Лабораторна робота №5

Система здоров’я для GameObjects

Мета: ознайомитися і набути навичок у створенні системи здоров'я, яка дозволить легко керувати життєвими показниками різних GameObjects у 2D грі.

Література

2D Game Kit Walkthrough: https://learn.unity.com/tutorial/2d-game-kit-walkthrough#

2D Game Kit Reference Guide: https://learn.unity.com/tutorial/2d-game-kit-reference-guide#5c7f8528edbc2a002053b76a

2D Game Kit: https://learn.unity.com/project/2d-game-kit

Зміст роботи

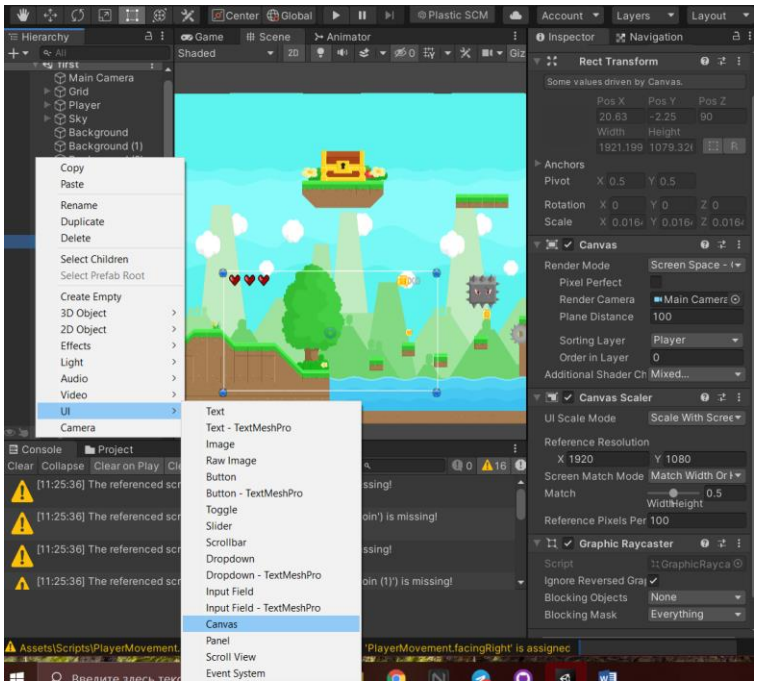
Завдання. Створити систему життя для персонажу.

У персонажа є 3 життя (три серця). При зіткненні з ворожим об'єктом життєві показники зменшуються. При підборі "аптечки" (монетки) життєві показники збільшуються.

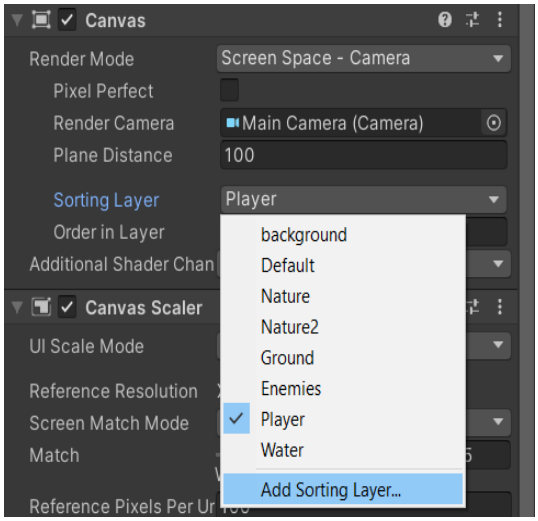
Умови: Якщо у персонажа максимальні життєві показники "аптечку" підняти не можна. Якщо життєві показники вичерпані персонаж припиняє своє існування, гра закінчується, відбувається перехід на сцену меню.

Методичні рекомендації

Додаємо об’єкт Canvas до сцени.

