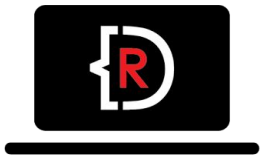
Game design



<{DevelopeR/>

Знищення персонажів. Меню. Світло.

Повторення

Для чого використовується структура Vector3(x, y, z)?

Ця структура використовується всередині Unity для передачі 3D-позицій та напрямків. Вона також містить функції для виконання загальних векторних операцій.

Повторення

Як реалізувати обмін даними для системи Scoring?

*Слід створити скрипт **ScoreManager**, через який відбуватиметься обмін данними і створити в ньому public змінну:*

```
public static int score;
```

Для того аби отримати змінну за межами класу, змінна повинна бути загальнодоступною і звертатися до неї слід через ім'я класу:

```
ScoreManager.score += 100;
```

Повторення

Що таке масив?

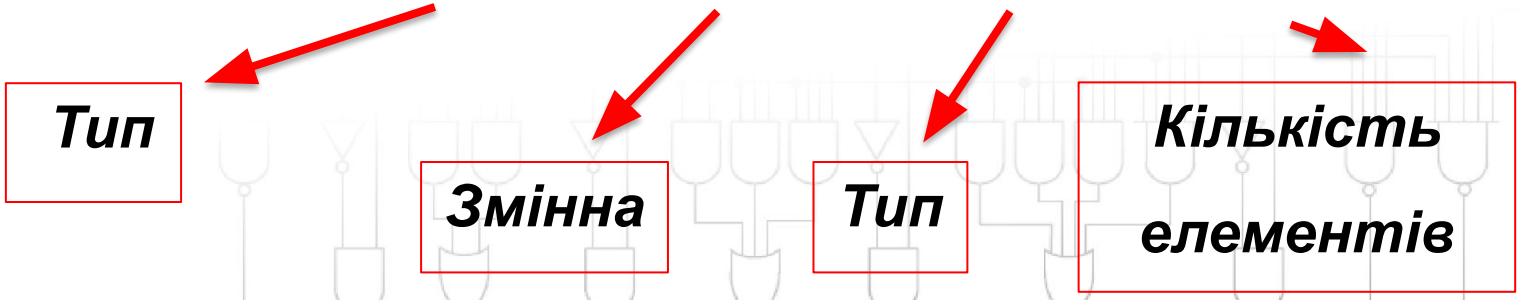
Масив *– сукупність однотипних елементів, які містяться в пам'яті послідовно та мають спільне ім'я.*

В мові C# оголошення масиву відбувається ось так:

```
ТИП[] ЗМІННА = new ТИП[Кількість елементів];
```

Тобто, щоб створити масив для наших елементів live, необхідно записати наступне:

```
private Transform[] hearts = new Transform[5];
```



Завдання

Відкрийте попередній проект. В ньому додайте об’єкт, що наноситиме урон головному герою (налаштування аналогічні падаючому об’єкту):

- *Додайте об’єкту скрипт падіння angryball (скопіюйте вміст скрипта «кульки», що падає).*
- *В ньому створіть об’єкт* `heart1` *класу* `heart`, *що викликатиме метод* Refresh().

Завдання


```
public float FallSpeed;
public GameObject explosion;
public Transform zubiCheckPoint;
public float zubiCheckRadius;
public LayerMask zubiCheckLayer;
private bool isTouchingzubi;
public Transform angryballs;

float fcoordX;
float fcoordY;

private heart heart1;
```

Швидкість падіння

Об'єкт вибуху

Переміщення точки перевірки

Радіус перевірки контакту

Шар, з яким очікуємо контакт

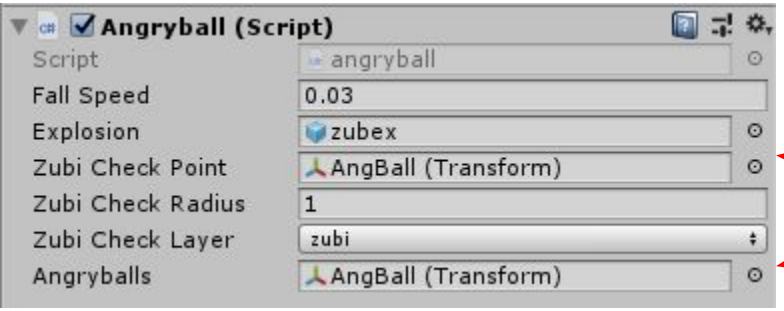
Змінна,що вказуватиме відбувся контакт чи ні

Переміщення об'єкта

Об'єкт класу heart

Завдання

В інспекторі :



Швидкість падіння

Prefab анімації вибуху

Префаб об'єкта, що падає

Перевірка контакту із шаром zubi (головний герой)

Завдання

В скрипті angryball, коли задетектовано контакт із персонажем, викликаємо знищення поточного об'єкта (this), виникнення анімації вибуху (минулий урок), зменшення hp головного персонажу та виклик метода Refresh() класу heart1:

```
 if (isTouchingzubi)
     {
        Instantiate(explosion, new Vector3(fcoordX, fcoordY, 0),
Quaternion.identity);
        Object.Destroy(this.gameObject);
        zubi.hp -= 1;
        heart1.Refresh();
     }
 else if (transform.position.y < -7f)
     {
        Object.Destroy(this.gameObject);
     }
```

