтерфейсу середовища Unity.
- Надайте класифікацію стандартних об’єктів Unity, охарактеризуйте їх властивості.
- Дайте визначення префаба та опишіть особливості його використання.
- Дайте визначення ассета та опишіть особливості його використання.
- Охарактеризуйте можливості Unity, пов’язані із імпортом зовнішніх об’єктів..

## 10.2 **Тема 2. Організація геймплея за допомогою скриптів в Unity. Динамічне створення об’єктів.**

### 10.2.1 **Мета та очікувані результати**

Ознайомити студентів з основами організації геймплея за допомогою скриптів.

Сформувати у студентів вміння, навички та основні прийоми роботи з об’єктами та подіями Unity із використанням скриптів C#.

Сформувати практичні навички динамічного створення об’єктів в процесі гри.

### 10.2.2 Лекція

Лекція знайомить з основами створення скриптів в Unity. Розглядаються базові класи та методи об’єктів Unity, представлені мовою C#. Розглядаються засоби прив’язки скриптів до об’єктів та подій ігрового середовища. Розглядаються можливості Unity з динамічного створення об’єктів в ході гри.

Мета лекції:

- Сформувати у студентів знання щодо знання створення, редагування та компіляції скриптів, їх прив’язки до об’єктів та подій сцени;
- Сформувати у студентів знання основних класів та методів мови C# для роботи з об’єктами Unity;
- Сформувати знання про особливості роботи з об’єктами, що динамічно створюються в процесі гри.

Основні результати лекції відповідають вище передбаченим цілям.

### 10.2.3 Лабораторна робота № 2. Використання скриптів Unity для створення геймплея.

Лабораторна робота орієнтована на отримання студентами практичних навичок роботи з об’єктами та подіями Unity із використанням скриптів C#.

Мета лабораторної роботи:

- Навчити студентів створювати, редагувати та компілювати скрипти засобами вбудованого редактора та із застосуванням середовища розробки Visual Studio;
- Навчити студентів використовувати основні класи та методи C# для роботи з об’єктами та подіями Unity;
- Навчити студентів прив’язувати скрипти до об’єктів та подій ігрового середовища.

Відповідно до наскрізного завдання студент визначає можливі події з об’єктами гри та створює необхідні скрипти для об’єктів та подій ігрового середовища, створених в рамках сцени ігрового додатку.

У разі успішного виконання лабораторної роботи студент буде вміти створювати, редагувати та компілювати скрипти, вміти використовувати основні класи та методи C# для роботи з об’єктами та подіями Unity, а також прив’язувати скрипти до об’єктів та подій ігрового середовища.

### 10.2.4 **Методичні матеріали та вказівки**

Методичні матеріали та вказівки доступні за посиланням http://89.185.3.253:9090/login , log: **guest**, pass: **gamehub ,** директорія **02_M_Game Applications Development Based on Unity Engine**.

Опорний конспект лекції № 2 дивись

02_M_Lec_2_Part_1_Game Applications Development Based on Unity Engine.pdf

Методичні вказівки щодо виконання лабораторної роботи дивись

02_M_Lab_ Game Applications Development Based on Unity Engine.pdf.

### 10.2.5 **Питання, що виносяться на іспит.**

- Надайте перелік та поясніть призначення основних операцій при створенні та редагування скрипта;
- Охарактеризуйте особливості прив’язки скриптів до об’єктів та подій ігрового середовища

## 10.3 **Тема 3: Створення інтерактивного геймплея з персонажем за допомогою засобів движка Unity. Організація користувальницького інтерфейсу засобами Unity.**

### 10.3.1 **Мета та очікувані результати**

Ознайомити студентів з особливостями створення інтерактивного геймплея з персонажем в середовищі Unity. Надати класифікацію персонажів та навести їх ключові особливості. Пояснити поняття тригеру та особливості використання тригерів при вирішенні задачі створення інтерактивного геймплею.

Сформувати практичні навички створення інтерактивного геймплея.

Сформувати практичні навички в створенні користувальницького інтерфейсу засобами движка Unity.

### 10.3.2 Л**екція**

Лекція знайомить з засобами Unity, що використовуються для створення інтерактивного геймплея з персонажем. Наводиться огляд вбудованих персонажів Unity та розглядаються засоби створення власних персонажів та керування ними в процесі гри. Дається поняття тригера та розглядаються особливості використання тригерів.

Розглядаються засоби Unity, що використовуються для створення користувальницького інтерфейсу (UI). Наводиться огляд засобів організації UI в Unity. Розглядаються особливості організації користувальницького меню.

Мета лекції:

- Ознайомити студентів з поняттям ігрового персонажу та їх класифікацією в Unity;
- Ознайомити студентів з поняттям тригеру та особливостями використання тригерів;
- Ознайомити студентів з поняттям користувальницького інтерфейсу та основними засобами його створення в Unity;
- Ознайомити студентів із засобами створення UI.

Основні результати лекції відповідають вище передбаченим цілям..

### 10.3.3 Лабораторна робота № 3. Створення інтерактивного геймплея з персонажем за допомогою засобів движка Unity. Організація взаємодії користувача.

Лабораторна робота орієнтована на отримання практичних навичок в створенні інтерактивного геймплея з персонажем засобами движка Unity, а також на отримання практичних навичок в створенні та адаптації користувальницького інтерфейсу.

Мета лабораторної роботи:

- Навчити студентів створити ігровий персонаж та організувати керування ним з клавіатури;
- Навчити студентів динамічно створювати об’єкти ігрового середовища безпосередньо в процесі гри;
- Навчити студентів використовувати тригери для організації взаємодії об’єктів ігрового середовища;
- Навчити студентів створювати елементи інтерфейсу користувача та прив’язувати до них необхідні події;
- Сформувати вміння змінювати позиціонування елементів інтерфейсу та інших властивостей елементів інтерфейсу;
- Навчити студентів створювати переходи між різними екранами меню та сценами гри;
- Навчити студентів робити збірку різних сцен в один проект.

Відповідно до наскрізного завдання студенти в результаті обговорення в групі визначають характеристики ігрового персонажу

(персонажів), способи його взаємодії з об’єктами ігрового середовища, та створюють ігровий персонаж для сцени ігрового додатку.

У разі успішного виконання лабораторної роботи студент буде вміти створювати ігровий персонаж та інші об’єкти і організовувати їх взаємодію з іншими статичними та динамічними об’єктами ігрового середовища; вміти створювати гру з інтерфейсом меню, забезпечувати переходи між різними екранами меню та сценами гри.

Успішне засвоєння матеріалів та виконання лабораторної роботи сприяє формування у студента наступних соціальних навичок та вмінь:

- ініціативність;
- вміння, необхідні для виконання групової та командної роботи;
- вміння дотримуватися регламентних рамок, оцінювати строки проведення та виконання роботи,
- вміння обґрунтувати той чи інший спосіб діяльності при виконанні практичних завдань;
- комунікаційна ефективність;
- вміння застосовувати раніше отримані знання як основу для засвоєння нових знань.

### 10.3.4 **Методичні матеріали та вказівки**

Методичні матеріали та вказівки доступні за посиланням http://89.185.3.253:9090/login , log: **guest**, pass: **gamehub ,** директорія **02_M_Game Applications Development Based on Unity Engine**.

Опорний конспект лекції № 3 дивись

02_M_Lec_3_Part_1_Game Applications Development Based on Unity Engine.pdf

Методичні вказівки щодо виконання лабораторної роботи дивись

02_M_Lab_ Game Applications Development Based on Unity Engine.pdf.

### 10.3.5 **Питання, що виносяться на іспит.**

- Надайте перелік вбудованих персонажів Unity та наведіть їх характеристики;
- Визначте поняття тригеру та опишіть особливості його використання при створенні інтерактивного геймплею.

- Опишіть послідовність дій для створення динамічних об’єктів безпосередньо в процесі гри.
- Охарактеризуйте засоби створення користувальницького інтерфейсу гри в Unity.
- Охарактеризуйте базові прийоми організації доступу до елементів інтерфейсу гри в Unity

## 10.4 Тема 4: Збірка та налаштування готового проекту.

### 10.4.1 Мета та очікувані результати

Сформувати у студентів знання про особливості створення готової гри з урахуванням програмного та апаратного оточення.

Сформувати практичні навички в адаптації користувальницького інтерфейсу засобами движка Unity відповідно до форм-фактору пристрою та використаної платформи.

### 10.4.2 Лекція

Лекція знайомить з особливостями позиціонування елементів інтерфейсу та адаптації інтерфейсу до пристроїв із різним форм-фактором. Надається огляд засобів Unity для адаптації та збірки гри під різне програмне та апаратне оточення.

Мета лекції:

- Ознайомити студентів із засобами адаптації UI для різних пристроїв та операційних систем;
- Сформувати поняття про засоби створення збірок ігор на базі одного проекту для різних умов функціонування (програмне та апаратне оточення).

Основні результати лекції відповідають вище передбаченим цілям.

### 10.4.3 Заключний практичний семінар. Демонстрація та обговорення ігрового додатку, створеного у середовищі Unity:

Заключний практичний семінар орієнтовано на обговорення створеного студентами при виконанні лабораторних робіт 1-3 прототипу закінченого 3D ігрового додатку у середовищі Unity, який зібрано із сцен, створених студентами відповідно до індивідуального варіанту завдання.

Мета практичного семінару:

- Оцінити правильність визначення основних елементів сцен ігрового додатку та їх реалізації у середовищі Unity.
- Оцінити правильність визначення основних подій та дій об’єктів (ігрової механіки) та їх реалізації у середовищі Unity.
- Оцінити працездатність сцен ігрового додатку в різних збірках.
- Оцінити показники інтерфейсу сцен гри (юзабіліті, відповідність евристичним показникам оцінки інтерфейсів та коректність відображення елементів і їх роботи в різних збірках).
- Надати рекомендації щодо подальшого розвитку розробленого ігрового додатку.

За результатами обговорення на практичному семінарі студенти отримають навички оцінки ігрових додатків, створених у середовищі Unity, навчаться виявляти недоліки та помилки, що були допущені на етапах розробки елементів сцен, інтерфейсу, ігрової механіки додатку, визначати подальші шаги розвитку додатку з урахуванням результатів обговорення, отримають навички формування зауважень щодо функціональних та нефункціональних вимог до гри, що сприяли б підвищенню якості геймплея та інтересу гравців.

Успішне засвоєння матеріалів практичного семінару сприяє формуванню у студента наступних соціальних навичок та вмінь: критично ставитися до результатів особистої роботи, вміти проводити самооцінку виконаних робіт, обґрунтувати той чи інший спосіб діяльності при виконанні практичних завдань, презентувати та аргументувати свої рішення, обговорювати в групі результати роботи. Форми прояву: приймати участь у груповому обговоренні, оцінюванні результатів виконаної робіти, вміння самостійно формулювати та доносити до співрозмовника свої думки, вміння ясно і конкретно висловлювати думки, слухати та розуміти співрозмовника, використовувати прийоми ділового спілкування (публічного мовлення, презентаційних виступів, доповідей) та раніше отримані знання як основу для засвоєння нових знань, дотримуватись етичних норм поведінки, ділового та партнерського спілкування.

### 10.4.4 **Методичні матеріали та вказівки**

Методичні матеріали та вказівки доступні за посиланням http://89.185.3.253:9090/login , log: **guest**, pass: **gamehub ,** директорія **02_M_Game Applications Development Based on Unity Engine**.

Опорний конспект лекції № 4 дивись

02_M_Lec_3_Part_2_Game Applications Development Based on Unity Engine.pdf

Методичні вказівки щодо виконання лабораторної роботи дивись

02_M_Lab_ Game Applications Development Based on Unity Engine.pdf.

### 10.4.5 **Питання, що виносяться на іспит.**

- Опишіть основні показники, які враховуються при адаптації гри в залежності від умов функціонування.
- Вкажіть основні елементи ігрового додатку, які потребують адаптації при зміні форм-фактору пристрою. Наведіть приклади.
- Наведіть перелік засобів адаптації гри в залежності від умов функціонування (оточення гри).

## 11 Рекомендована література (інтернет посилання)

1 Sue Blackman. Beginning 3D Game Development with Unity: The World’s Most Widely Used Multiplatform Game Engine, 2011. – 986 p.

2 Steven Goodwin. Polished Game Development: From First Steps to Final Release , 2016. – 269 p.

3 Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.

4 Офіційний сайт Unity3d https://unity3d.com/ru

5 Сайт розробників ігор http://gamesmaker.ru/3d-game-engines/unity3d/

# МЕТОДИЧНІ ВКАЗІВКИ
# до ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

**Дисципліна: Інструментальна підтримка розробки комп'ютерних ігрових додатків**

**Модуль: Розробка ігрових додатків на базі движка Unity**

# 1 Вступ. Загальні положення

Комплекс лабораторних робот модуля «Розробка ігрових додатків на базі движка Unity» виконується відповідно [1], передбачає послідовне (step by step) виконання наскрізного завдання та має на меті відпрацювання навичок і вмінь щодо розробки типового ігрового додатку з реалізацією типових ігрових елементів у відповідності до платформи середовища Unity [3]. До комплексу входить заключний практичний семінар – заняття, на якому студент презентує ігровий додаток, що був розроблений згідно наскрізного завдання. Інші студенти групи оцінюють результат, виступаючи у ролі опонентів та перших гравців. За результатами обговорення на семінарі сформулюються наступні зміни, що дозволять грі стати більш функціональною та привабливою для широкої аудиторії.

Заліковий звіт – надання докладного письмового звіту та презентація в групі ігрового додатка, розробленого в програмному середовищі Unity, при послідовному виконанні наскрізного завдання лабораторних робіт 1-3.

Самостійна робота студентів передбачає вивчення додаткової літератури [2, 4], підготовку до лабораторних робіт, а також підготовку та оформлення залікового звіту та презентації за результатами виконання лабораторних робіт.

## 1.1 Графік виконання лабораторних робіт

| **Номер тижня** | **Виконання** | **Оцінювання (максимальний бал)** |
|---|---|---|
| 1 | Лабораторна робота №1 | 4 |
| 2-3 | Лабораторна робота №2 | 4 |
| 4-5 | Лабораторна робота №3 | 4 |
| 6 | Презентація та обговорення отриманих результатів на заключному практичному семінарі | 4 |
| | Заліковий звіт | 2 |
| | Підсумкове тестування по модулю | 2 |
| Максимальний сумарний бал | | 20 |

## 1.2 Оцінка виконання комплексу лабораторних робіт.

Критерії оцінювання виконання лабораторної роботи наведені в додатку А.

Критерії оцінювання обговорення результатів на заключному практичному семінарі наведені в додатку Б.

Критерії оцінювання заключного звіту наведені в додатку В.

Інформація щодо оцінки змістовного модуля в цілому надається студентам на 13 тижні навчання.

## 1.3 Контактні дані для on-line допомоги та консультування:

Викладачі :

- асистент Скрипник Т. В., tetiana.skrypnyk@donntu.edu.ua
- асистент Тихонова О.А., oksana.tykhonova@donntu.edu.ua.

# 2 Наскрізне завдання

Група слухачів розбивається на 3 - 5 команд (по 3 - 5 осіб). Кожна команда повинна створити новий проект.

Визначитися зі структурою проекту.

Кожна команда повинна:

1. Розробити ескіз основного ігрового об'єкта.
2. Спланувати властивості і можливості основного ігрового об'єкта.
3. Визначити фон простору, де буде відбуватися дія гри.
4. Розробити ескіз додаткового ігрового об'єкта.
5. Спланувати властивості і можливості додаткового ігрового об'єкта.
6. Продумати поведінку додаткових ігрових об'єктів за умови, що їх буде кілька.
7. Проаналізувати, що розвиває гра у гравця, які виробляє навички.

Визначити цільову аудиторію гри.

# 3 Обладнання

Для виконання лабораторних робіт необхідно використовувати персональний комп’ютер із встановленою операційною системою Windows та програмою Unity .

Мінімальні вимоги до технічних характеристик персонального комп’ютера:

- процесор Intel Core i5-2500K з частотою 3,3 ГГц або AMD FX-8350 з частотою 4,0 ГГц.
- 6 Гбайт оперативної пам'яті,
- 50 Гбайт вільного простору на жорсткому диску,
- відеокарта NVIDIA GeForce GTX 680 або AMD Radeon HD 7970 з 2 Гбайт відеопам'яті,
- звукова карта з підтримкою DirectX 9.0c
- Windows-сумісна клавіатура і миша або геймпад
- інтернет з’єднання.

Вимоги до операційної системи

- Microsoft Windows 7 (Service Pack 1) або вище,
  - 32-bit або 64-bit версія.

# 4 Попередні підготовчі кроки до виконання лабораторних робіт.

Для виконання лабораторних робіт необхідно встановити Unity 5 або версію вище.

.

## 4.1 Установка Unity

Перш за все, необхідно перейти сайт розробника https://unity3d.com/ru/.

Завантажити дистрибутив Unity (веб установник), для цього на головній сторінці сайту необхідно натиснути на кнопку " ОТРИМАТИ Unity", рисунок 4.1.

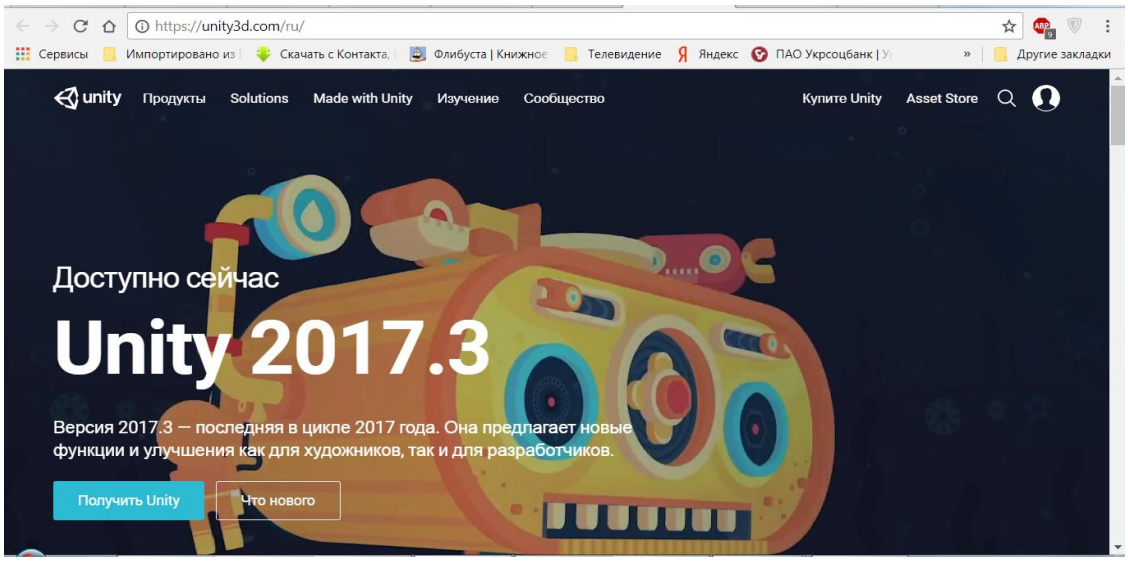



Рисунок 4.1

На наступному екрані перераховуються можливості безкоштовної та платної версій. Необхідно натиснути кнопку "БЕСПЛАТНАЯ ЗАГРУЗКА".

На наступній сторінці знаходитиметься кнопка ЗАГРУЗИТЬ УСТАНОВЩИК. Так само тут можна побачити версію останнього релізу, дату випуску, розмір установника і вибрати платформу Windows або MacOS.

Завантажити і запустити установник, рисунки 4.2-4.4:


Рисунок 4.2

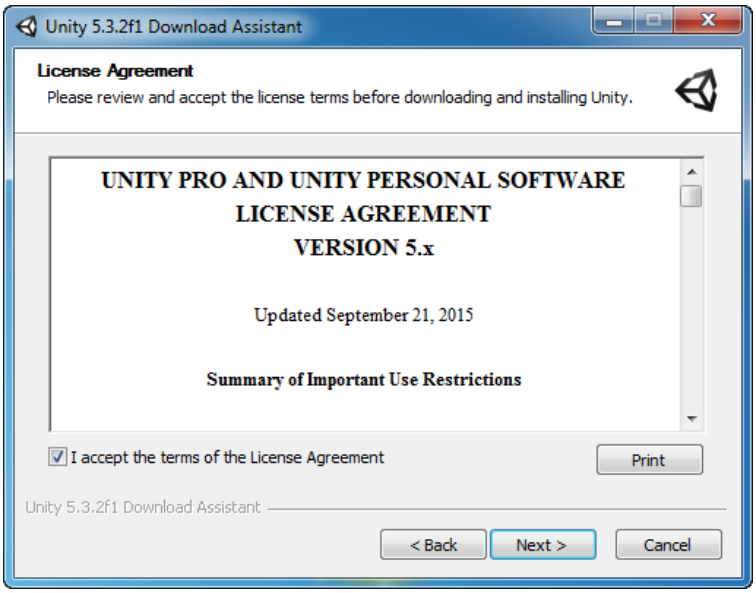



Рисунок 4.3

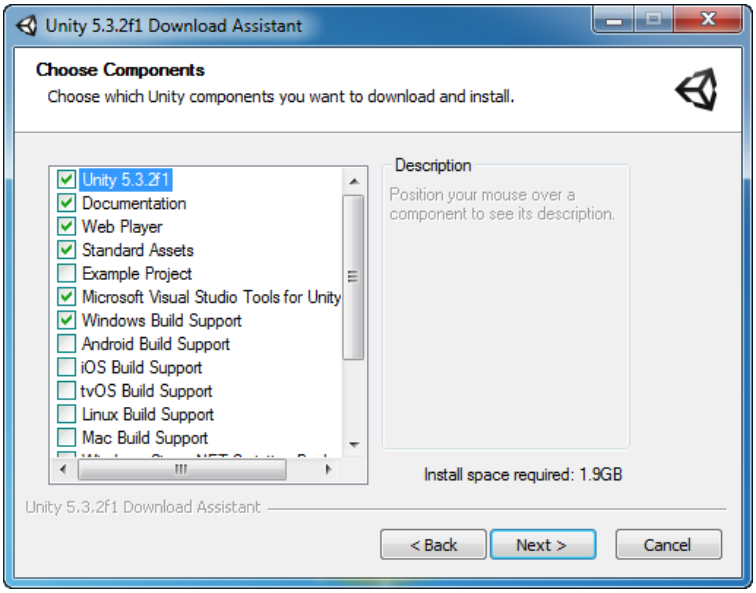



Рисунок 4.4

У вікні Choose Components можна вибрати пакети, які необхідно завантажити для зручної роботи в Unity. Деякі прапорці вже стоять:

**Unity 5.3.2** - це власне саме середовище розробки;

**Documentation** - тут знаходяться досить докладний матеріал по роботі з Unity;

**Web Player** - інструмент для створення браузерних додатків та ігор;

**Standart Assets** - так званий стандартний Ассет, в ньому знаходяться приклади використання скриптів, фізичних об'єктів, текстури, матеріали і багато іншого;

**Microsoft Visual Studio Tools for Unity** - вбудований редактор скриптів;

**Windows Build Support** - служить для компіляції проекту під ОС Windows. Якщо ігри будуть створюватися під інші операційні системи, слід поставити прапорці в установнику у відповідних пунктах.

Натиснути кнопку Next.

У наступному вікні вибрати місце установки програми і натиснути копку Next (рисунок 4.5)

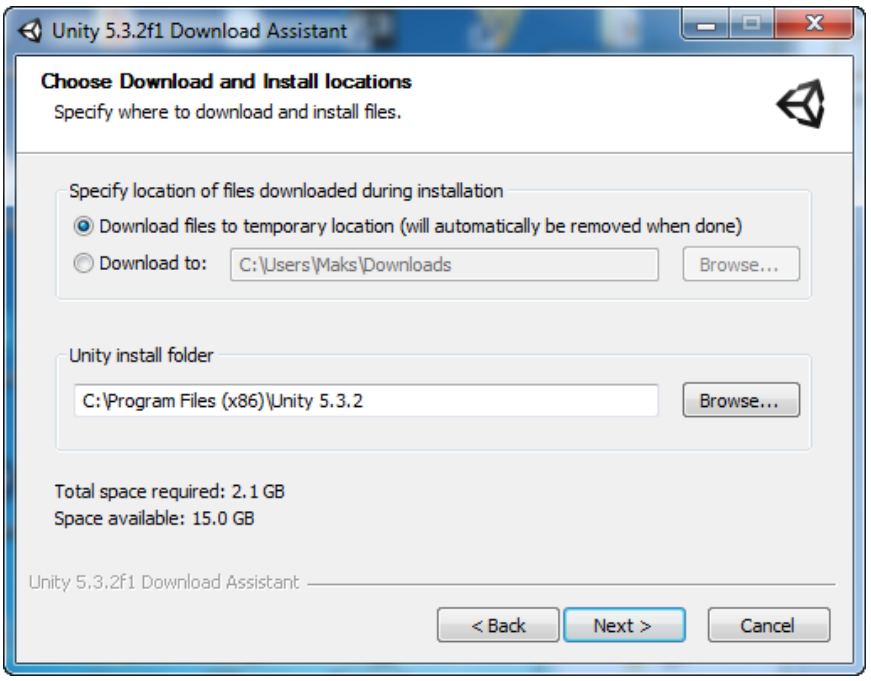



Рисунок 4.5

Після всіх пройдених етапів завантаження і установки, на робочому столі з'явиться ярлик Unity.

При першому запуску програми з'являється вікно з введенням логіна і пароля, якщо є обліковий запис створена на сайті Unity можна вводити свої дані, якщо немає, потрібно зареєструватися або працювати offline, натиснувши на кнопку work offline, рисунок 4.6.

Для створення нововго проекту необхідно натиснути на кнопку NEW. Ввести назву проекту і місце його розташування, так само вибрати 3D або 2D (за замовчуванням стоїть 3D) і натиснути Create project. Створений новий проект з'явиться у вікні Project, для його запуску досить натиснути на нього і Unity запуститься, рисунки 4.7 – 4.9.

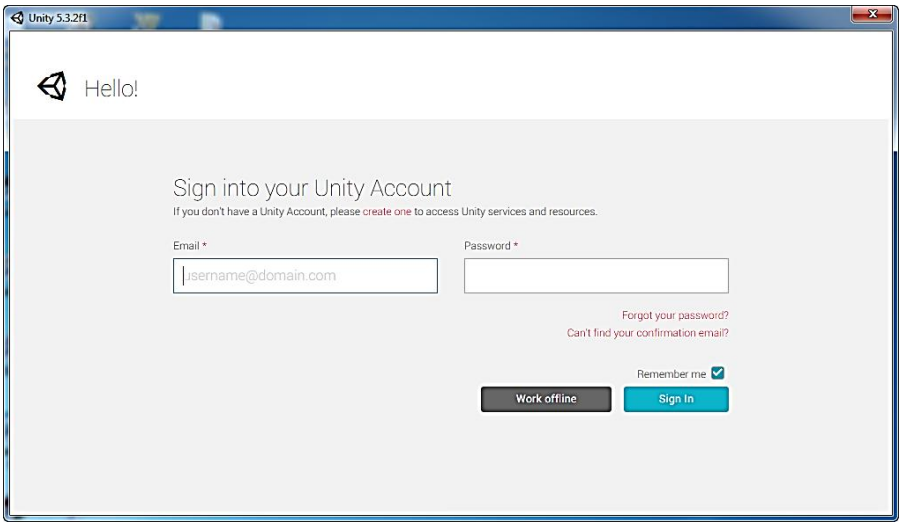



Рисунок 4.6

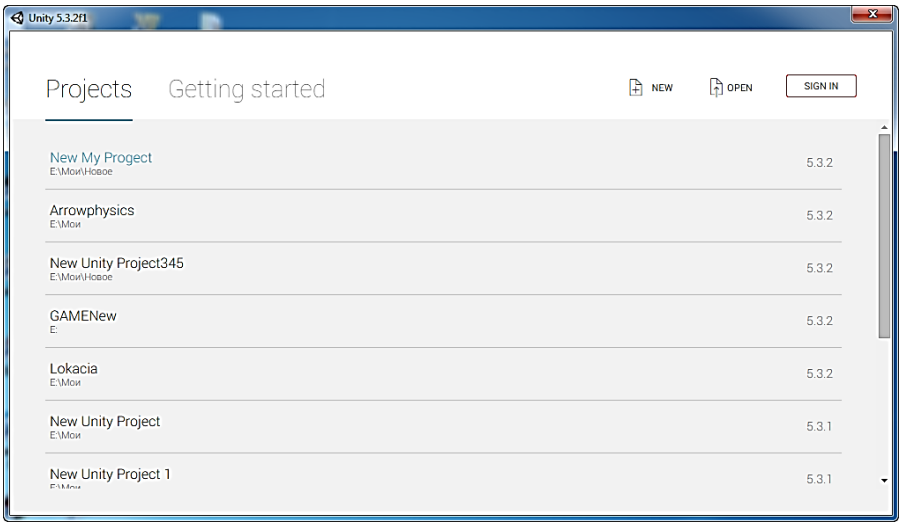



Рисунок 4.7

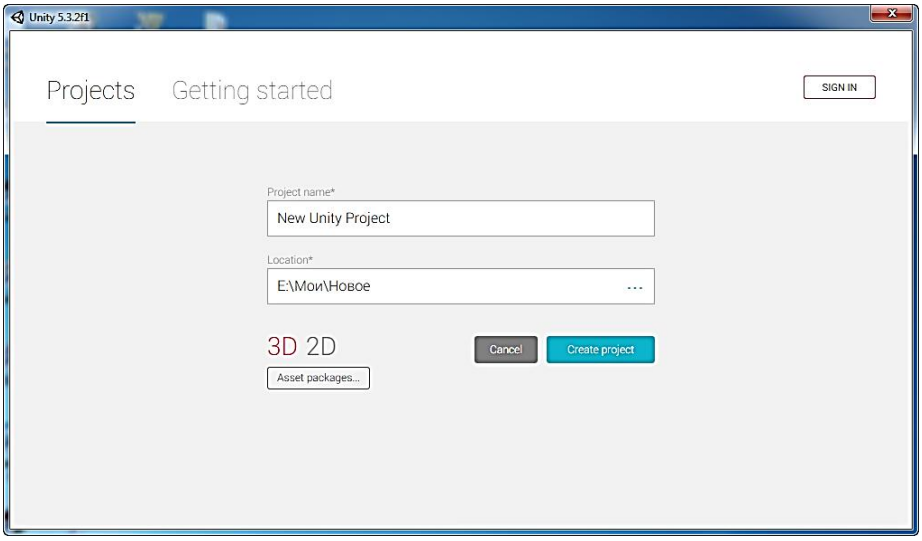



Рисунок 4.8

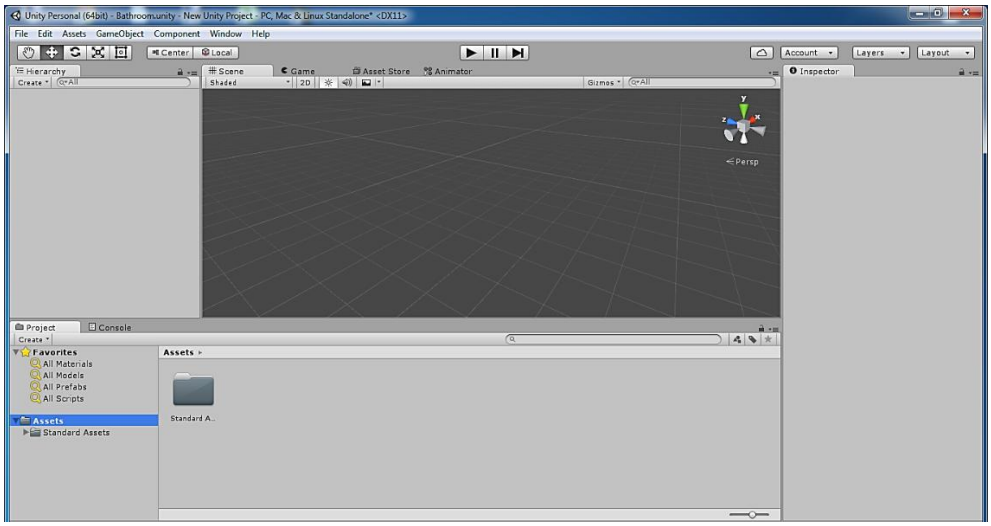


Рисунок 4.9.

## 4.2 Основні елементи інтерфейсу Unity

Головне вікно редактора складається з декількох вкладок, які називаються Видами (Views). У Unity є кілька типів видів - всі вони призначені для конкретних цілей. Це означає, що зовнішній вигляд редактора може відрізнятися і буде залежати від вподобань розробника і від проекту.

Як правило, робоче вікно Unity розбите на 6 взаємозв'язаних областей, рисунок 4.10:

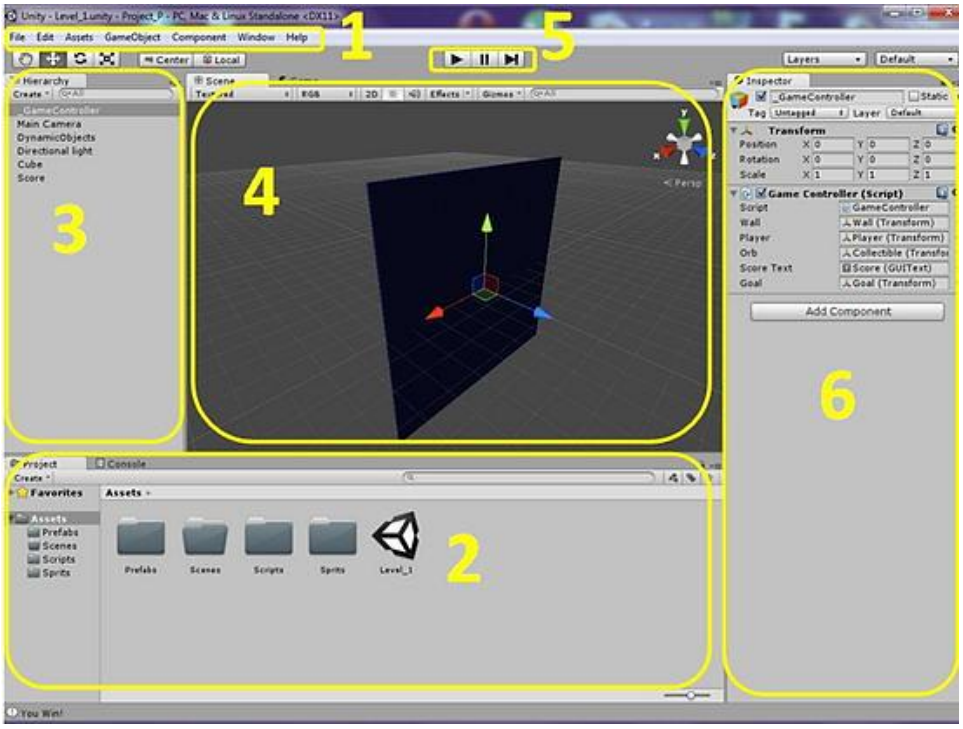



Рисунок 4.10 – робоче вікно Unity

**1. Main menu** (Головне меню) – Рядок тексту зверху, де розташовуються всі команди, доступні в програмі. Багато команд продубльовано кнопками і меню в робочих областях, тому головне меню не обов'язково використовувати.

**2. Project View** (Огляд проекту) – Список, в якому показані всі використовувані файли в грі: файл сцени, файл програмного коду, графічні і аудіофайли.

**3. Hierarchy** (Ієрархія) – Список, де перелічено всі об'єкти, додані на сцену. Тут можна працювати з об'єктами, копіювати їх, перейменовувати, видаляти.

**4. Scene** (Сцена) – Область, де відображається ігровий світ або ігрова сцена. Тут можна додавати нові об'єкти, перетягувати їх, змінювати вид.

5. **Game** (Гра) – Область попереднього перегляду, де видно, як сцена буде виглядати в грі. Тут можна налаштовувати різні настройки екрану і режиму відео.

6. **Inspector** (Інспектор) – Список, що складається з декількох різних по виду розділів. Показує всі властивості обраного об'єкта: розміри, моделі, текстури, скрипти.