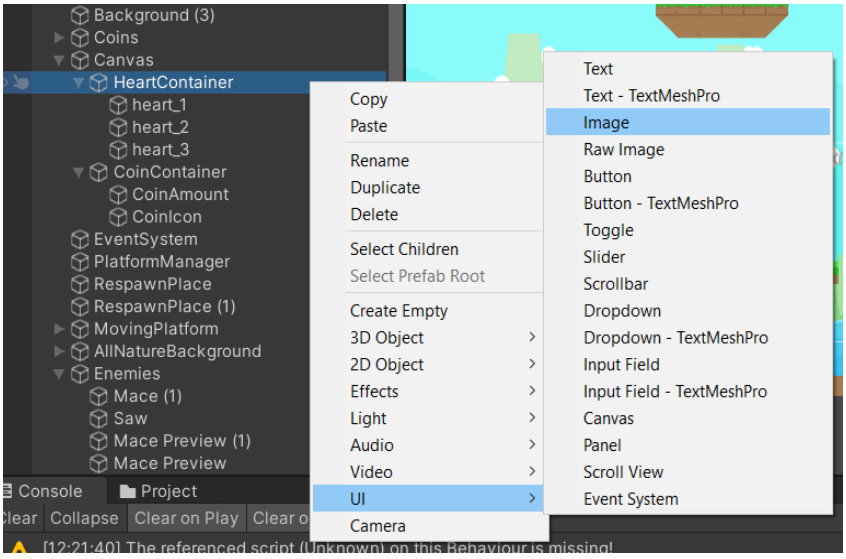


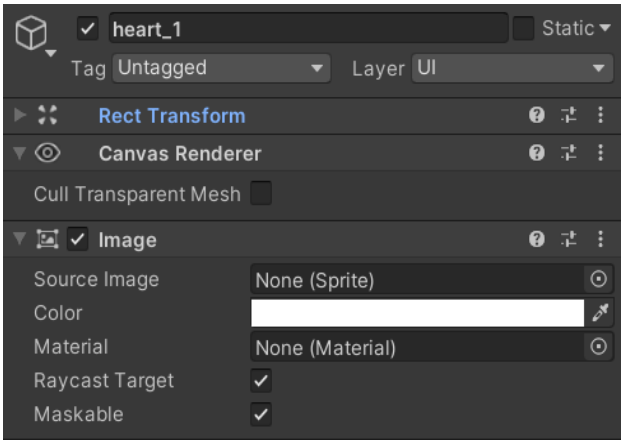
Обираємо Render Mode для Canvas – Screen Space – Camera



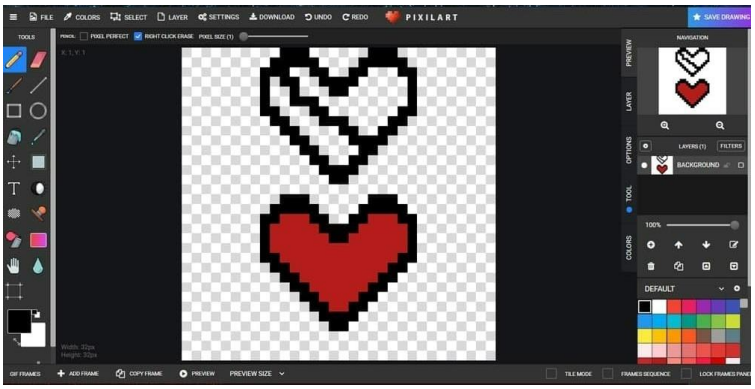
Створюємо Sorting Layer відмінний від навколишнього середовища. Цей шар може бути таким самим , як і на гравці .

Додаємо пустий об’єкт (контейнер) HeartContainer до сцени, в якому будуть знаходитись наші серця. Також створюємо контейнер CoinContainer для монет. Далі в залежності від бажаної кількості життів додаємо об’єкти Image до контейнеру Hearts. Для контейнеру Coins додаємо об’єкт Text та Image.