Завдання

На початку скрипта angryball (void Start()) нам слід передати створенному об'єкту heart1 *об'єкт класу* heart:

```
void Start()
    {
        heart1 = FindObjectOfType<heart>();
    }
```

В тілі скрипта angryball присвоїмо значення необхідним змінним та задамо рух об'єкта:

```
  fcoordX = angryballs.position.x;
  fcoordY = angryballs.position.y;
  transform.Translate(0, -FallSpeed, 0);
  isTouchingzubi =
Physics2D.OverlapCircle(zubiCheckPoint.position,zubiCheckRadius,
zubiCheckLayer);
```

Завдання

В тілі angryball, коли задетектовано контакт із персонажем, викликаємо знищення поточного об'єкта (this), виникнення анімації вибуху (дивіться минулий урок), зменшення hp головного персонажу та виклик метода Refresh() класу heart1:

```
 if (isTouchingzubi)
     {
        Instantiate(explosion, new Vector3(fcoordX, fcoordY, 0),
Quaternion.identity);
        Object.Destroy(this.gameObject);
        zubi.hp -= 1;
        heart1.Refresh();
     }
 else if (transform.position.y < -7f)
     {
        Object.Destroy(this.gameObject);
     }
```

Завдання

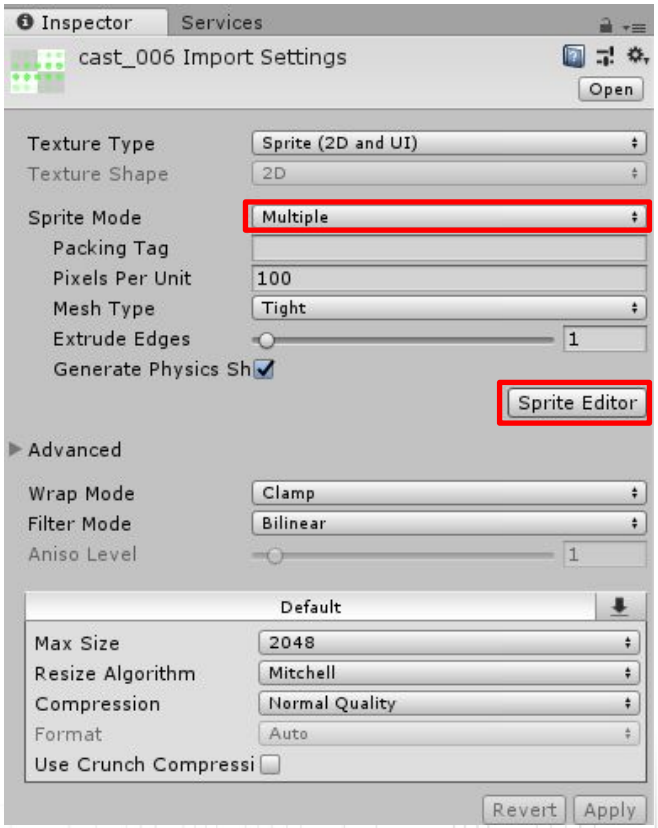
Залишилось створити анімацію вибуху та скрипт генерації «ворожих» об'єктів (randAnball):

```
    public GameObject anball;
    float offsetX = 0;
    float offsetY = 10f;
    float fTime = 1f;
void FixedUpdate()
    {
        if (fTime < Time.realtimeSinceStartup)
        {
            offsetX = Random.Range(-7f, 9f);
            Instantiate(anball, new Vector3(offsetX, offsetY, 0),
Quaternion.identity);
            fTime = Time.realtimeSinceStartup + Random.Range(4f, 6f);
        }

   }
```

Завдання

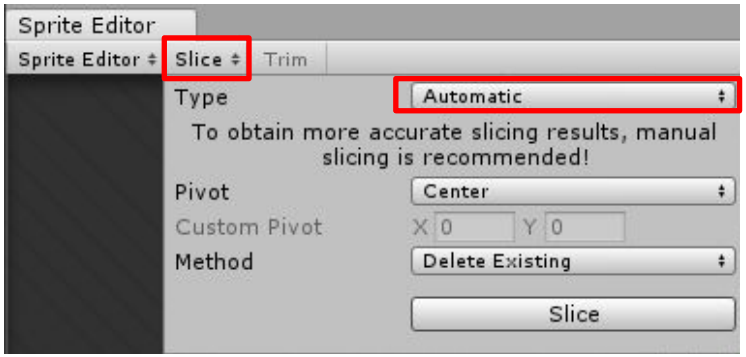
Для створення анімації є простіший метод:

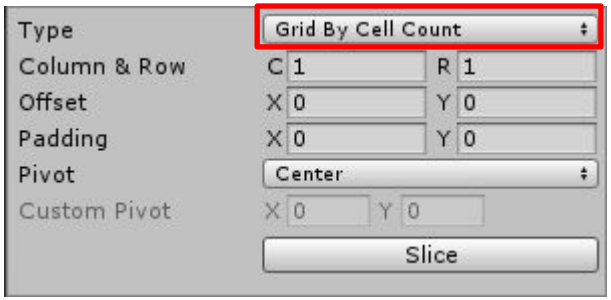
- *Скачайте Sprite Sheet.*
- *Sprite Mode->Multiple.*
- *Зайдіть в Sprite Editor*