# 5 Лабораторна робота №1. Створення простого ігрового середовища засобами Unity.

Лабораторна робота № 1 орієнтована на ознайомлення з середовищем Unity та отримання практичних навичок в створенні простого ігрового середовища засобами движка Unity.

Мета лабораторної роботи:

- Ознайомити студентів з основними елементами інтерфейсу движка Unity; пояснити призначення пунктів меню, панелей інструментів та робочих вікон;
- Навчити студентів створювати проект гри;
- Навчити студентів створювати об’єкти ігрового середовища та редагувати їх властивості;
- Навчити студентів створювати працювати зі стандартними ассетами, префабами, текстурами та матеріалами;

- Навчити студентів створювати власні матеріали та текстури; вміння експортувати двовимірні та тривимірні об’єкти, створені в різних графічних редакторах.

Відповідно до наскрізного завдання студент створює сцену в проекті гри та визначає необхідні елементи для подальшого розвитку проекту в цілому та конкретної сцени відповідно до індивідуального завдання.

У разі успішного виконання лабораторної роботи студент буде вміти запускати додаток Unity, створювати новий проект гри Unity, зберігати та завантажувати його; створювати та редагувати нові сцени гри в Unity; інтегрувати створені сцени в існуючий ігровий додаток; буде знати та вміти застосовувати засоби Unity для створення ігрового середовища.

Успішне засвоєння матеріалів та виконання лабораторної роботи сприяє формування у студента наступних соціальних навичок та вмінь:

- вміння реалізувати свою діяльність творчо, проявляти пізнавальну активність;
- вміння, необхідні для виконання групової та командної роботи;
- вміння критично ставитися до результатів групової та особистої роботи;
- вміння проводити самооцінку та групове оцінювання виконаних робіт
- вміння дотримуватися регламентних рамок, оцінювати строки проведення та виконання роботи,
- вміння обґрунтувати той чи інший спосіб діяльності при виконанні практичних завдань;
- комунікаційна ефективність;
- вміння застосовувати раніше отримані знання як основу для засвоєння нових знань.

## 5.1 Завдання до лабораторної роботи

- Створити основний ігровий об'єкт.
- Створити фон.
- Створити ігрове середовище.

## 5.2 Підготовка до лабораторної роботи № 1

Підготовка до виконання лабораторної робити № 1 включає

- Встановити Unity (дивись 4.1).
- Ознайомитись з інтерфейсом Unity (дивись 4.2).

- Ознайомитись з методичними вказівками до виконання лабораторної роботи (дивись 5.2)
- Сформувати папки для зберігання зразків ігрових об'єктів, фону, текстури.
- Розробити в будь-якому графічному редакторі макети ігрових об'єктів.

## 5.3 Контрольні питання для допуску до лабораторної роботи № 1

- Охарактеризуйте призначення основних елементів інтерфейсу середовища Unity.
- Охарактеризуйте можливості Unity, пов’язані із імпортом зовнішніх об’єктів.
- Для чого створюються папки в проекті?
- Як ввести в проект об'єкт створений в іншому графічному додатку?
- Для чого використаються шари в проекті?
- Чи можна файл із зображенням фону відразу помістити в 3D-об'єкт?

## 5.4 Методичні вказівки до виконання лабораторної роботи № 1

### 5.4.1 Створення проекту

1. Завантажити Unity. У головному меню вибирати «**File -- New Project**». У вікні вказати місце на жорсткому диску для збереження проекту.

У нижній частині вікна вибирати тип гри "двомірна" «Setup defaults for: 2D».

Після цього натиснути кнопку «Create». (рис. 5.1 ).

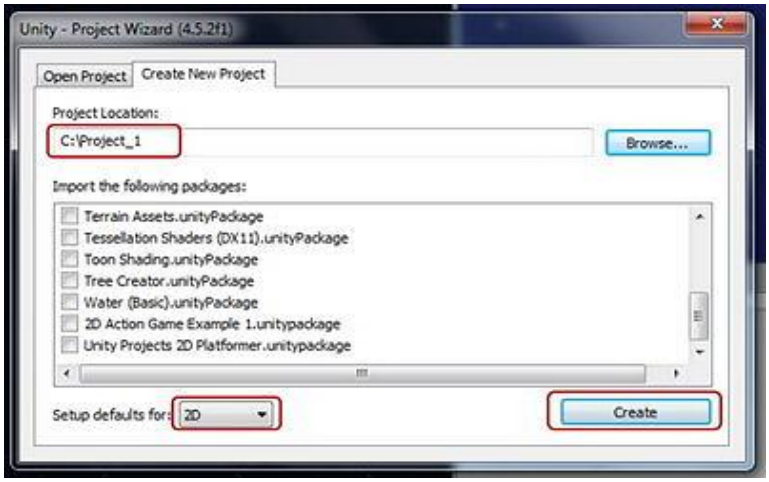



Рисунок 5.1 Створення проекту.

**Для командної роботи необхідно дотримуватися всі стандарти розробки проекту: чітка ієрархія, осмислені назви змінних і коментарі в програмному коді (на майбутнє).**

2. Створити чітку ієрархію проекту - кожному виду файлів створити свою папку:

1) У вікні «Project» (верхній лівий кут) натиснути кнопку «Create».

2) У списку вибрати «Folder». У кореневій папці «Assets» з'явиться нова папка.

3). Створити чотири папки:

Prefabs (масиви об'єктів),

Scripts (програмний код),

Sounds (для звукового супроводу ігрових дій),

Sprites (зображення для ігрових об'єктів).

Результат дій показано на рисунку 5.2:


Рисунок 5.2 - Ієрархія проекту.

### 5.4.2 Створення ігрового середовища:

#### *1.1.1.1 Створення Sprite (спрайт)*

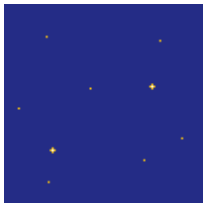
1. Запустити програму Paint. У властивостях зображення змінити розмір на «100 х 75». Намалювати корабель з найпростіших фігур,

зафарбувати, навколо корабля залишити білий колір. Зберегти файл під ім'ям «playerShip.png».

2. Створений файл відкрити в програмі «Gimp» (вільно поширюваний растровий графічний редактор). Зліва вибрати інструмент «чарівна паличка», на зображенні вибирати білий ділянку навколо корабля. У головному меню, на вкладці «Зображення» вибрати команду «Color to alfa-cannel». Після цього навколо корабля буде прозорий колір. Якщо на зображенні ще залишилися білі ділянки, повторити дії. Зберегти файл.

В аналогічний спосіб створити графічні файли прибульця (ворога), космосу і пострілу. У таблиці 5.1 показані можливі зображення гри:

Таблиця 5.1 Зображення для створення гри.

| Корабель **playerShip.png** | Космос **background.png** | Прибулець **enemy.png** | Постріл **laser.png** |
|---|---|---|---|
|  |  |  |  |

### *1.1.1.2 Створення ігрового об'єкта.*

1. Спосіб перший. В окремому вікні відкрити папку зі створеним графічним файлом. Перетягнути його у вікно Unity в зону «Project» на папку «Sprites». Файл підвантажиться в папку. Після цього виділити завантажений файл, у вікні «Inspector» в рядку «Texture Type», вибрати значення «Sprite (2D and UI)». Натиснути кнопку «Apply». (Див. рисунок 5.3)

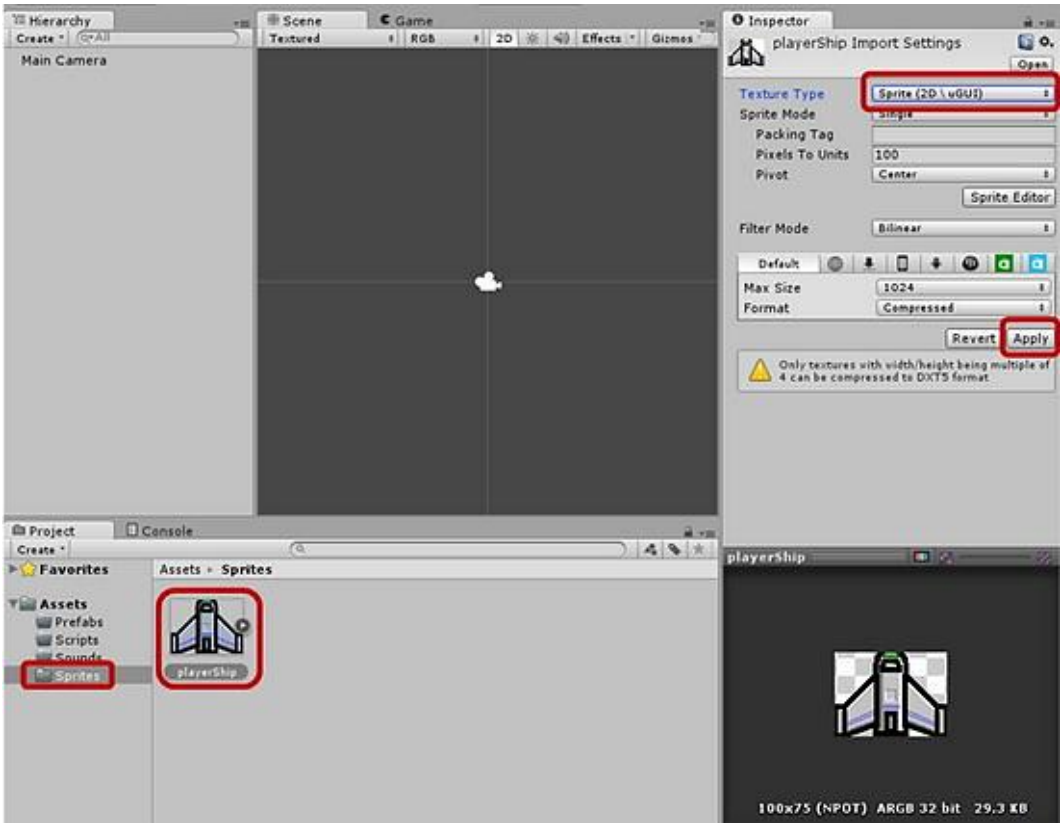



Рисунок 5.3 – Створення ігрового об'єкту.

Спосіб другий. У зоні «Project» вибрати папку «Sprites». Викликати для неї контекстне меню. У списку вибрати команду «Import New Asset». У вікні, що відкриється, указати місце розташування файлу зображення.

2. Вибрати завантажений файл в зоні «Project» і перемістити його в зону «Scene» (темно-сіра частина робочого вікна Unity, де є зображення відеокамери).

Можливо об'єкт встановився не рівно. У зоні «Inspector» (праворуч) є характеристики «Transform» - це координати об'єкта в ігровому просторі. Якщо в координатах X і Y вказані не цілі, а дробові числа, їх необхідно виправити. Для цього викликати контекстне меню для «Transform», вибрати команду «Reset Position». Після цього зображення встане рівно по центру світу (X = 0, Y = 0). Див. рисунок 5.4.

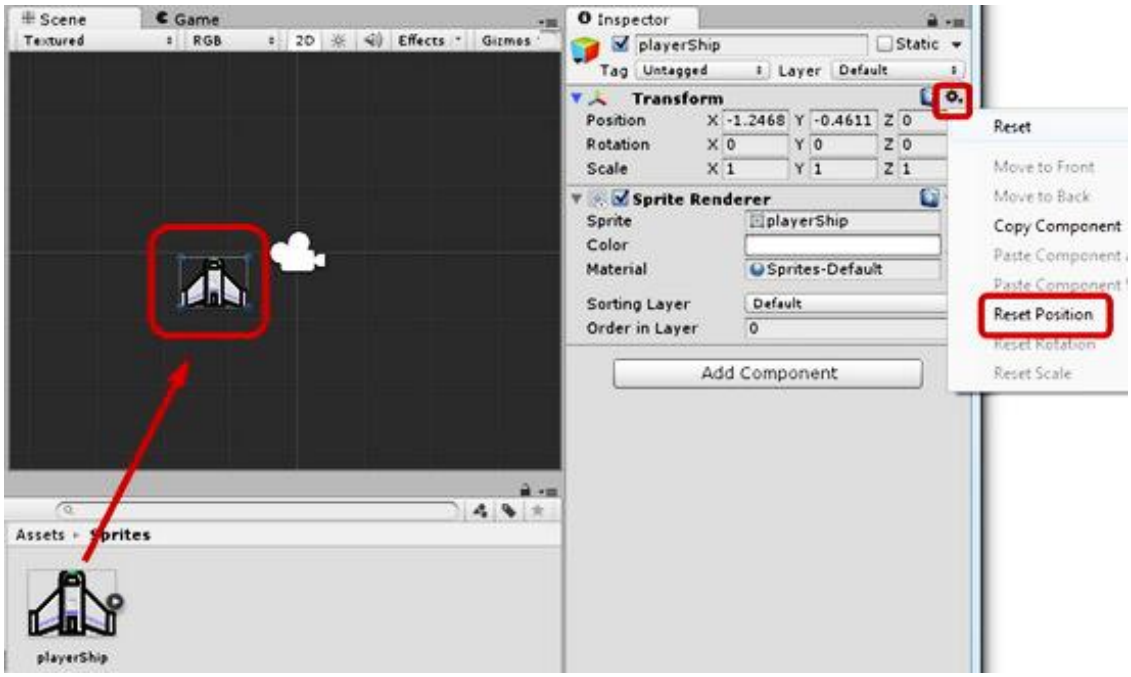



Рисунок 5.4 - Вирівнювання координат об'єкту

### *1.1.1.3 Створення фону.*

1. Намалювати зображення зоряного простору - темний фон і кілька зірок. Розмір зображення - 100 x 100, формат - .png.

2. Створений файл додати у вікно Unity в папку «Sprites».

3. Спосіб перший. Перетягнути файл у вікно ігрової сцени. Видно, що нове зображення встало поверх старого, і корабля тепер майже не помітно. У Unity влаштовано так, що новий елемент ставиться поверх старого. Однак фон повинен бути на задньому плані. Щоб виправити

черговість зображень, вибрати ігровий об'єкт з фоном. У вікні «Inspector» у властивостях «Transform» встановити значення «Z = 1». У зображення корабля має стояти властивість «Z = 0», де Z - це глибина розташування об'єкта в 2D іграх. Чим більше значення Z, тим далі об'єкт знаходиться від ігрової камери. Рисунок 5.5.

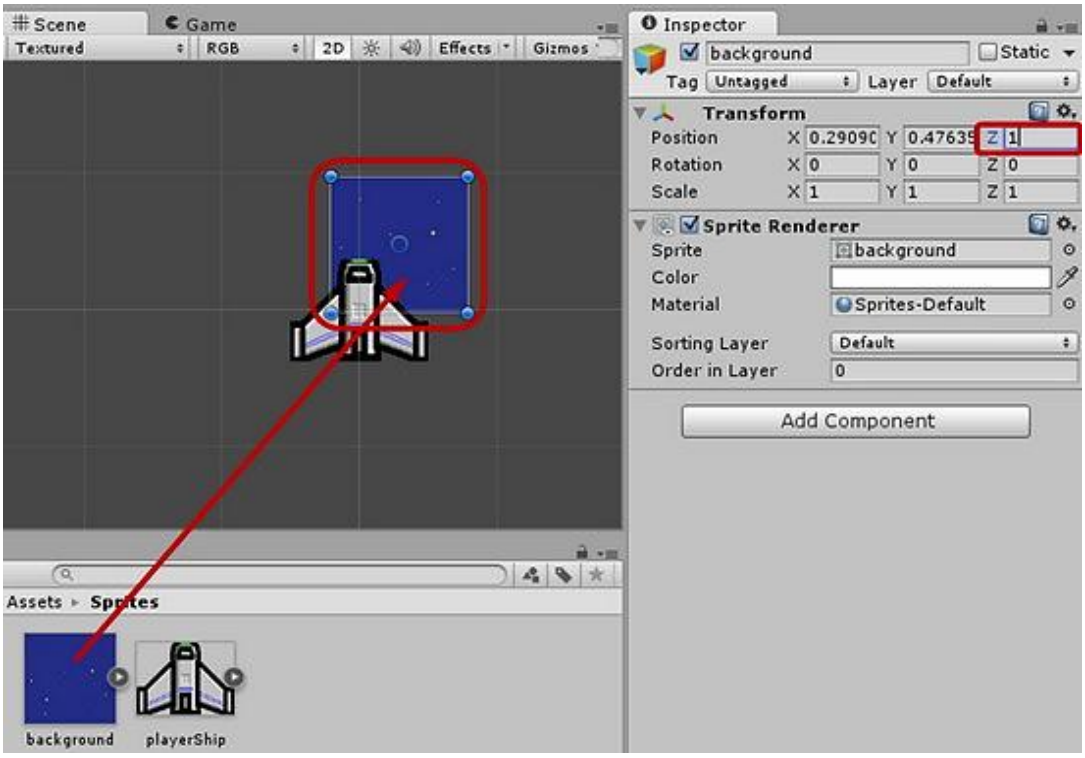



Рисунок 5.5 – Зміна черговості зображень

4. Спосіб другий. На сцені вибрати об'єкт фону, у вікні «Inspector» знайти властивості «Sprite Renderer», в рядку «Sorting Layer» натиснути кнопку і вибрати «Add Sorting Layer». Після цього з'явиться новий рядок з назвою шару «Layer 1». Перейменувати його в «Background». Так само створити ще два шари: «Foreground» і «GUI».

5. Вибрати об'єкт корабля, у вікні «Inspector» в рядку «Sorting Layer» виставити йому шар «Foreground». Вибрати об'єкт фону, виставити йому «Background». Тепер корабель буде відображатися поверх фону.

Фонове зображення прикладу зараз має розмір 100 х 100 пікселів. Для фону цього буде мало. Можна було б створити багато копій цього зображення, і закрити ними весь ігровий екран, але це дуже довгий і неефективний спосіб. Професійніше перетворити спрайт в текстуру.

6. Виділити об'єкт фону зі списку «Hierarchy» і видалити його (клавіша «delete»), щоб прибрати фон з ігрової сцени. У зоні «Project» вибрати зображення фону, у вікні «Inspector» замінити його тип «Texture Type» на «Texture». У рядку «Wrap Mode» встановити значення «Repeat». Натиснути кнопку «Apply». Див. рисунок 5.6.

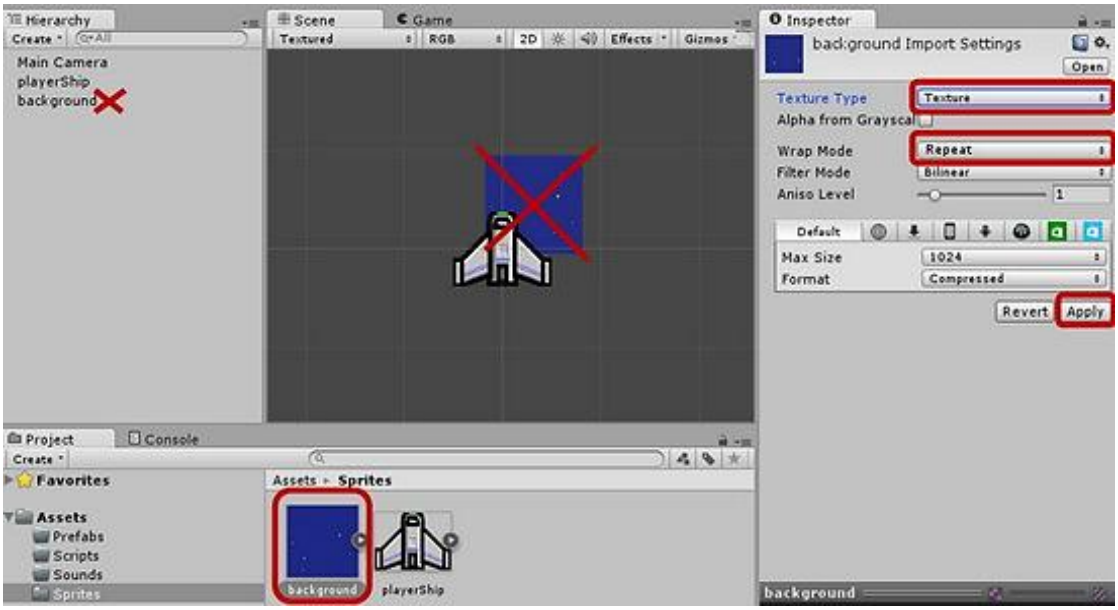



Рисунок 5.6 – Видалення об'єкта фону.

7. Є текстура, тепер необхідно створити для неї відповідний ігровий об'єкт. У головному меню вибрати рядок «Game Object | Create Other | Cube ». Змінити ім'я об'єкта, який з'явився, на «Background». У властивостях «Transform» змінити розташування «Position: 0, 0, 1» і розмір «Scale: 100, 100, 1».

Створився великий куб на задньому плані, який заслонив всю частину ігрового світу, що відображена на екрані монітора.

8. У властивостях куба видалити розділ «Box Collider» (обробник зіткнень). Для цього викликати контекстне меню на розділ. У меню вибрати команду «Remove Component». Див. рисунок 5.7.

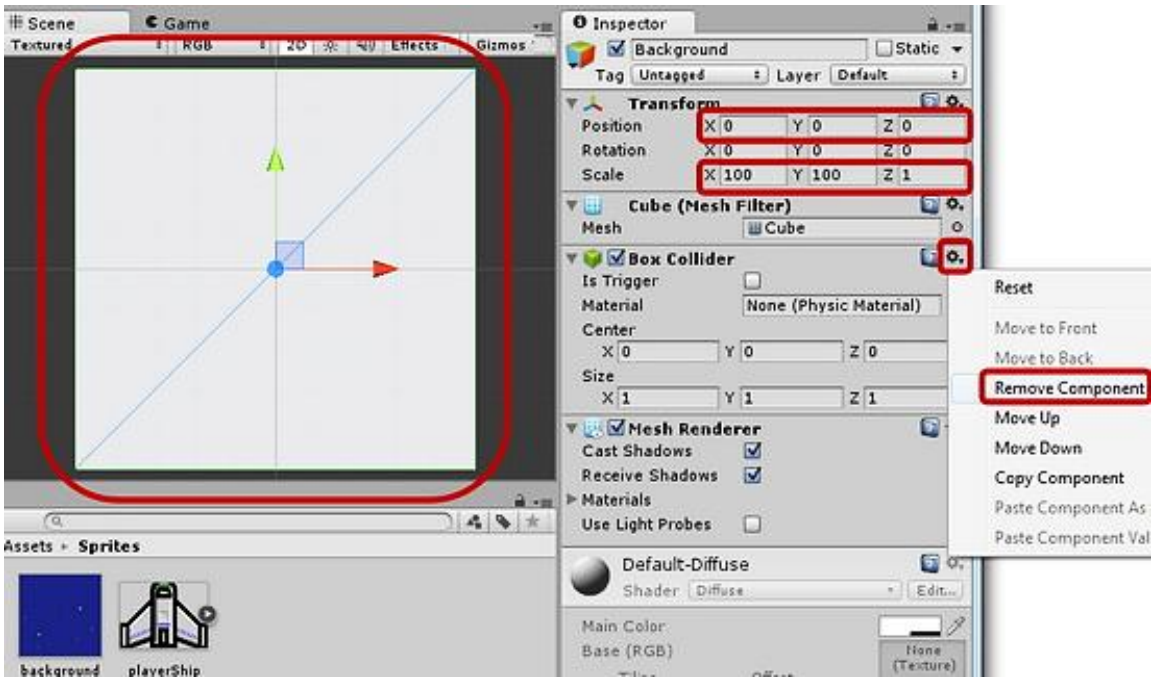



Рисунок 5.7 - Видалення розділу «Box Collider».

9. Зображення фону не можна безпосередньо помістити в 3D-об'єкт. Для початку зображення потрібно перетворити в матеріал. У зоні «Project» зверху натиснути «Create | Material ». Матеріал, який з'явився, назвати «BackgroundMaterial». У властивостях «Shader» натиснути на меню, що випадає, вибрати «Unlit | Texture ». У правій частині властивостей клікнути по квадрату з зображенням текстури «Texture box», із списку вибрати текстуру створеного фону (або ж можна просто перетягнути сюди файл текстури із зони «Project»). У властивості «Tiling» встановити значення x = 25, y = 25. Див. рисунок 5.8.

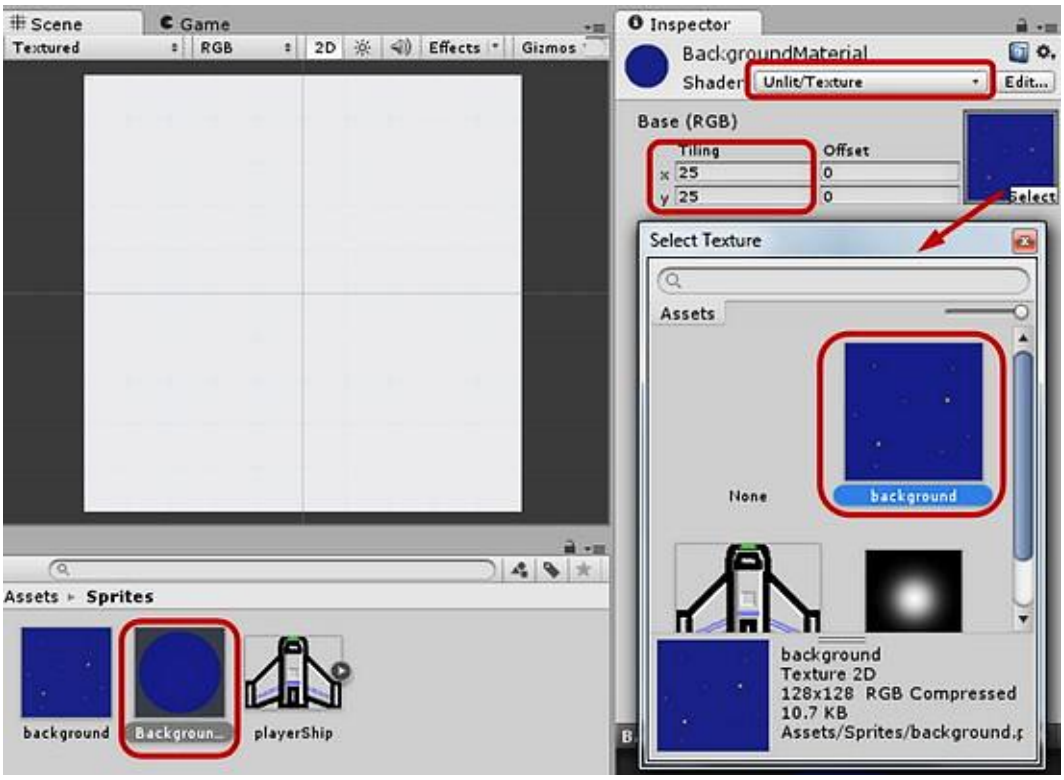



Рисунок 5.8 – Створення текстури.

10. У зоні «Hierarchy» вибрати об'єкт «Background». В його властивості, під компонентом «Mesh Renderer» відкрити «Materials», змінити значення «Element 0» на матеріал «BackgroundMaterial». Після цього на ігровій сцені з'явиться повноцінний фон. Рисунок 5.9.

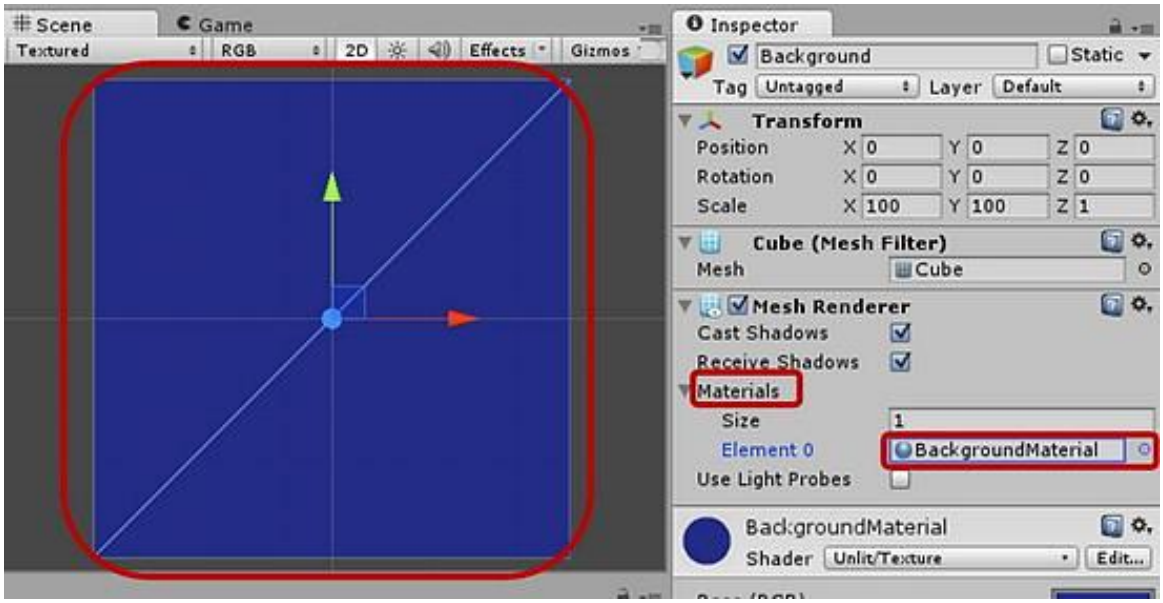



Рисунок 5.9 – Установка фону.

# 6 Лабораторна робота № 2. Використання скриптів Unity для створення геймплея.

Лабораторна робота орієнтована на отримання студентами практичних навичок роботи з об’єктами та подіями Unity із використанням скриптів C#.

Мета лабораторної роботи:

- Навчити студентів створювати, редагувати та компілювати скрипт засобами вбудованого редактора та із застосуванням середовища розробки Visual Studio;
- Навчити студентів використовувати основні класи та методи C# для роботи з об’єктами та подіями Unity;
- Навчити студентів прив’язувати скрипти до об’єктів та подій ігрового середовища.

Відповідно до наскрізного завдання студент визначає можливі події з об’єктами гри та створює необхідні скрипти для об’єктів та подій ігрового середовища, створених в рамках сцени ігрового додатку.

У разі успішного виконання лабораторної роботи студент буде вміти створювати, редагувати та компілювати скрипти, вміти використовувати основні класи та методи C# для роботи з об’єктами та подіями Unity, а також прив’язувати скрипти до об’єктів та подій ігрового середовища.

## 6.1 Завдання до лабораторної роботи № 2

Кожна команда повинна створити скрипт, за допомогою якого головний ігровий об'єкт зможе повертатися за покажчиком мишки і переміщуватися при натисканні на кнопки управління курсора на клавіатурі (стрілки).

## 6.2 Підготовка до лабораторної роботи № 2

При підготовці до виконання лабораторної роботи необхідно:

- Запустити графічний редактор Unity.
- Ознайомитись з методичними вказівками до виконання лабораторної роботи (дивись 6.4)
- Ознайомитись з додатковою програмою «MonoDevelop».

## 6.3 Контрольні питання і попередні матеріали для допуску до лабораторної роботи № 2

### 6.3.1 Контрольні питання

- Для чого використовуються скрипти в Unity?
- Як відкрити вікно для написання скриптів?
- З чого складається стандартний шаблон скрипта?
- Як прив'язати скрипт до об'єкта?

### 6.3.2 Попередні матеріали

Для допуску к виконанню лабораторної роботи необхідно представити:

- Проект в якому створено ігрове середовище ;
- Продумати, згідно варіанту, які дії повинен виконувати головний ігровий об'єкт.

## 6.4 Методичні вказівки до виконання лабораторної роботи № 2

### 6.4.1 Реалізація пересувань гравця

1. В області «Проекту» викликати контекстне меню на папці «Scripts», в якому вибрати команду «Create» - «C # Script». (Або ж можна натиснути на кнопку «Create» у верхньому лівому куті області, і створити скрипт там). Назвати створений файл - «PlayerScript» (рисунок 6.1).

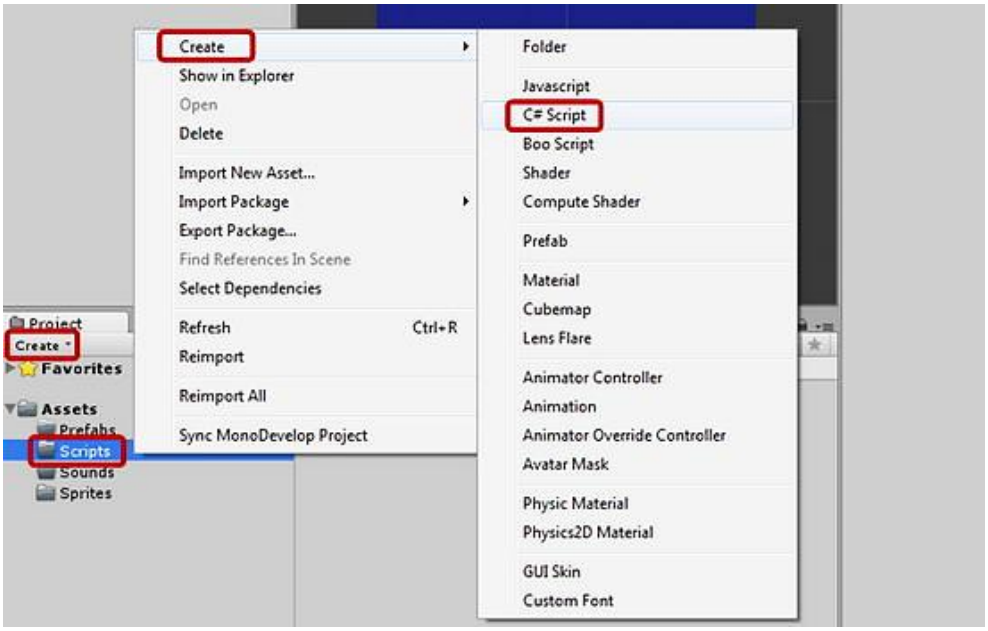



Рисунок 6.1

2. Виконати подвійний клік по файлу скрипта, після цього відкриється вікно додаткової програми «MonoDevelop» (це програма з комплекту Unity, призначена для написання скриптів). Після запуску «MonoDevelop» бачимо, що частина програмного коду вже створена автоматично. Див. рисунок 6.2.

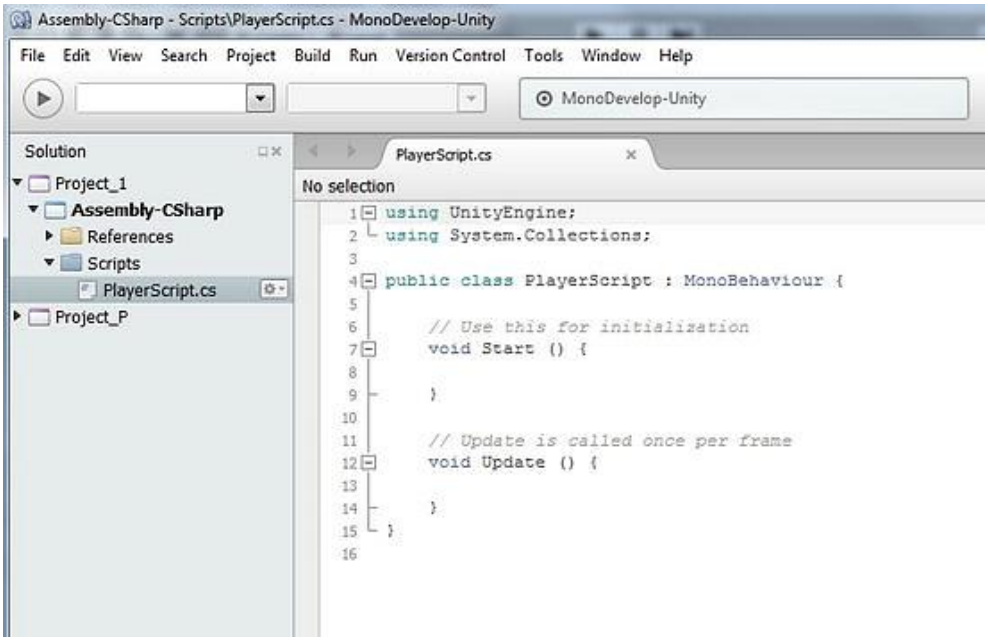



Рисунок 2 - Вікно програми «MonoDevelop»

### 6.4.2 **Аналіз шаблону скрипта**

У самому верху є два рядки:

```
using UnityEngine;
using System.Collections.Generic;
```

У програмному коді необхідно оперувати різними класами об'єктів. Описи класів містяться в спеціальних бібліотеках. Ось ці бібліотеки і підключаються до нашого коду командою «using».