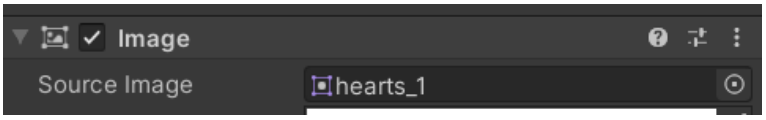




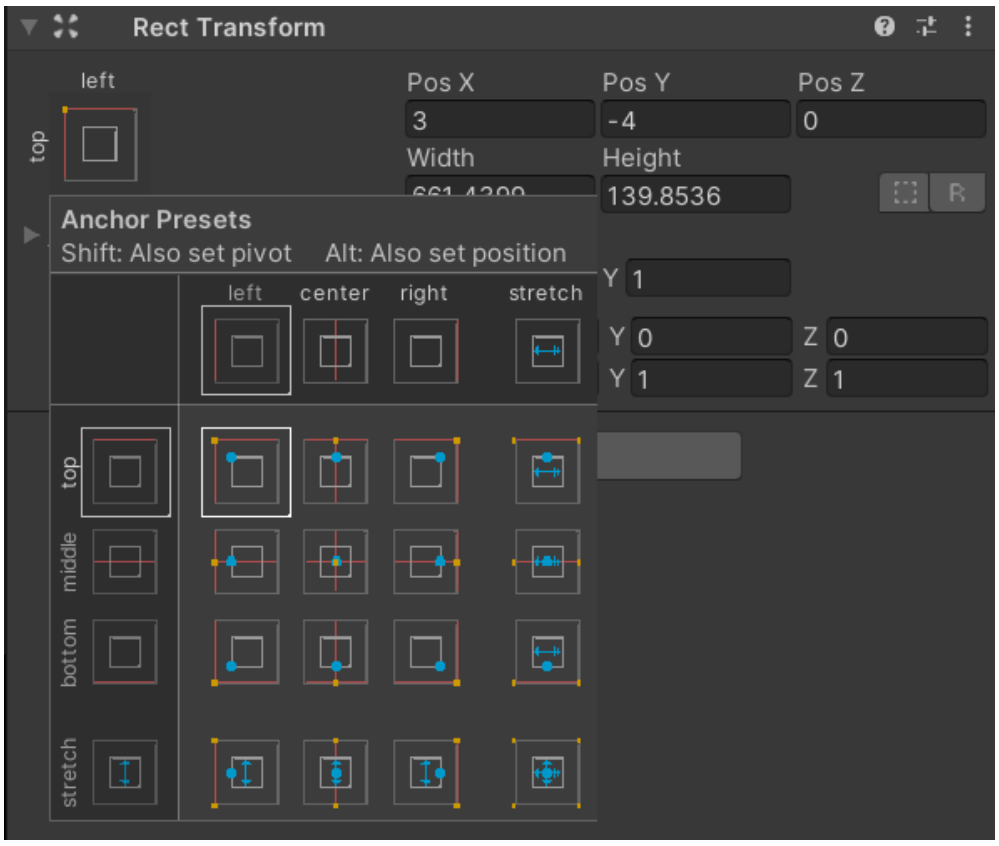
Далі потрібно створити іконки заповнених та пустих сердець. Для цього можна обрати графічний редактор яким ви вмієте користуватись, чи використати безкоштовний он-лайн ресурс для створення піксельної графіки https://www.pixilart.com/draw.



Після чого встановлюємо sprite mode – multiple, та нарізаємо наші спрайти. Тепер ми можемо додати спрайти до об’єктів Image контейнеру HeartContainer. Для цього відкриваємо об’єкт з компонентом Image та встановлюємо його поле Source Image.

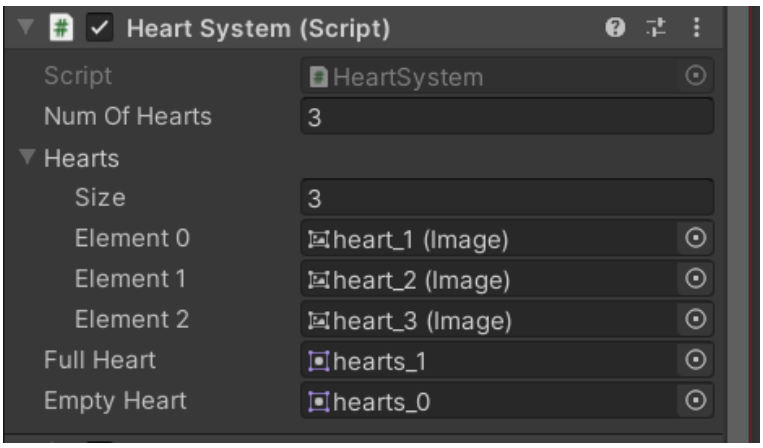


Щоб правильно розташувати HeartContainer, використаємо компонент Rect Transform. По-перше, потрібно щоб всі дочірні об’єкти нашого контейнера знаходились в позиції (0,0,0). По-друге, потрібно збільшити розміри самого контейнера, щоб вмістити усі серця. Далі натискаємо на квадрат, який знаходиться на компоненті Rect Transform, та натиснувши Shift+Alt обираємо до якого кута буде прив’язано HeartContainer.



Для кожного дочірнього об’єкту HeartContainer також можна встановити необхідну прив’язку в межах самого контейнера.

Створюємо скрипт HeartsSystem для гравця та додаємо його до об’єкту.