Завдання

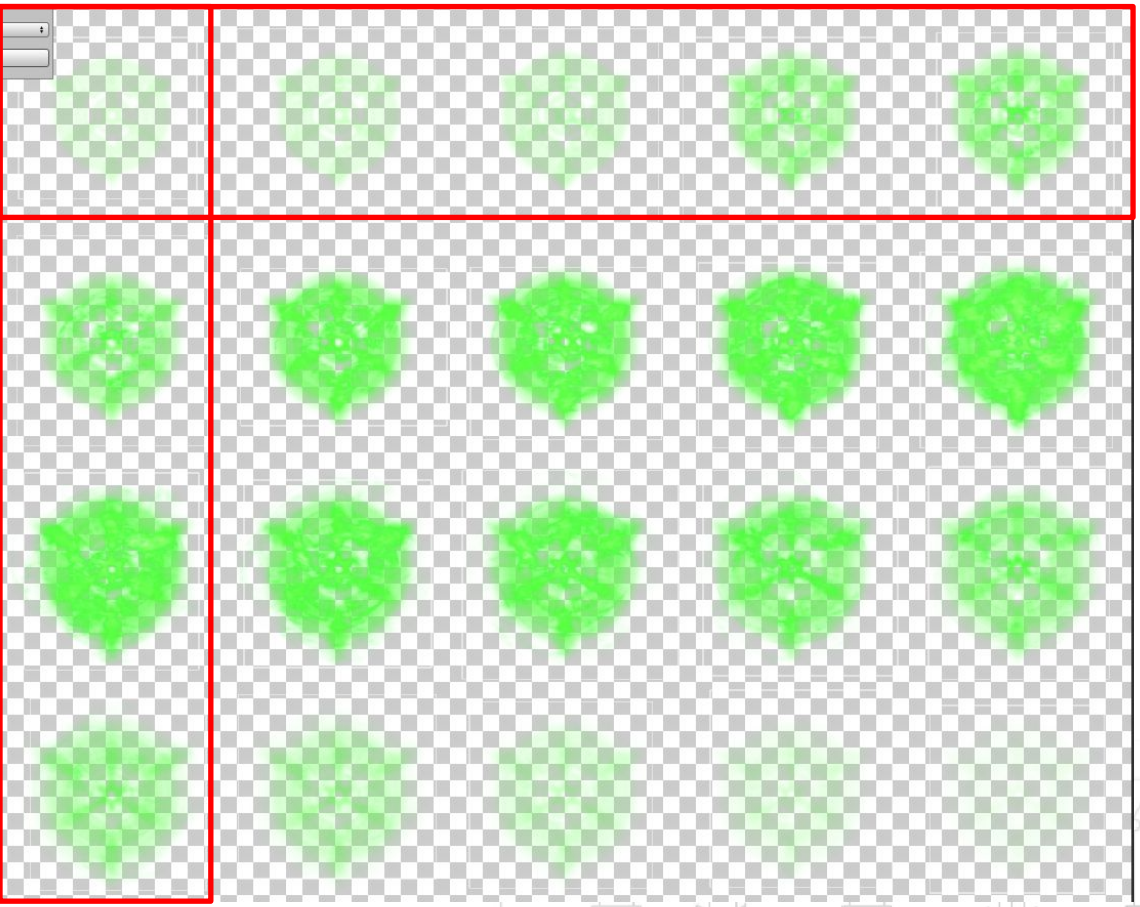
Тип Slice оберіть Grid by Cell Count:

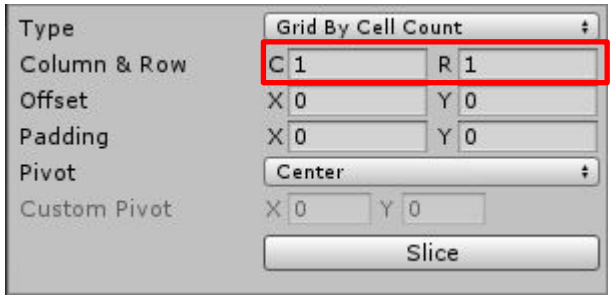




Завдання

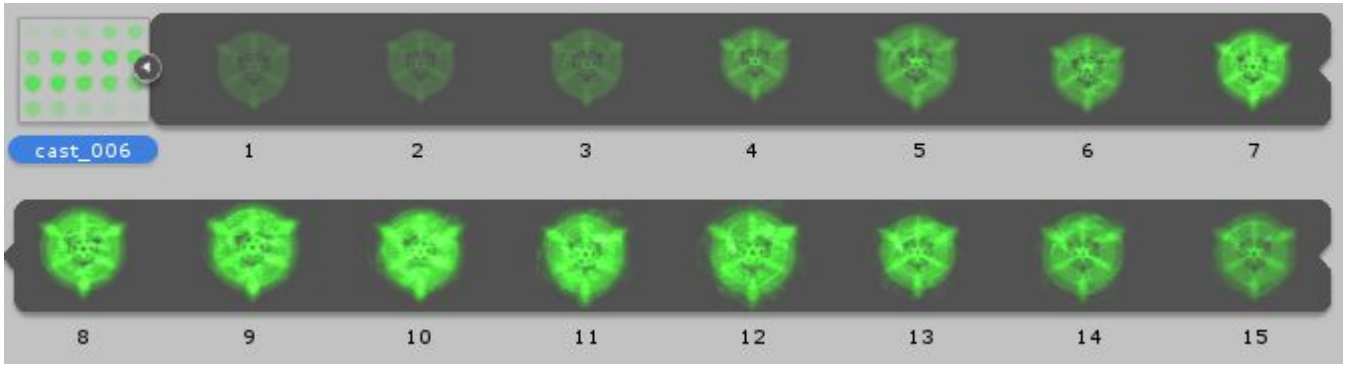
Підрахуйте кількість стовпчиків та колонок і введіть у відповідні поля:



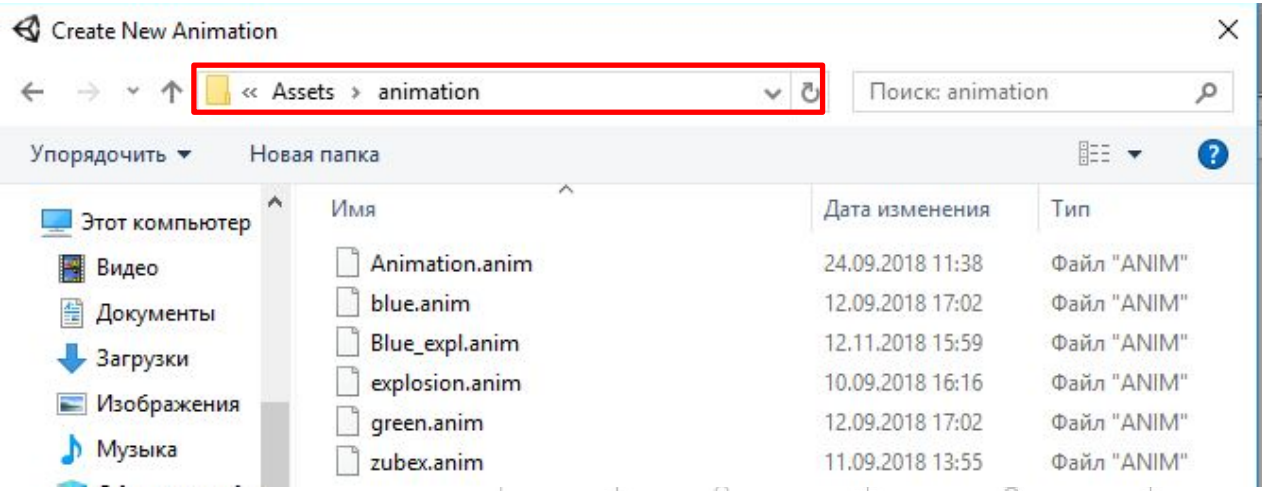


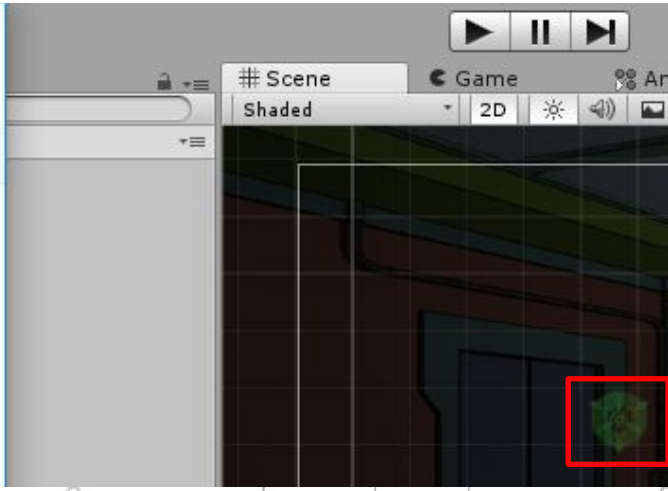
Завдання

Як наслідок – отримаєте розкадровку:



Щоб додати необхідні параметри анімації – просто перетягніть об’єкт на екран:





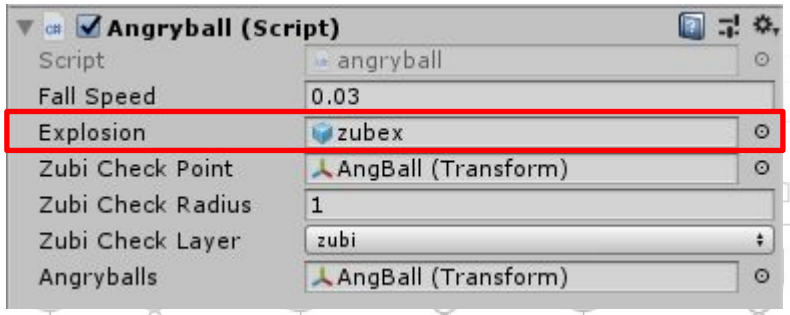
Завдання

Додайте до об'єкта анімації скрипт знищення (щоб вона не повторювалась безкінечно):

```
void FixedUpdate () {
        Destroy(gameObject, 1.1f);
}
```

Тривалість анімації

Збережіть префаб анімації та додайте його в інспектор «ворожого об'єкта»:



Меню

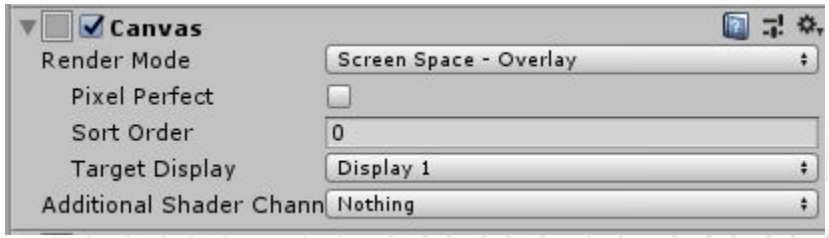
Створюємо сцену з назвою Menu (пкм –> Create –> Scene). Відкриваємо File/Build Settings. У вікні Scene In Build повинні бути розміщенні сцени гри та меню в наступному порядку:

Меню

Переходимо у сцену Menu.