У бібліотеці «UnityEngine» містяться описи всіх стандартних об'єктів всередині движка Unity (об'єкти, їх властивості, файли, префаб, система успадкування, зв'язку між об'єктами).

У бібліотеці «System.Collections.Generic» містяться найпростіші логічні конструкції (класи, списки, переліки, масиви, таблиці, вектори), а так само джерела зовнішніх даних для майбутньої гри (натискання клавіш клавіатури, кнопок миші, властивості екрану).

Далі в тексті є рядок:

```
public class PlayerScript : MonoBehaviour {
```

Це заголовок створеного скрипта. Видно, що третє слово «PlayerScript» - це назва скрипта, воно відповідає тому, як був названий файл «PlayerScript.cs». Після двокрапки вказаний клас скрипта - «MonoBehaviour». Це стандартний клас для всіх скриптів Юніті.

Після символу «{» починається перелік команд всередині скрипта. У самій останній сходинці скрипт обов'язково повинен завершиться символом «}».

Всередині скрипта видно рядки:

```
// Use this for initialization
void Start () {
}
// Update is called once per frame
void Update () {
}
```

Це дві порожні стандартні функції. «Void» - це команда виклику функції. «Start» і «Update» - назви функцій.

«()» - означає що це процедурна функція, для неї не потрібні зовнішні значення, і вона не видає результат, а просто виконує певні дії.

«{}» - початкові і кінцеві межі функцій, між цими символами повинні міститися рядки функції, але поки там порожньо.

Перед функціями бачимо рядки тексту, що починаються з символів «//». Так позначаються коментарі до програмного коду. Ці записи ніяк не впливають на сам код, але вони допомагають розібратися в ньому.

### 6.4.3 **Зміна скрипта.**

3. Рядок «using System.Collections;» змінюється, щоб отримати більше можливостей при розробці. Дописується категорія «.Generic»:

```
using System.Collections.Generic;
```

4. Функція «Start» виконується один раз при створенні об'єкта в грі, а функція «Update» повторюється кожну мить в процесі гри. «Start» не потрібна, її можна видалити.

5. У проекті будуть потрібні змінні значення. Створення: після рядка «public class PlayerScript: MonoBehaviour {» додати наступний текст:

```
// Зміна швидкості переміщення героя
public float playerSpeed = 2.0f;
// Поточна швидкість переміщення
private float currentSpeed = 0.0f;

// Створення змінних для кнопок
public List<KeyCode> upButton;
public List<KeyCode> downButton;
public List<KeyCode> leftButton;
public List<KeyCode> rightButton;

// Збереження останнього переміщення
private Vector3 lastMovement = new Vector3();
```

Опис:

«public» – глобальний тип змінної (її зможуть змінювати інші ігрові об'єкти).

«float» – тип значення, що зберігається в змінній, в даному випадку - число з дробовим значенням.

«playerSpeed» – назва змінної (можна назвати по іншому).

«= 2.0f» – початкове значення, яке зберігається в змінній. Дробове число написано в такому форматі - число з крапкою, а в кінці буква «f», щоб комп'ютеру було зрозуміло, що це не звичайна цифра, а число з дробовим значенням. Такий тип змінних використовується для координат об'єкту в просторі.

«private» – локальний тип змінної (таку змінну може змінювати тільки сам об'єкт, змінна для внутрішнього користування).

«List<KeyCode>» – тип змінної «масив з декількох значень», в масиві містяться посилання на клавіші клавіатури. «UpButton», «downButton», … - назви застосовуваних клавіш.

«Vector3» – тип змінної «тривимірний вектор». «New Vector3 ()» - створення порожнього вектору (обов'язково для ініціалізації такого типу змінної).

### 6.4.4 **Прив'язування скрипта до об'єкту.**

6. Зберегти зміни в скрипті. Це можна зробити, натиснувши комбінацію клавіш «Ctrl + S». Згорнути вікно «MonoDevelop», повернутися на екран Unity.

7. У «ієрархії» вибрати об'єкт корабля. Перетягнути файл скрипта у властивості корабля в вікні «інспектора». Там з'явиться нова властивість об'єкта «Player Script (Script)». У цьому місці можна побачити, що всі публічні змінні відображаються у властивостях об'єкта, і їх можна поміняти, без повернення до програмного коду (рисунок 6.3).

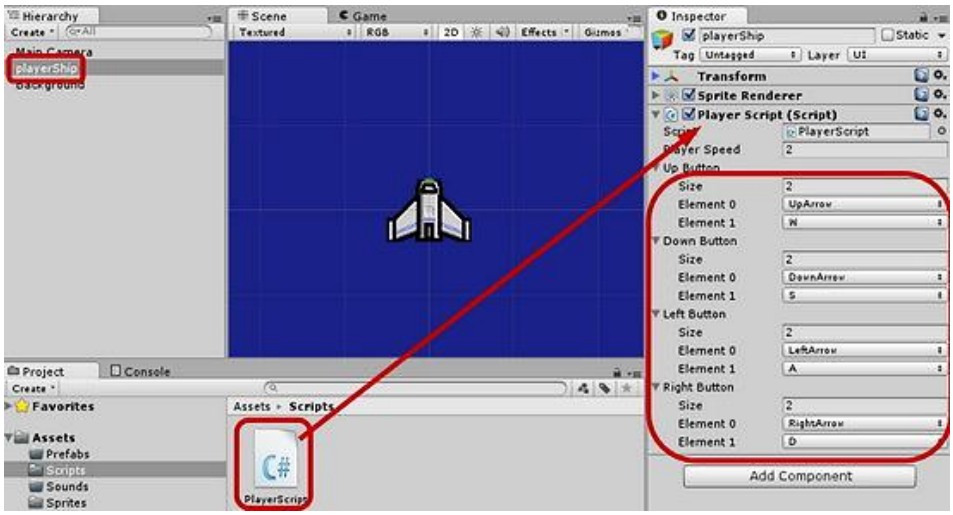



Рисунок 6.3 - Властивість об'єкта «Player Script (Script)

8. Налаштування управління кораблем. У властивостях кожної змінної-кнопки «Up Button», «Down Button», «Left Button», «Right Button» в рядку «Size» встановити значення «2». Після цього з'являються два списки «Element 0» і «Element 1», в них вибрати ті клавіші, які будуть відповідати цієї змінної. Для «Up Button» це «UpArrow» (клавіша зі стрілкою вгору на клавіатурі) і «W». У підсумку необхідно призначити змінним всі кнопки-стрілки і клавіші «W, A, S, D», як це показано на рисунку. (У списку слів на клавіатурі можна натискати клавішу з першою літерою назви клавіші, щоб швидко знайти її в величезному списку).

Таким чином управління переміщенням буде продубльовано і на «стрілках» і на буквених клавішах, а гравець вже сам буде вибирати, чим йому користуватися.

### 6.4.5 **Функція переміщення об'єкта.**

9. Повернутися в «MonoDevelop». Усередині функції «Update» прописати ще дві функції. Розміщення функцій в «Update» означає, що вони будуть повторюватися знову і знову, на протязі всієї гри:

```
// Update is called once per frame
void Update () {
// Поворот героя до мишки
Rotation();
// Переміщення героя
Movement();
}
```

10. Вище було написано лише виклик функцій. Тепер нижче, потрібно описати, що ж власне виконуватимуть ці функції. Після символу «}», який закриває функцію «Update», і перед останнім символом «}» додати код:

```
// Поворот героя до мишки
void Rotation() {
    // Показуємо гравцеві, де мишка
    Vector3 worldPos = Input.mousePosition;
    worldPos = Camera.main.ScreenToWorldPoint(worldPos);
    // Зберігаємо координати покажчика миші
    float dx = this.transform.position.x - worldPos.x;
    float dy = this.transform.position.y - worldPos.y;
    // Обчислюємо кут між об'єктами «Корабель» і «Покажчик»
    float angle = Mathf.Atan2(dy, dx) * Mathf.Rad2Deg;
    // Трансформуємо кут в вектор
    Quaternion rot = Quaternion.Euler(new Vector3(0, 0, angle + 90));
    // Змінюємо поворот героя
```

```
    this.transform.rotation = rot;
}
```

11. Опис функції руху корабля «Movement»:

```
// Рух героя до мишки
void Movement() {
    // Необхідний рух
    Vector3 movement = new Vector3();
    // Перевірка натиснутих клавіш
    movement += MoveIfPressed(upButton, Vector3.up);
    movement += MoveIfPressed(downButton,
Vector3.down);
    movement += MoveIfPressed(leftButton,
Vector3.left);
    movement += MoveIfPressed(rightButton,
Vector3.right);
    // Якщо натиснуто кілька кнопок, обробляємо це
    movement.Normalize();
    // Перевірка натискання кнопки
    if(movement.magnitude > 0)
    {
        // Після натискання рухаємося в цьому
напрямку
        currentSpeed = playerSpeed;
        this.transform.Translate(movement *
Time.deltaTime * playerSpeed, Space.World);
        lastMovement = movement;
    }
    else
    {
        // Якщо нічого не натиснуто
        this.transform.Translate(lastMovement *
Time.deltaTime * currentSpeed, Space.World);
```

```
            // Уповільнення згодом
            currentSpeed *= 0.9f;
        }
    }

    // Повертає рух, якщо натиснута кнопка
    Vector3 MoveIfPressed(List<KeyCode> keyList, Vector3
    Movement) {
        // Перевіряємо кнопки зі списку
        foreach (KeyCode element in keyList)
        {
            if(Input.GetKey (element))
            {
                // Якщо натиснуто, залишаємо функцію
                return Movement;
            }
        }
        // Якщо кнопки не натиснуті, то не рухаємося
        return Vector3.zero;
    }
```

12. Зберегти файл скрипта, повернутися у вікно Unity, зберегти сцену (в головному меню натиснути «File | Save Scene». Тепер можна запустити гру. На вкладці «Game» натиснути кнопку «Maximize on Play», щоб гра запускалася на все вікно Unity. Натиснути клавішу «Play» у верхній частині екрану для включення гри. (Ця ж кнопка для відключення гри). Див. рисунок 6.4.

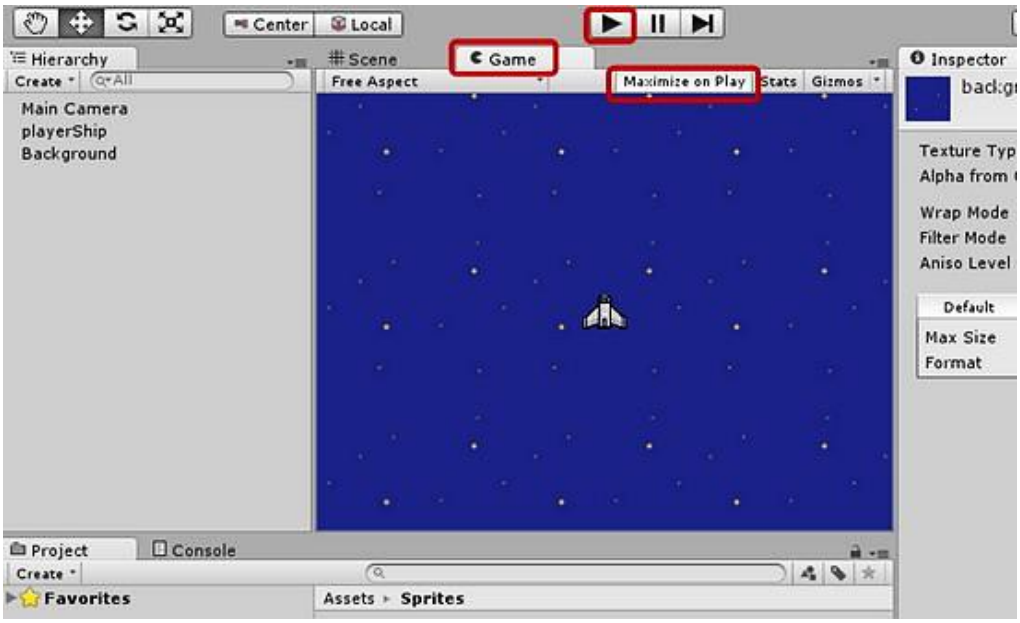



Рисунок 6.4 – Запуск гри.

Очікувані **практичні** результати роботи:

- У разі успішного виконання лабораторної роботи студент буде вміти створювати скрипти и прив'язувати їх до ігрових об'єктів.

# 7 Лабораторна робота №3 . Створення інтерактивного геймплея з персонажем за допомогою засобів движка Unity. Організація взаємодії користувача.

Лабораторна робота орієнтована на отримання практичних навичок в створенні інтерактивного геймплея з персонажем засобами движка Unity, а також на отримання практичних навичок в створенні та адаптації користувальницького інтерфейсу.

Мета лабораторної роботи:

- Навчити студентів створювати ігровий персонаж та організовувати керування ним з клавіатури;
- Навчити студентів динамічно створювати об’єкти ігрового середовища безпосередньо в процесі гри;
- Навчити студентів використовувати тригери для організації взаємодії об’єктів ігрового середовища;
- Навчити студентів створювати елементи інтерфейсу користувача та прив’язувати до них необхідні події;
- Сформувати вміння змінювати позиціонування елементів інтерфейсу та інших властивостей елементів інтерфейсу;
- Навчити студентів створювати переходи між різними екранами меню та сценами гри;
- Навчити студентів робити збірку різних сцен в один проект.

Відповідно до наскрізного завдання студенти в результаті обговорення в групі визначають характеристики ігрового персонажу (персонажів), способи його взаємодії з об’єктами ігрового середовища, та створюють ігровий персонаж для сцени ігрового додатку.

У разі успішного виконання лабораторної роботи студент буде вміти створювати ігровий персонаж та інші об’єкти і організовувати їх взаємодію з іншими статичними та динамічними об’єктами ігрового середовища; вміти створювати гру з інтерфейсом меню, забезпечувати переходи між різними екранами меню та сценами гри.

Успішне засвоєння матеріалів та виконання лабораторної роботи сприяє формування у студента наступних соціальних навичок та вмінь:

- ініціативність;

- вміння, необхідні для виконання групової та командної роботи;
- вміння дотримуватися регламентних рамок, оцінювати строки проведення та виконання роботи,
- вміння обґрунтувати той чи інший спосіб діяльності при виконанні практичних завдань;
- комунікаційна ефективність;
- вміння застосовувати раніше отримані знання як основу для засвоєння нових знань.

## 7.1 Завдання до лабораторної роботи

Кожна команда повинна додати ворогів до основного персонажу, які будуть нападати порціями.

## 7.2 Підготовка до лабораторної роботи № 3

При підготовці до виконання лабораторної роботи необхідно:

При підготовці до виконання лабораторної роботи необхідно:

- Запустити графічний редактор Unity.
- Ознайомитись з методичними вказівками до виконання лабораторної роботи (дивись 6.4)
- Ознайомитись засобами Unity, що використовуються для створення інтерактивного геймплея з персонажем.

## 7.3 Контрольні питання і попередні матеріали для допуску до лабораторної роботи № 3

### 7.3.1 Контрольні питання

- Як змусити новий об'єкт рухатися в потрібному напрямку?
- Яку властивість необхідно налаштувати, щоб об'єкт не "падав вниз"?
- Чи можуть скрипти звертатися один до одного і змінювати внутрішні змінні?
- Як об'єднати кілька об'єктів в групу?

### 7.3.2 Попередні матеріали

Для допуску к виконанню лабораторної роботи необхідно представити:

- Проект в якому створено ігрове середовище, та головний ігровий об'єкт до якого прив'язані скрипти дій.

- Продумати, згідно варіанту, які дії повинні виконувати додаткові ігрові об'єкти.

## 7.4 Методичні вказівки до виконання лабораторної роботи № 3

1. У вікні проекту в папку «Sprites» додати спрайт «enemy.png».

2. Перетягнути цей спрайт на ігрову сцену, щоб створити об'єкт. Рисунок 7.1.

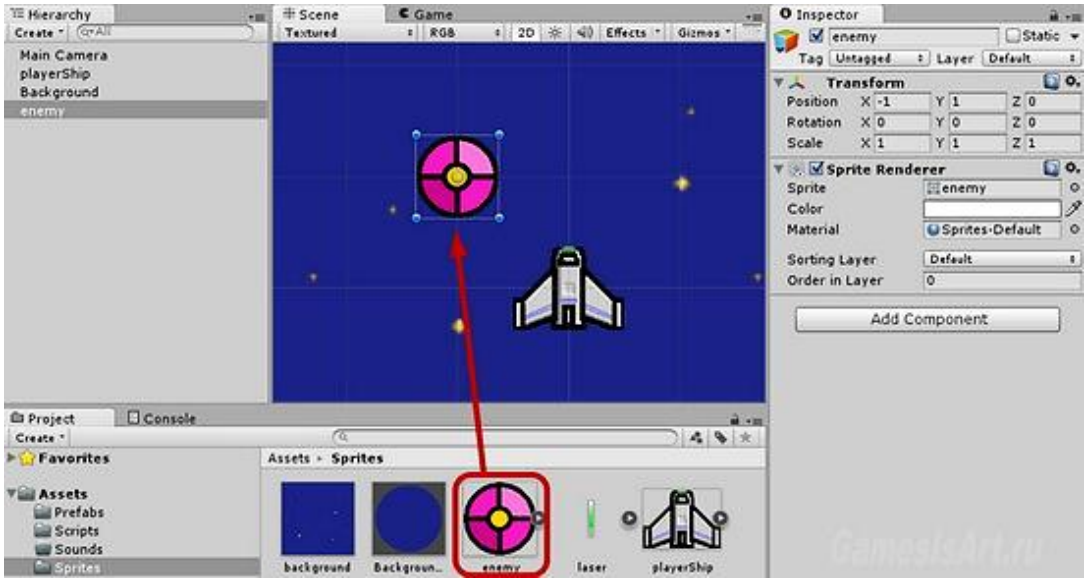



Рисунок 7.1. Створення об'єкту.

3. В папці «Scripts» створити новий скрипт, називати його «MoveTowardsPlayer». Перейти в «MonoDevelop», щоб прописати текст скрипта:

```
using UnityEngine;
using System.Collections;

public class MoveTowardsPlayer : MonoBehaviour
    {
    // Змінна для координат об'єкту player
    private Transform player;

    // Швидкість руху ворога
    public float speed = 1.5f;
```

```
    // Use this for initialization
    void Start ()
    {
        player                                                    =
GameObject.Find("playerShip").transform;
    }

    // Update is called once per frame
    void Update ()
    {
Vector3 delta = player.position - transform.position;
delta.Normalize();
float moveSpeed = speed * Time.deltaTime;
transform.position = transform.position + (delta *
moveSpeed);
    }
}
```

4. Перетягнути створений скрипт в якості ворожого об'єкту.

5. Створити у властивостях ворожого об'єкту новий компонент: в головному меню вибирати «Component | Physics 2D | Circle Collider 2D ». В його властивості в рядку «Radius» встановити «0.350». Рисунок 7.2.

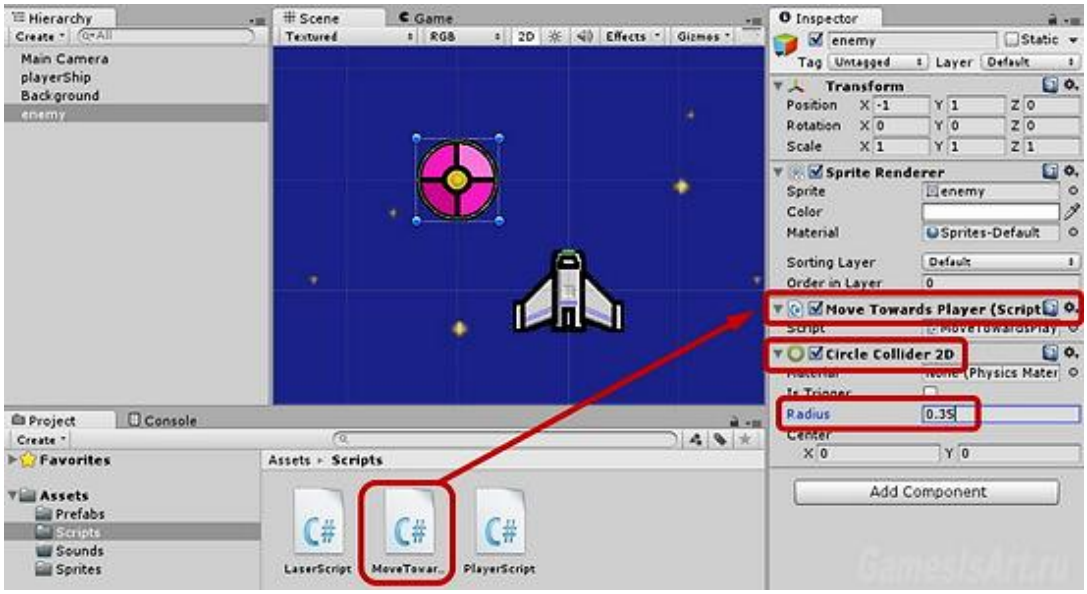



Рисунок 7.2. Налаштування руху.

6. Запустити гру для перевірки.

Ворожий об'єкт постійно рухається в бік головного персонажа. Швидкість у нього трохи менше, ніж у головного, але якщо він його наздожене, то потрапить під його спрайт і буде рухатися разом з ним. Це потрібно доопрацювати.

## 7.5 Створення скрипта, щоб у ворога можна було стріляти і знищувати його

7. У папці «Scripts» клікнути на кнопку «Create», вибрати «C # Script», назвати новий скрипт «EnemyScript». Перейти в «MonoDevelop», написати:

```
using UnityEngine;

public class EnemyScript : MonoBehaviour
{
    // Скільки разів потрібно потрапити у ворога, щоб знищити його
    public int health = 2;

    void OnCollisionEnter2D(Collision2D theCollision)
    {
        //Перевіряємо колізію з об'єктом типу «лазер»
        if(theCollision.gameObject.name.Contains("laser"))
        {
        LaserScript laser = theCollision.gameObject.GetComponent("LaserScript") as LaserScript;
        health -= laser.damage;
        Destroy (theCollision.gameObject);
        }
        if (health <= 0)
        {
            Destroy (this.gameObject);
```

```
        }
    }
}
```

8. Зберегти скрипт, повернутися у вікно Юніті. Приєднати скрипт «EnemyScript» до ворожого об'єкту.

9. Для того щоб відбувалася подія «колізія» (зіткнення об'єктів), потрібно створити для ворога фізичне тіло. У компонентах додати «Component | Physics 2D | Rigidbody 2D ». У цьому компоненті змінити властивість «Gravity Scale» на значення «0» (це щоб на ворога не діяли гравітація, інакше він постійно буде падати вниз). Рисунок 7.3.

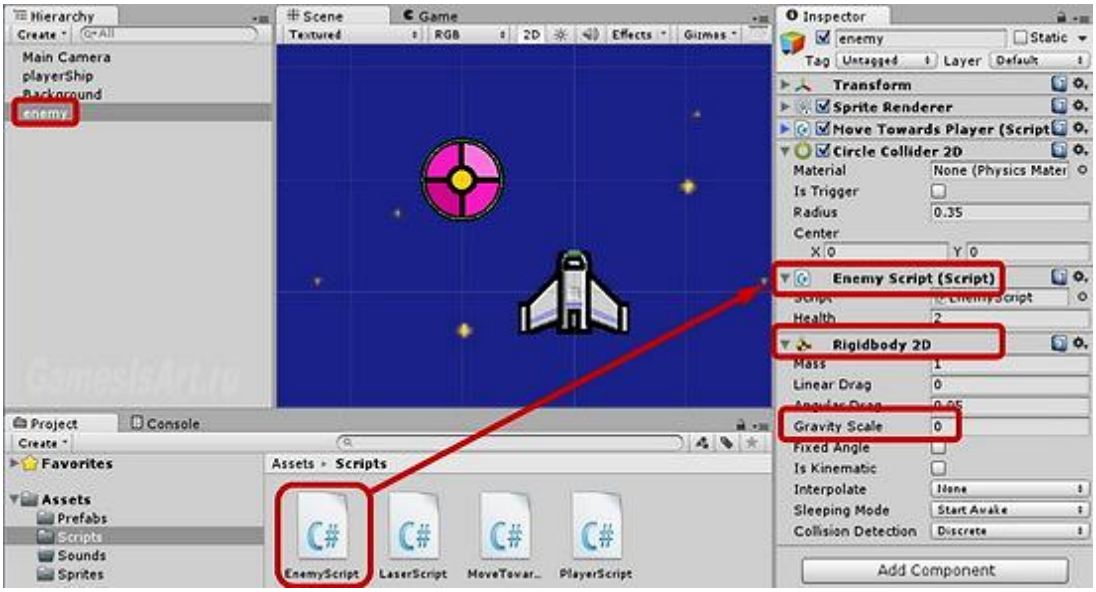



Рисунок 7.3. Налаштування «колізії».

## 7.6 Контролер для створення хвиль ворогів

1. Створити порожній ігровий об'єкт: в головному меню вибрати «GameObject | Create Empty ». Назвати об'єкт «GameController». В його властивості встановити «Position» = (0, 0, 0). Під ім'ям знайти рядок «Tag», за замовчуванням там стоїть значення «Untagged», замінити його на «GameController» («Tag» - це спосіб об'єднати кілька об'єктів в групу).

2. У властивостях об'єкта створити новий скрипт «Add Component | New Script », вибирати мову« CSharp », називати скрипт« GameController ». Натиснути кнопки «Create» і «Add». (Це ще один із способів створення файлів скриптів, але при цьому скрипт з'явиться в кореневій папці «Assets», перетягнути його в папку «Scripts»). Рисунок 7.4.

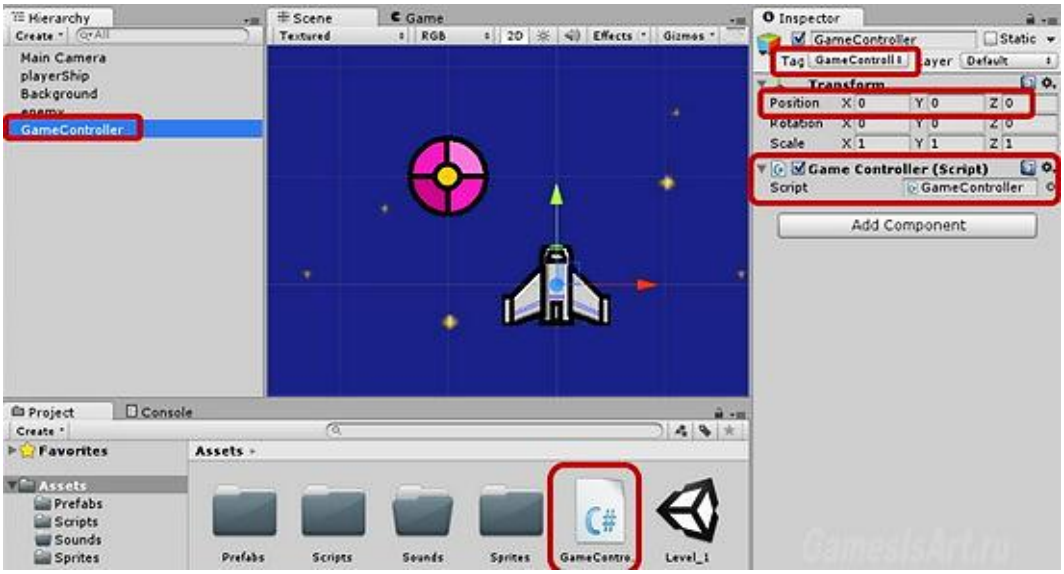



Рисунок 7.4. Створення нового ігрового об'єкта.

3. Вибрати щойно створений скрипт «GameController» в папці «Assets \ Scripts». Перейти в «MonoDevelop» для редагування тексту скрипта. У розділі опису змінних додати наступні рядки:

```
// Створення змінної «ворог»
public Transform enemy;
```

4. Тепер потрібно помістити в створену змінну об'єкт «enemy». Спочатку потрібно одиничний екземпляр об'єкту перетворити в префаб: перетягнути об'єкт «enemy» з ієрархії в розділ «Проект» в папку «Prefabs». Видалити об'єкт «enemy» з ігрової сцени.

Перетягнути створений префаб в змінну «enemy» у властивостях скрипта. Рисунок 7.5.

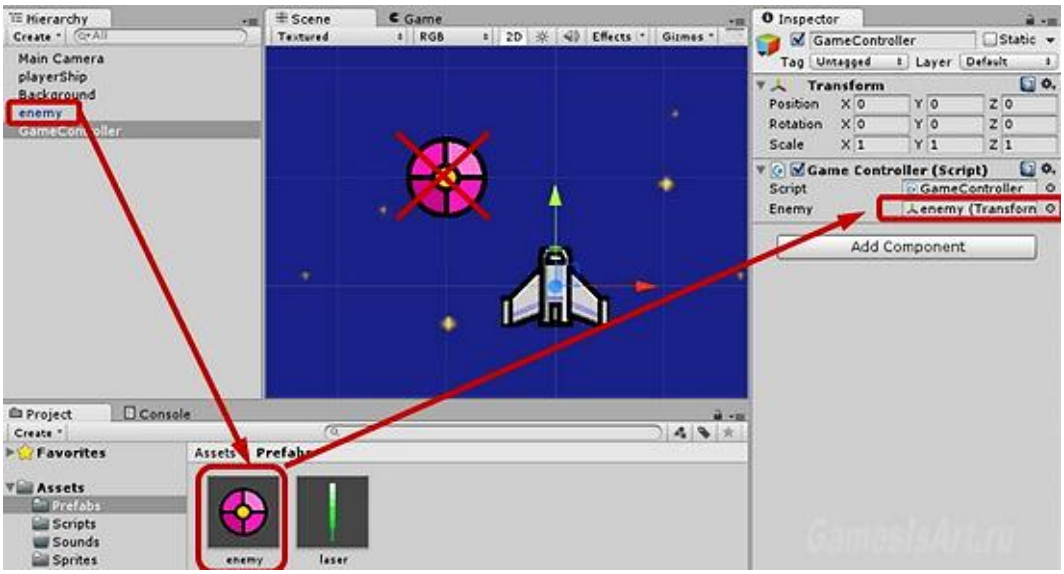



Рисунок 7.5. Перетворення об'єкта в префаб.

5. Повернутися до редагування скрипта «GameController». Створити змінні, які знадобляться:

```
// Тимчасові проміжки між подіями, к-ть ворогів
public float timeBeforeSpawning = 1.5f;
public float timeBetweenEnemies = 0.25f;
public float timeBeforeWaves = 2.0f;
public int enemiesPerWave = 10;
private int currentNumberOfEnemies = 0;
```

6. Створимо функцію для створення нових ворогів і назвемо її SpawnEnemies. Потрібно зробити так, щоб вороги з'являлися не всі відразу, а хвилями по 10 штук.

Усередині функції «Start» написати рядок:

```
StartCoroutine (SpawnEnemies());
```

Функція Coroutine призупиняє або продовжує виконання дій, що спрацьовують через якийсь проміжок часу. В даному випадку запускається лічильник для виконання функції «SpawnEnemies».

7. Нижче (після функції Update, але до останньої дужки яка закривається) прописується сама функція створення ворога:

```
// Поява хвиль ворогів
IEnumerator SpawnEnemies()
```

```
{
    // Початкова затримка перед першою появою ворогів
    yield return new WaitForSeconds (timeBeforeSpawning);
    // Коли таймер закінчиться, починаємо виробляти ці дії
    while(true)
    {
        // Не створювати нових ворогів, поки не знищені старі
        if (currentNumberOfEnemies <= 0)
        {
            float randDirection;
            float randDistance;
            // Створити 10 ворогів в випадкових місцях за екраном
            for (int i = 0; i < enemiesPerWave; i++)
            {
     // Задаємо випадкові змінні для відстані та напрямку
                randDistance = Random.Range (10, 25);
                randDirection = Random.Range (0, 360);
     // Використовуємо змінні для завдання координат появи ворога
      float posX = this. transform.position.x + (Mathf.Cos((randDirection) * Mathf.Deg2Rad) * randDistance);
      float posY = this. transform.position.y + (Mathf.Sin((randDirection) * Mathf.Deg2Rad) * randDistance);
      // Створюємо ворога на заданих координатах
      Instantiate (enemy, new Vector3 (posX, posY, 0), this.transform.rotation);
```

```
            currentNumberOfEnemies++;
            yield return new WaitForSeconds
(timeBetweenEnemies);
                }
            }
            // Очікування до наступної перевірки
            yield return new WaitForSeconds
(timeBeforeWaves);
        }
}
```

8. Прописати дії, які відбуватимуться при знищенні ворога. Для цього необхідно змінювати змінну «currentNumberOfEnemies», але це внутрішня змінна, і вона може бути змінена тільки всередині класу «GameController». Створити нову функцію в класі «GameController»:

```
// Процедура зменшення кількості ворогів у змінній
public void KilledEnemy()
{
    currentNumberOfEnemies--;
}
```

9. Відредагувати скрипт «EnemyScript». Усередині функції «onCollisionEnter2D» після функції знищення ворога «Destroy (this.gameObject)» додати пару рядків:

```
GameController controller =
GameObject.FindGameObjectWithTag("GameController").Ge
tComponent("GameController") as GameController;
controller.KilledEnemy();
```

10. Зберегти обидва скрипта, зберегти ігрову сцену, запустити гру. Тепер на основного гравця поступово будуть нападати 10 ворогів, а коли вони будуть знищені, з'явиться наступна група ворогів.

# 8 Заключний практичний семінар.

Заключний практичний семінар орієнтовано на обговорення створеного студентами при виконанні лабораторних робіт 1-3 прототипу закінченого ігрового додатку у середовищі Unity, який зібрано із сцен, створених студентами відповідно до індивідуального варіанту завдання.

**Мета практичного семінару:**

- Оцінити правильність визначення основних елементів сцен ігрового додатку та їх реалізації у середовищі Unity.
- Оцінити правильність визначення основних подій та дій об’єктів (ігрової механіки) та їх реалізації у середовищі Unity.
- Оцінити працездатність сцен ігрового додатку в різних збірках.
- Оцінити показники інтерфейсу сцен гри (юзабіліті, відповідність евристичним показникам оцінки інтерфейсів та коректність відображення елементів і їх роботи в різних збірках).
- Надати рекомендації щодо подальшого розвитку розробленого ігрового додатку.

За результатами обговорення на практичному семінарі студенти отримають навички оцінки ігрових додатків, створених у середовищі Unity, навчаться виявляти недоліки та помилки, що були допущені на етапах розробки елементів сцен, інтерфейсу, ігрової механіки додатку, визначати подальші шаги розвитку додатку з урахуванням результатів обговорення, отримають навички формування зауважень щодо функціональних та нефункціональних вимог до гри, що сприяли б підвищенню якості геймплея та інтересу гравців.

На заключному семінарі кожна група студентів виступає з презентацією своїх розробок в яких повинно:

- Надати відомості щодо завдання розробки відповідно до свого варіанту.
- Надати аналіз поведінки ігрових та неігрових персонажів, станів гри та розвитку подій у грі, обраної при виконанні лабораторних робіт
- Продемонструвати розроблені під час сумісної роботи ігровий проект.
- Визначити основні вади і недоліки розробленого додатку.
- Показати загальну перспективу розвитку проекту.

Крім того доповідач від групи повинен надати можливість тестування розробленого додатку. Представник будь-якої іншої групи вибирається доповідачем, який тестує розроблений ігровий додаток і виступає як опонент доповідача, висвітлюючи вади та недоліки розробки.

В ході обговорення студенти інших груп оцінюють ігровий проект який обговорюється, аналізують його, та визначають рекомендації щодо подальшого їх розвитку.

Результати виступу доповідача та «опонента» оцінюються за критеріями, що наведені в додатку В.

Успішне засвоєння матеріалів практичного семінару сприяє формуванню у студента наступних соціальних навичок та вмінь: критично ставитися до результатів особистої роботи, вміти проводити самооцінку виконаних робіт, прагнення вирішувати поточні проблеми самостійно, або із залученням членів команди, презентувати та аргументувати свої рішення, обговорювати в групі результати роботи.