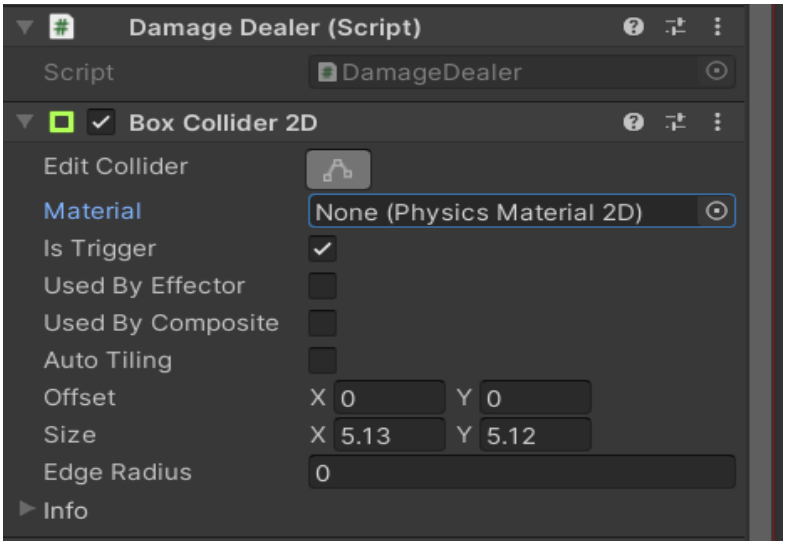
Класс HeartSystem має містити масив Hearts типу Image. Розташовані серці у сцені додаються у масив. Також Клас містить Sprite заповненого та порожнього серця.

В Update перевіряється кількість здоров’я. Якщо кількість сердець зменшується , тоді ми змінюємо sprite нашої картинки.

```
hearts[i].sprite = fullHeart;
```

Також потрібно створити функцію, яка буде зменшувати кількість сердець. Функція має перевіряти відсутність здоров’я та відображати відповідну сцену.

До перепон які будуть завдавати шкоди гравцю додаємо BoxColider2D та створюємо скрипт DamageDealer.

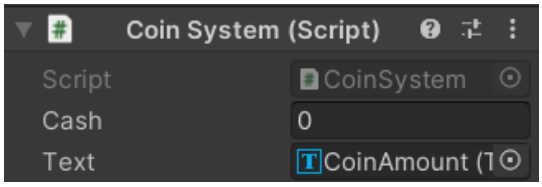


У скрипті потрібно створити функцію OnTriggerEnter2D. Ця функція буде спрацьовувати коли об’єкти з колайдерами зіткнуться. В параметрах функції можна отримати колайдер з яким зіткнувся об’єкт, та через ігровий об’єкт отримати інші його компоненти. Потрібно описати як саме буде відображатись отримання пошкодження в функції DealDamage.

```
private void OnTriggerEnter2D(Collider2D otherCollider)
{
    if (otherCollider.GetComponent<playerBehavior2>() != null)
    {
        otherCollider.gameObject.GetComponent<HeartSystem>().DamagePlayer(1);
    }

    DealDamage(otherCollider);
}
```

Відповідно до класу HeartSystem потрібно описати CoinSystem для збору монет.



Контрольні питання:

1. Які типи GameObjects у грі потребують системи здоров'я?
2. Які механіки отримання та відновлення здоров'я можна реалізувати у 2D гри?
3. Як можна візуально відображати стан здоров'я гравця та інших об'єктів?
4. Які події, пов'язані зі зміною стану здоров'я і як їх можна обробляти?
5. Як забезпечується гнучкість системи здоров'я, щоб її можна було легко налаштовувати для різних GameObjects?
6. Які компоненти Unity можна використовувати для реалізації системи здоров'я?
7. Як можна забезпечити ефективність системи здоров'я, особливо для великої кількості GameObjects?
8. Як можна провести тестування системи здоров'я, щоб переконатися в її коректності та надійності?
9. Які особливості 2D гри потрібно врахувати при розробці системи здоров'я?
10. Як можна інтегрувати систему здоров'я з іншими компонентами гри, такими як система ушкоджень та система інвентарю?
11. Які можливі проблеми можуть виникнути при реалізації системи здоров'я, і як їх вирішити?
12. Як забезпечити масштабованість системи здоров'я для великої кількості GameObjects?
13. Які існують альтернативні підходи до реалізації системи здоров'я в Unity, і чому ви обрали саме цей підхід?