Створюємо полотно Canvas (ПКМ->UI->Canvas).

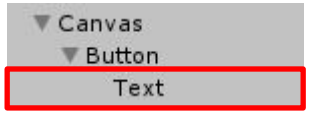
В інспекторі Canvas Render Mode вказуємо Screen Space-Overlay.

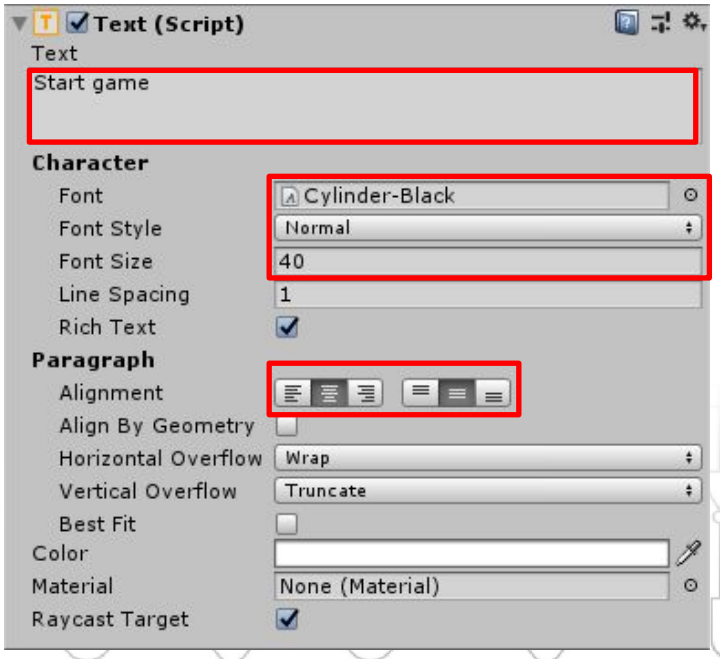


Меню

В Canvas створюємо 2 об'єкти Button (UI -> Button), кнопку входу в гру та виходу на робочий стіл:

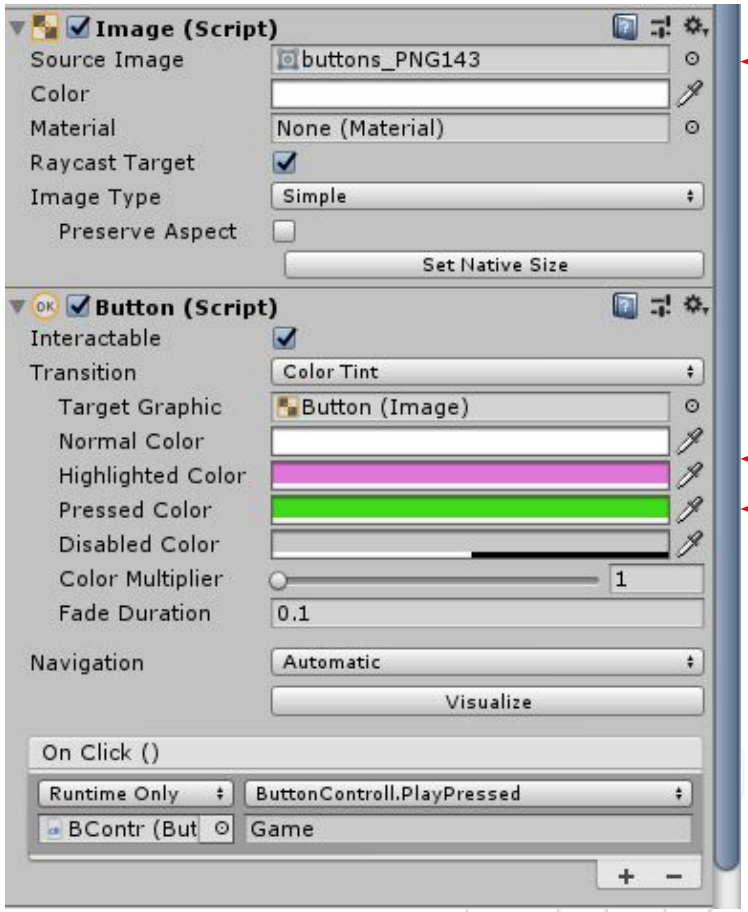
В полі Text можна задати назву кнопки, шрифт та положення напису:





Меню

В об'єкті Button можна задати зміну вигляду кнопки та зміну підсвітки при наведені/кліку:



Зображення кнопки

Підсвітка при наведенні

Підсвітка при кліку

SceneManager

Для керування переходу між сценами по натисненню на кнопки, слід створити новий скрипт ButtonControll, *для реалізації керування сценами на початку скрипта слід підключити бібліотеку:*

```
using UnityEngine.SceneManagement;
```

Щоб об'єкти кнопок отримали доступ до методів(функцій) даного скрипта, їх слід створювати як public:

```
public void PlayPressed(string NewLvl)
    {
   //~~~~~~~~~~~~~~~~~~Тіло функції~~~~~~~~~~~~~~//
    }
```

SceneManager

Керування сценою під час виконання.