Форми прояву: приймати участь у груповому обговоренні, оцінюванні результатів виконаної робіти, вміння самостійно формулювати та доносити до співрозмовника свої думки, вміння ясно і конкретно висловлювати думки, слухати та розуміти співрозмовника, використовувати прийоми ділового спілкування (публічного мовлення, презентаційних виступів, доповідей) та раніше отримані знання як основу для засвоєння нових знань, дотримуватись етичних норм поведінки, ділового та партнерського спілкування.

## 9 Заключний звіт. Вимоги

У заключний звіт з виконання комплексу лабораторних роботі повинні бути включені наступні пункти:

- титульна сторінка;
- мета роботи;
- наскрізне завдання до комплексу лабораторних робіт;
- короткі теоретичні відомості;
- попередні ескізи загального вигляду кімнати, ескізи спрайтів головного героя, перешкод, фруктів, ворога.

- опис використаного технічного і програмного забезпечення;
- опис закінченого проекту з зазначенням всіх об’єктів, подій і дій;
- аналіз тестових прогонів гри, опис виявлених помилок, багів, тощо;
- пропозиції щодо подальшого розвитку створеного ігрового додатку;
- висновки.

Вимоги до змісту окремих частин звіту з лабораторної роботи

**Мета роботи** повинна відображати тему лабораторної роботи, а також конкретні завдання, поставлені студенту на період виконання роботи. За обсягом мета роботи в залежності від складності та багатозадачності роботи становить від кількох рядків до 0,5 сторінки.

**Короткі теоретичні відомості**. У цьому розділі викладається короткий теоретичний опис досліджуваного в роботі явища або процесу, наводяться також необхідні розрахункові формули. Матеріал розділу не повинен копіювати зміст методичного посібника або підручника з даної теми, а обмежується викладом основних понять і законів, розрахункових формул, таблиць, потрібних для подальшої обробки отриманих експериментальних результатів. Обсяг літературного огляду не повинен перевищувати 1/3 частини всього звіту.

**Опис експериментальної установки** і методики виконання роботи. В даному розділі наводиться структура установки з описом її роботи і детально викладається методика виконання роботи, процес отримання даних і спосіб їх обробки. Якщо використовуються стандартні пакети комп'ютерних програм, то необхідно обґрунтувати можливість і доцільність їх застосування. В цьому розділі необхідно також описати математичну модель і комп'ютерні програми, що використовуються для виконання роботи.

**Експериментальні результати.** У цьому розділі наводяться безпосередньо результати, отримані в ході проведення лабораторних робіт: результаті виконання розроблених програм, процедур, тощо, певні значення величин, скріншоти, графіки, таблиці, діаграми.

**Аналіз результатів роботи.** Розділ звіту повинен містити докладний аналіз отриманих результатів. Слід порівняти отримані результати з

очікуваними, обговорити їх відповідність тестовим розрахункам, ескізам. Якщо виявлено невідповідність отриманих і передбачуваних результатів і тестових розрахунків, необхідно обговорити можливі причини цих невідповідностей.

**Висновки.** У висновках коротко викладаються результати роботи: отримані результати, вказується їх відповідність або невідповідність завданню (очікуваним результатом), можливі причини невідповідності.

Заключний звіт оформлюється відповідно до ДСТУ [7]. Зразок написання титульного аркуша звіту наведено в додатку Д.

# 10 Література

1. Довідник модуля «Розробка ігрових додатків на базі движка Unity» дисципліни «Інструментальна підтримка розробки комп'ютерних ігрових додатків»
2. Документація, посібник по Unity:
   https://docs.unity3d.com/ru/current/Manual/index.html/
3. Офіційний сайт Unity3d: https://unity3d.com/ru.
4. Розділ навчальних матеріалів по Unity:
   https://unity3d.com/ru/learn/tutorials
5. Steven Goodwin. Polished Game Development: From First Steps to Final Release , 2016. – 269 p.
6. Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.
7. Sue Blackman. Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine, 2011. – 986 p.
8. Сайт розробників ігор http://gamesmaker.ru/3d-game-engines/unity3d/
9. ДСТУ 3008-2015 «Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлення».

## Додаток А. Критерії оцінювання виконання лабораторної роботи

| | | | | |
|---|---|---|---|---|
| Хід виконання | Самостійність виконання | Робота виконується повністю самостійно (без допомоги викладача) з елементами інновації, креативності<br>Робота виконується повністю самостійно<br>Робота викується з незначною допомогою викладача<br>Робота не може бути виконана без допомоги викладача | 3<br>2<br>1<br>0 | |
| | Виконання лабораторної роботи в передбачені терміни | Робота виконана з випередженням терміну<br>Робота виконана у відведений термін<br>Робота виконана із незначним (менше ніж тиждень) порушенням терміну<br>Робота виконана із значним (більш ніж тиждень) порушенням терміну | 3<br>2<br>1<br>0 | |
| Результат виконання | Відповідність результату теоретичним посилання | Знає та використовує методи, способи, технології інтелектуального аналізу та обробки даних, здійснює опрацювання, інтерпретацію та узагальнення отриманих результатів<br>Знає але не в повній мірі використовує методи, способи, технології інтелектуального аналізу та обробки даних, здійснює опрацювання, інтерпретацію та узагальнення отриманих | 3<br>2 | |

| | | | | |
|---|---|---|---|---|
| | | результатів<br>Має часткові знання щодо методів, способів, технологій інтелектуального аналізу та обробки даних. Не в повній мірі здійснює опрацювання, інтерпретацію та узагальнення отриманих результатів<br>Не здійснює опрацювання, інтерпретацію та узагальнення отриманих результатів | 1<br>0 | |
| | Збіг отриманих результатів з тестовими розрахункам | Отримані результати повністю співпадають з тестовими розрахунками, виконано аналіз трудомісткості отримання результату<br>Отримані результати повністю співпадають з тестовими розрахунками.<br>Отримані результати частково співпадають з тестовими розрахунками.<br>Отримані результати не співпадають з тестовими розрахунками. | 3<br>2<br>1<br>0 | |

Підсумкова оцінка виконання лабораторної роботи

| | | |
|---|---|---|
| 24 - 20 | Перевищує вимоги | 3 |
| 19 - 13 | Відповідає вимогам | 2 |
| 12 - 5 | Майже відповідає вимогам | 1 |
| 4 -0 | Не відповідає вимогам | 0 |

## Додаток Б. Критерії оцінювання презентації результатів виконання наскрізного завдання на заключному практичному семінарі

| | | | | |
|---|---|---|---|---|
| Зміст<br>Студент пояснює процес та висновки проекту та демонструє отримані знання. | Визначення теми | □ Чітко визначає тему чи основні питання, їх значення.<br>□ Чітко визначає тему.<br>□ Визначає тему.<br>□ Не визначає тему. | 3<br>2<br>1<br>0 | |
| | Доказовість | □ Підтверджує достовірність основних результатів шляхом аналізу відповідних та точних доказів<br>□ Підтверджує достовірність основних результатів шляхом використання необхідних доказів<br>□ Підтверджує достовірність результатів шляхом використання мінімально необхідних доказів<br>□ Не підтверджує результати доказами | 3<br>2<br>1<br>0 | |
| | Джерела | □ Використовує технології та інструментарії пошукових систем. Надає докази обширного та достовірного дослідження з використанням численних та різноманітних джерел. | 3<br>2<br>1 | |

| | | | | |
|---|---|---|---|---|
| | | Не використовує технології та інструментарії пошукових систем. Надає докази достовірності дослідження за різноманітними джерелами.<br>Надає докази достовірності дослідження за рядом джерел.<br>Надає незначну кількість доказів дослідження (або не надає взагалі) за сторонніми джерелами. | 0 | |
| | Вирішення проблеми | Генерує нові ідеї вирішення складної проблеми та виходить з поясненнями за межі навчання.<br>Надає різноманітні свідчення вирішення складної проблеми та виходить з поясненнями за межі навчання.<br>Надає деякі свідчення вирішення проблеми.<br>Надає мало свідчень вирішення проблеми та виходу за межі навчання. | 3<br>2<br>1<br>0 | |
| | Ідейна база | Поєднує і оцінює існуючі ідеї для формування нових поглядів.<br>Поєднує існуючі ідеї для формування нових поглядів.<br>Поєднує існуючі ідеї.<br>Незначною мірою поєднує існуючі ідеї. | 3<br>2<br>1<br>0 | |
| Організація матеріалу | Представлення теми | Представляє тему чітко та творчо<br>Представляє тему чітко | 3<br>2 | |

| | | | | |
|---|---|---|---|---|
| презентації<br>Студент демонструє логічність та організованість | | □ Представляє тему<br>□ Не представляє тему | 1<br>0 | |
| | Утримання фокусу на темі | □ Підтримує чіткий фокус на темі.<br>□ Підтримує фокус на темі.<br>□ Деякою мірою підтримує фокус на темі.<br>□ Не підтримує фокус на темі. | 3<br>2<br>1<br>0 | |
| | Використання переходів | □ Ефективно використовує поступові переходи для поєднання ключових моментів.<br>□ Використовує поступові переходи для поєднання ключових моментів.<br>□ Використовує деякі переходи для поєднання ключових моментів.<br>□ Використовує деякі переходи, що рідко дозволяють поєднати ключові моменти | 3<br>2<br>1<br>0 | |
| | Висновки | □ Презентація закінчується логічним, ефективним і відповідним висновком.<br>□ Презентація закінчується відповідним очевидним висновком<br>□ Презентація закінчується очевидним висновком.<br>□ Презентація закінчується без висновку | 3<br>2<br>1<br>0 | |

| | | | | |
|---|---|---|---|---|
| Питання й відповіді | Представлення теми | Має систематичні знання в досліджуваній тематиці, демонструє вміння осмислювати тему і робити обґрунтовані висновки. Точно та належним чином на всі запитання аудиторії<br>Показує широкі знання теми, відповідаючи впевнено, точно та належним чином на всі запитання та зауваження аудиторії.<br>Демонструє знання теми, відповідаючи точно та відповідним чином на питання та відгуки.<br>Демонструє неповне знання теми, відповідаючи неточно і невідповідно на питання та відгуки. | 3<br>2<br>1<br>0 | |
| Використання мови та стиль вираження себе в усній формі<br>Студент ефективно передає свої ідеї | Зоровий контакт | Ефективно використовує зоровий контакт.<br>Підтримує зоровий контакт.<br>Є деякий зоровий контакт, але він не підтримується.<br>Неефективний зоровий контакт. | 3<br>2<br>1<br>0 | |
| | Мова | Говорить чітко, по суті і впевнено, використовуючи правильні обсяг і темп<br>Говорить чітко, використовуючи правильні обсяг і темп<br>Мова частково чітка, частково нечітка<br>Мова нечітка, не підходящий темп | 3<br>2<br>1<br>0 | |
| | Спілкування | Демонструє знання граматичних, стилістичних особливостей | 3 | |

| | | | | |
|---|---|---|---|---|
| | | лексики, термінології, лексичних структур. Використовує типові для тематичної комунікації лексико-синтаксичні моделі.<br>□ Демонструє знання граматичних, стилістичних особливостей лексики, термінології, лексичних структур. Не використовує типові для тематичної комунікації лексико-синтаксичні моделі.<br>□ Частково відсутні знання граматичних, стилістичних особливостей лексики, термінології.<br>□ Відсутні знання граматичних, стилістичних особливостей лексики, термінології. | 2<br>1<br>0 | |
| | Залучення аудиторії | □ Повністю залучає аудиторію.<br>□ Старається залучити аудиторію.<br>□ Випадкове залучення аудиторії.<br>□ Аудиторія не залучається. | 3<br>2<br>1<br>0 | |
| | Одяг | □ Одягнений належним чином<br>□ Одягнений неналежним чином | 1<br>0 | |

Підсумкова оцінка презентації та виступу на заключному практичному семінарі

| Сумарний бал | | Зарахований бал |
|---|---|---|
| 44 - 35 | Перевищує вимоги | 4 |

| | | |
|---|---|---|
| 34 – 27 | Відповідає вимогам | 3 |
| 26 - 20 | Частково відповідає вимогам | 2 |
| 19 10 | Майже відповідає вимогам | 1 |
| 9 -0 | Не відповідає вимогам | 0 |

## Додаток В. Критерії оцінювання заключного звіту

| | | | | |
|---|---|---|---|---|
| Завдання роботи | Формулювання завдання роботи | Завдання на роботу сформульовано докладно, з можливістю перевірки зрозуміння виконавцем<br>Завдання на роботу сформульовано, можливість перевірки розуміння утруднена<br>Завдання на роботу сформульовано частково, перевірка розуміння не можлива<br>Завдання на роботу не сформульовані, перевірка розуміння не можлива | 3<br>2<br>1<br>0 | |
| | Очікувані результати | Очікувані результати сформульовані повністю, дозволяють легке порівняння з отриманими результатами, наведені повні тестові розрахунки<br>Очікувані результати сформульовані повністю, наведені частково тестові розрахунки, що не дозволяє легкого порівняння з отриманими результатами<br>Очікувані результати сформульовані частково, наведені частково тестові розрахунки, що не дозволяє порівняння з отриманими результатами | 3<br>2<br>1 | |

| | | | | |
|---|---|---|---|---|
| | | □ Очікувані результати не сформульовані, тестові розрахунки не наведені | 0 | |
| Опис процедури (процесу) виконання роботи | Опис устаткування | □ Наведено докладний перелік та опис використаного устаткування (комп'ютери, засоби виводу графічної інформації, комунікаційне обладнання, тощо)<br>□ Наведено перелік та частковий опис використаного устаткування (комп'ютери, засоби виводу графічної інформації, комунікаційне обладнання, тощо)<br>□ Наведено частковий перелік використаного устаткування (комп'ютери, засоби виводу графічної інформації, комунікаційне обладнання, тощо)<br>□ Перелік та опис використаного устаткування (комп'ютери, засоби виводу графічної інформації, комунікаційне обладнання, тощо) відсутній | 3<br>2<br>1<br>0 | |
| | Опис використаного програмного продукту | □ Наведено докладний перелік та опис використаного стандартного програмного продукту (операційна система, середа розробки, прикладні пакети, тощо)<br>□ Наведено перелік та частковий опис використаного стандартного програмного продукту (операційна система, середа розробки, прикладні пакети, тощо)<br>□ Наведено частковий перелік використаного стандартного | 3<br>2<br>1 | |

| | | | | |
|---|---|---|---|---|
| | | програмного продукту (операційна система, середа розробки, прикладні пакети, тощо)<br>□ Перелік використаного стандартного програмного продукту (операційна система, середа розробки, прикладні пакети, тощо) не наведено | 0 | |
| | Процедура виконання | □ Усі кроки докладно перераховані та легко можуть бути повторені<br>□ Усі кроки докладно перераховані, але не можуть бути легко повторені<br>□ Не всі кроки перераховані і не можуть бути легко повторені<br>□ Процедури виконання не приведена | 3<br>2<br>1<br>0 | |
| | Наведено опис розроблених за завданням програм (алгоритмів) | □ Наведено блок-схеми та докладний текстовий опис розроблених програм (алгоритмів)<br>□ Наведено блок-схеми та частковий текстовий опис розроблених програм (алгоритмів)<br>□ Наведено блок-схеми розроблених програм (алгоритмів). Текстовий опис розроблених алгоритмів не наведено.<br>□ Опис розроблених за завданням прогам та алгоритмів не наведено | 3<br>2<br>1<br>0 | |
| | Наведено опис отриманих за завданням | □ Наведені отримані результати та їх порівняння очікуваними результатами та з тестовими розрахунками | 3 | |

| | | | | |
|---|---|---|---|---|
| | результаті | □ Наведені отримані результати. Їх порівняння з очікуваними результатами та з тестовими розрахунками не наведено.<br>□ Відсутній опис отриманих результатів | 2<br>0 | |
| Заключна частина | Висновки | □ Обґрунтовано підтверджуються теоретичні положення, підтверджена правильність роботи програми, алгоритму<br>□ Теоретичні положення підтверджуються частково, правильність роботи алгоритму (програми) підтверджена частково<br>□ Теоретичні положення , правильність роботи алгоритму (програми) не підтверджені<br>□ Відсутні висновки за роботою | 3<br>2<br>1<br>0 | |
| | Подальший розвиток за напрямом роботи | □ Надано розширений опис шляхів удосконалення запропонованих при виконанні роботи рішень (процедур, алгоритмів, програм), представлені напрямки подальшого розвитку підходів щодо рішень поставленої проблеми.<br>□ Надано опис шляхів удосконалення запропонованих при виконанні роботи рішень (процедур, алгоритмів, програм), представлені напрямки подальшого розвитку підходів щодо рішень поставленої проблеми.<br>□ Надано опис шляхів удосконалення запропонованих при виконанні роботи рішень (процедур, алгоритмів, програм), напрямки подальшого розвитку підходів щодо рішень | 3<br>2<br>1 | |

| | | | | |
|---|---|---|---|---|
| | | поставленої проблеми не представлені.<br>□ Шляхи удосконалення запропонованих при виконанні роботи рішень не запропоновані, напрямки подальшого розвитку підходів щодо рішень поставленої проблеми не представлені. | 0 | |
| Звіт як технічний текст | Містить:<br>назву, прізвище студента, прізвище викладача, дату | □ Немає відсутніх компонентів<br>□ Відсутня 1 компонента<br>□ Відсутні 2 -4 компоненти<br>□ Відсутні більш ніж 4 компоненти | 3<br>2<br>1<br>0 | |
| | Акуратно оформлена | □ Звіт оформлений акуратно<br>□ Звіт оформлено<br>□ Звіт оформлено не акуратно | 2<br>1<br>0 | |
| | Помилки | □ Помилки відсутні<br>□ Невелика кількість граматичних та синтаксичних помилок (на сторінці не більш ніж 1 помилка)<br>□ Велика кількість граматичних та синтаксичних помилок (кількість помилок більше ніж кількість сторінок звіту) | 2<br>1<br>0 | |
| | Технічні аспекти | □ Відсутні помилки в пунктуації, використанні заголовних букв і орфографії.<br>□ Майже немає помилок в пунктуації, використанні заголовних букв і орфографії. | 3<br>2 | |

| | | | | |
|---|---|---|---|---|
| | | Багато помилок в пунктуації, використанні заголовних букв і орфографії.<br>Численні помилки в пунктуації, використанні заголовних букв і орфографії, що не дають можливості зрозуміти думку автора | 1<br>0 | |
| | Використання слів | Речення побудовані без помилок, всі слова використовуються вірно.<br>Майже немає помилок в структурі речень і використанні слів.<br>Багато помилок в побудові структури речень структури і використанні слів.<br>Численні і відволікаючі помилки в структурі речень і використанні слова | 3<br>2<br>1<br>0 | |

Підсумкова оцінка заключного звіту

| Сумарний бал | | Зарахований бал |
|---|---|---|
| 40 – 27 | Відповідає вимогам | 2 |
| 26 -13 | Майже відповідає вимогам | 1 |
| 13 -0 | Не відповідає вимогам | 0 |

**Додаток Д. Приклад оформлення титульної сторінки заключного звіту.**

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Кафедра прикладної математики та інформатики

**ЗАКЛЮЧНИЙ ЗВІТ**

З виконання комплексу лабораторних робіт

з дисципліни: Інструментальна підтримка розробки комп'ютерних ігрових додатків, модуль: Розробка ігрових додатків на базі движка Unity

Виконав студент гр. ІПЗ 14
_____ Іванов І.І.
«____»_____2018 р.

Прийняв_____
Викладач кафедри ПМІ
_____ Петров В.О.
«____»_____2018 р.

Покровськ - 2018

Навчальне видання

Навчальний модуль «Розробка ігрових додатків на базі движка Unity» з дисципліни «Інструментальна підтримка розробки комп'ютерних ігрових додатків» підготовки студентів освітнього рівня «магістр» спеціальності 121 «Інженерія програмного забезпечення»

Укладачі: Скрипник Тетяна Володимирівна
Тихонова Оксана Анатоліївна

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Donetsk National Technical University

Модуль «Розробка ігрових додатків на базі движка Unity»

Асистент Скрипник Т.В.

**DonNTU GameHub**

***561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Лекція 1. Частина 1.

- Огляд сучасних інструментальних засобів розробки ігрових додатків.
- Основи роботи в середовищі Unity.
- Створення простого ігрового середовища засобами Unity.

Co-funded by the Erasmus+ Programme of the European Union



# Лекція 1. Частина 1.

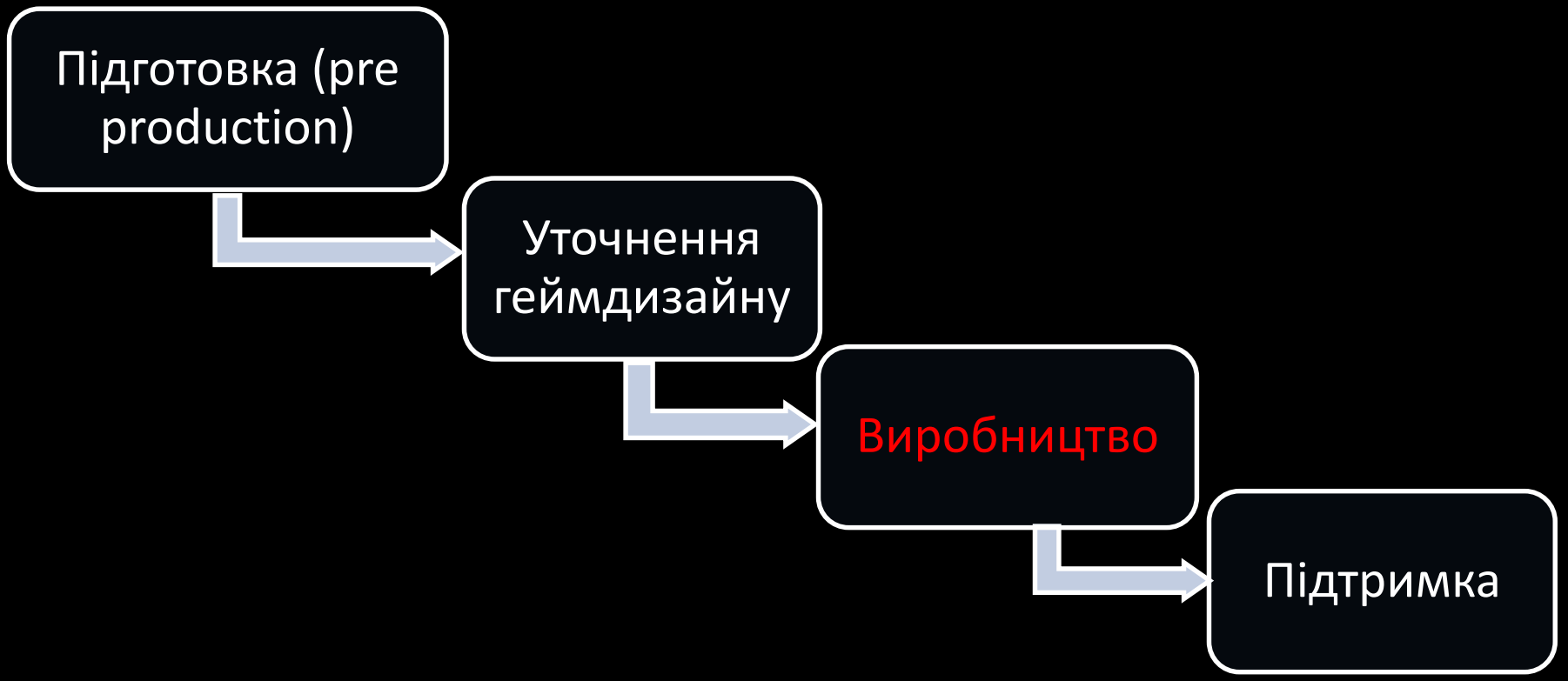
- Середовище Unity: основні об’єкти та елементи інтерфейсу
- Класифікація і опис властивостей стандартних об’єктів Unity.

Co-funded by the Erasmus+ Programme of the European Union



# Основні інструментальні засоби створення ігрових додатків

## Етапи створення гри



Co-funded by the Erasmus+ Programme of the European Union



# Процес виробництва гри

- Написання програмного коду (програмісти)
- Створення графіки (художники)
- Розробка звукових ефектів (звукооператори)
- Написання музики (композитори)
- Доповнення та зміни геймдизайну (геймдизанери)
- Створення рівнів (дизайнери рівнів)
- Написання діалогів (письменники)

Co-funded by the Erasmus+ Programme of the European Union



# Поняття ігрового рушія

**Ігровий рушій –** інструментарій для створення комп'ютерних та відеоігор або інших інтерактивних додатків із графікою, що обробляється в реальному часі.

**Призначення:**

- **Спрощення розробки ігор**
- **Можливість повторного використання одного ігрового рушія для створення різних ігор**
- **Підтримка різних платформ**

Co-funded by the Erasmus+ Programme of the European Union



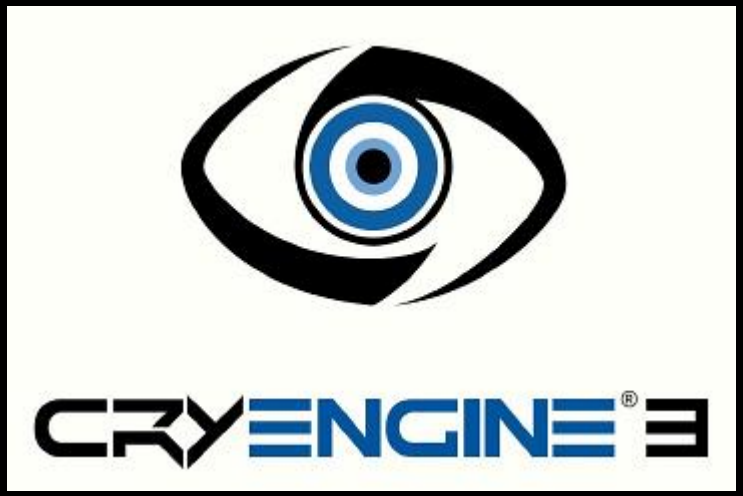
# Движок Phaser

- Рушій  з відкритим вхідним кодом
- Призначений для розробки десктопих та мобільних HTML5 ігор

Co-funded by the Erasmus+ Programme of the European Union



# Движок CryEngine




- Комерційний рушій
- Існує безкоштовна освітня ліцензія, що не видається у відриві від навчального закладу
- Підтримує ігрові приставки PlayStation 3, Xbox 360 та їхніх нащадків
- Приклади ігор: Giant, Sniper II: Ghost Warrior та інші

Co-funded by the Erasmus+ Programme of the European Union



# Движок Unreal Engine


- Може використовуватися для роботи з графікою в кінематографі, наприклад, для створення освітлювальних спецефектів
- Підтримує різні платформи
- Є безкоштовна версія, але з 5% оплатою рояті, якщо прибуток перевищить $3000

Co-funded by the Erasmus+ Programme of the European Union



# Движок Unity


- Відсутня прив’язка до певного стилю гри
- Підтримує 24 платформи: десктоп, мобільні пристрої, VR, консолі і веб-платформи
- Існує безкоштовна версія
- Приклади ігор: Temple Run, Deus Ex: The Fall, Assassin’s Creed: Identity

Co-funded by the
Erasmus+ Programme
of the European Union

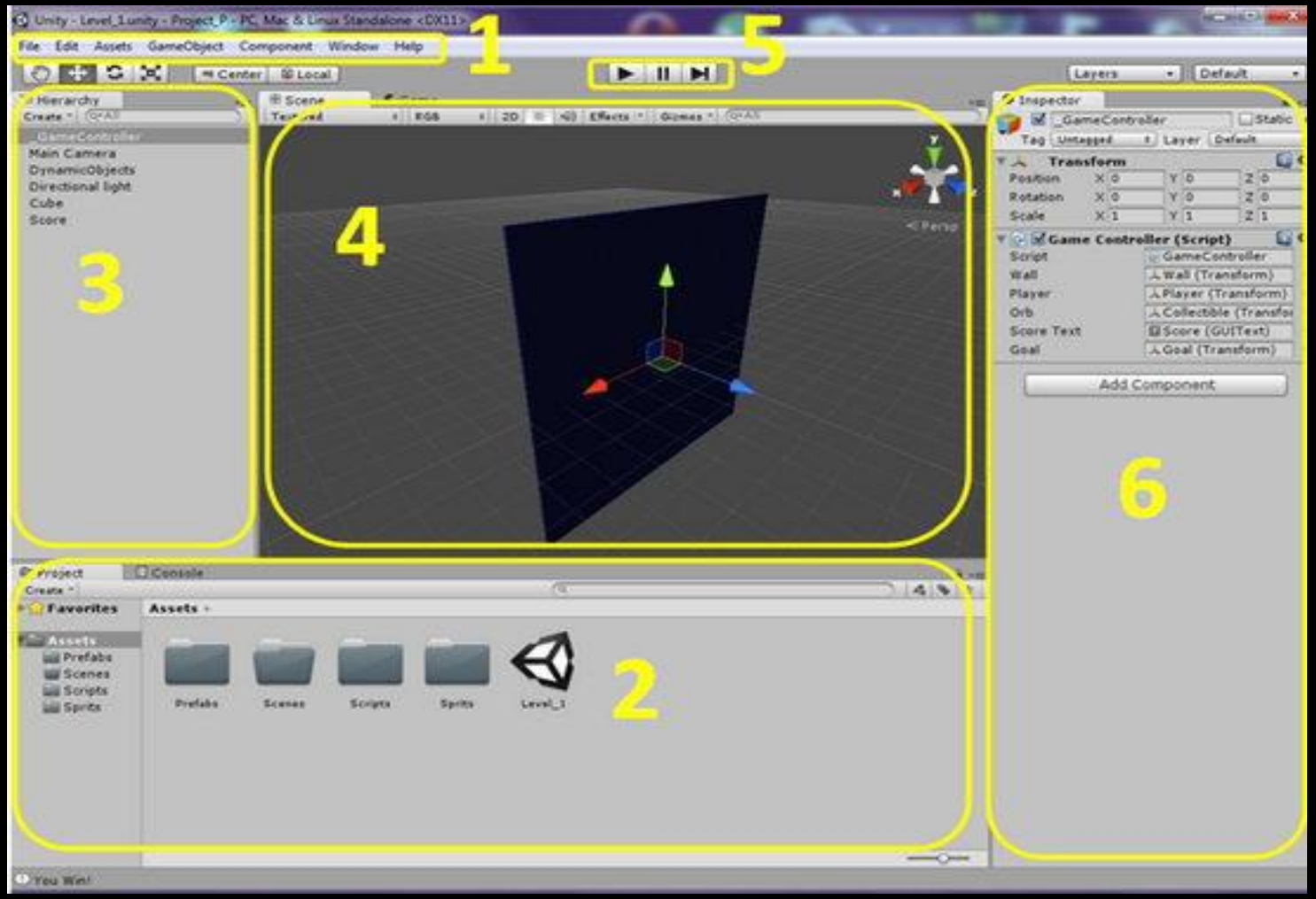


# Інтерфейс програми Unity

Co-funded by the
Erasmus+ Programme
of the European Union



# Загальний вид



Co-funded by the Erasmus+ Programme of the European Union



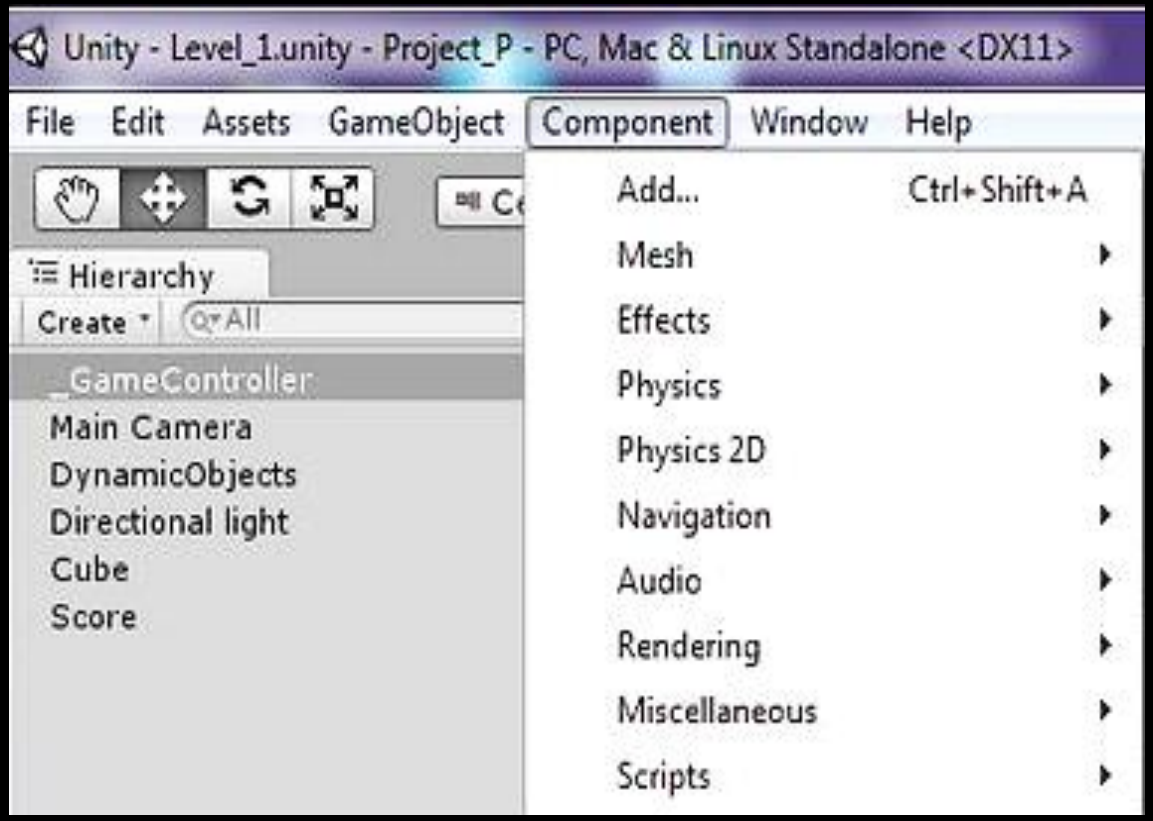
# Перелік всіх робочих областей вікна Unity:

| | |
|---|---|
| 1. Main menu (Головне меню) | Рядок тексту зверху, де розташовано всі команди, доступні в програмі. Багато команд продубльовано кнопками і меню в робочих областях, тому головне меню не обов'язково використовувати. |
| 2. Project View (Огляд проекту) | Список, в якому показано всі використовувані в грі файли: файл сцени, файл програмного коду, графічні і аудіофайли. |
| 3. Hierarchy (Ієрархія) | Список, де перераховані всі об'єкти, додані на сцену. Тут можна працювати з об'єктами, копіювати їх, перейменовувати, видаляти. |
| 4.Scene (Сцена) | Область, де відображається ігровий світ або ігрова сцена. Тут можна додавати нові об'єкти, перетягувати їх, змінювати вид. |
| 5.Game (Гра) | Область попереднього перегляду, де видно, як сцена буде виглядати в грі. Тут можна налаштовувати різні параметри екрану і режиму відео. |
| 6.Inspector (Інспектор) | Список, що складається з декількох різних за виглядом розділів. Показує всі властивості обраного об'єкта: розміри, моделі, текстури, скрипти. |