Статичний метод LoadScene завантажує сцену по назві або по індексу в налаштуваннях збірки.

Синтаксис:

```
SceneManager.LoadScene («OtherSceneName», LoadSceneMode.Additive
);
```

Назва сцени

Параметри

SceneManager

Скрипт матиме наступний вигляд:

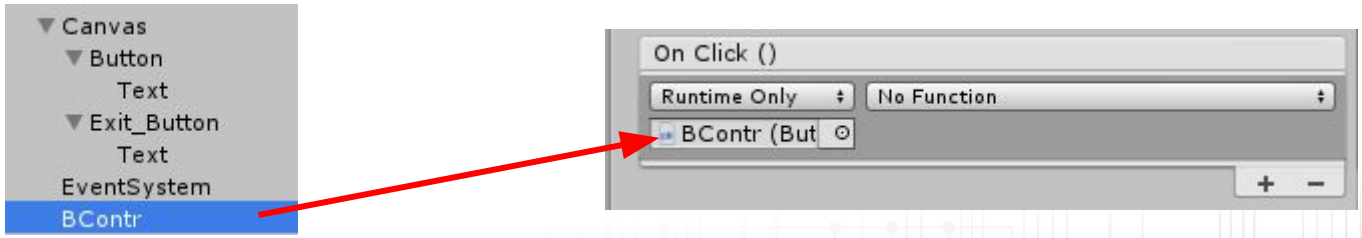
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class ButtonControll : MonoBehaviour
{
    public void PlayPressed(string NewLvl)
    {
        SceneManager.LoadScene(NewLvl);
    }
    public void ExitPressed()
    {
        Application.Quit();
    }
}
```

Строковий параметр, що міститиме ім'я завантажуваної сцени

SceneManager

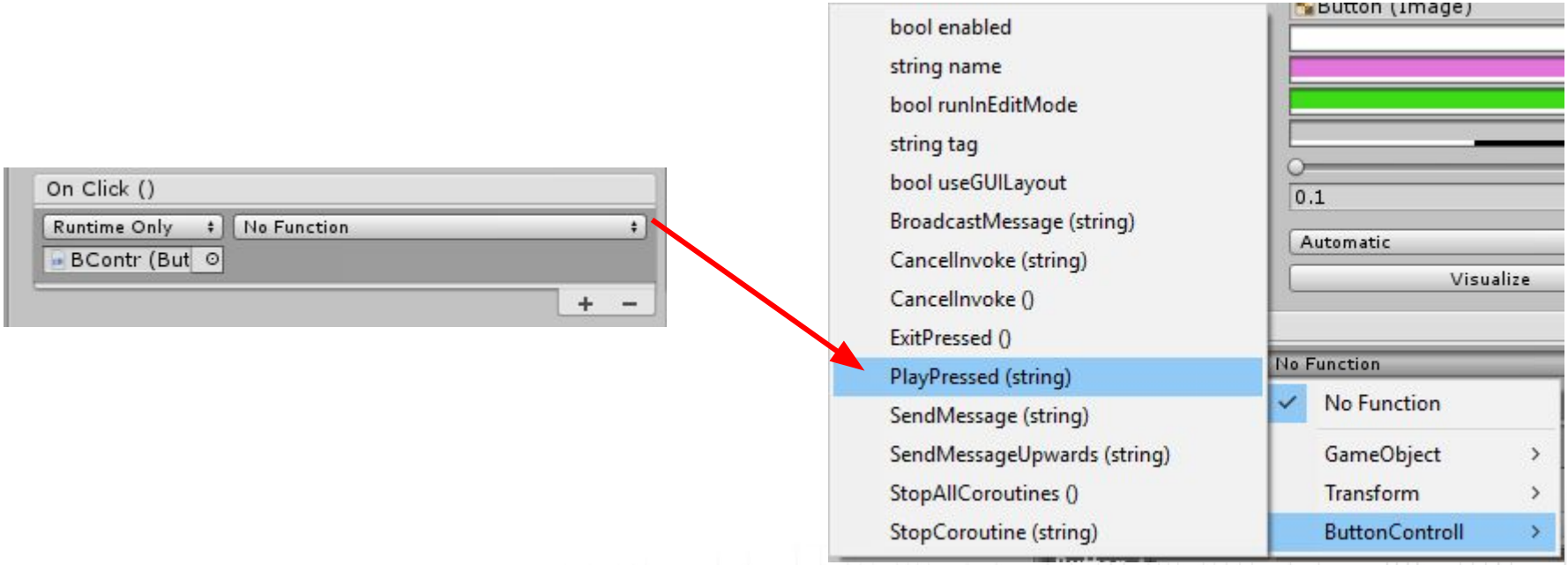
Скрипт приєднайте до об'єкта, що не знищується (краще створіть порожній об'єкт).