Co-funded by the
Erasmus+ Programme
of the European Union



# Головне меню (Main menu)



Co-funded by the
Erasmus+ Programme
of the European Union

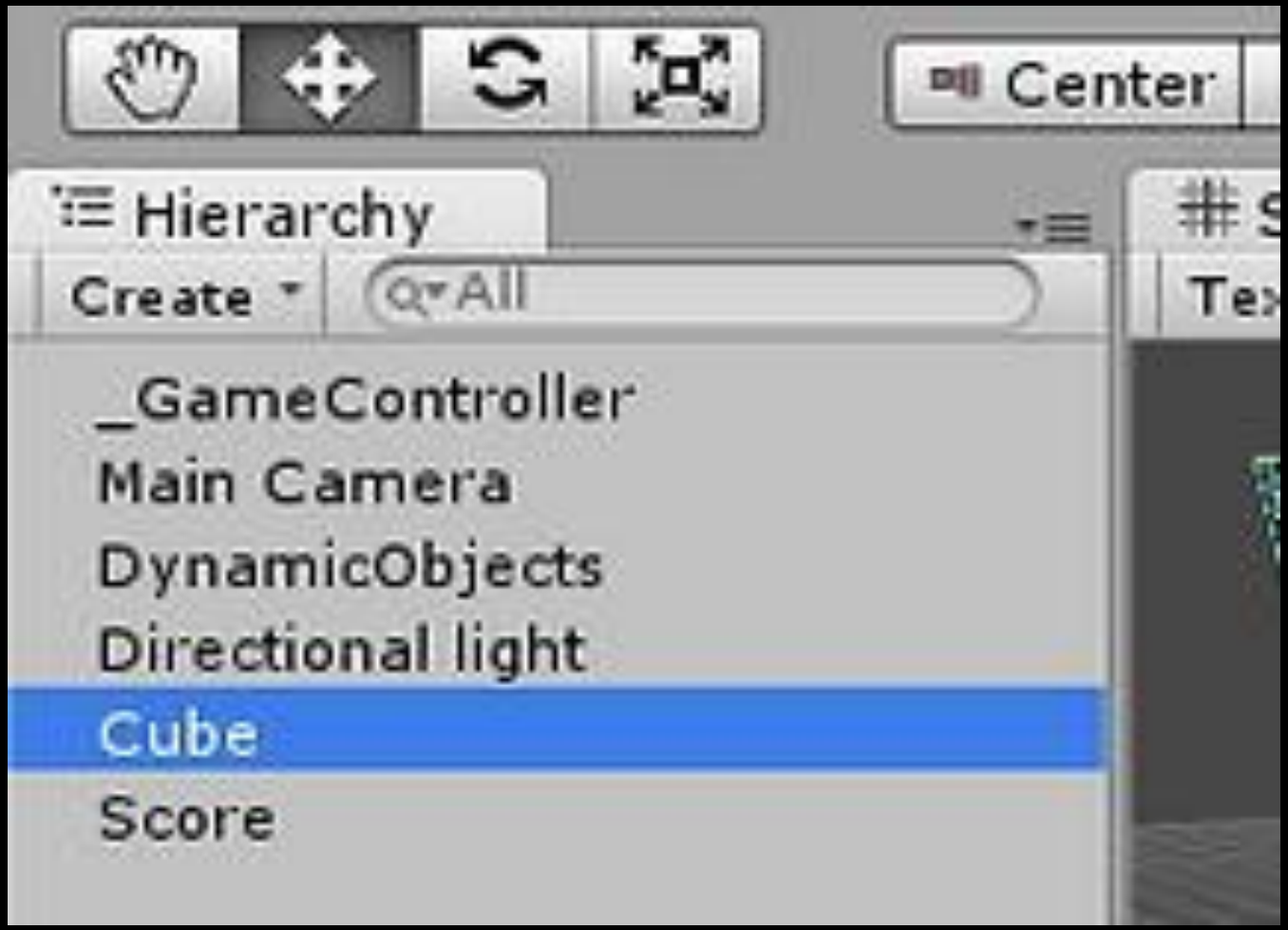
GAMEHUB University-Enterprises Cooperation
In Game Industry In Ukraine

# Огляд проекту (Project View)

Co-funded by the Erasmus+ Programme of the European Union



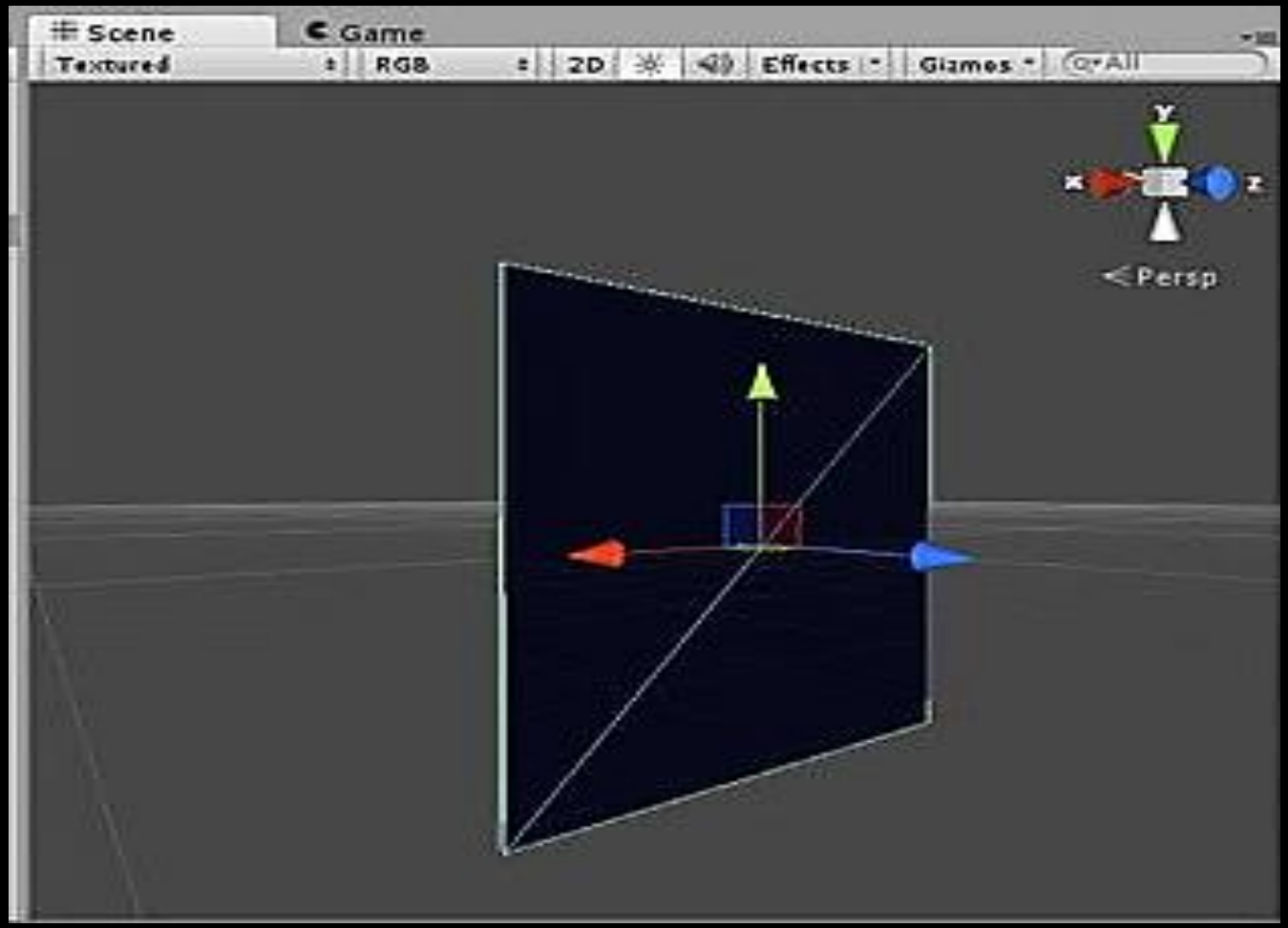
# Ієрархія (Hierarchy)



Co-funded by the
Erasmus+ Programme
of the European Union



# Сцена (Scene View)



Co-funded by the
Erasmus+ Programme
of the European Union

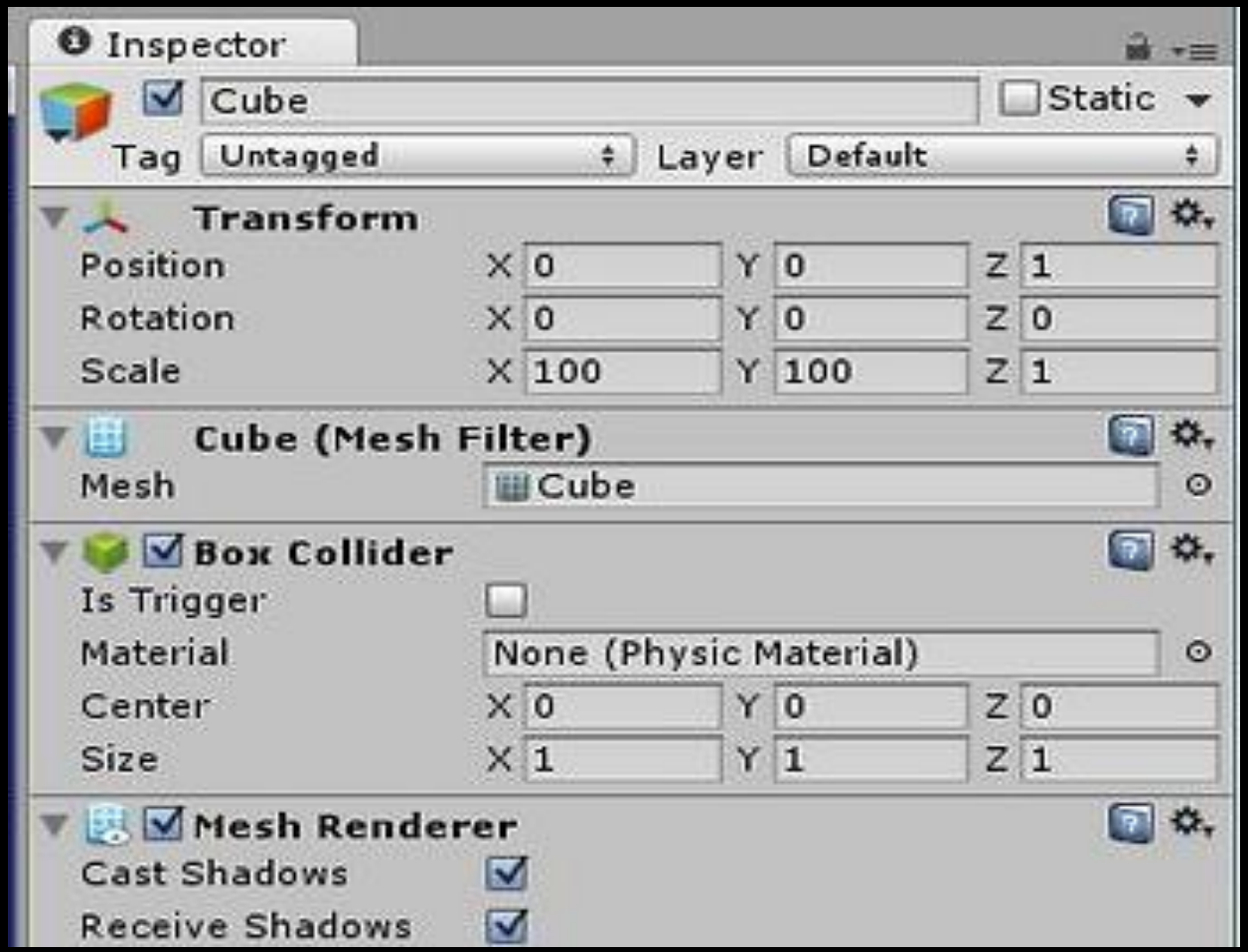


# Ігровий вид (Game View)

Co-funded by the Erasmus+ Programme of the European Union



# Інспектор (Inspector)



Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Що ми вивчили?

**Основні ресурси:**

- Основні інструментальні засоби створення ігрових додатків.
- Порівняльний аналіз та огляд сучасних движків для створення ігор.
- Основні елементи та структура інтерфейсу Unity 3D.
- Налаштування інтерфейсу Unity 3D.

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Що ми вивчили?

**Класифікація стандартних об’єктів інтерфейсу.**

Параметри та призначення:

- ассетів.
- префабів.
- текстур.
- матеріалів.

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Студент має :

- Пояснити роль комп'ютерної графіки у створення ігрових додатку.
- Охарактеризувати параметри та основні функціональні можливості та відмінності існуючих ігрових движків.
- Знати етапи та послідовність створення 3D гри
- Знати графічні основні технології створення ігор та пояснити їх відмінності, переваги, недоліки.
- Надати визначення 3D гри.

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Студент має:

- Unity 3D. Знати структуру та призначення робочих областей інтерфейсу.
- Unity 3D. Знати правила іменування об’єктів / ресурсів гри відповідно до прийнятих правил іменування..
- Unity 3D. Вказати порядок операцій при створенні графічного зображення ассету (трансформація, обробка, анімація).

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Заключення. Лабораторія

Лабораторна робота №1. Створення простого ігрового середовища засобами Unity.

Дивись «Методичні вказівки до виконання лабораторних робіт»

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.
# Рекомендовані ресурси.

## Документація

Офіційний сайт Unity3d https://unity3d.com/ru

https://docs.unity3d.com/ru/

Sue Blackman. Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine, 2011. – 986 p.

Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.

## *Дякую!*
## *Питання?*

***The lection was performed with support of the Erasmus+ Programme of the European Union (561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP). The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Donetsk National Technical University

Модуль «Розробка ігрових додатків на базі движка Unity»

Асистент Скрипник Т.В.

**DonNTU GameHub**

***561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Лекція 1. Частина 2.

- Основи роботи в середовищі Unity.
- Створення простого ігрового середовища засобами Unity.

Co-funded by the Erasmus+ Programme of the European Union



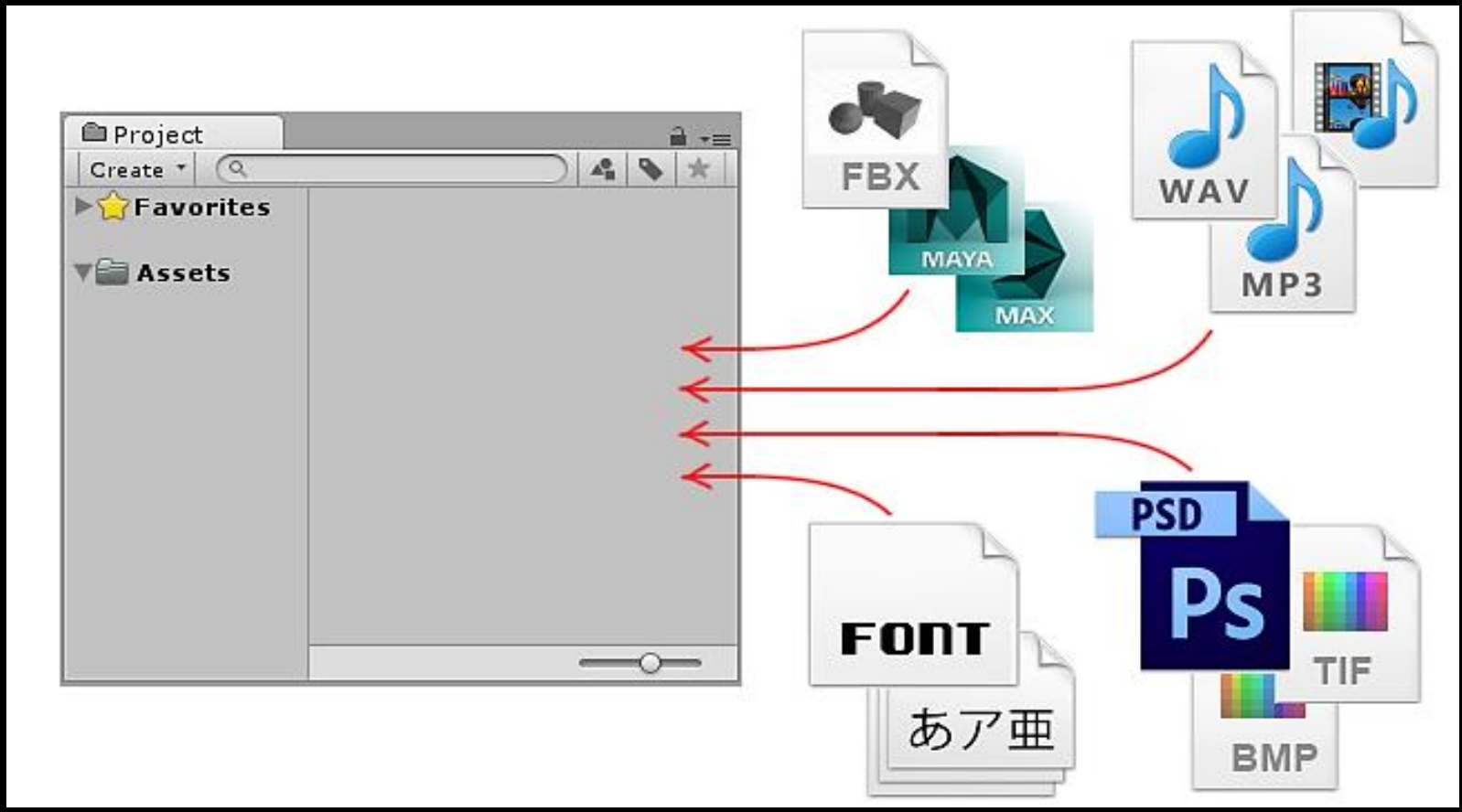
# Лекція 1. Частина 2.

- Середовище Unity: основні об’єкти та елементи інтерфейсу
- Класифікація і опис властивостей стандартних об’єктів Unity.
- Робота з ассетами.

Co-funded by the
Erasmus+ Programme
of the European Union



# Робота з ассетами (Asset Workflow)



Co-funded by the Erasmus+ Programme of the European Union



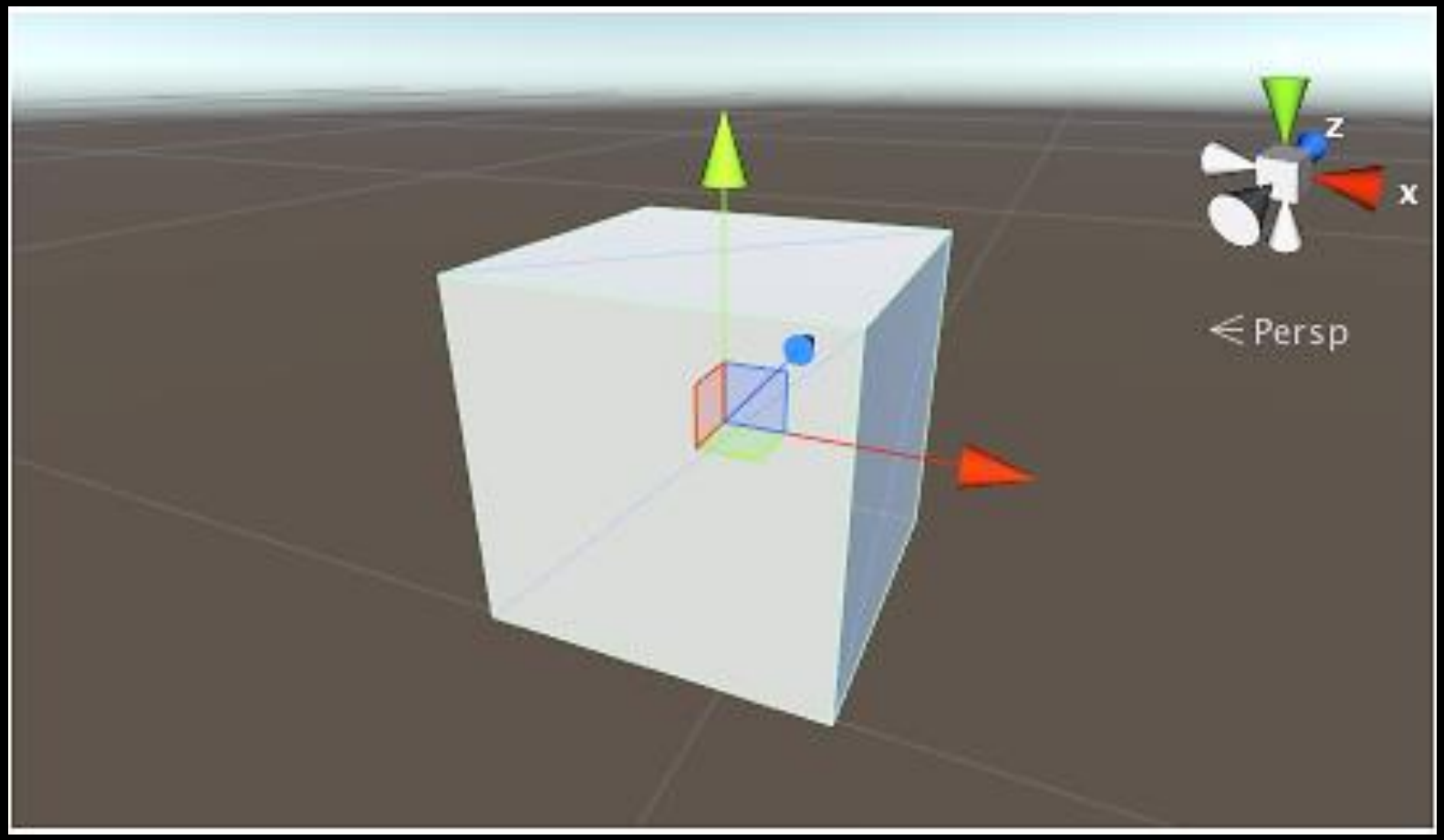
# Ассети: Примітиви

- Куб (Cube)
- Сфера (Sphere)
- Капсула (Capsule)
- Циліндр (Cylinder)
- Площина (Plane)
- Квадрат (Quad)

Co-funded by the
Erasmus+ Programme
of the European Union



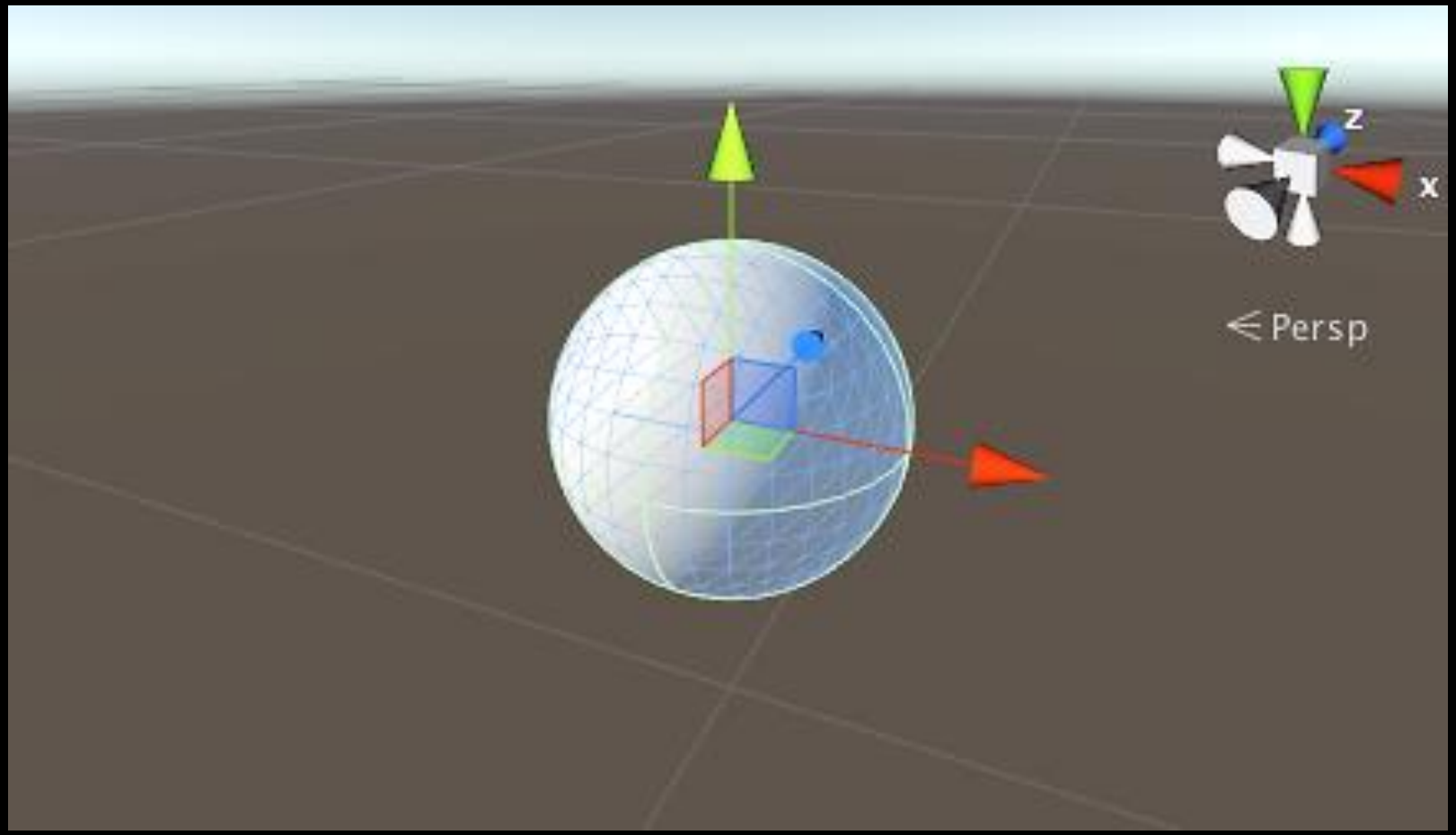
# Примітиви: Куб



Co-funded by the
Erasmus+ Programme
of the European Union



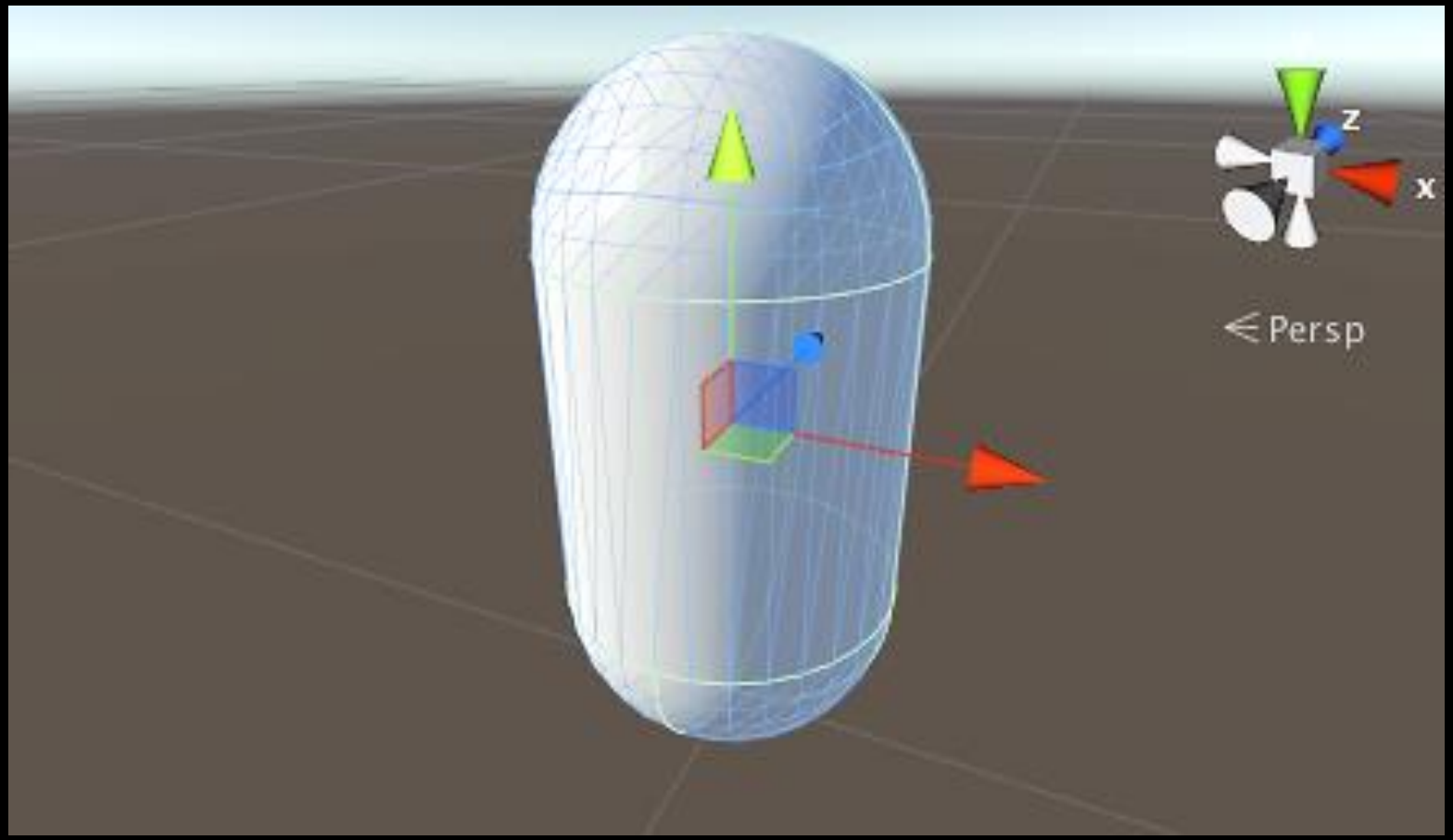
# Примітиви: Сфера



Co-funded by the
Erasmus+ Programme
of the European Union



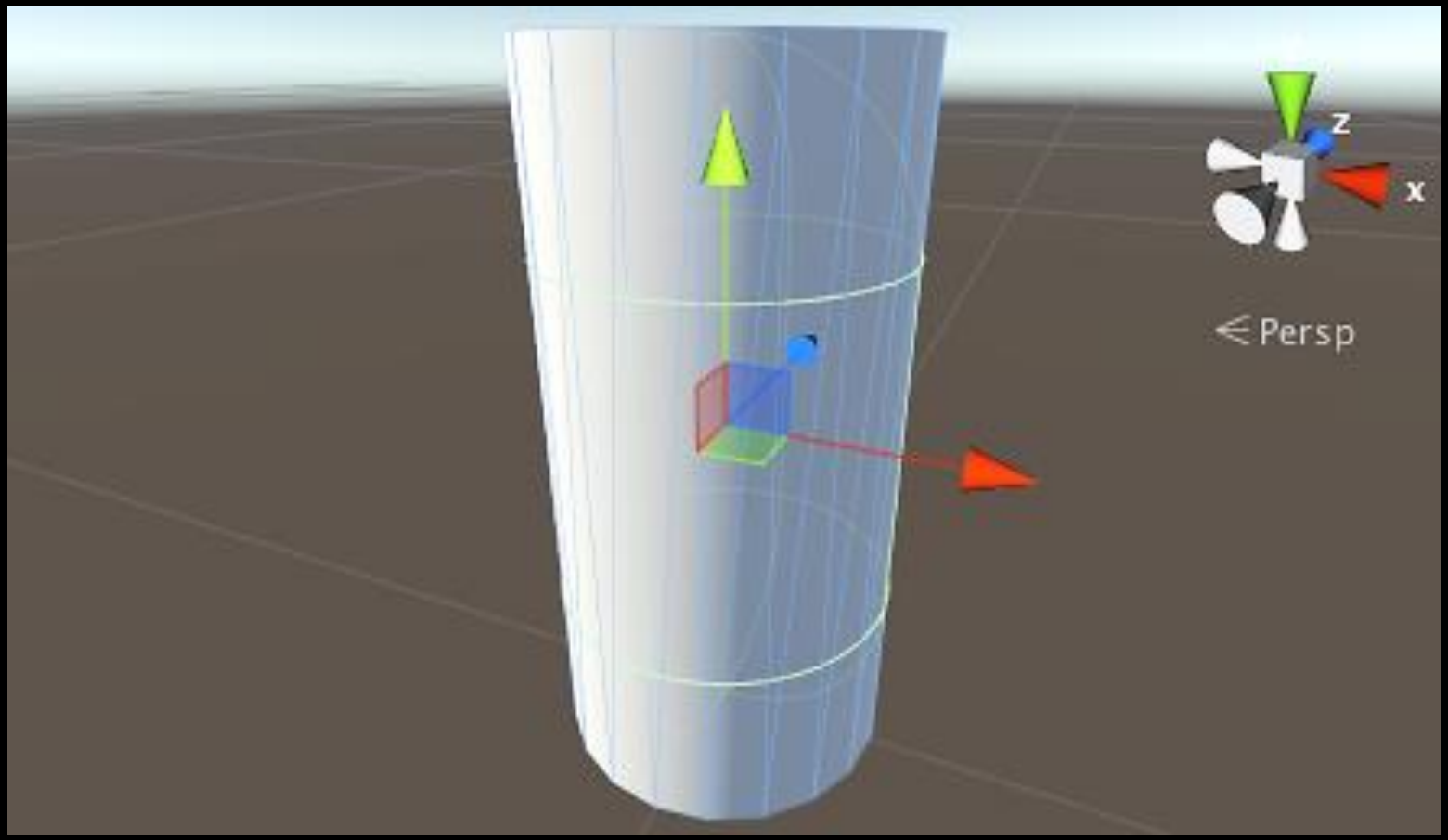
# Примітиви: Капсула



Co-funded by the
Erasmus+ Programme
of the European Union



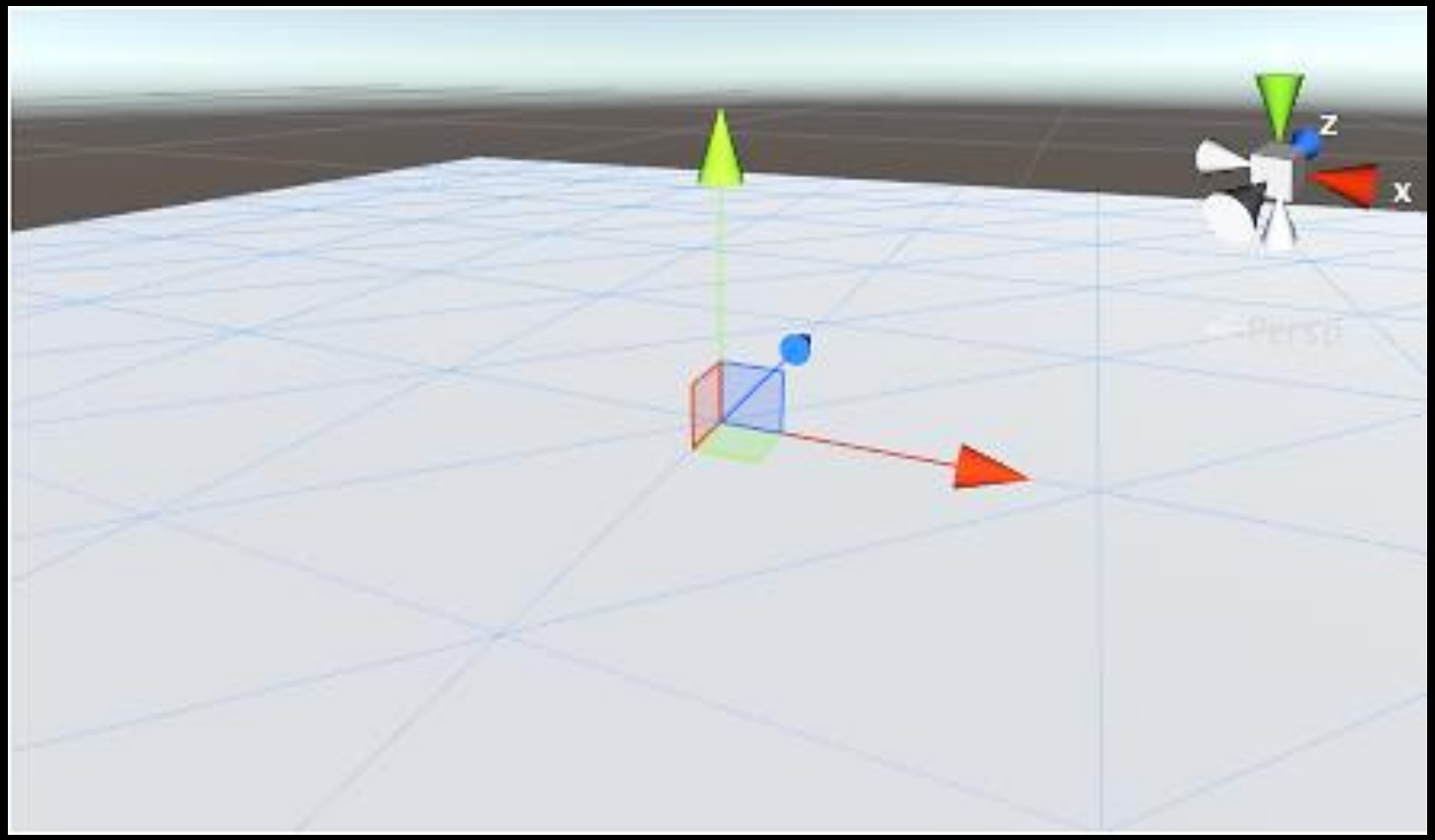
# Примітиви: Циліндр



Co-funded by the
Erasmus+ Programme
of the European Union



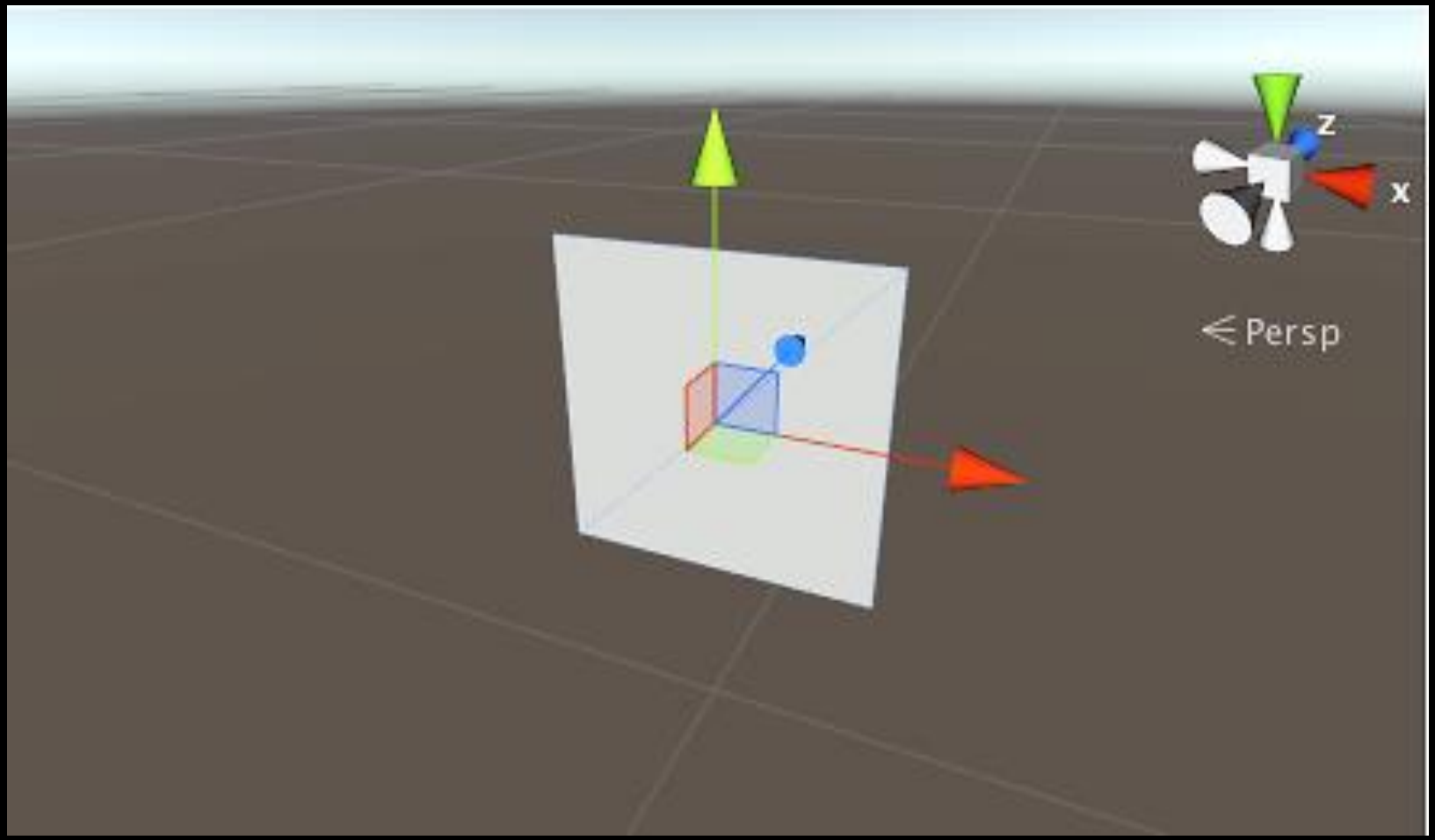
# Примітиви: Площина



Co-funded by the
Erasmus+ Programme
of the European Union



# Примітиви: Квадрат

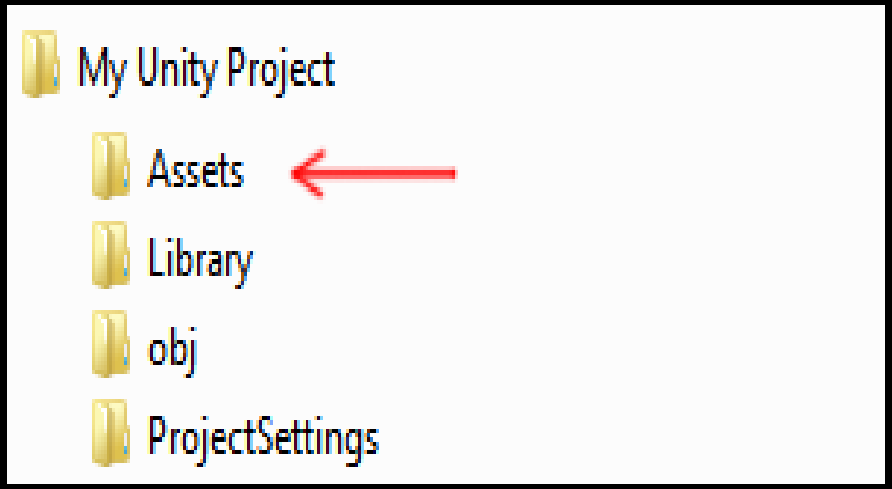


Co-funded by the Erasmus+ Programme of the European Union



# Імпорт ассетів

При створенні проекту Unity необхідно створити папку з ім'ям нового проекту, яка містить наступні піддиректорії:

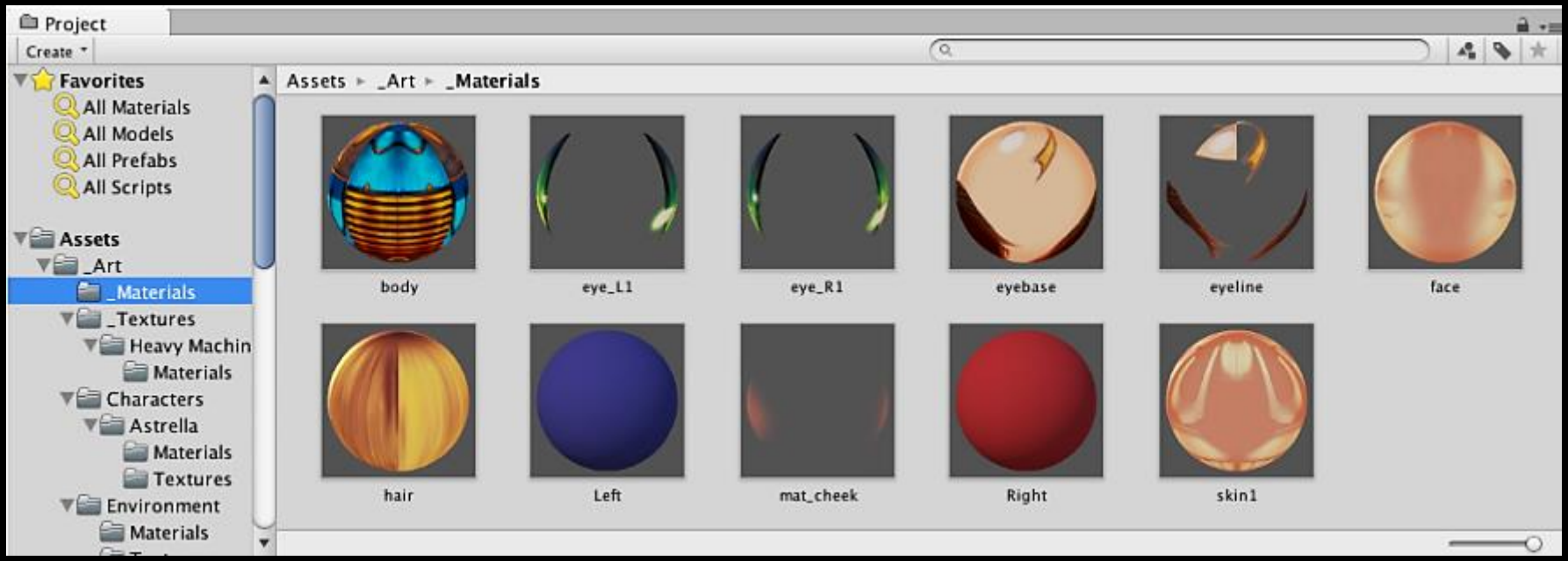



Це основна файлова структура проекту Unity

Co-funded by the
Erasmus+ Programme
of the European Union

GAMEHUB University-Enterprises Cooperation
In Game Industry In Ukraine

# Імпорт ассетів: Вікно Project




Вміст вікна Project в Unity показує елементи в папці «Assets» .

Co-funded by the Erasmus+ Programme of the European Union



# Ассети: Загальні типи ассетів

- Файли зображень
- Звукові файли
- Меші та анімації
- Аудіофайли

Co-funded by the Erasmus+ Programme of the European Union



# Ассети: Префаби

Це особливий тип асету, що дозволяє зберігати весь ігровий об’єкт (GameObject) з усіма компонентами і значеннями властивостей. Префаб виступає у ролі шаблону для створення екземплярів об'єкта, що зберігається в сцені. Будь-які зміни в префабі негайно відображаються і на всіх його примірниках, при цьому можна перевизначати компоненти і налаштування для кожного екземпляра окремо.

Створити префаб можна, обравши Asset → Create Prefab і перетягнувши об'єкт зі сцени в "порожній" префаб, що з'явився в проекті.

Co-funded by the Erasmus+ Programme of the European Union



# Асети: Префаби

## Створення префабів в деяких основних ситуаціях:

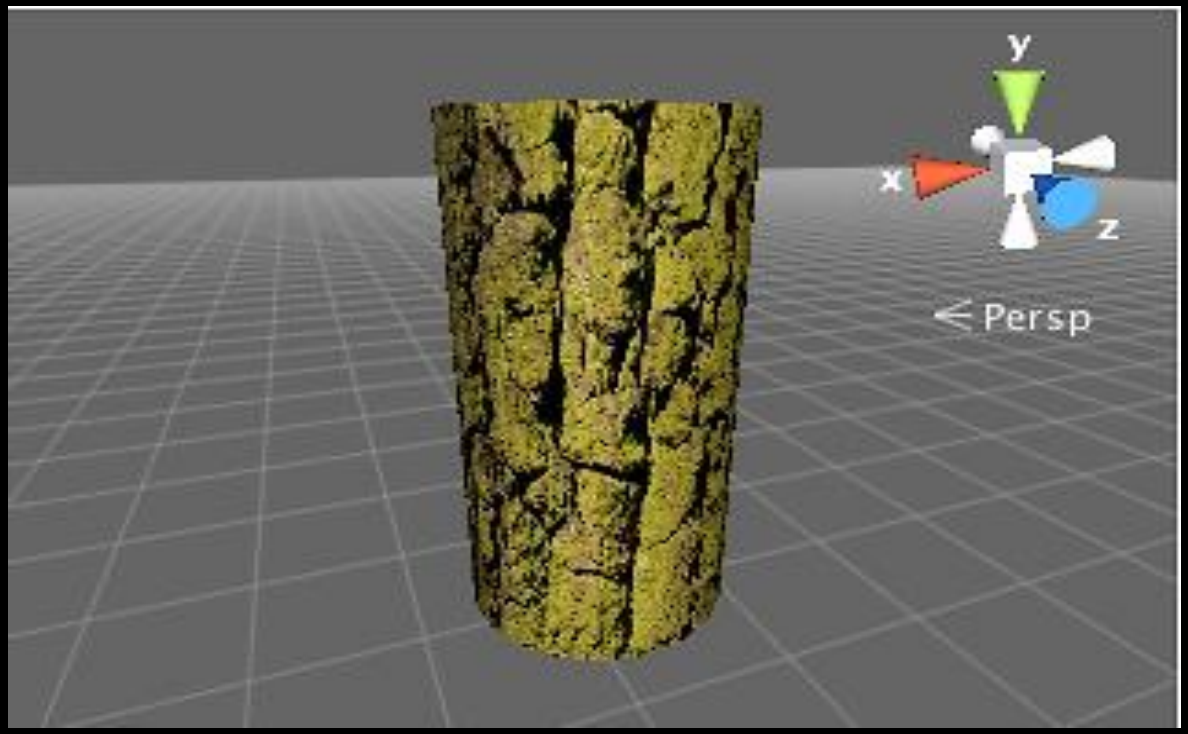
1. Побудова стіни з одного префабу "цеглини" шляхом створення його кілька разів в різних позиціях.
2. Ракетна установка створює екземпляр префабу ракети, коли користувач тисне кнопку атаки. Префаб містить меш, Rigidbody, коллайдер і дочірній GameObject, що має систему часток для сліду
3. Робот руйнується на шматки. Початковий цілий робот знищується і замінюється на префаб зламаного, що складатиметься з робота, розділеного на багато частин, маючих Rigidbody і системи часток. Ця техніка дозволяє підірвати робота на дрібні шматочки одним рядком коду, замінюючи один об'єкт на префаб.

Co-funded by the Erasmus+ Programme of the European Union



# Текстури та відео. Текстури для 3D моделей

Текстура - це стандартне растрове зображення, яке накладається зверху на поверхню мешу(сітки).




Циліндр з корою дерева

Co-funded by the Erasmus+ Programme of the European Union



# Текстури для 3D моделей

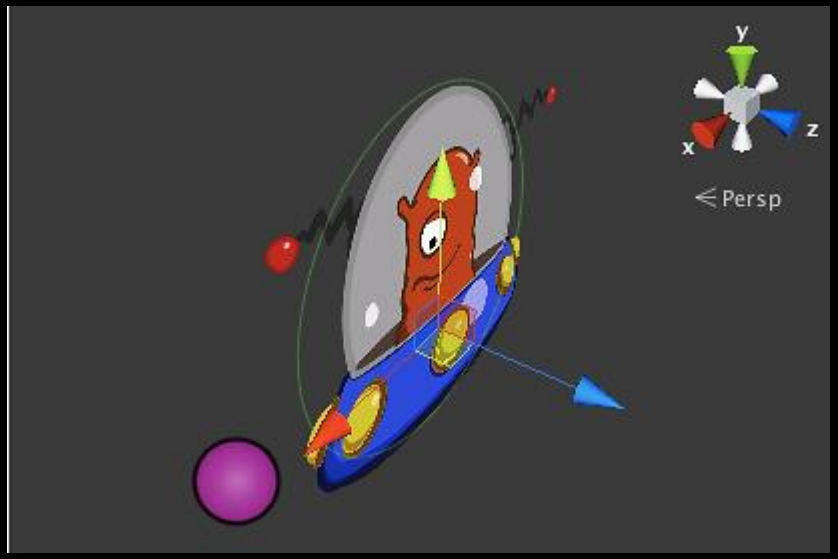
Текстури, що створюються, повинні мати розміри, що являють собою степені двійки (наприклад, 32x32, 64x64, 128x128, 256x256 і т. д.). Просто розмістіть їх в папку Ассети вашого проекта, цього буде достатньо, щоб потім вони з'явились в вікні Project View.

Co-funded by the Erasmus+ Programme of the European Union



# Текстури для 2D моделей

В 2D-іграх Sprites реалізовані з використанням текстур, що застосовуються до пласких мешів, що апроксимують форми об'єктів.



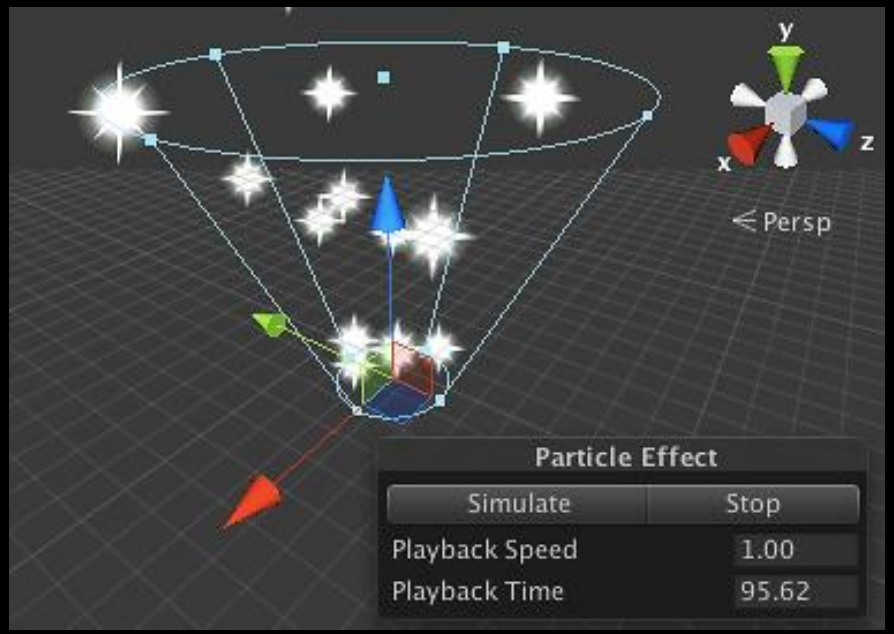

Спрайт з 3D-точки зору

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Частинки

**Частинки (Particle Systems)** – невеликий графічний 2D об'єкт, який представляє собою невелику частину чогось, що є в основному рідиною або газом, наприклад, як хмара диму.



Co-funded by the Erasmus+ Programme of the European Union



# Shader assets. Шейдерні ассети

Шейдери - це ассети, що містять код і інструкції для виконання графічної карти. Матеріали посилаються на шейдери і налаштовують їх параметри (текстури, колір).

Unity містить вбудовані шейдери, що завжди доступні в проекті (наприклад, стандартний шейдер - Standard shader).

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Що ми вивчили?

## Основні ресурси:

**Класифікація стандартних об’єктів інтерфейсу.**

Параметри та призначення:

- ассетів.
- префабів.
- текстур.
- матеріалів.

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Студент має:

- Unity 3D. Знати правила іменувати об’єктів / ресурсів гри відповідно до прийнятих правил іменування.
- Unity 3D. Вказати порядок операцій при створенні графічного зображення ассету (трансформація, обробка, анімація).
- Unity 3D. Знати та вміти пояснити призначення властивостей ассету.

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Лабораторія

Лабораторна робота №1. Створення простого ігрового середовища засобами Unity.

Дивись «Методичні вказівки до виконання лабораторних робіт»

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.
# Рекомендовані ресурси.

## Документація

Офіційний сайт Unity3d https://unity3d.com/ru

https://docs.unity3d.com/ru/

Sue Blackman. Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine, 2011. – 986 p.

Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.

## *Дякую!*
## *Питання?*

***The lection was performed with support of the Erasmus+ Programme of the European Union (561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP). The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Donetsk National Technical University

Модуль «Розробка ігрових додатків на базі движка Unity»

Асистент Скрипник Т.В.

**DonNTU GameHub**

***561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Лекція 2. Частина 1.

- Організація геймплея за допомогою скриптів в Unity.
- Динамічне створення об’єктів.

Co-funded by the Erasmus+ Programme of the European Union



# Лекція 2. Частина 1.

- Базові класи та методи об’єктів Unity, що представлені мовою C#.
- Засоби прив’язки скриптів до об’єктів та подій ігрового середовища.
- Можливості Unity з динамічного створення об’єктів в ході гри.

Co-funded by the Erasmus+ Programme of the European Union



# Скриптинг. Основні поняття.

Co-funded by the Erasmus+ Programme of the European Union



# Скрипти

## Призначення:

- Дозволяють активізувати ігрові події
- Дозволяють змінювати параметри компонентів
- Забезпечують можливість організації введення даних користувачем будь-яким чином

## Мови програмування:

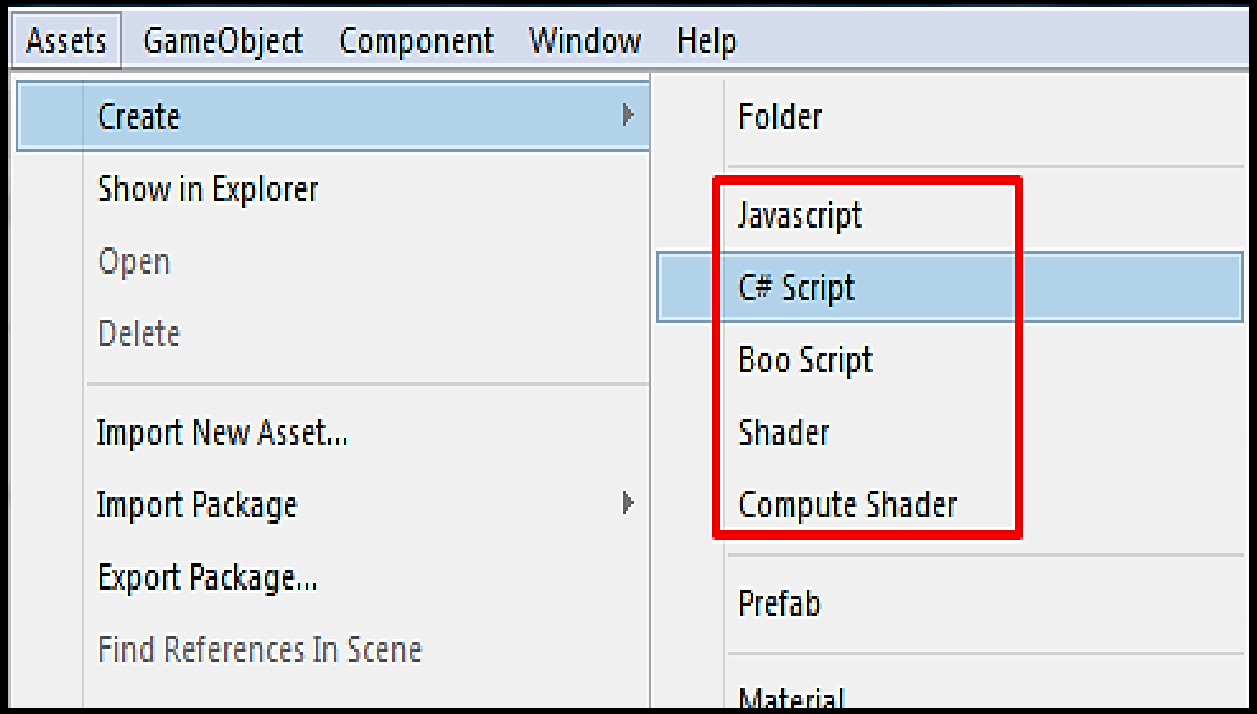
- C#
- UnityScript (за зразком JavaScript)
- Boo, інші мови, що здатні генерувати DLL

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Створення скриптів

Скрипти можна створювати безпосередньо в Unity, вибравши **Assets → Create → C# Script (або JavaScript / Boo скрипт)** в головному меню.



Co-funded by the Erasmus+ Programme of the European Union



# Загальна структура файлу скрипту на C#

```
using UnityEngine;
using System.Collections;

public class MainPlayer : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {

    }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Загальна структура файлу скрипту

- **MonoBehaviour**–базовий вбудований клас
- **Функція Start** –викликається Unity до початку ігрового процесу, може бути використана для ініціалізації
- **Функція Update** –відповідає за обробку оновлення кадру ігрового об’єкту

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Загальна структура файлу скрипту UnityScript

```
#pragma strict

function Start () {

}

function Update () {

}
```

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Керування ігровим об’єктом

Екземпляр скрипту виглядає так само, як і інші компоненти у вікні Inspector: -


Приклад коду у функції Start: -

```
// Use this for initialization
void Start () {
    Debug.Log("I am alive!");
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Основні поняття і принципи скриптингу

- **Команди**
- Змінні
- **Область видимості змінних**
- Оператори
- Консоль Unity

Co-funded by the Erasmus+ Programme of the European Union



# Змінні. Типи даних

Найбільш уживані типи змінних:

- int,
- float,
- bool,
- String
- Vector3

Co-funded by the Erasmus+ Programme of the European Union



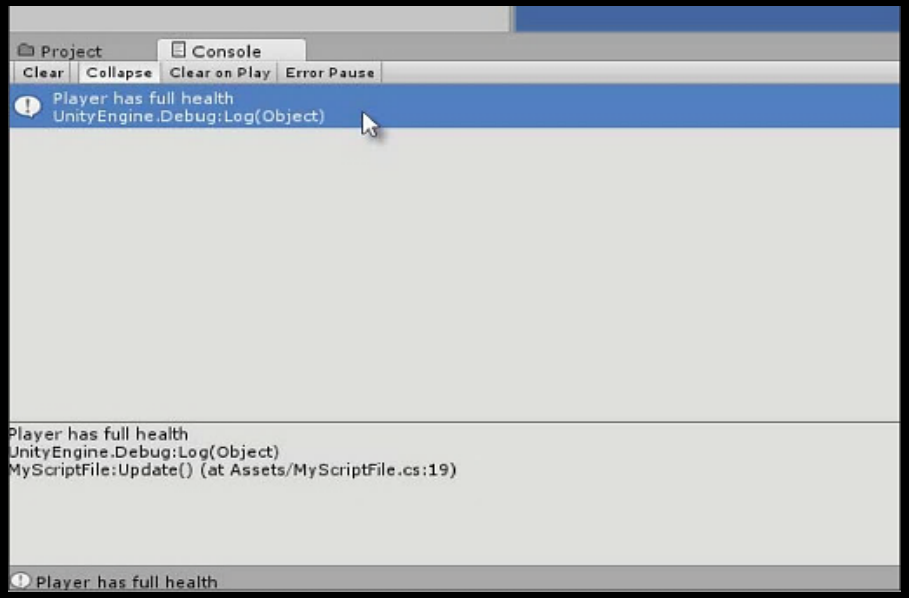
# Приклад коду скрипту

```
01  using UnityEngine;
02  using System.Collections;
03
04  public class MyNewScript: MonoBehaviour
05 {
06  public string PlayerName = "";
07  public int PlayerHealth = 100;
08  public Vector3 Position = Vector3.zero;
09
10  // Використовується це для ініціалізації
11  void Start () {
12
13  }
14
15  // Оновлення викликається один раз за кадр
16  void Update () {
17
18  }
19  }
```

Co-funded by the Erasmus+ Programme of the European Union



# Консоль Unity




Консоль Unity зручно використовувати для виводу повідомлень в процесі налагодження.

За допомогою оператора «!=» можна перевірити нерівність змінної заданому значенню. Крім того, можна навіть об'єднати декілька перевірок за допомогою операторів && (І) та || (АБО). Наприклад, наступний оператор if виконує блок коду між дужками {}, тільки якщо значення змінної PlayerHealth знаходиться в інтервалі між 0 і 100, і не дорівнює 50:

```
if (PlayerHealth >= 0 && PlayerHealth <= 100 && PlayerHealth !=50)
{
Debug.log   ("Player has  full  health");
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Що ми вивчили?

**Основні ресурси:**

- Основні базові класи та методи об’єктів Unity
- Основні поняття скриптингу.
- Створення скриптів і їхня прив’язка до об’єктів.
- Консоль Unity.

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Студент має :

- Пояснити роль скриптів при створенні ігрових додатків за допомогою Unity.
- Знати основні класи та методи мови C# для роботи з об’єктами Unity.
- Знати про особливості роботи з об’єктами, що динамічно створюються в процесі гри

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Заключення. Лабораторія

Лабораторна робота №2. Використання скриптів Unity для створення геймплея.

Дивись «Методичні вказівки до виконання лабораторних робіт»

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D. Рекомендовані ресурси.

## Документація

Офіційний сайт Unity3d https://unity3d.com/ru

https://docs.unity3d.com/ru/

Sue Blackman. Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine, 2011. – 986 p.

Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.

## *Дякую!*
## *Питання?*

***The lection was performed with support of the Erasmus+ Programme of the European Union (561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP). The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Donetsk National Technical University

Модуль «Розробка ігрових додатків на базі движка Unity»

Асистент Скрипник Т.В.

**DonNTU GameHub**

***561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Лекція 2. Частина 2.

- Організація геймплея за допомогою скриптів в Unity.
- Динамічне створення об’єктів.

Co-funded by the Erasmus+ Programme of the European Union



# Лекція 2. Частина 2.

- Базові класи та методи об’єктів Unity, що представлені мовою C#.
- Засоби прив’язки скриптів до об’єктів та подій ігрового середовища.
- Можливості Unity з динамічного створення об’єктів в ході гри.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Керування ігровими об'єктами (GameObjects) за допомогою компонентів

Звернення до компонентів за допомогою функції GetComponent:

```
void Start () {
    Rigidbody rb = GetComponent<Rigidbody>();
}
```

В UnityScript синтаксис дещо відрізняється:

```
function Start () {
    var rb = GetComponent.<Rigidbody>();
}
```

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Функції подій. Звичайні Update події.

## Функція Update

Гра - це щось на зразок анімації, де кадри генеруються на ходу. Ключовий концепт в програмуванні ігор полягає в зміні позиції, стану і поведінки об'єктів в грі прямо перед відмальовкою кадру. Такий код в Unity зазвичай розміщують у функції Update. Update викликається перед відмальовкою кадру і перед розрахунком анімацій.

```
void Update() {
    float distance = speed * Time.deltaTime * Input.GetAxis("Horizontal");
    transform.Translate(Vector3.right * distance);
}
```

Co-funded by the
Erasmus+ Programme
of the European Union

GAMEHUB University-Enterprises Cooperation
In Game Industry In Ukraine

# Функції подій: Функція FixedUpdate

Фізичний движок також оновлюється фіксованими за часом кроками, аналогічно тому як працює відмальовка кадру. Окрема функція події FixedUpdate викликається прямо перед кожним оновленням фізичних даних. Оскільки оновлення фізики і кадру відбувається не з однаковою частотою, то виходять більш точні результати від коду фізики, якщо помістити його в функцію FixedUpdate, а не в Update.

```
void FixedUpdate() {
    Vector3 force = transform.forward * driveForce * Input.GetAxis("Vertical");
    rigidbody.AddForce(force);
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Функції подій: Функція LateUpdate

У ситуації, коли код скрипта повинен перевизначити ефект анімації (припустимо, змусити голову персонажа повернутися до цільового об'єкту в сцені), то тоді можна використовувати функцію LateUpdate.

```
void LateUpdate() {
    Camera.main.transform.LookAt(target.transform);
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Функції подій: Функція OnGUI

У Unity є система для відтворення елементів управління GUI поверх всього, що відбувається в сцені і реагування на кліки по цих елементах. Цей код обробляється трохи інакше, ніж звичайне оновлення кадру, так що він повинен бути поміщений в функцію OnGUI, яка буде періодично викликатися.

```
void OnGUI() {
    GUI.Label(labelRect, "Game Over");
}
```

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Функції подій: події фізики

- OnCollisionEnter
- OnCollisionStay
- OnCollisionExit
- OnTriggerEnter
- OnTriggerStay
- OnTriggerExit

```
void OnCollisionEnter(otherObj: Collision) {
    if (otherObj.tag == "Arrow") {
        ApplyDamage(10);
    }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Керування часом і кадрами

Задачу переміщення об'єкта вперед поступово, по одному кадру за раз

```
//C# script example
using UnityEngine;
using System.Collections;

public class ExampleScript : MonoBehaviour {
    public float distancePerFrame;

    void Update() {
        transform.Translate(0, 0, distancePerFrame);
    }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Керування часом і кадрами

## Приклад масштабування розміру руху на час кадру

```
//C# script example
using UnityEngine;
using System.Collections;

public class ExampleScript : MonoBehaviour {
    public float distancePerSecond;

    void Update() {
        transform.Translate(0, 0, distancePerSecond * Time.deltaTime);
    }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Створення і знищення ігрових об'єктів

Деякі ігри мають постійну кількість об'єктів на сцені, але зазвичай персонажі, скарби та інші об'єкти створюються і видаляються під час гри. У Unity ігровий об'єкт (GameObject) створюється за допомогою функції Instantiate, що копіює існуючий об'єкт.

```
public GameObject enemy;

void Start() {
    for (int i = 0; i < 5; i++) {
        Instantiate(enemy);
    }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Створення і знищення ігрових об'єктів

Функція Destroy знищує об'єкт після того, як завантаження кадру буде завершено або опціонально після короткої паузи: -

```
void OnCollisionEnter(Collision otherObj) {
    if (otherObj.gameObject.tag == "Missile") {
        Destroy(gameObject,.5f);
    }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Приклад скрипту для ігрового персонажу

Управління діями персонажа у ролі ворога за допомогою переліку (оператор **switch**)

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class MyScriptFile : MonoBehaviour
05 {
06   // Визначення можливих станів ворога у вигляді переліку
07   public enum EnemyState {CHASE, FLEE, FIGHT, HIDE};
08
09   // Поточний стан ворога
10   public EnemyState ActiveState = EnemyState.CHASE;
11
12   // Цей метод виконує ініціалізацію
13   void Start () {
14   }
15
```

Co-funded by the Erasmus+ Programme of the European Union



# Приклад скрипту (продовження)

```
16  // Виклик при відображенні кожного кадру
17  void Update ()
18  {
19   // Перевірка змінної ActiveState
20   switch(ActiveState)
21   {
22    case EnemyState.FIGHT:
23     {
24              // Реалізувати бій
25              Debug.log ("Entered fight state");
26          }
27          break;
28
29
30    case EnemyState.FLEE:
31    case EnemyState.HIDE:
32     {
33      // Втеча і засідка реалізуються однаково
34      Debug.log ("Entered flee or hide state");
```

Co-funded by the Erasmus+ Programme of the European Union



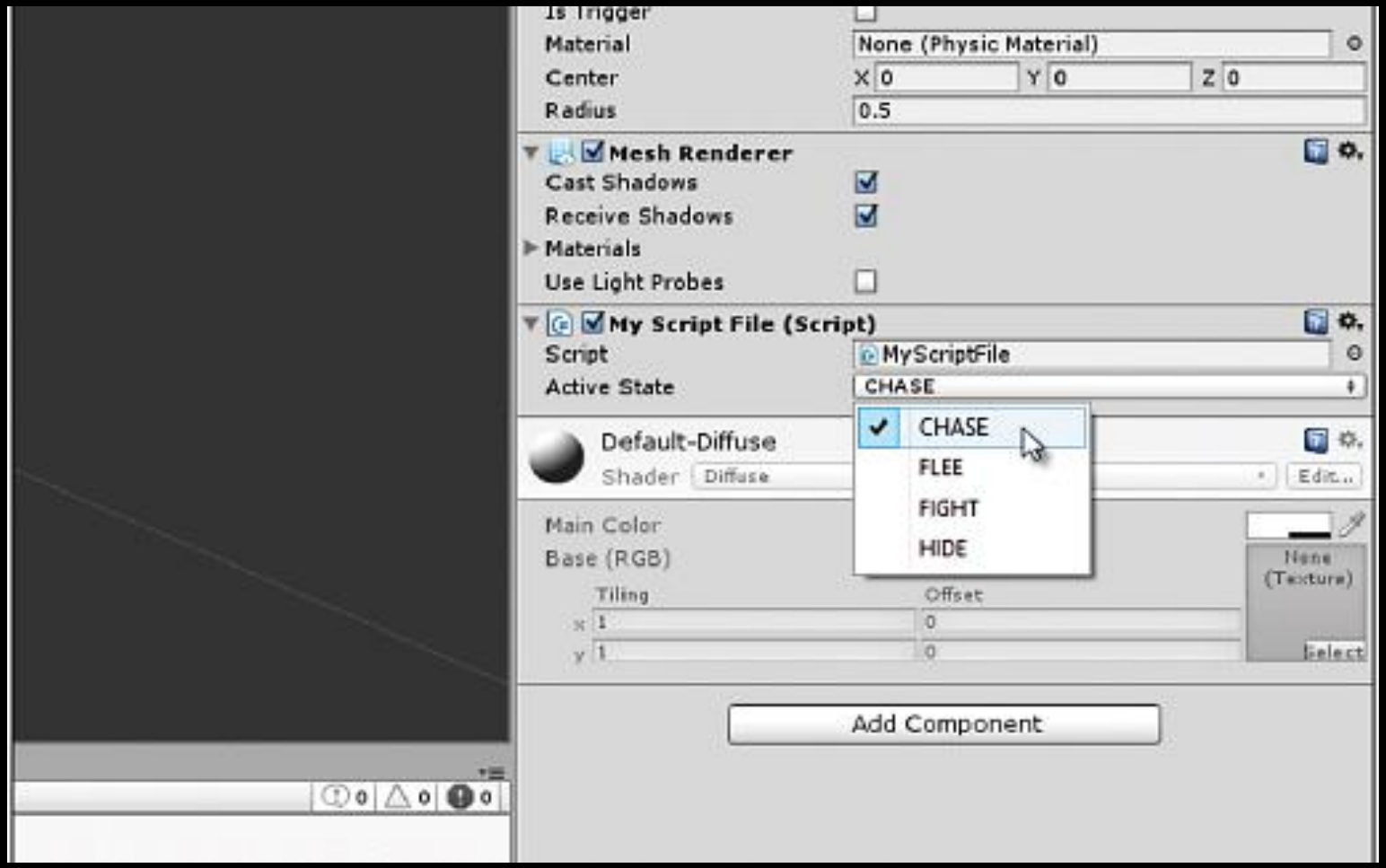
# Приклад скрипту (продовження)

```
35      }
36     break;
37
38     default:
39      {
40      // Випадок за замовчуванням, коли ніякий інший випадок
41      // не підходить. Зараз обробляється стан гонитви.
42      Debug.log ("Entered chase state");
43      }
44     break;
45    }
46  }
47 }
```

Co-funded by the
Erasmus+ Programme
of the European Union



# Інспектор об’єктів зі списком перерахувань



Co-funded by the Erasmus+ Programme of the European Union



# Спеціальні папки і порядок компіляції скриптів

Фази компіляції:

- Фаза 1: Виконуються скрипти з папок з іменами Standard Assets, Pro Standard Assets і Plugins.
- Фаза 2: Всі інші скрипти, які не перебувають в папці Editor.
- Фаза 3: Решта скриптів (тобто ті, що знаходяться в папці Editor).

Додатково, будь-який скрипт, що знаходиться в папці WebPlayerTemplates, в самому верху папки Assets, взагалі не буде скомпільовано. Ця поведінка трохи відрізняється для вкладених папок з іншими іменами (наприклад, Scripts / Editor працює як папка для скриптів редактора і Scripts / WebPlayerTemplates не завадить компіляції).

Co-funded by the Erasmus+ Programme of the European Union



# Платформно залежна компіляція

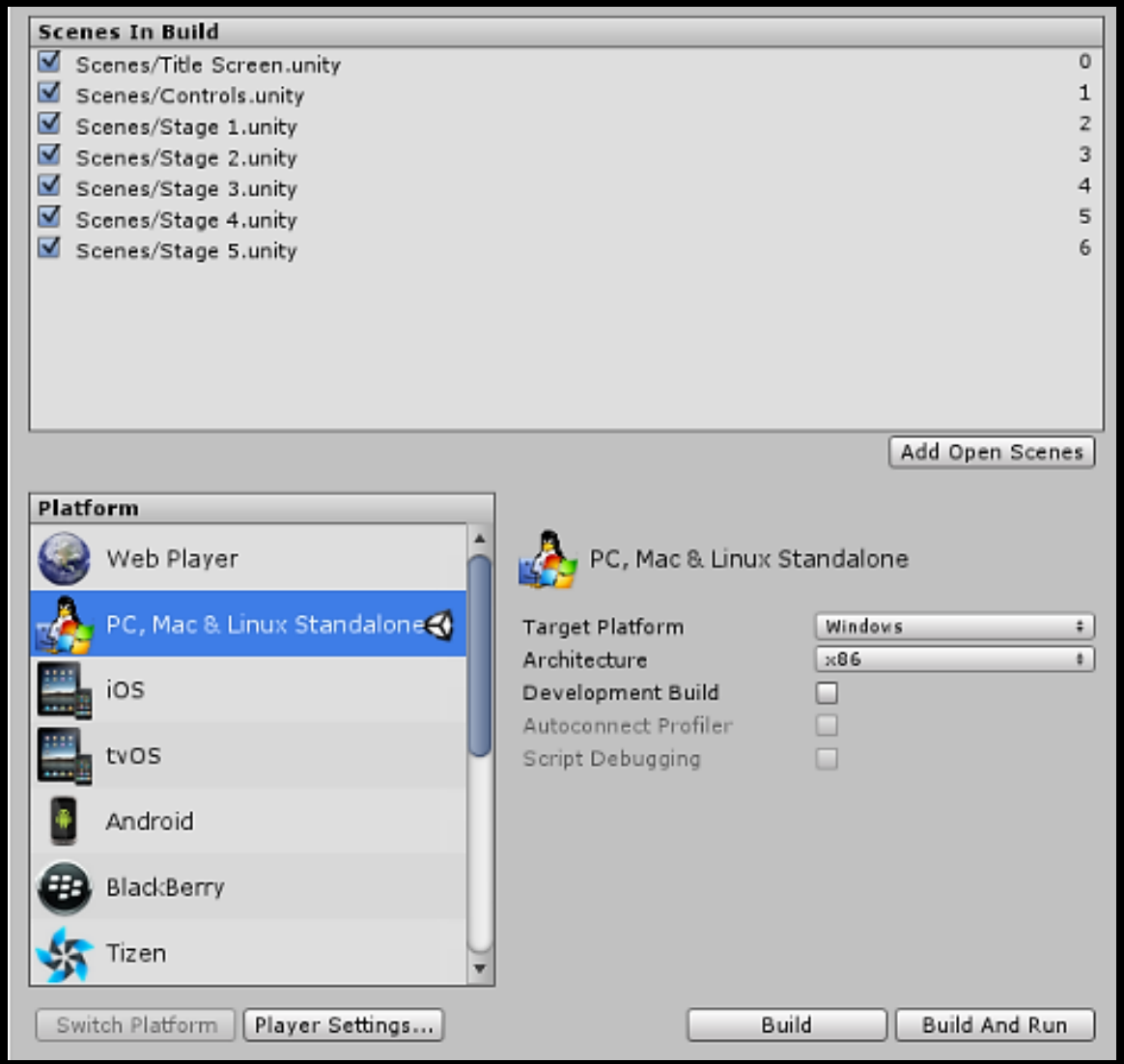
## Визначення платформ для скриптів, які підтримує Unity:

| Властивість: | Функція: |
|---|---|
| UNITY_EDITOR | Визначення для виклику скриптів редактора Unity з вашого ігрового коду. |
| UNITY_EDITOR_WIN | Визначення платформи для редактора коду на Windows. |
| UNITY_EDITOR_OSX | Визначення платформи для редактора коду на Windows. |
| UNITY_STANDALONE_OSX | Визначення платформи для компіляції / виконання коду спеціально для Mac OS (це включає в себе Universal, PPC і Intel архітектури). |
| UNITY_STANDALONE_WIN | Використовуйте, коли ви бажаєте скомпілювати / виконати код для програми на Windows. |
| UNITY_STANDALONE_LINUX | Використовуйте, коли ви бажаєте скомпілювати / виконати код для програми на Linux. |
| UNITY_STANDALONE | Використовуйте, коли ви бажаєте скомпілювати / виконати код для будь-якої платформи Mac, Windows або Linux. |
| UNITY_WII | Визначення платформи для компіляції / виконання коду для консолі Wii. |
| UNITY_IOS | Визначення платформи для компіляції / виконання коду для iPhone. |
| UNITY_ANDROID | Визначення платформи для Android. |
| … | … |
| UNITY_ANALYTICS | Визначення для виклику скриптів редактора Unity з вашого ігрового коду. |
| UNITY_ASSERTIONS | #define директива для процесу контролю за твердженнями. |

Co-funded by the Erasmus+ Programme of the European Union



# Тестування прекомпільованого коду.



Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Тестування прекомпільованого коду.

Створіть скрипт і скопіюйте цей код: -

```
// C#
using UnityEngine;
using System.Collections;

public class PlatformDefines : MonoBehaviour {
  void Start () {

    #if UNITY_EDITOR
      Debug.Log("Unity Editor");
    #endif

    #if UNITY_IOS
      Debug.Log("Iphone");
    #endif

    #if UNITY_STANDALONE_OSX
    Debug.Log("Stand Alone OSX");
    #endif

    #if UNITY_STANDALONE_WIN
      Debug.Log("Stand Alone Windows");
    #endif

  }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Що ми вивчили?

**Структура файлу скрипту.**

- команди.
- функції подій.
- компіляція, платформно залежна компіляція.
- тестування.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Заключення. Студент має :

- Пояснити роль скриптів при створенні ігрових додатків за допомогою Unity.
- Знати основні класи та методи мови C# для роботи з об’єктами Unity.
- Знати про особливості роботи з об’єктами, що динамічно створюються в процесі гри
- Знати про порядок компіляції скриптів
- Знати про особливості платформно залежної компіляції скриптів.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Заключення. Лабораторія

Лабораторна робота №2. Використання скриптів Unity для створення геймплея.

Дивись «Методичні вказівки до виконання лабораторних робіт»

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D. Рекомендовані ресурси.

## Документація

Офіційний сайт Unity3d https://unity3d.com/ru

https://docs.unity3d.com/ru/

Sue Blackman. Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine, 2011. – 986 p.

Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.

## *Дякую!*
## *Питання?*

***The lection was performed with support of the Erasmus+ Programme of the European Union (561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP). The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Donetsk National Technical University

Модуль «Розробка ігрових додатків на базі движка Unity»

Асистент Скрипник Т.В.

**DonNTU GameHub**

***561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP***

Co-funded by the Erasmus+ Programme of the European Union



# Лекція 3. Частина 1.

- Створення інтерактивного геймплея з персонажем за допомогою засобів движка Unity.
- Організація користувальницького інтерфейсу засобами Unity.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Лекція 3. Частина 1.

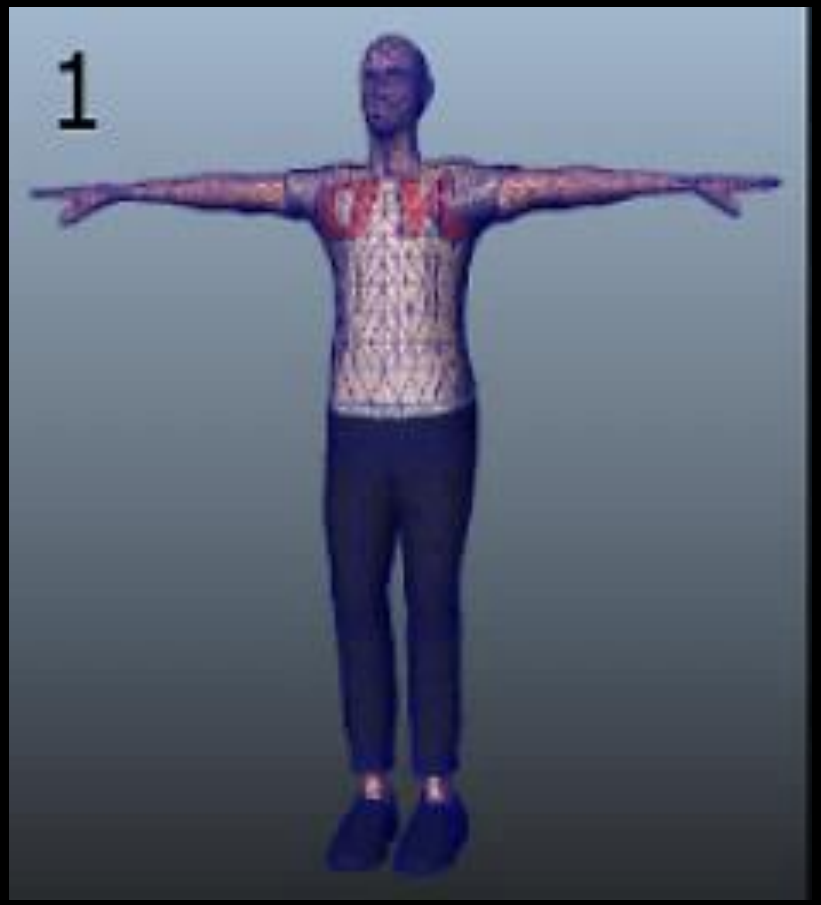
- Огляд вбудованих персонажів Unity, засоби створення власних персонажів та керування ними в процесі гри.
- Поняття тригера та особливості використання тригерів.

Co-funded by the Erasmus+ Programme of the European Union



# Використання Гуманоїдних Персонажів.

1. Модель персонажу, як правило, складається з багатокутників (полігонів) у 3D-пакеті або перетворюється в багатокутник або триангульовану сітку (меш) з більш складного типу сітчастих елементів перед експортом

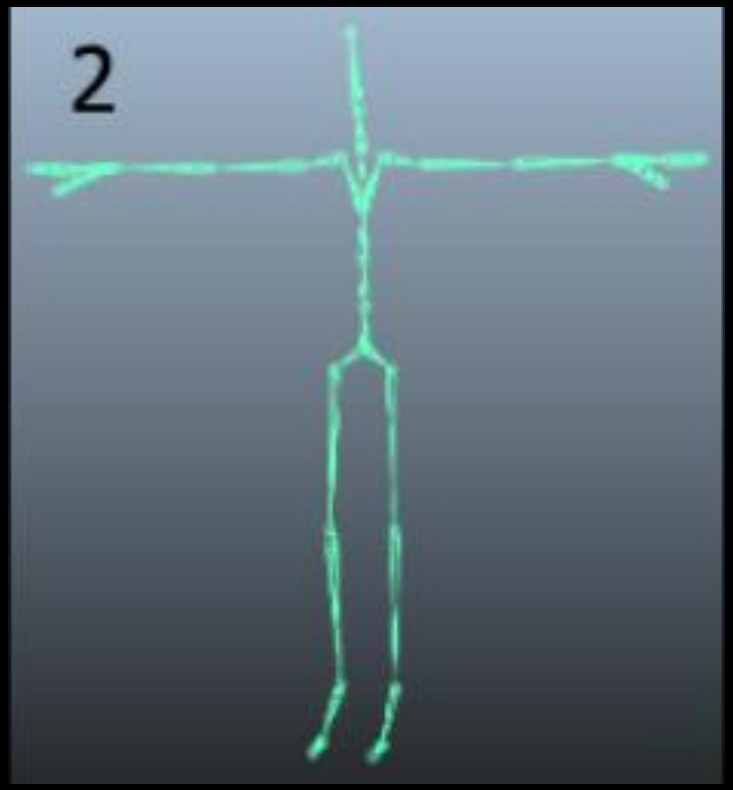


Co-funded by the Erasmus+ Programme of the European Union



# Використання Гуманоїдних Персонажів.

2. Для управління рухом персонажа повинна бути створена спільна ієрархія або скелет, який визначає кістки всередині мешу і їх рух по відношенню один до одного. Процес створення спільної ієрархії відомий як **ріггінг**

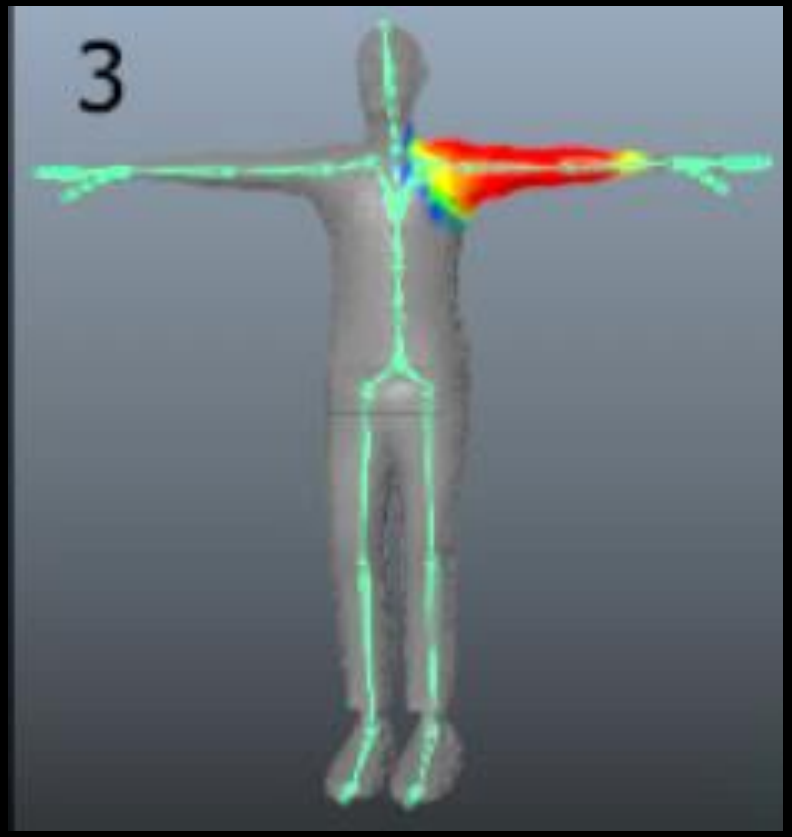


Co-funded by the Erasmus+ Programme of the European Union



# Використання Гуманоїдних Персонажів.

3. Потім меш або шкіра повинні бути підключені до ієрархії суглобів, щоб визначити, які частини мешу персонажу рухаються, коли анімований даний суглоб. Процес з'єднання скелета з сіткою відомий як **скінінг**.



Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Отримання гуманоїдних моделей.

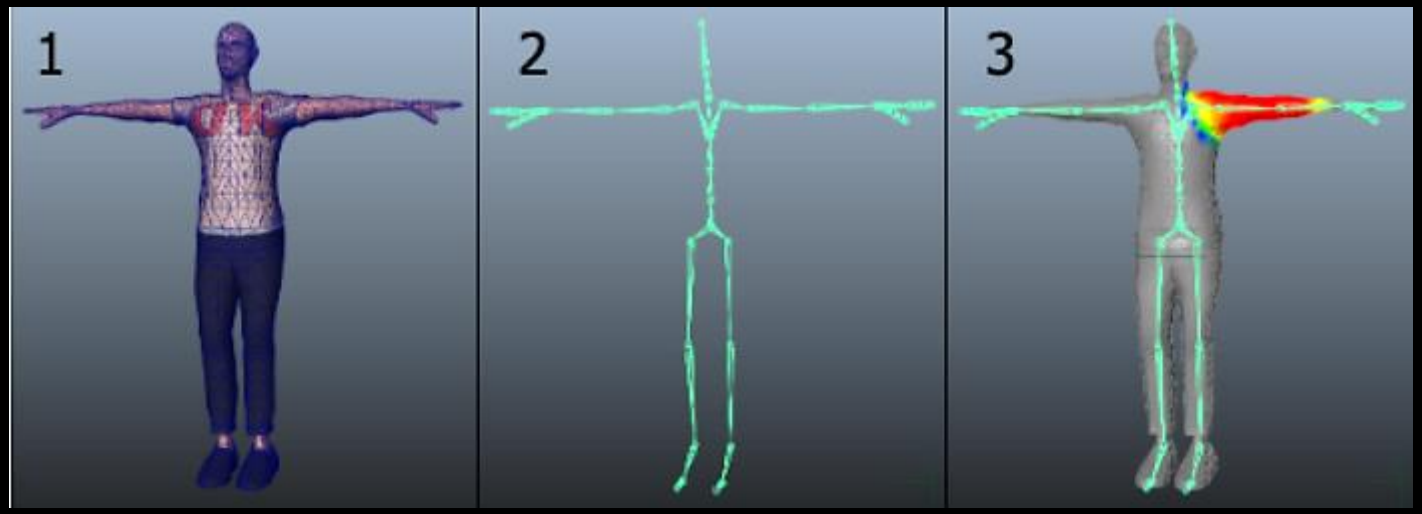
1. Використання процедурної системи персонажів або генератор персонажів, таких як Poser, Makehuman або Mixamo.
2. Придбання демонстраційних прикладів і вмістів персонажів з Unity Asset Store.
3. Підготовка власного персонажу з нуля.

Co-funded by the Erasmus+ Programme of the European Union



# Підготовка власного персонажу

- **моделювання**
- **ріггінг**
- **скінінг**



Co-funded by the Erasmus+ Programme of the European Union



# Підготовка власного персонажу.
# **Моделювання**

- Дотримування розумної топології.
- Масштабування мешу
- Розташування мешу певним чином
- Моделювання в Т-позі
- Тримати модель в порядку

Co-funded by the Erasmus+ Programme of the European Union



# Підготовка власного персонажу.
# Ріггінг

## Можливі структури ієрархії:

- Таз - хребет - грудна клітка - плечі - рука - передпліччя – кисть
- Таз - хребет - грудна клітка - шия – голова
- Таз - стегно - нога - ступня - палець - кінець_пальця

Co-funded by the Erasmus+ Programme of the European Union



# Підготовка власного персонажу.
# Скінінг

## Основні рекомендації для цього процесу:

- Автоматизація початкового налаштування скінінгу
- Створення простих анімацій або імпорт готових
- Поступове редагування
- Використання не більш 4-х впливів при м’якій прив’язці

Co-funded by the Erasmus+ Programme of the European Union



# Використання контролерів та тригерів в Unity3D

Приклад створення анімаційного контролера:

1. Створити нову сцену.
2. Створити об'єкт 3D Cube Game і помістити камеру на нього.
3. Перейти в меню вікна і натиснути кнопку «Animation». Ця дія відкриє вікно анімації.

Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

4. У вікні анімації натиснути кнопку «Create/Створити» в середині вікна.

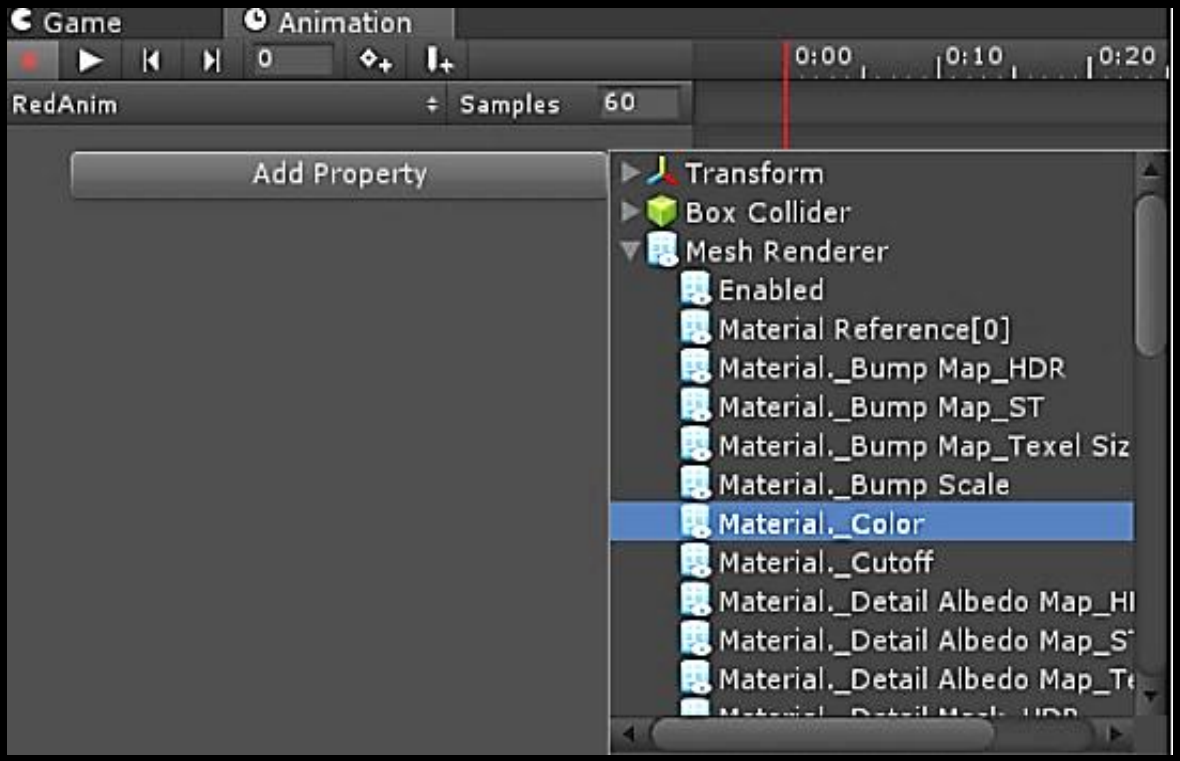

5. З'явиться вікно з проханням вказати ім'я файлу. Назвати це можна, як завгодно. Наприклад, «RedAnim».

Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

6. У вікні анімації натиснути кнопку «Add Property/Додати властивість». Вибрати «Mesh Renderer». Знайти «Material._Color» і прокрутивши праворуч у вікні, обрати піктограму "+", щоб додати її до списку властивостей.

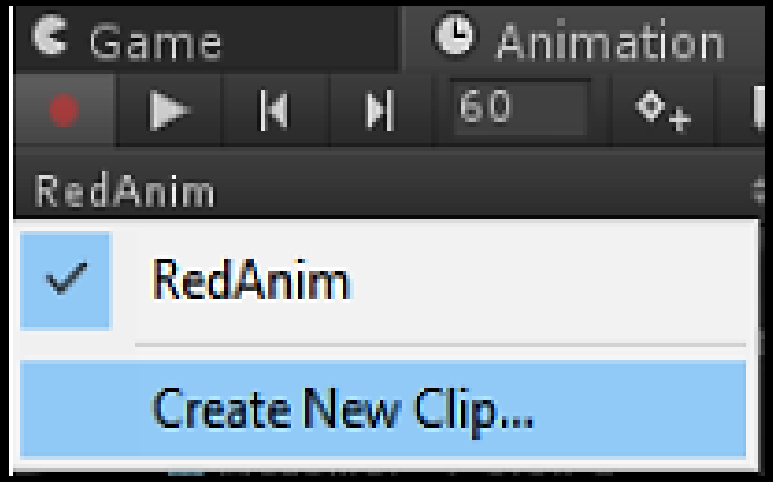


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

7. Це все, що потрібно для базової анімації і гарантує, що куб білого кольору.
8. Слід зробити ще одну анімацію для базового кольору, натиснувши кнопку заголовка «RedAnim», обрати «Create New Clip…/Створити новий кліп ...»:

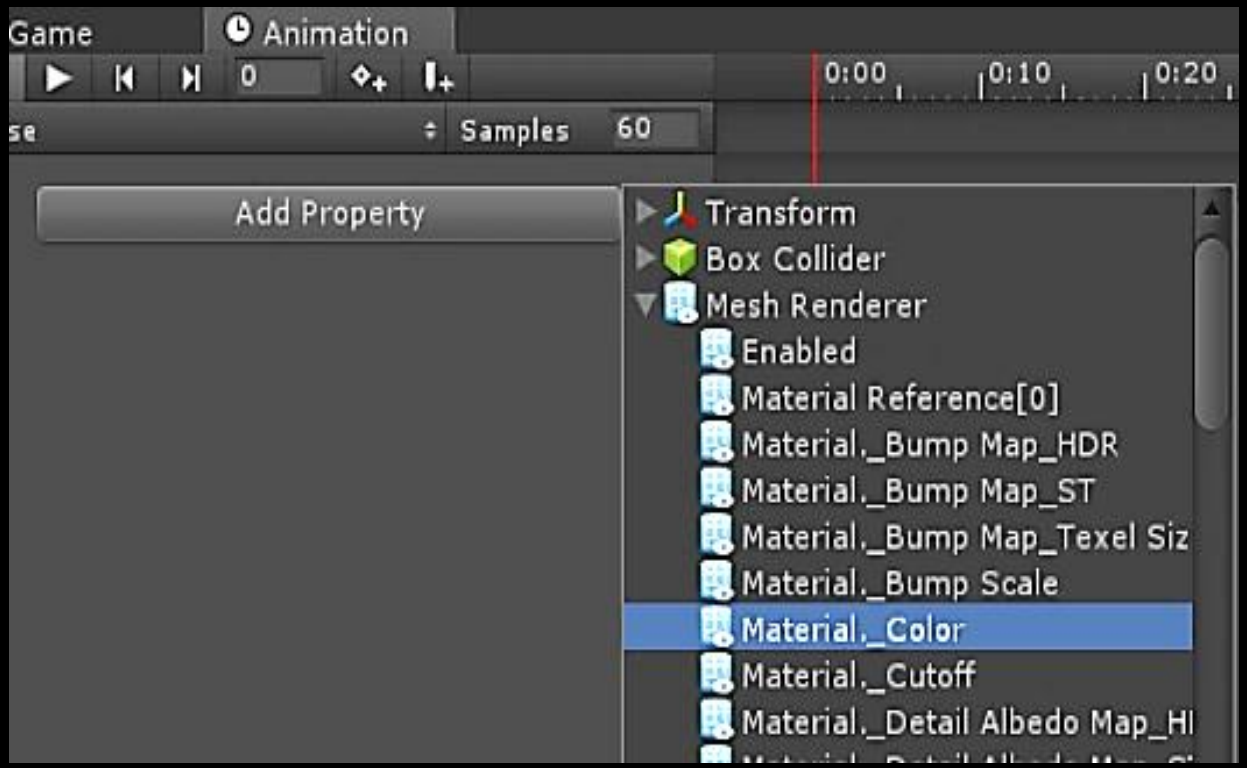


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

9. З'явиться інше діалогове вікно з питанням, як зберегти файл. Наприклад, «Base». Знову додамо властивість кольору, натиснувши «Add Property» і вибравши опцію Mesh Renderer → Material._Color

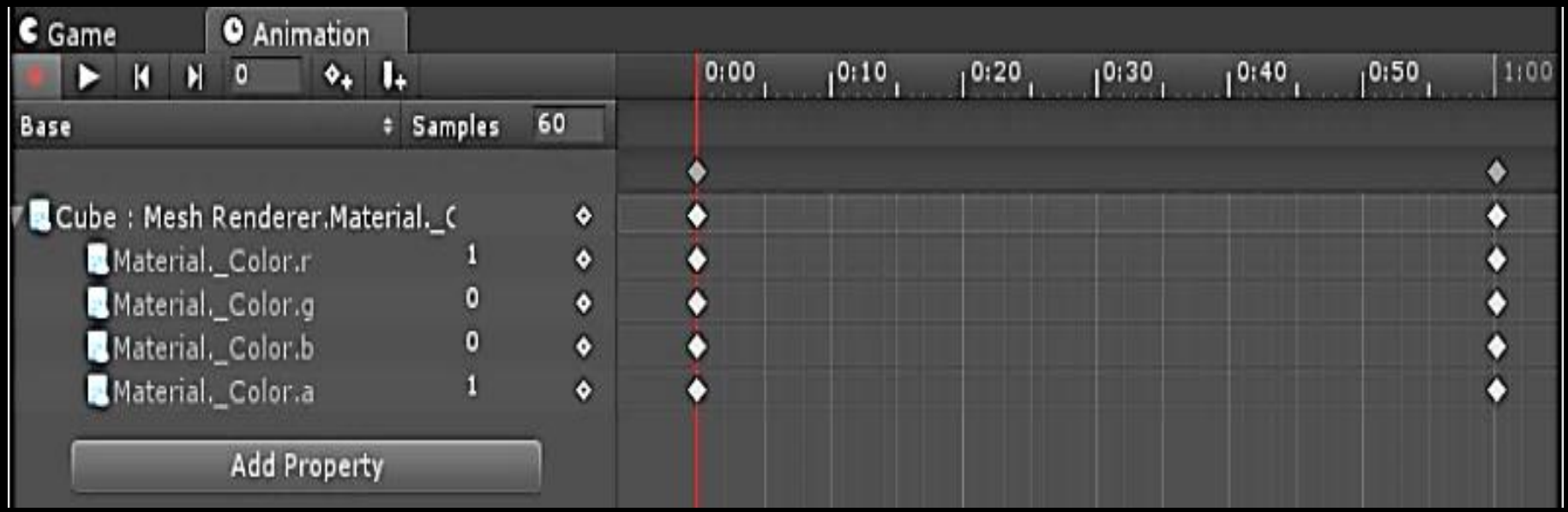


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

10. Слід переконатись, що часова шкала знаходиться в 0:00 і змінюємо значення g і b на 0.



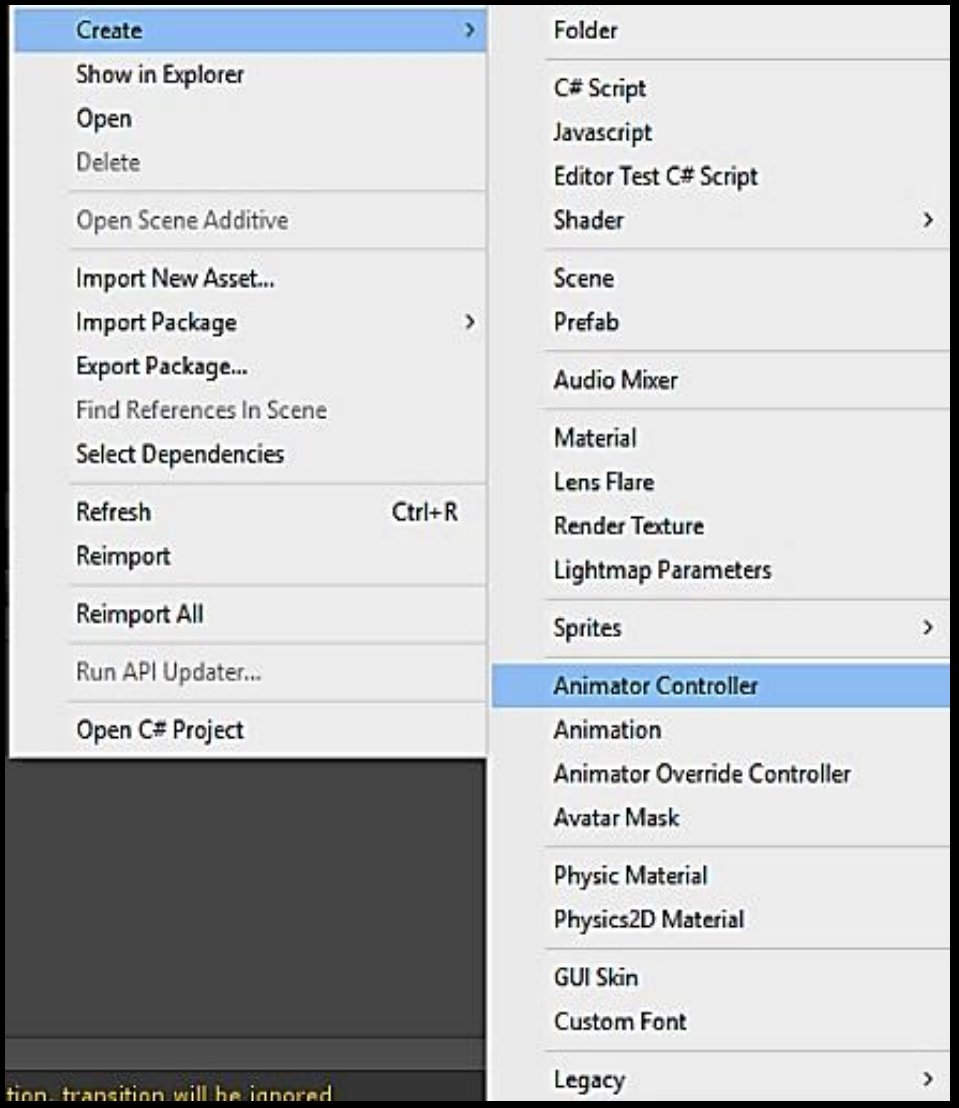

Це призведе до більш природнього переходу від червоного до білого.

Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

11. Тепер потрібно створити контролер анімації: клікнути правою кнопкою миші у вікні проекту і вибрати «Create» → «Animator Controller». Ім’я може бути довільним.

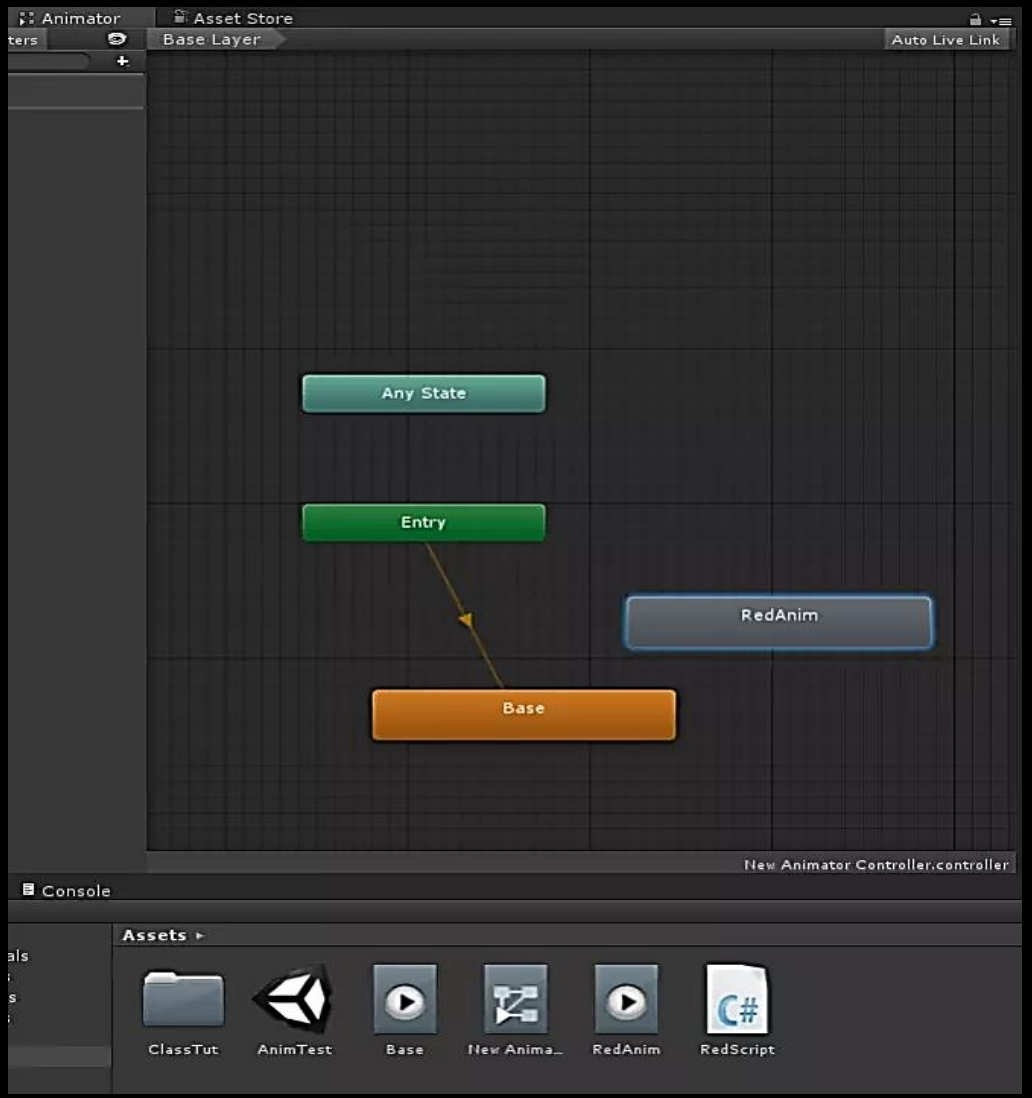


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

12. Двічі клікніть контролер аніматора, щоб відкрити редактор контролера аніматора. Як тільки він буде відкритий, перетягніть обидві ваші анімації в вікно контролера аніматора.

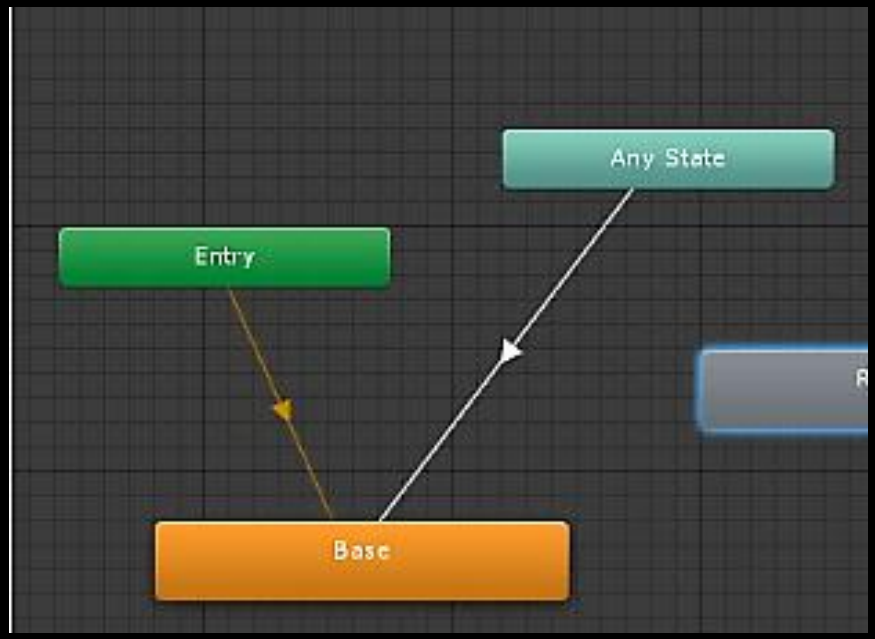


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

13. Створення логіки переходу для тепер існуючих анімацій таке: клікнути правою кнопкою миші на «Any State» і обрати «Make Transition», навести курсор миші над вузлом «Base» і клікнути лівою кнопкою миші по ньому. Це з'єднає стрілку з будь-якого стану в базову анімацію, наприклад:

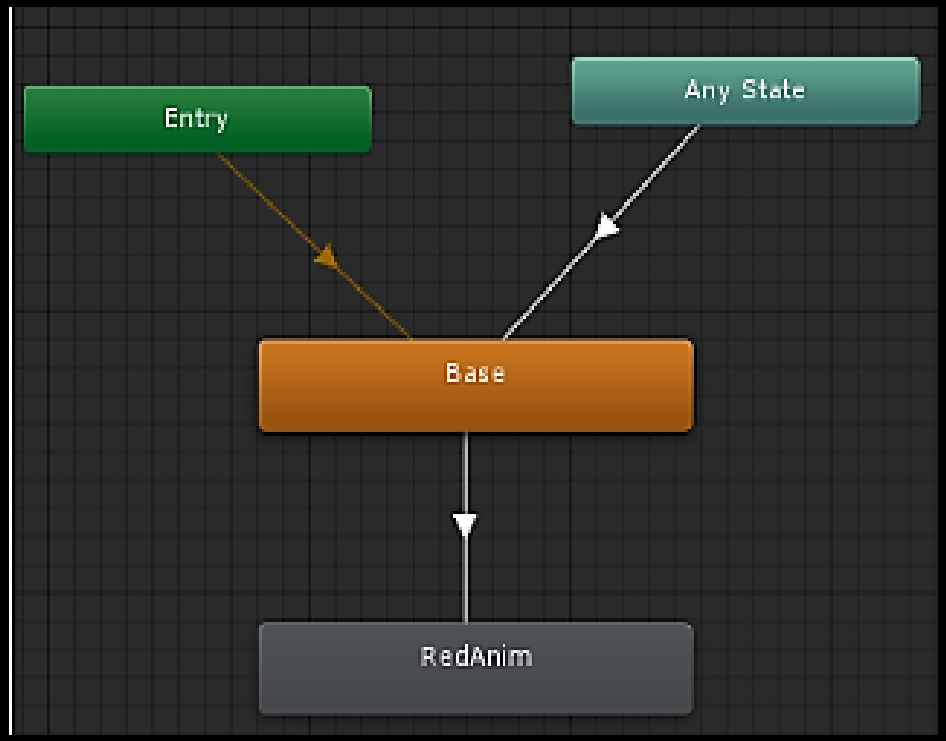


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

14. Тепер потрібно мати можливість переходити з базового стану в стан RedAnim. Для цього потрібно клікнути правою кнопкою миші на «Base», вибрати «Make Transition /Зробити перехід» і натиснути «RedAnim».

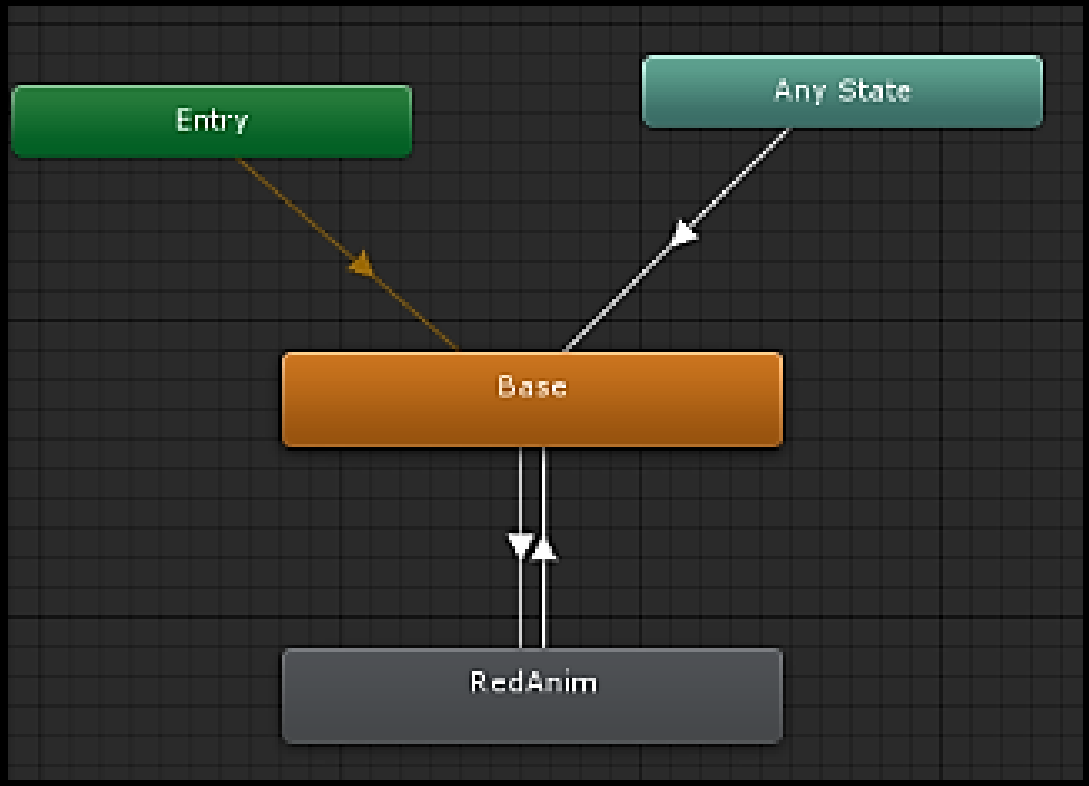


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

15. Повертаємось в базовий стан з RedAnim, клікнувши правою кнопкою миші на RedAnim і обравши «Make Transition» підключити його до вузла «Base».

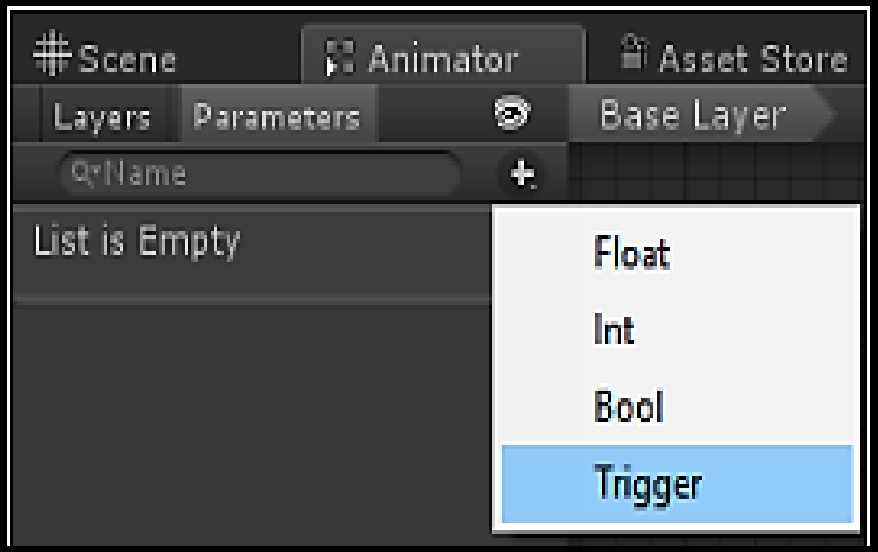


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

16. Отримано блок-схему. Створюємо змінну, яку можна використовувати в коді, щоб зробити перехід. Для цього треба натиснути значок «+» в лівому верхньому кутку вікна контролера аніматора і вибрати «Trigger» із списку, що випаде:

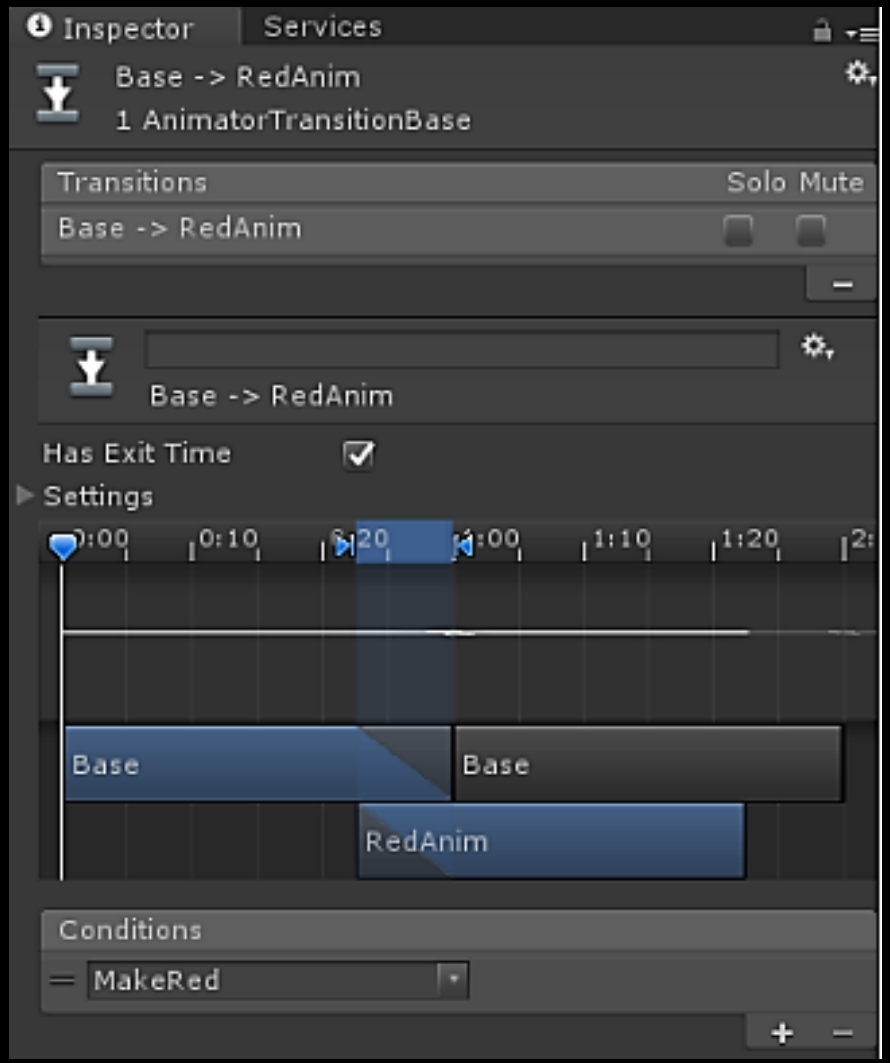


Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

17. Назвемо тригер «MakeRed», щоб викликати його через C#. Далі треба натиснути лівою кнопкою миші по стрілці, що йде від бази до RedAnim -- перехід з'явиться в інспекторі з правого боку Unity, де існує розділ «Conditions», та натиснути значок «+» в цьому розділі, який додасть тригер «MakeRed» в список умов.



Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

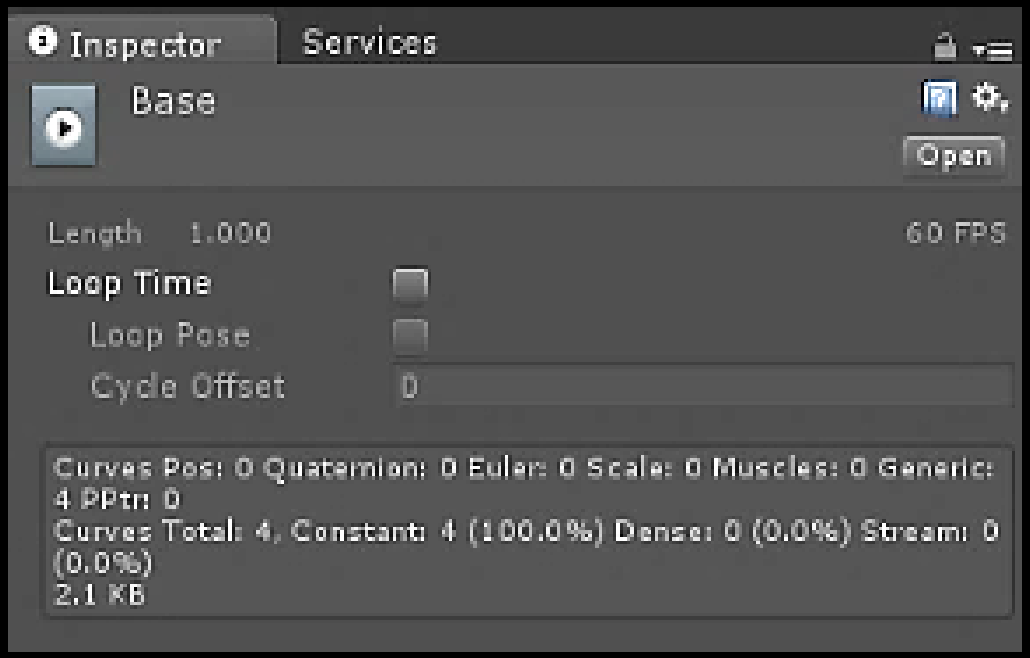
18. Аналогічні дії потрібно зробити при переході з RedAnim на Base.
19. Поставимо контролер аніматора на куб. Список компонентів куба повинен виглядати так:

Co-funded by the Erasmus+ Programme of the European Union



# Приклад створення анімаційного контролера:

20. Перш ніж створювати скрипт, потрібно зробити такі дії - клікнути лівою кнопкою миші по анімації у вікні проекту, в інспекторі зняти прапорець «Loop Time/Час циклу». Це забезпечить запуск циклу в цих анімаціях, і це потрібно зробити для обох анімацій.



Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Приклад створення анімаційного контролера:

21. Тепер потрібно створити новий C# - скрипт з ім'ям RedScript. Це виглядає так:

```
using UnityEngine;
using System.Collections;

public class RedScript : MonoBehaviour {

    Animator anim;

    void Start ()
    {
        anim = GetComponent<Animator>();
    }

    void Update ()
    {
        if (Input.GetKeyDown(KeyCode.Space)) anim.SetTrigger("MakeRed");
    }
}
```

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Що ми вивчили?

**Основні ресурси:**

- Використання персонажів
- Підготовка власного персонажу.
- Використання контролерів та тригерів.
- Поняття анімаційного контролеру.

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Студент має :

- Пояснити роль ігрового персонажу при створенні ігрових додатків за допомогою Unity.
- Знати, як створити власний персонаж гуманоїдного типу.
- Знати про особливості створення анімаційних контролерів.
- Знати про призначення тригерів в Unity

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Заключення. Лабораторія

Лабораторна робота №3. Створення інтерактивного геймплея з персонажем за допомогою засобів движка Unity. Організація взаємодії користувача.

Дивись «Методичні вказівки до виконання лабораторних робіт»

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D. Рекомендовані ресурси.

## Документація

Офіційний сайт Unity3d https://unity3d.com/ru

https://docs.unity3d.com/ru/

Sue Blackman. Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine, 2011. – 986 p.

Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.

## *Дякую!*
## *Питання?*

***The lection was performed with support of the Erasmus+ Programme of the European Union (561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP). The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Donetsk National Technical University

Модуль «Розробка ігрових додатків на базі движка Unity»

Асистент Скрипник Т.В.

**DonNTU GameHub**

***561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP***

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Лекція 3. Частина 2.

- Організація користувальницького інтерфейсу засобами Unity.
- Особливості організації користувальницького меню.

Co-funded by the
Erasmus+ Programme
of the European Union



# Користувальницький інтерфейс GUI

Co-funded by the Erasmus+ Programme of the European Union



# Користувальницький інтерфейс GUI. Canvas (Полотно)

Canvas (полотно) - це область, всередині якої знаходяться всі елементи UI (користувальницького інтерфейсу). Полотно - це ігровий об'єкт (Game Object), з доданим до нього компонентом Canvas. Всі елементи UI повинні бути дочірніми цьому Canvas.

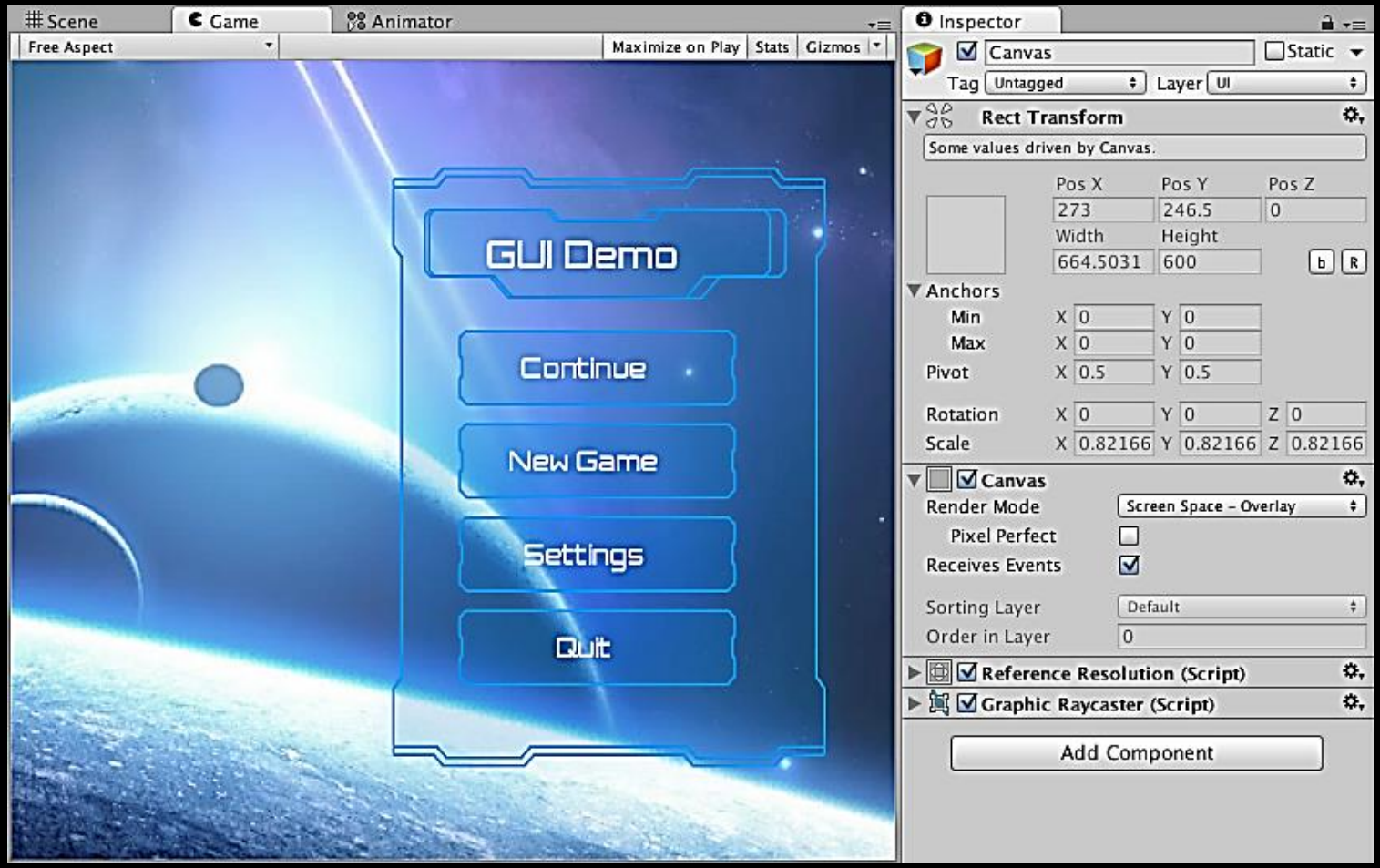
При створенні нового елемент UI, наприклад, зображення (Image), використовуючи меню GameObject → UI → Image, разом з ним автоматично створюється і Canvas, якщо до цього на сцені його ще не було. Елемент UI створюється дочірнім цьому Canvas.

Co-funded by the Erasmus+ Programme of the European Union



# Режими відображення Canvas

## Простір екрану - Перекриття (Screen Space - Overlay)

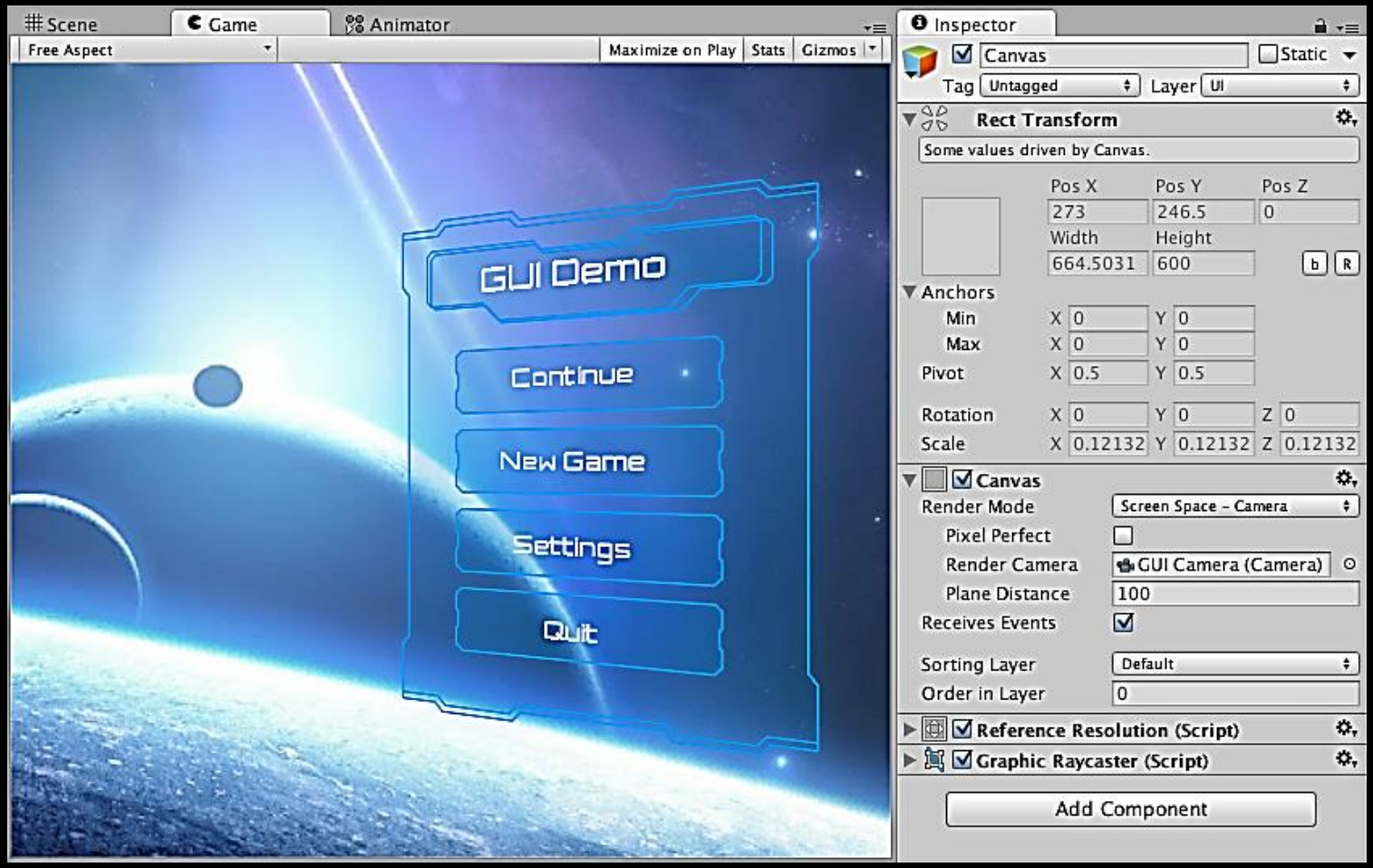


Co-funded by the Erasmus+ Programme of the European Union



# Режими відображення Canvas

## Простір екрану - Камера (Screen Space - Camera)

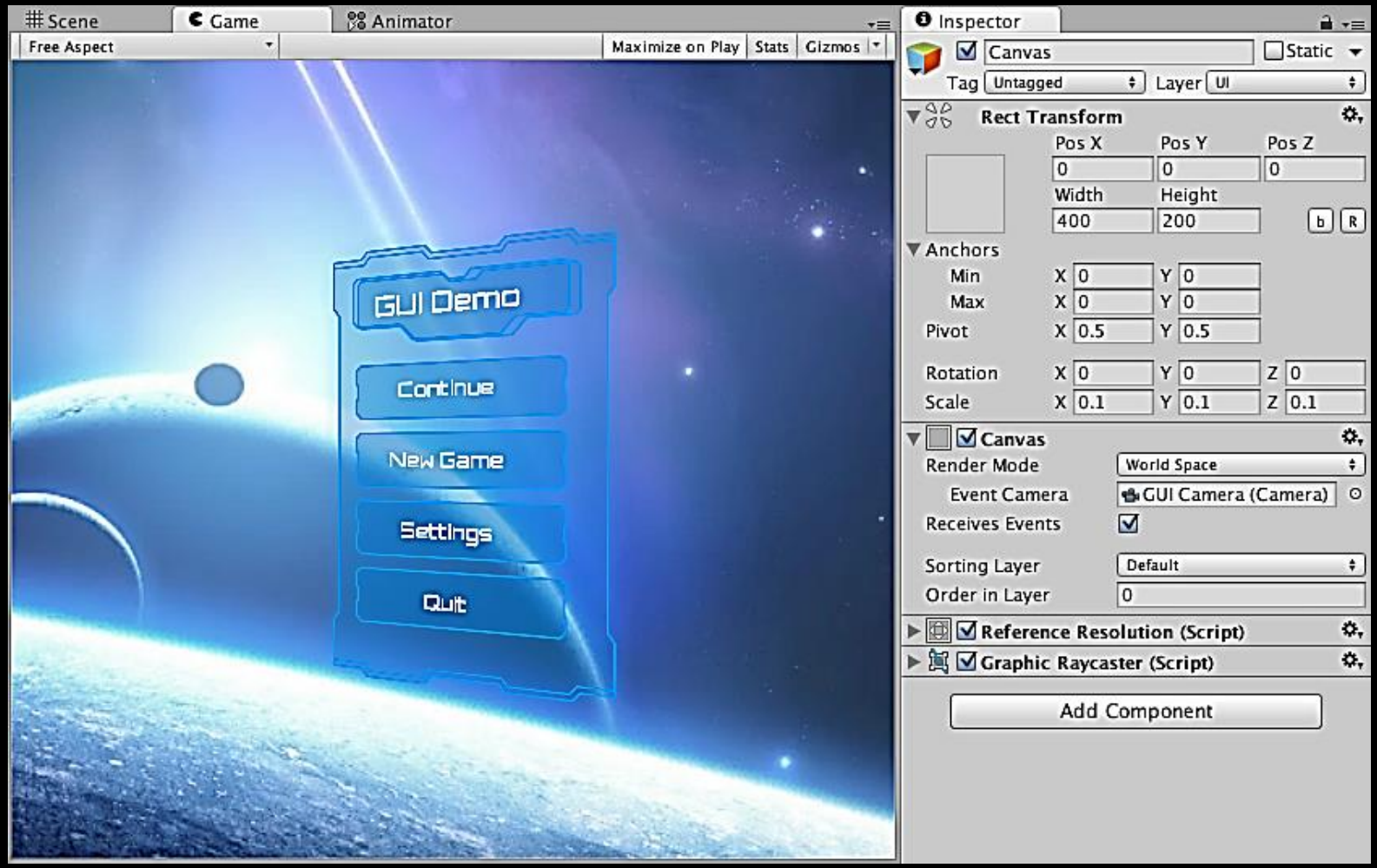


Co-funded by the Erasmus+ Programme of the European Union



# Режими відображення Canvas

## Простір ігрового світу (World Space)



Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Користувальницький інтерфейс GUI. Компоненти взаємодії

| Назва (вид) | Призначення |
|---|---|
| **Кнопка**   | **Кнопка має OnClick UnityEvent, щоб визначити, що вона робитиме при натисканні.** |
| **Тумблер (перемикач)**   | **У Toggle встановлений прапорець Is On, який визначає, чи увімкнене Toggle. Він також має OnValueCHanged UnityEvent, щоб визначити, що він робитиме, коли значення буде змінено.** |
| **Toggle Group (група перемикачів)**   | **Toggle група може бути використана для групування набору перемикачів, так що тільки один з них може бути обраний одночасно - вибір одного з них автоматично скасовує вибір всіх інших.** |

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Користувальницький інтерфейс GUI. Компоненти взаємодії (продовження)

| Назва (вид) | Призначення |
|---|---|
| **Повзунок (Slider)**  | **Повзунок може бути горизонтальним або вертикальним, має десяткове числове значення, яке користувач може перетягнути від мінімального до максимального значення. Він має OnValueChanged UnityEvent для визначення, що він буде робити, якщо значення буде змінено.** |
| **Полоса прокрутки (Scrollbar)**  | **Полоса прокрутки має значення десяткового числа між 0 і 1, що змінюється відповідним чином, коли користувач перетягує покажчик прокрутки.** |
| **Випадаючий список (Dropdown)**   | **У випадаючому списку є перелік варіантів на вибір.** |

Co-funded by the Erasmus+ Programme of the European Union



# Користувальницький інтерфейс GUI. Створення UI

У Unity 4.6 додана абсолютно нова система UI для створення елементів HUD (heads-up display) у грі з використанням тексту, панелей, віджетів і ін. Додавання тексту в HUD вашої гри зводиться до вибору GameObject → UI → Text і завданням шрифту і тексту. Якщо потрібно пізніше управляти текстом в коді, наприклад для поновлення рахунку, то просто використовуєте код:

```
// Отримуємо компонент UnityEngine.UI.Text
var score = GetComponent<Text>();
score.text = "Score:0";
```

Co-funded by the Erasmus+ Programme of the European Union



# Користувальницький інтерфейс GUI. Створення UI


UI із зображенням і текстами в HUD

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Що ми вивчили?

**Основні ресурси:**

- Користувальницький інтерфейс і його призначення.
- Компоненти взаємодії

Co-funded by the Erasmus+ Programme of the European Union



# Заключення. Студент має :

- Знати про особливості створення користувальницького інтерфейсу ігрового додатку.
- Вміти використовувати компоненти взаємодії за їхнім призначенням

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Заключення. Лабораторія

Лабораторна робота №3. Створення інтерактивного геймплея з персонажем за допомогою засобів движка Unity. Організація взаємодії користувача.

Дивись «Методичні вказівки до виконання лабораторних робіт»

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D. Рекомендовані ресурси.

## Документація

Офіційний сайт Unity3d https://unity3d.com/ru

https://docs.unity3d.com/ru/

Sue Blackman. Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine, 2011. – 986 p.

Joe Hocking. Unity in Action: Multiplatform Game Development in C#, 2015. – 352 p.

Co-funded by the Erasmus+ Programme of the European Union

GAMEHUB University-Enterprises Cooperation In Game Industry In Ukraine

# Unity 3D.

## *Дякую!*
## *Питання?*

***The lection was performed with support of the Erasmus+ Programme of the European Union (561728-EPP-1-2015-1-ES-EPPKA2-CBHE-JP). The European Commission support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.***