Щоб викликати функції зі скрипта ButtonControll, при натисненні кнопки, слід виділити об'єкт кнопки і в функцію OnClick() перетягнути об'єкт зі скриптом:



SceneManager

У вікні функції оберіть назву скрипта, і знайдіть фукцію яку ви хочете викликати при натисненні на кнопку (у нас це public void PlayPressed(string NewLvl)):



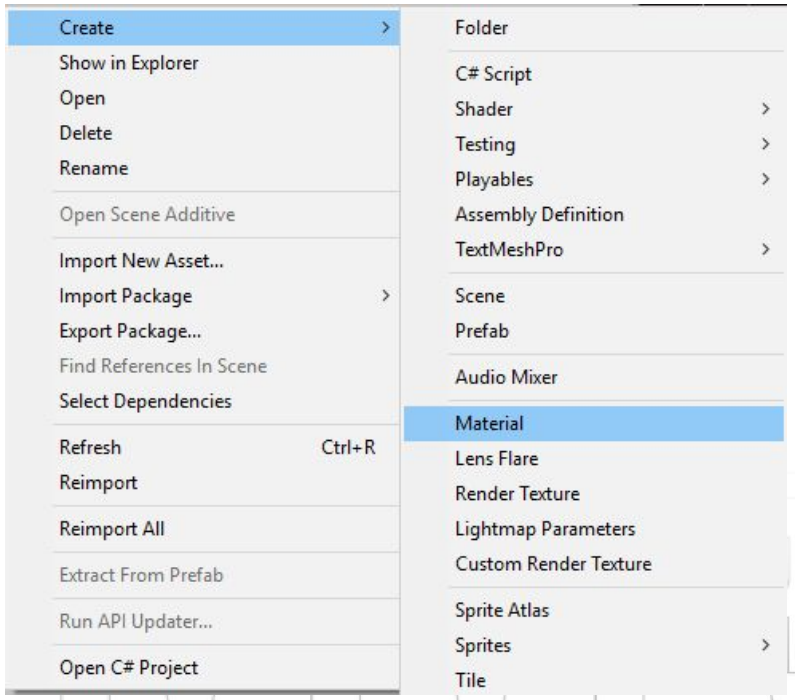
SceneManager

З'явиться місце для вводу назви сцени, яку ви хочете завантажити:

ЗВЕРНІТЬ УВАГУ!! Ім'я має співпадати з ім'ям сцени:

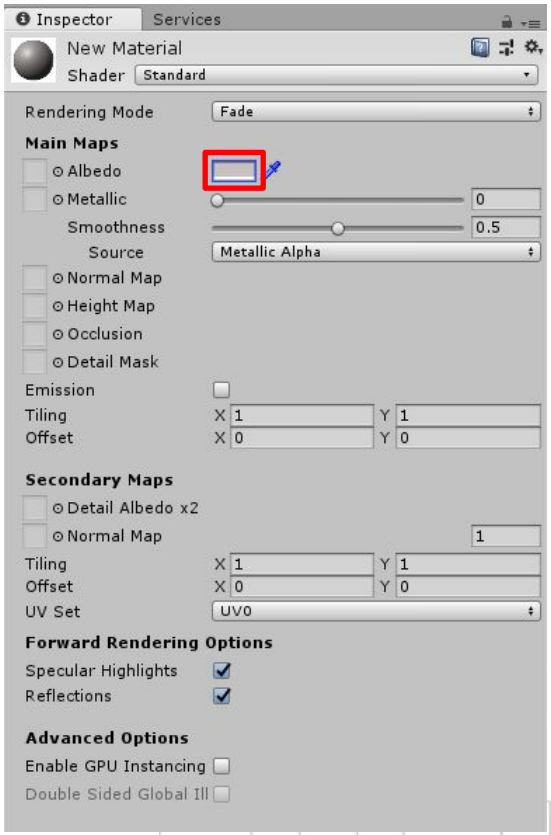
Point Light

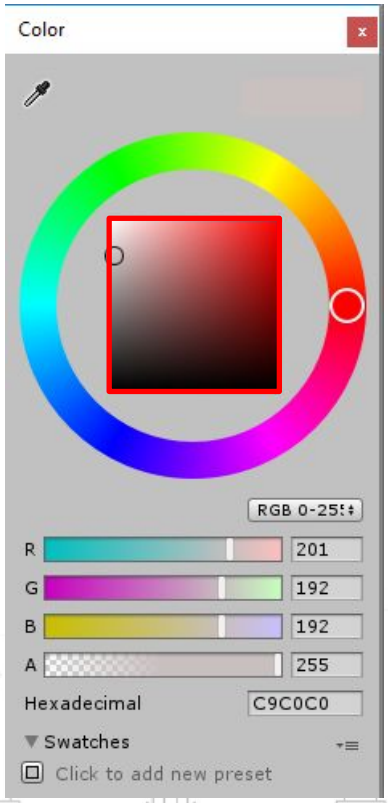
Об’єкти в Unity використовують матеріал з підсвіткою за замовчанням, якщо ми хочемо регулювати підсвітку об’єктів, то слід створити новий матеріал. Перейдіть в сцену з грою, в Assets ПКМ-> Create->Material:



Point Light

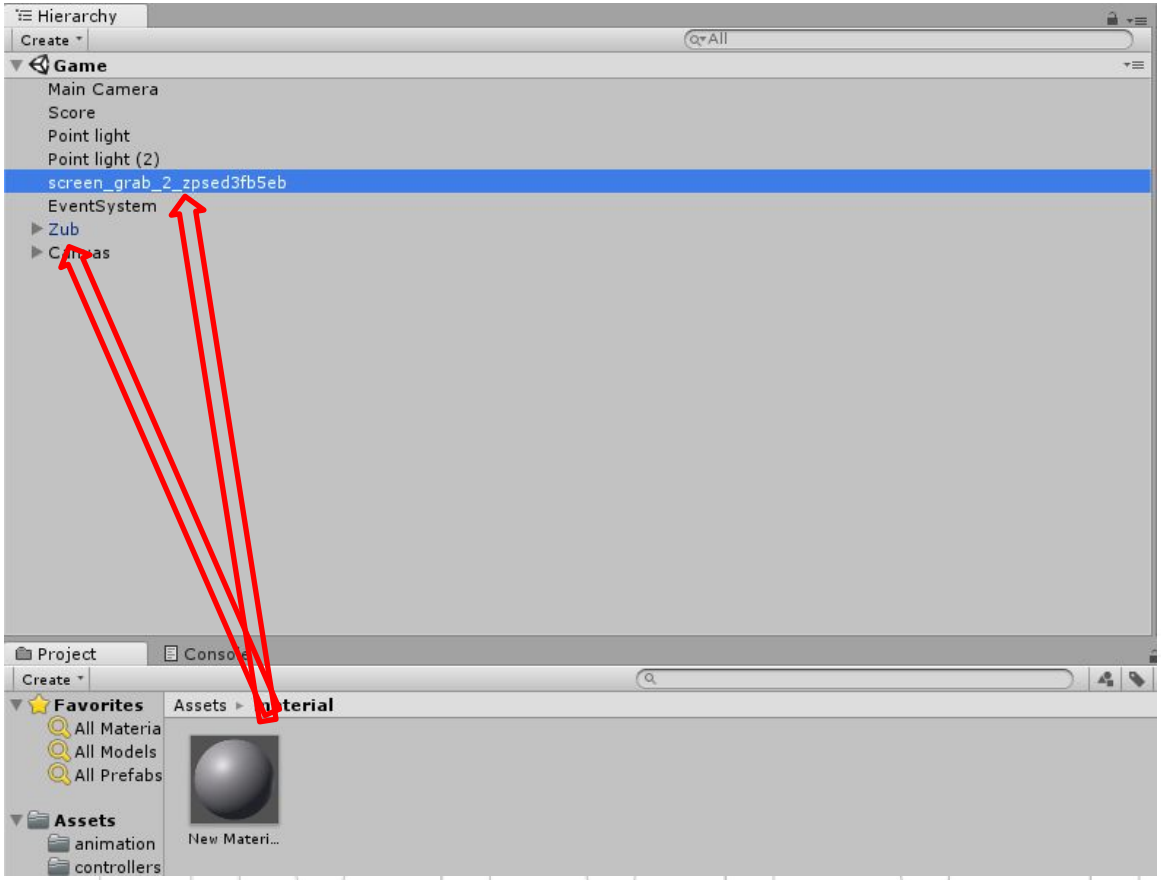
Використавши стандартні шейдери, за допомогою Albedo відрегулюйте яскравість та колір підсвітки:





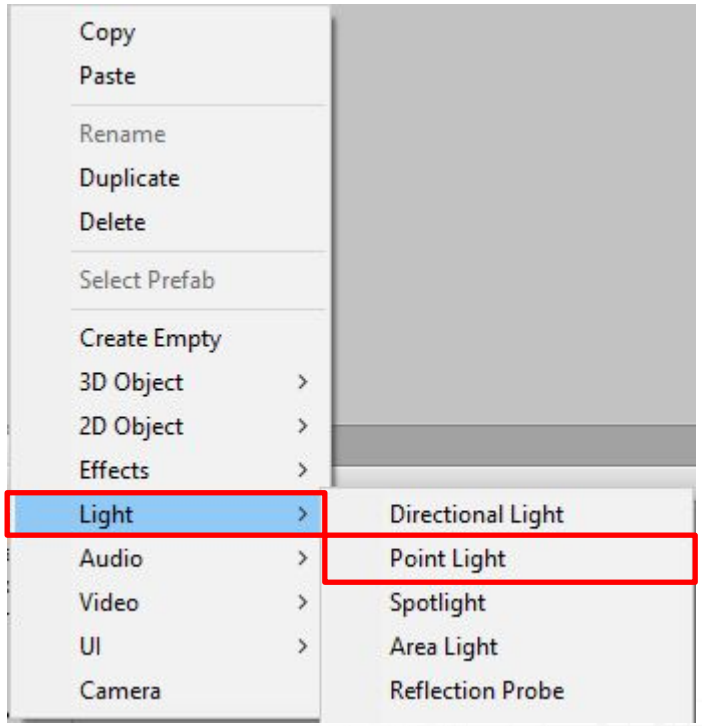
Point Light

Перетягніть цей матеріал на об’єкти, які ви хочете затемнити і збережіть їх префаби:



Point Light

Створимо точкові джерела світла (в ієрархії ПКМ -> Light -> PointLight):



Розмістіть точкові джерела світла за своїм бажанням (їх навіть можна приєднувати на об’єкти, що рухаються)

Build Exe

Перейдіть Файл -> Build Settings, в Scene In Build впевніться, що підключені всі необхідні сцени і першою йде меню. Оберіть платформу (PC), натисніть Build: